Wind Energ. Sci., 11, 2567–2583, 2026
https://doi.org/10.5194/wes-11-2567-2026

eawe european academy of wind energy
Open Access WIND ENERGY SCIENCE

Large-eddy simulation of thermally stratified atmospheric boundary layers with a lattice Boltzmann method

Henry Korb[1], **Henrik Asmuth**[1], **Martin Schönherr**[2], **Martin Geier**[2], **and Stefan Ivanell**[1]

[1]Wind Energy Division, Department of Earth Sciences, Uppsala University, Visby, Sweden
[2]Institute for Computational Modeling in Civil Engineering, TU Braunschweig, Braunschweig, Germany

Correspondence: Henry Korb (henry.korb@geo.uu.se)

Received: 24 September 2025 – Discussion started: 20 October 2025
Revised: 13 June 2026 – Accepted: 24 June 2026 – Published: 21 July 2026

Abstract. Thermal stratification plays an important role in wind farm flows and must therefore be included in simulations of such flows. At the same time, wind farms are covering larger areas, requiring very large domains and leading to exceptional computational costs for large-eddy simulations (LESs). The lattice Boltzmann method (LBM) is a novel approach to LES of wind farm flows that is particularly efficient and suitable for massively parallel hardware, such as graphics processing units (GPUs). In this work, we present a novel model for LES-LBM of stratified atmospheric boundary layers, using a so-called double-distribution function approach. We develop a novel boundary condition to apply Monin–Obukhov similarity theory and implement a number of other components required for simulations of stratified boundary layers in the GPU-resident version of the open-source LBM solver VirtualFluids. The model is validated for conventionally neutral and stably stratified boundary layers. Results agree closely with numerical references. The model is able to simulate conventionally neutral boundary layers with parameters typical for wind energy applications of the order of real time on a single GPU. Future work will include development of a precursor–successor method for wind farm flow simulations and improvements to the collision operator of the temperature model.

1 Introduction

Wake recovery (Abkar and Porté-Agel, 2015), the persistence of wind farm wakes (Platis et al., 2022), and the occurrence of gravity waves (Lanzilao and Meyers, 2024) all depend on the thermal stratification of the atmospheric boundary layer (ABL). Large-eddy simulation (LES) enables the most accurate examination of these phenomena that is feasible with currently available hardware. Wind farms or clusters of wind farms cover large areas, yet simulations need to be well resolved to fully capture the behavior of the boundary and inversion layer (Allaerts and Meyers, 2017). The resulting simulations thus feature extremely large numbers of degrees of freedom. Current LES models for the ABL, typically CPU-resident finite-volume or pseudospectral solvers, require days or months of computing time on very large compute clusters to perform such simulations.

A new generation of solvers seeks to alleviate the immense computational cost by utilizing graphics processing units (GPUs), such as MicroHH (van Heerwaarden et al., 2017), AMR-Wind (Kuhn et al., 2025b), and FastEddy (Sauer and Muñoz-Esparza, 2020). van Heerwaarden et al. (2017) report that 32 CPU cores are necessary to achieve the same computational performance as one NVIDIA Quadro K6000. Sauer and Muñoz-Esparza (2020) have already reported that 1 GPU equals the performance of 256 CPU cores, highlighting the rapidly increasing speed of GPUs.

However, a different approach, based on the lattice Boltzmann method (LBM), has also been introduced to wind energy and boundary layer research over the last decade. A recent review can be found in Korb et al. (2026). The LBM's mathematical structure is well suited for the use of massively parallel hardware, such as GPUs. In the LBM, fluid is described as a set of populations on the nodes of a Carte-

sian grid. The LBM then follows a two-step algorithm. First, the populations at a node collide, then they are advected to neighboring nodes (Krüger et al., 2017). The collision step is potentially non-linear but is a local operation, while the advection step simply consists of moving memory on a computer. While initial formulations of the LBM were unstable at high Reynolds numbers, recent advances in the formulation of the collision step have rendered it a suitable method for high-Reynolds flows and LES (Geier et al., 2015, 2020; Jacob et al., 2018).

LES based on the LBM has been used now for over a decade to simulate large-scale boundary layer flows. Onodera et al. (2013) present a simulation of $10\,\mathrm{km} \times 10\,\mathrm{km}$ of Tokyo's urban area at a 1 m resolution on up to 1000 GPUs, demonstrating the method's suitability for very large problems. Further examples of simulations utilizing GPUs include King et al. (2017) and Lenz et al. (2019). Both report near-real-time computational performance, demonstrating the high computational efficiency of the LBM on GPUs. None of the aforementioned simulations include wall models. Asmuth et al. (2021) introduced a wall-modeling approach suitable for atmospheric boundary layers and showed very good agreement with reference results.

All of the aforementioned models only consider isothermal boundary layers, and very few models considering thermal stratification have previously been presented in the literature. The temperature equation can be discretized either via "traditional" approaches, such as finite difference or finite volume, or via a modified LBM. The former is named a hybrid approach, while the latter is referred to as a double-distribution function approach (DDF). A hybrid approach was applied in one of the earliest studies of stratified boundary layer flows with the LBM, the THELMA project, described in a series of publications (Obrecht et al., 2012, 2013, 2015). Another solver designed for atmospheric boundary layers implementing a hybrid approach is presented in Feng et al. (2021). PROLB employs the hybrid recursive regularized collision model (Jacob et al., 2018) and the wall-modeling approach by Malaspinas and Sagaut (2014). However, it is not mentioned that it utilizes GPUs, and no remarks on its computational performance are given. An overview of advection–diffusion LBM models can be found in Gruszczyński and Łaniewski-Wołłk (2022), where the authors also compare a number of more advanced LBM models. They find that models based on the cascaded LBM yield the most accurate results. A similar model is applied in Alihussein et al. (2021) to simulate the dissolution in porous media. Wang et al. (2020) present a method using the DDF approach for simulation of the stratified flow over a ridge. Both LBM models (for momentum and temperature) employ a multiple-relaxation time method (d'Humières, 1994).

In this paper, we propose a novel model to simulate the stratified atmospheric boundary layer via a DDF LBM model based on the cumulant LBM for momentum and a factorized cascaded model for the advection–diffusion equation. We describe the methodology in Sect. 2. We compare results obtained with our method to reference data for both a neutral and a stably stratified test case in Sect. 3 and give our concluding remarks in Sect. 4. As with any model development, we tried many different variants until we converged on the model we present here. We document some of those approaches in Appendix A.

2 Methodology

We will be begin by describing the fundamentals of the cumulant lattice Boltzmann method and its modifications to simulate atmospheric boundary layers. Subsequently, we present the method used to simulate the advection–diffusion of the potential temperature. Finally, we discuss novel formulations for boundary conditions and other aspects specific to modeling thermally stratified boundary layers.

2.1 Governing equations

Our aim is to simulate the filtered incompressible Navier–Stokes equations with Coriolis forces coupled to an advection–diffusion equation of potential temperature via the Boussinesq approximation (see Stoll et al., 2020, and references therein):

$$\frac{\partial u_i}{\partial x_i} = 0, \tag{1}$$

$$\frac{\partial u_i}{\partial t} + u_j \frac{\partial u_i}{\partial x_j} = -\frac{1}{\rho_0}\frac{\partial p}{\partial x_i} + \frac{\partial}{\partial x_j}\left(\nu_\mathrm{e}\frac{\partial u_i}{\partial x_j}\right) + F_i^\mathrm{C} + F_i^\mathrm{B}, \tag{2}$$

$$\frac{\partial \theta}{\partial t} + u_j \frac{\partial \theta}{\partial x_j} = \frac{\partial}{\partial x_j}\left(D_\mathrm{e}\frac{\partial \theta}{\partial x_j}\right). \tag{3}$$

Here, the coordinate system is denoted as $\boldsymbol{x} = [x, y, z]^\mathrm{T}$; t is time; ρ_0 is the density; $\boldsymbol{u}$ is the filtered velocity; p is the pressure deviation from the background pressure; θ is the filtered potential temperature; the Coriolis and buoyancy forces are $\boldsymbol{F}^\mathrm{C}$ and $\boldsymbol{F}^\mathrm{B}$, respectively; and subgrid stresses and heat flux are parameterized via an effective viscosity and diffusivity, ν_e and D_e, respectively. We use Einstein's summation convention. The Coriolis force is given by

$$F_i^\mathrm{C} = -\epsilon_{ij3}(G_j - u_j) f_\mathrm{c}, \tag{4}$$

where ϵ_{ijk} is the Levi–Civita symbol, $\boldsymbol{G} = G[\cos\alpha, \sin\alpha, 0]^\mathrm{T}$ is the geostrophic wind with the geostrophic wind speed G and direction α, and $f_\mathrm{c} = 2\Omega\sin\varphi$ is the Coriolis parameter that depends on the rotational speed of the Earth Ω and latitude φ. The Boussinesq approximation assumes $F_i^\mathrm{b} = g\frac{\theta-\theta_r}{\theta_r}\delta_{i3}$, where g is the gravitational acceleration, θ_r is a reference temperature, and δ_{ij} is the Kronecker delta. However, to remove the hydrostatic pressure, we take the horizontal average, denoted by $\langle\cdot\rangle$, of Eq. (2) with $i = 3$, following, e.g., Deardorff (1970). We find that $\frac{1}{\rho_0}\frac{\partial\langle p\rangle}{\partial z} = \frac{\langle\theta\rangle-\theta_0}{\theta_0}$, since the continuity equation dictates $\langle u_3\rangle = 0$. Splitting p

into $p' + \langle p \rangle$ and inserting the previous result in Eq. (2), we can write

$$F_i^{\mathrm{B}} = g \frac{\theta - \langle\theta\rangle(z)}{\theta_{\mathrm{r}}} \delta_{i3} \tag{5}$$

if we replace p by p'. Contrary to "classical" computational fluid dynamics, we do not discretize these equations but instead solve them via the lattice Boltzmann method implemented in the GPU-resident version of the open-source solver VIRTUALFLUIDS (Geier et al., 2025).

2.2 The cumulant lattice Boltzmann method

The fundamental variable of the lattice Boltzmann method is the particle distribution function (PDF) f. The PDF describes the probability of encountering a particle with velocity $\boldsymbol{\xi}$ at time t and location $\boldsymbol{x}$. The discretization of velocity space to a lattice of discrete velocities $\boldsymbol{c}_{\iota\kappa\lambda} = (\iota\delta_{i1} + \kappa\delta_{i2} + \lambda\delta_{i3})c$, with $c = \frac{\Delta x}{\Delta t}$ as lattice velocity, leads to the discrete populations $f_{\iota\kappa\lambda}(\boldsymbol{x}, t) := f(\boldsymbol{c}_{\iota\kappa\lambda}, \boldsymbol{x}, t)$. Following Geier et al. (2015), we denote lattice directions with triplets of Greek indices corresponding to their directions in space and define $\bar{\iota} := -\iota$. Note that Greek indices are not subject to the summation convention, and triplet indices in parentheses represent all possible permutations of that triplet. We employ a D3Q27 lattice, i.e., the set of 27 lattice directions of all permutations with $\iota, \kappa, \lambda \in \{-1, 0, 1\}$. The lattice speed of sound is $c_{\mathrm{s}} = \frac{c}{\sqrt{3}}$, and each $\boldsymbol{c}_{\iota\kappa\lambda}$ has an associated weight $w_{\iota\kappa\lambda}$. Macroscopic quantities, that is quantities on the scale of continuum mechanics, are obtained by taking different order moments of $f_{\iota\kappa\lambda}$, for example density (zeroth order) and velocity (first order):

$$\rho = \sum f_{\iota\kappa\lambda}, \tag{6}$$

$$\rho \boldsymbol{u} = \sum \boldsymbol{c}_{\iota\kappa\lambda} f_{\iota\kappa\lambda} + \frac{\boldsymbol{F}}{2}. \tag{7}$$

$\boldsymbol{F}$ is the total force density. The evolution of the PDF is described by the Boltzmann equation. Replacing the continuous PDFs with the discrete populations and integration along the characteristic $\boldsymbol{x} = \boldsymbol{c}_{\iota\kappa\lambda} t$ from t to $t + \Delta t$ yields the lattice Boltzmann equation:

$$f_{\iota\kappa\lambda}(\boldsymbol{x} + \boldsymbol{c}_{\iota\kappa\lambda}\Delta t, t + \Delta t) = f_{\iota\kappa\lambda}(\boldsymbol{x}, t) + \Delta t \Omega_{\iota\kappa\lambda}(\boldsymbol{x}, t), \tag{8}$$

where Ω is the collision operator, which is discussed later on. Asymptotic analysis shows that the moments of the lattice Boltzmann equation yield the weakly compressible Navier–Stokes equations, which approximate the incompressible Navier–Stokes equations with an error proportional to $\mathcal{O}(\mathrm{Ma}^3)$, with the Mach number $\mathrm{Ma} = \frac{V_0}{c_{\mathrm{s}}}$ and V_0 being a reference velocity. The size of this error is effectively controlled by the size of the time step and the grid spacing since

$$\mathrm{Ma} = \frac{\sqrt{3} V_0 \Delta t}{\Delta x}. \tag{9}$$

By limiting $\mathrm{Ma} < 0.1$, we ensure that the error is small. The collision operator is of great importance for the accuracy and stability of the method. In this work we employ the cumulant collision operator (Geier et al., 2015, 2017); here we present a short overview. Generally, collision operators relax the populations towards an equilibrium. The cumulant collision operator performs this relaxation in cumulant space, eliminating many shortcomings of traditional multi-relaxation time methods, mainly due to the fact that cumulants are statistically independent and can therefore be relaxed at independent rates. First, the populations in continuous form undergo a Laplace transformation to wave number space:

$$F(\boldsymbol{\Xi}) = \mathcal{L}\left\{ \sum_{\iota\kappa\lambda} f_{\iota\kappa\lambda} \delta(\iota c - \xi)\delta(\kappa c - \upsilon)\delta(\lambda c - \zeta) \right\}, \tag{10}$$

where δ is the Dirac delta function. Thereafter, cumulants $c_{\alpha\beta\gamma}$ are obtained from the cumulant generating function

$$c_{\alpha\beta\gamma} = c^{-\alpha-\beta-\gamma} \frac{\partial^\alpha \partial^\beta \partial^\gamma}{\partial \Xi^\alpha \partial \Upsilon^\beta \partial \mathrm{Z}^\gamma} \ln\left(F(\boldsymbol{\Xi})\right)\Big|_{\Xi=\Upsilon=\mathrm{Z}=0}. \tag{11}$$

Note that transformation from populations to cumulants is implemented via the chimera transform, which greatly reduces the computational cost while significantly improving the numerical precision of the computation (Geier et al., 2015, Appendix I). After transformation, the cumulants are relaxed towards their respective equilibrium $c^{\mathrm{eq}}_{\alpha\beta\gamma}$:

$$c^*_{\alpha\beta\gamma} = \omega_{\alpha\beta\gamma} c^{\mathrm{eq}}_{\alpha\beta\gamma} + (1 - \omega_{\alpha\beta\gamma}) c_{\alpha\beta\gamma}, \tag{12}$$

where $c^*_{\alpha\beta\gamma}$ denotes the post-collision cumulant. Finally, the post-collision cumulants are transformed back to populations. The relaxation rates $\omega_{\alpha\beta\gamma}$ are computed according to Geier et al. (2017). The relaxation rate of the second-order cumulants $\omega_{(110)}$ is related to the kinematic viscosity by

$$\frac{1}{\omega_{(110)}} = \frac{\nu}{c_{\mathrm{s}}^2 \Delta t} + \frac{1}{2}. \tag{13}$$

In this study, we employ an eddy-viscosity model to explicitly model the subgrid scales of the LES. Some studies, e.g., Geier et al. (2020) and Gehrke and Rung (2022), suggest using the cumulant operator alone to conduct implicit LES. However, we have found this method unsuitable for performing LES of the atmospheric boundary layer due to the very high Reynolds number, as discussed in Asmuth et al. (2021).

2.3 The lattice Boltzmann method for advection–diffusion

To solve the advection–diffusion equation, one can either use a so-called hybrid solver that solves the Navier–Stokes equations via the LBM and the advection–diffusion equation via finite differences or use the finite-volume

method. The other possibility is to use another LBM solver to solve the advection–diffusion equation with a double-distribution function (DDF) approach. The hybrid method has the advantage that it requires significantly less memory, since for every node in the grid we only have to save one quantity, as compared to the DDF, which needs to save 27 (if one uses a D3Q27 lattice) quantities per node. Therefore, we also implemented a hybrid approach first, but, despite much effort, it was not successful. We discuss the details of the approaches we tried in Appendix A. Instead, we pivoted to a DDF approach: We can describe a scalar, such as the potential temperature θ, with a second set of populations, $g_{\iota\kappa\lambda}$, and define

$$\theta = \sum g_{\iota\kappa\lambda}. \tag{14}$$

During collision, only the zeroth-order moment, i.e., θ, is conserved. The lattice Boltzmann equation for the advection–diffusion problem is analogous to the formulation for momentum:

$$g_{\iota\kappa\lambda}(\boldsymbol{x} + \boldsymbol{c}_{\iota\kappa\lambda}\Delta t, t + \Delta t) = g_{\iota\kappa\lambda}(\boldsymbol{x}, t) + \Delta t \Omega^{\mathrm{AD}}_{\iota\kappa\lambda}(\boldsymbol{x}, t). \tag{15}$$

In this work, we employ the factorized central moment-based collision operator described in Yang et al. (2016). Similar to the cumulant method, the populations $g_{\iota\kappa\lambda}$ first undergo a Laplace transform,

$$G(\Xi) = \mathcal{L}\left\{\sum_{\iota\kappa\lambda} g_{\iota\kappa\lambda}\delta(\iota c - \xi)\delta(\kappa c - \upsilon)\delta(\lambda c - \zeta)\right\}. \tag{16}$$

Central moments $\tilde{\kappa}$ are then obtained from the moment-generating function,

$$\tilde{\kappa}_{\alpha\beta\gamma} = c^{-\alpha-\beta-\gamma}\frac{\partial^\alpha\partial^\beta\partial^\gamma}{\partial\Xi^\alpha\partial\Upsilon^\beta\partial Z^\gamma}e^{-\boldsymbol{u}\cdot\Xi}G(\Xi)\bigg|_{\Xi=\Upsilon=Z=0}. \tag{17}$$

Again, the computation is performed via the chimera transform. To obtain the factorized central moments $\kappa_{\alpha\beta\gamma}$, the following orthogonalization is applied:

$$\begin{aligned}
\kappa_{000} &= \tilde{\kappa}_{000}\\
\kappa_{(100)} &= \tilde{\kappa}_{(100)}\\
\kappa_{(110)} &= \tilde{\kappa}_{(110)}\\
\kappa_{111} &= \tilde{\kappa}_{111}\\
\kappa_{(200)} &= \tilde{\kappa}_{(200)} - \frac{1}{3}\kappa_{000}\\
\kappa_{(210)} &= \tilde{\kappa}_{(210)} - \frac{1}{3}\kappa_{(010)}\\
\kappa_{(211)} &= \tilde{\kappa}_{(211)} - \frac{1}{3}\kappa_{(011)}\\
\kappa_{(220)} &= \tilde{\kappa}_{(220)} - \frac{1}{3}\kappa_{000}\\
\kappa_{(221)} &= \tilde{\kappa}_{(221)} - \frac{1}{9}\kappa_{(001)}\\
\kappa_{222} &= \tilde{\kappa}_{222} - \frac{1}{27}\kappa_{000}.
\end{aligned}$$

The factorized central moments are then relaxed towards their equilibria:

$$\kappa^*_{\alpha\beta\gamma} = \omega_{\alpha\beta\gamma}\kappa^{\mathrm{eq}}_{\alpha\beta\gamma} + (1 - \omega_{\alpha\beta\gamma})\kappa_{\alpha\beta\gamma}. \tag{18}$$

All equilibria are zero, except

$$\kappa^{\mathrm{eq}}_{(200)} = c_{\mathrm{s}}^2\kappa_{000}, \tag{19}$$

$$\kappa^{\mathrm{eq}}_{(220)} = c_{\mathrm{s}}^4\kappa_{000}, \tag{20}$$

$$\kappa^{\mathrm{eq}}_{(222)} = c_{\mathrm{s}}^6\kappa_{000}. \tag{21}$$

The first-order relaxations are related to the diffusivity by

$$\frac{1}{\omega_{(100)}} = \frac{D}{c_{\mathrm{s}}^2\Delta t} + \frac{1}{2}, \tag{22}$$

while we set all other relaxation rates to one. After the relaxation, the factorized central moments have to be transformed back to central moments and then populations.

2.4 Boundary conditions

We require two boundary conditions for both momentum and potential temperature fields. At the top of the domain we set a slip condition for the fluid flow and either a Neumann- or a Dirichlet-type boundary condition for potential temperature. At the bottom we set a combined stress and flux boundary condition computed from a wall model that is discussed in Sect. 2.5. Boundary conditions in the LBM have to be specified for the populations; therefore a variety of methods can be found resulting in the same macroscopic boundary condition. For the sake of completeness, we describe the boundary conditions for the fluid in more detail in Appendix B. A Dirichlet-type boundary condition for the scalar can be implemented via the anti-bounce-back rule as described in Krüger et al. (2017, p. 318). A Neumann boundary condition

can be implemented via this approach as well. However, in preliminary studies, we found this approach to cause spurious oscillations at the top of the domain.

Instead, we present here a formulation for a flux boundary condition, which will also be used as a Neumann boundary condition at the top of the domain. We first recall a few basic relations. The total flux $\boldsymbol{j}$ is the sum of the diffusive and advective fluxes $\boldsymbol{j}^{\mathrm{D}}$ and $\boldsymbol{j}^{\mathrm{A}}$:

$$\boldsymbol{j} = \boldsymbol{j}^{\mathrm{D}} + \boldsymbol{j}^{\mathrm{A}} = -D\nabla\theta + \boldsymbol{u}\theta. \tag{23}$$

Our goal is now to set a specified wall flux q_{w}, which will be computed either from a wall model or, in the case of a Neumann boundary condition, from $q_{\mathrm{w}} = -D\frac{\partial\theta}{\partial n}$, where $\frac{\partial\theta}{\partial n}$ is the specified gradient in the wall normal direction $\boldsymbol{n}$, with $\boldsymbol{n}$ pointing to the fluid domain.

We compute the diffusive flux at the node from the first-order moment of $\boldsymbol{g}$:

$$j_i^{\mathrm{D}} = \sum g_{\iota\kappa\lambda} c_{\iota\kappa\lambda,i} - u_i\theta. \tag{24}$$

We then prescribe the flux at the wall $\boldsymbol{j}^{\mathrm{w}}$ to be equal to the diffusive flux in the tangential direction and equal to q_{w} in the wall normal direction:

$$j_i^{\mathrm{w}} = j_i^{\mathrm{D}} - \left(j_j^{\mathrm{D}} n_j\right) n_i + q_{\mathrm{w}} n_i. \tag{25}$$

Finally, we employ the bounce-back rule to compute the missing distributions $g_{\overline{\iota\kappa\lambda}}$:

$$g_{\overline{\iota\kappa\lambda}} = g_{\iota\kappa\lambda} - 2w_{\iota\kappa\lambda}\frac{c_{\iota\kappa\lambda,i}\, j_i^{\mathrm{w}}}{c_{\mathrm{s}}^2}. \tag{26}$$

2.5 Wall model

At the bottom boundary, we make use of the standard Monin–Obukhov similarity theory (MOST) to determine the wall shear stress τ_{w} and heat flux q_{w}:

$$\zeta(z) = \frac{z}{L} = -\frac{z\kappa g q_{\mathrm{w}}}{u_*^3\theta_{\mathrm{r}}}, \tag{27}$$

$$u(z) = \frac{u_*}{\kappa}\left(\ln\frac{z}{z_0} - \psi_{\mathrm{M}}(\zeta)\right), \tag{28}$$

$$\theta(z) - \theta_0 = -\frac{q_{\mathrm{w}}}{u_*\kappa}\left(\ln\frac{z}{z_{0,\mathrm{H}}} - \psi_{\mathrm{H}}(\zeta)\right), \tag{29}$$

where ζ is a stability parameter based on the Obukhov length L; $u_* = \sqrt{\frac{\tau_{\mathrm{w}}}{\rho}}$ is the friction velocity; κ is the von Kármán constant; z_0 and $z_{0,\mathrm{H}}$ are roughness lengths for momentum and temperature, respectively; and θ_0 is the surface temperature (Arya, 2001). The similarity functions for momentum ($\psi_{\mathrm{M}}(\zeta)$) and heat ($\psi_{\mathrm{H}}(\zeta)$) have to be determined experimentally. We use the classical Businger–Dyer relations (Businger et al., 1971):

$$\phi_{\mathrm{M}}(\zeta) = (1 - \gamma_{\mathrm{M}}\zeta)^{-1/4}, \tag{30}$$

$$\phi_{\mathrm{H}}(\zeta) = (1 - \gamma_{\mathrm{H}}\zeta)^{-1/2}, \tag{31}$$

$$\psi_{\mathrm{M}}(\zeta) = \begin{cases} \ln\left[\left(\frac{1+\phi_{\mathrm{M}}^2}{2\phi_{\mathrm{M}}^2}\right)\left(\frac{1+\phi_{\mathrm{M}}}{2\phi_{\mathrm{M}}}\right)^2\right] \\ \quad -2\tan^{-1}\left(\frac{1}{\phi_{\mathrm{M}}}\right) + \frac{\pi}{2}, \quad \zeta < 0, \\ -\beta_{\mathrm{M}}\zeta, \quad \zeta \geq 0 \end{cases} \tag{32}$$

$$\psi_{\mathrm{H}}(\zeta) = \begin{cases} 2\ln\left(\frac{1+\phi_{\mathrm{H}}}{2\phi_{\mathrm{H}}}\right), \quad \zeta < 0 \\ -\beta_{\mathrm{H}}\zeta, \quad \zeta \geq 0 \end{cases}. \tag{33}$$

To facilitate the comparison with reference data from Beare et al. (2006) in Sect. 3.2, we set $\beta_{\mathrm{M}} = 4.8$ and $\beta_{\mathrm{H}} = 7.8$. However, no values for unstable stratification are reported in Beare et al. (2006). We therefore set $\gamma_{\mathrm{M}} = \gamma_{\mathrm{H}} = 15$, as suggested by Arya (2001). Based on a user-specified distance z_{EL}, we sample an exchange-location temperature θ_{EL} and velocity $\boldsymbol{u}_{\mathrm{EL}}$. Additionally, the exchange-location quantities are exponentially averaged over time, as is recommended by Yang et al. (2017). We do not apply any spatial averaging. The user can choose to prescribe either the surface heat flux or the surface temperature. We employ a variation of algorithm(1) or algorithm(2) from Basu et al. (2008), depending on the prescribed quantity, displayed in Algorithm 1. In the following, the subscript t denotes vectors tangential to the wall, overbars denote temporal averages, and quantities at the first node in the fluid domain are denoted with the subscript 1. Superscripts indicate the time step.

Algorithm 1 Algorithm for computing wall shear stress and kinematic heat flux.

$u_*^{\mathrm{t}} \leftarrow u_*^{t-1}$, $q_{\mathrm{w}}^{\mathrm{t}} \leftarrow q_{\mathrm{w}}^{t-1}$
repeat
  $u_*^{\mathrm{Old}} \leftarrow u_*^{\mathrm{t}}$
  compute ζ via Eq. (27)
  compute ψ_{H} and ψ_{M} via Eqs. (32) and (33)
  $u_*^{\mathrm{t}} = \kappa\left|\overline{\boldsymbol{u}_{\mathrm{EL,t}}^{\mathrm{t}}}\right|\left(\ln\frac{z_{\mathrm{EL}}}{z_0} - \psi_{\mathrm{M}}\right)^{-1}$
  if surface temperature given **then**
    $q_{\mathrm{w}}^{\mathrm{t}} = -u_*^{\mathrm{t}}\kappa(\overline{\theta}_{\mathrm{EL}} - \theta_0)\left(\ln\frac{z_{\mathrm{EL}}}{z_{0,\mathrm{H}}} - \psi_{\mathrm{H}}\right)^{-1}$
  end if
until $\frac{u_*^{\mathrm{t}} - u_*^{\mathrm{Old}}}{u_*^{\mathrm{Old}}} < 10^{-4}$
$\boldsymbol{\tau}_{\mathrm{w}} = \rho(u_*^{\mathrm{t}})^2\frac{\boldsymbol{u}_{1,\mathrm{t}}}{|\boldsymbol{u}_{1,\mathrm{t}}|}$

The combined boundary condition, which we refer to as the *surface layer boundary condition*, is executed in the following steps:

1. Load $f_{\iota\kappa\lambda}^*$ at the boundary node, and compute ρ_1 and $\boldsymbol{u}_1$ via Eqs. (6) and (7).

2. Load $g^*_{\iota\kappa\lambda}$ at the boundary node, and compute θ_1 via Eq. (14).

3. Compute τ_w and q_w according to Algorithm 1 using $\boldsymbol{u}_{EL}$, θ_{EL}, z_0, $z_{0,H}$, z_{EL}, and θ_0.

4. Apply the inverse momentum exchange method to determine $\boldsymbol{u}^w$ and $f_{\overline{\iota\kappa\lambda}}$.

5. Apply the flux boundary condition to determine $g_{\overline{\iota\kappa\lambda}}$.

Thus, the boundary condition is entirely local, with the exception of $\boldsymbol{u}_{EL}$ and θ_{EL}. Furthermore, the populations $g_{\overline{\iota\kappa\lambda}}$ can be computed independently, making the boundary condition easily adaptable to curved boundaries. However, the populations $f_{\overline{\iota\kappa\lambda}}$ cannot be computed independently.

2.6 Further models

A number of further modifications to VirtualFluids had to be implemented in order for it to be fully equipped to conduct simulations of atmospheric boundary layers. Namely, Coriolis and buoyancy forces have to be computed, and a Rayleigh damping layer has to be implemented.

2.6.1 Coriolis force

The Coriolis force can be computed directly from Eq. (4) based on a user-prescribed geostrophic wind and Coriolis parameter. The Coriolis force is simply added to the body force field in our implementation. Further potential for optimization by combining the computation with the collision kernel was left for future work.

2.6.2 Buoyancy force

As described in the beginning, we model the effect of buoyancy via the Boussinesq approximation. After the collision kernel for g, we add a number of models to compute the buoyancy. For simplicity's sake, we allocate an array with the size of the grid for a local reference temperature. The constant-buoyancy provider only computes a buoyancy force according to Eq. (5) and adds it to the body force field. Thus we can implement a constant-reference-temperature profile simply by changing the way that the reference temperature is initialized. The second variant computes buoyancy relative to the horizontally averaged temperature, as described by Eq. (5).

2.6.3 Damping layer

Since the free atmosphere is essentially an undamped oscillator, spurious oscillations that can arise at the top of the capping inversion propagate throughout the domain. To mitigate these waves, it is common practice to use Rayleigh damping layers (Khan et al., 2025). The force of in the damping layer F_D is computed by

$$F_i^D = -f\left(\frac{z - z_s}{z_e - z_s}\right) w\delta_{i3}. \tag{34}$$

The function $f(\tilde{z})$ of height normalized between start and end heights z_s and z_e of the damping layer can be chosen freely in our implementation but is normally set to $f(\tilde{z}) = f_R \sin^2\frac{\pi}{2}\tilde{z}$, with a damping factor roughly $f_R = 1 \times 10^{-4}\,1\,s^{-1}$.

2.7 Turbulence models

As mentioned in Sect. 2.1, we parameterize subgrid-scale fluxes with effective viscosity and diffusivity models. A few turbulent viscosity models have been implemented in previous studies, namely the Smagorinsky–Lilly (Smagorinsky, 1963; Lilly, 1966), QR (Verstappen, 2011), and anisotropic minimum dissipation (AMD) (Rozema et al., 2015) models. Note that due to the relation between second-order cumulants and the stress tensor, the Smagorinsky and QR models can be computed very efficiently during the collision step. In addition, we have implemented a number of turbulent diffusivity models for this study, two standalone diffusivity models, namely a constant turbulent Prandtl number model and the model suggested by Moeng (1984). Finally, we have also implemented the stratified AMD model proposed by Abkar et al. (2016), which augments the original AMD model for turbulence viscosity with a term modeling the effect of buoyancy on turbulence and includes a turbulence diffusivity. As this model requires the full velocity gradient tensor, which is only available in between collisions, we compute the effective turbulence viscosity and diffusivity for computing the collision at time step t from the velocity and temperature field at time $t-1$. In preliminary studies, we found the stratified AMD model to yield the best results; therefore we use only this model in the remainder of this study.

3 Results

We validate our model against reference data in two stability conditions, namely conventionally neutral and stable conditions. We provide a convergence study of the advection–diffusion model in Appendix C and find the order of convergence to be slightly above 2 for both advection and diffusion.

3.1 Conventionally neutral boundary layer

We begin validation of our model for thermally stratified boundary layers by comparing them to the conventionally neutral boundary layer (CNBL) simulation described in Berg et al. (2020). This case has also been used to validate the LES solver AMR-Wind, and we therefore have two references to compare them to. The geostrophic wind speed G is $5\,m\,s^{-1}$, the Coriolis parameter is $10^{-4}\,1\,s^{-1}$, and the lapse rate of the free atmosphere is $\partial\theta/\partial z = 3\,K\,km^{-1}$.

The reference temperature is $\theta_r = 290\,\mathrm{K}$, and gravitational acceleration is $9.81\,\mathrm{m\,s^{-1}}$. The domain has an extent of $2560\,\mathrm{m} \times 2560\,\mathrm{m} \times 896\,\mathrm{m}$. We conduct simulations at two resolutions, $\Delta x = 7\,\mathrm{m}$ and $\Delta x = 3.5\,\mathrm{m}$, corresponding to grids B and C of the original publication, respectively. The boundaries in the streamwise and lateral directions are periodic. At the top we employ a slip condition for momentum and the Neumann condition, as described in Sect. 2.4, for the potential temperature, with a temperature gradient equal to the free atmosphere lapse rate. The bottom boundary is a surface layer boundary condition with a prescribed heat flux of $0\,\mathrm{K\,m\,s^{-1}}$ and a roughness length of $z_0 = 0.05\,\mathrm{m}$. The domain is initialized with constant geostrophic wind speed and a constant temperature gradient equal to the lapse rate of the free atmosphere. Further details of the reference case can be found in Berg et al. (2020). We assume an eddy turnover time $T_E = 1700\,\mathrm{s}$ and average from 55 to 65 T_E. We use the stratified AMD model with a model constant set to $1/3$. Since we use an explicit subgrid-scale model, we "turn off" the limiter of the cumulants by setting it to 10^5. No damping is activated.

The original study employs a pseudospectral code developed at the National Center for Atmospheric Research (NCAR) over the last 40 years, with pseudospectral spatial discretization in horizontal directions and second-order finite differences in the vertical direction. Time stepping is performed with a third-order Runge–Kutta scheme. In addition to the results from the original publication, we also compare our results to the results published in the Exawind benchmark database (Kuhn et al., 2025a) obtained with AMR-WIND. AMR-WIND utilizes a combination of finite-volume and finite-element methods for spatial discretization and second-order accurate time stepping; details on AMR-WIND can be found in Kuhn et al. (2025b). We compare our results to results obtained on grids C and D, with a resolution of $\Delta x = 3.5\,\mathrm{m}$ and $\Delta x = 1.75\,\mathrm{m}$, respectively, as these are the resolutions available in the Exawind benchmark database.

We provide a qualitative impression of the simulation in Figs. 1 and 2, where we show instantaneous horizontal wind speed and vertical velocity at $z = 35\,\mathrm{m}$ and $z = 333\,\mathrm{m}$, equivalent to 10 % and 90 % of the boundary layer height, respectively. Similar plots are shown in Berg et al. (2020). At the lower height we see the dominance of small-scale turbulent structures in both horizontal and vertical directions, as expected due to the proximity of the wall. At higher resolution we can observe that smaller scales are resolved. Close to the inversion height, we find much fewer small scales; instead, larger structures dominate. Comparing the direction of the mean horizontal velocity indicated by the black arrow, we observe the expected veer.

Moving to a more quantitative analysis, we show the horizontal averages of wind speed S, wind direction ϕ, and potential temperature of our model, referred to as VF, alongside respective results from references in Fig. 3. Overall, we find very good agreement between our results and the references. Within the boundary layer in particular we observe very close

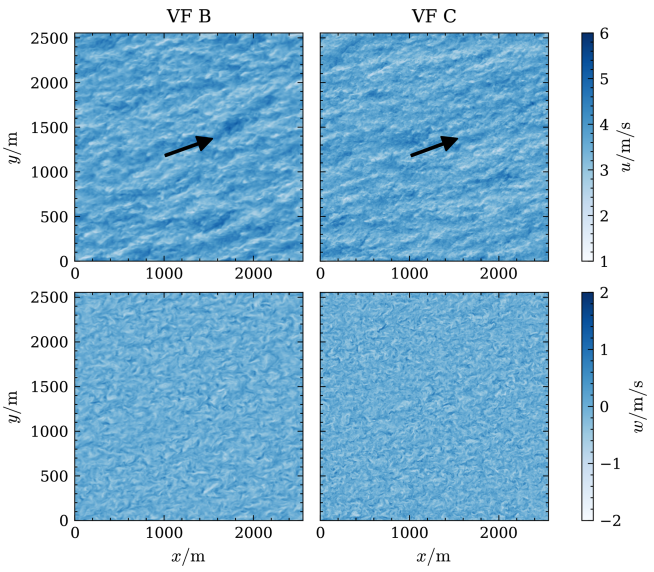

Figure 1. Instantaneous velocity fields of the CNBL case for grids B (left) and C (right). Horizontal wind speed (top) and vertical velocity (bottom) in the plane at $z = 37\,\mathrm{m}$ are shown. The black arrow in the top row indicates the average wind direction in the plane.

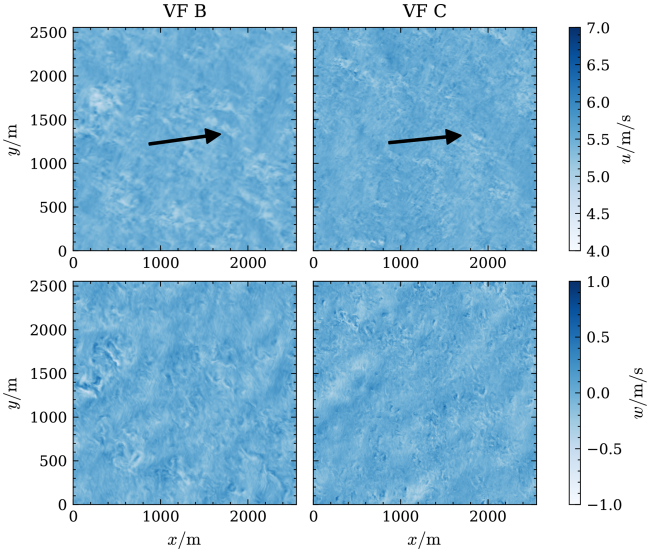

Figure 2. Instantaneous velocity fields of the CNBL case for grids B (left) and C (right). Horizontal wind speed (top) and vertical velocity (bottom) in the plane at $z = 333\,\mathrm{m}$ are shown. The black arrow indicates the average wind direction in the plane.

agreement in all quantities, indicating that the boundary condition and other models behave correctly. However, we observe that the B grid is not able to properly resolve the upper edge of the capping inversion, resulting in a weaker gradient. This is in line with observations in Berg et al. (2020) and a range of other literature, for example van Heerwaarden et al. (2017). Furthermore, we observe that the wind direction in the free atmosphere is not exactly aligned with the

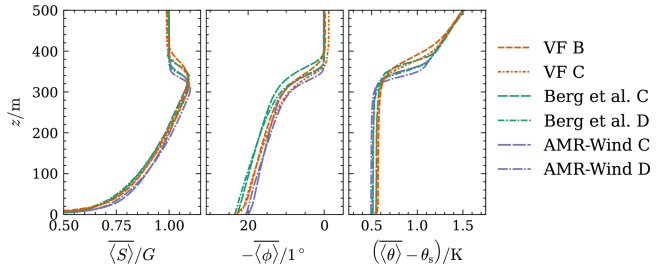

Figure 3. Vertical profiles of averaged velocity (left), wind veer (center), and temperature (right) of the CNBL reference case.

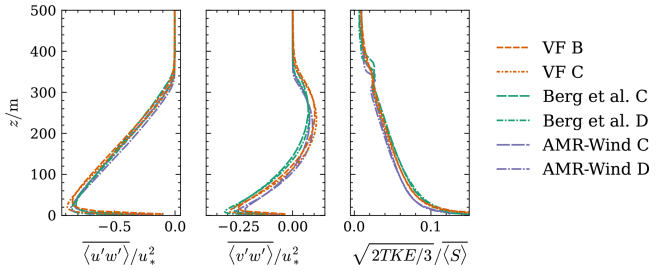

Figure 4. Vertical profiles of resolved vertical momentum flux in the streamwise (left) and lateral (center) directions and resolved turbulence intensity (right) from the CNBL reference case.

Table 1. Friction velocity and inversion height of the CNBL case compared to references.

Quantity	VIRTUALFLUIDS		Berg et al. (2020)		AMR-WIND	
	B	C	C	D	C	D
$u_*/\mathrm{m\,s^{-1}}$	0.207	0.205	0.225	0.221	0.208	0.203
z_i/m	381.5	362.3	372	350	352.1	337.0

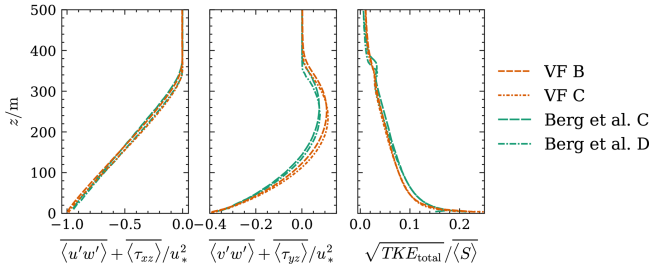

Figure 5. Vertical profiles of total vertical momentum flux in the streamwise (left) and lateral (center) directions and total turbulence intensity (right).

geostrophic wind for the higher-resolution case. This inaccuracy is due to the forcing of the geostrophic wind being very small compared to the streamwise velocity in the free atmosphere. See Appendix E for more details. Nevertheless, the error is small and seems to have a negligible effect on the wind direction below the inversion.

From the averaged results we can compute an inversion height z_i as the height with the maximum temperature gradient. We list our results next to the results reported by Berg et al. (2020) and AMR-WIND in Table 1. Furthermore, we list the friction velocity computed from the wall model. All results agree closely. The inversion height decreases with higher resolution for all solvers. AMR-WIND reports the lowest inversion heights, while our results for grid C are in the middle. We report the lowest friction velocity, while Berg et al. (2020) report the highest. Nevertheless, the results agree within 10 % of each other.

We show second-order statistics in Fig. 4, where we present the vertical momentum flux in the streamwise ($u'w'$) and lateral directions ($v'w'$) and turbulence intensity based on the horizontally averaged wind speed, computed as $\sqrt{2\mathrm{TKE}/3}/\overline{\langle S\rangle}$, where $\mathrm{TKE} = \frac{1}{2}(\overline{\langle u'u'\rangle} + \overline{\langle v'v'\rangle} + \overline{\langle w'w'\rangle})$ is the resolved turbulence kinetic energy. Again, we find very good agreement with both references. The streamwise flux even shows excellent agreement. We can see the influence of the mismatch in wind direction in the lateral vertical momentum flux, which is slightly too high in the upper region of the boundary layer. The turbulence intensity is slightly higher near the ground than the results from AMR-WIND, while the agreement with the results from Berg et al. (2020) is very close. The prominent increase in turbulence intensity at the inversion height observed in AMR-WIND is significantly less prominent in our results, even at the same resolution.

To evaluate the performance of the SGS model in more detail, we present the total flux as the sum of resolved and subgrid-scale fluxes $\tau_{ij} = \nu_e(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i})$ and the total turbulence intensity $\mathrm{TKE}_\mathrm{total} = \mathrm{TKE} + \frac{1}{2}\overline{\langle \tau_{ii}\rangle}$ in Fig. 5. Note that total fluxes are only available from Berg et al. (2020). Again, we find very close agreement with the reference data. Only the lateral flux deviates somewhat from the reference results in the upper half of the boundary layer.

Finally, we compare spatial velocity spectra obtained at three different heights ($z = 37$, 186, and 333 m) in Fig. 6. We compute spectra from horizontal planes following the procedure described in Berg et al. (2020). First, we compute the spectral tensors Φ_{11}, Φ_{22}, and Φ_{33} from the covariance function $R_{ij}(r_x, r_y, z) = \langle u_i'(x, y, z)u_j'(x + r_x, y + r_y, z)\rangle$:

$$\Phi_{ij}(k_\mathrm{x}, k_\mathrm{y}, z) = \frac{1}{(2\pi)^2}\iint R_{ij}e^{\hat{\mathrm{I}}(r_\mathrm{x}k_\mathrm{x}+r_\mathrm{y}k_\mathrm{y})}\mathrm{d}r_\mathrm{x}\mathrm{d}r_\mathrm{y}, \tag{35}$$

where $\hat{\mathrm{I}}$ is the imaginary unit.

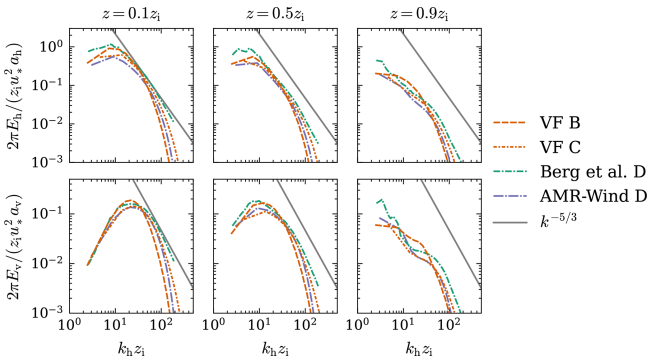

Figure 6. Horizontal (top) and vertical (bottom) spectra at $z = 37$ m (left), $z = 186$ m (center), and $z = 333$ m (right).

Then, we compute the ring-averaged energy in horizontal and vertical spectra $E_h(k_h)$ and $E_v(k_h)$ with $k_h = \sqrt{k_x^2 + k_y^2}$:

$$E_h(k_h) = \frac{1}{2}\int_0^{2\pi} \Phi_{11}(k_h, \phi) + \Phi_{22}(k_h, \phi) \mathrm{d}\phi, \tag{36}$$

$$E_v(k_h) = \int_0^{2\pi} \Phi_{33}(k_h, \phi) \mathrm{d}\phi. \tag{37}$$

Results are binned into 50 equally sized bins. Wavelengths are normalized with the computed inversion height, and spectra are normalized by $\frac{2\pi}{z_i u_* a_h}$ and $\frac{2\pi}{z_i u_* a_v}$, with $a_h = 0.54(55/18)a$, $a_v = 0.61(55/18)a$, and $a = 0.5$, in accordance with Berg et al. (2020). As a final step, we average the spectra computed from 10 time steps to reduce noise.

Our results match the results from AMR-WIND very closely at all heights despite the lower resolution. This demonstrates the lower dissipativity of the cumulant LBM compared to finite-volume solvers. At lower wavenumbers there is also very good agreement with the results from Berg et al. (2020). At high wavenumbers, the LBM is more dissipative than the pseudospectral solver. Close to the inversion height, the small wavenumbers are lower due to the capping inversion, and a characteristic hump is visible in the vertical spectra, which we could accurately reproduce with our solver.

Overall we find very good agreement in all examined quantities with the reference data, even at lower resolutions. In general, the accuracy of the model is positioned between the two reference models. The newly developed surface boundary condition is able to model the wall region accurately in neutral conditions.

Our simulations were carried out using a single NVidia RTX A6000 GPU on a workstation computer. Simulating 1.225×10^5 s with grid B required 2.7×10^4 s of wall time, while the simulation of grid C ran for 4.3×10^5 s, which is approximately a 16-fold increase as expected. Hence, we are able to run full boundary layer simulations using a workstation computer of the order of real time. Further details of the computational efficiency when scaled to multiple GPUs can be found in Appendix D. Compared to isothermal simulations on the same hardware, the computational speed is reduced by around 26 % due to the double-distribution approach, which essentially doubles the memory accessed per node as well as the additional models for Coriolis and buoyancy forces. Future work will combine the different forces and collision kernels to minimize memory access and increase computational efficiency.

3.2 Stably stratified boundary layer

To examine the performance of our model in stable boundary layer simulations, we compare it with the well-known GABLS1 benchmark (Beare et al., 2006). The domain has an extent of $400\,\mathrm{m} \times 400\,\mathrm{m} \times 400\,\mathrm{m}$, with periodic boundaries in the streamwise and lateral directions. The geostrophic wind is set to $8\,\mathrm{m\,s^{-1}}$ and the Coriolis parameter to $1.39 \times 10^{-4}\,\mathrm{1\,s^{-1}}$. The domain is initialized with a constant velocity equal to the geostrophic wind and a two-layered temperature profile. The lowest 100 m is initialized with a constant temperature $\theta_0 = 265$ K, above which sits an inversion layer with a temperature gradient of $0.01\,\mathrm{K\,m^{-1}}$. In the lowest 50 m, the temperature is superimposed with random fluctuations with an amplitude of 0.1 K. The surface temperature is also initialized with θ_0, and a constant cooling rate of $0.25\,\mathrm{K\,h^{-1}}$ is applied. The roughness length is set to $z_0 = 0.1$ m. The reference temperature is set to 263.5 K, density to $1.3223\,\mathrm{kg\,m^{-3}}$, and gravity to $9.81\,\mathrm{m\,s^{-2}}$. A Rayleigh damping layer is used at the top, with the damping factor set to $1.6 \times 10^{-3}\,\mathrm{1\,s^{-1}}$. We apply the same boundary conditions as in the previous case, but now the surface temperature is prescribed in the surface layer boundary condition. The total simulated time is 9 h, and averages are computed over the last hour. We simulate two grids, one with $\Delta x = 2$ m and a finer resolution of $\Delta x = 1$ m.

We compare our results to data from the original benchmark and a later publication by Gadde et al. (2021). In the original paper, a number of different models are compared, with a large variety of formulations. Gadde et al. (2021) employ a pseudospectral discretization in horizontal directions and second-order finite differences in the vertical direction and Adam–Bashforth time stepping. They compare three different turbulence models, the Smagorinsky model, the AMD model, and the Lagrangian-averaged scale-dependent model (LASD) (Bou-Zeid et al., 2005). We compare our results only to the results obtained with the Smagorinsky and LASD models since the results differ only marginally between LASD and AMD. Gadde et al. (2021) conduct all simulations at an isotropic resolution of 2.08 m.

As with the previous case, we first show an instantaneous view of the simulation in Fig. 7. The capping inversion is clearly visible in the velocity field, exhibiting a super

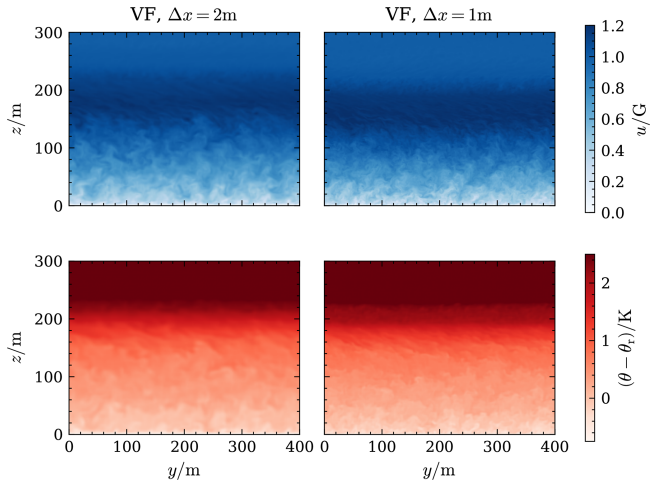

Figure 7. Instantaneous velocity and temperature fields at $x = 200$ m of simulations of the GABLS1 reference case with both resolutions at $t = 9$ h.

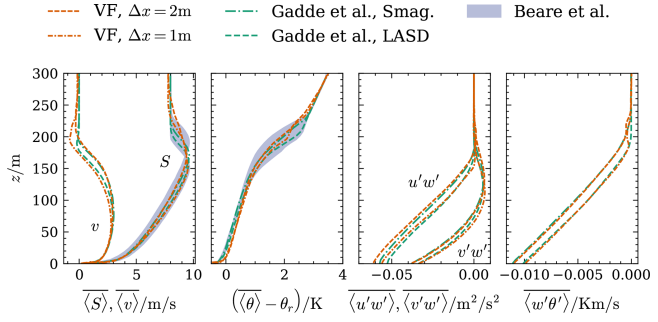

Figure 8. Planar-averaged first- and second-order statistics of the GABLS1 reference case. From left to right: wind speed and lateral velocity, temperature, total momentum fluxes, and total vertical temperature flux. The area covered by all results at 2 m resolution from Beare et al. (2006) is shown as a blue-shaded area.

geostrophic wind and high veer. Above the inversion, a clear reduction in turbulence can be observed. At the higher resolution the transition from boundary layer to free atmosphere is sharper, which is discussed in more detail later on. The temperature field shows a stable stratification and the presence of a strong capping inversion.

We show profiles of horizontally averaged quantities of simulating the GABLS1 reference case in Fig. 8. The wind speed below the inversion agrees well with the reference data in both cases. The case with lower resolution exhibits a significantly lower velocity gradient than the higher-resolution case. This is in line with findings from Beare et al. (2006), where simulations at a lower resolution also exhibited this behavior. At the higher resolution, our results match those of Gadde et al. (2021) closely. We find higher negative veer in the inversion layer compared to Gadde et al. (2021), particularly at higher resolution, which also results in higher wind speed in that region. However, there seems to be only a very limited effect on the flow in the boundary layer. The temperature profile at the lower resolution agrees well with the reference data within the boundary layer. In the inversion layer the gradient is slightly lower than that of Gadde et al. (2021) but still well within the range of results from Beare et al. (2006). At the higher resolution we find very good agreement. The vertical momentum fluxes agree very well with the reference results. Only results from Gadde et al. (2021) are available. The vertical temperature fluxes give a similar picture, although they are slightly smaller than the reference data. Furthermore, there exists a small hump in the results from the case with lower resolution. We believe this is connected to the velocity profile, where the top of the inversion had a significantly lower gradient.

A comparison of some quantities of interest is shown in Table 2. Note that the friction velocity and buoyancy flux are computed from the total fluxes at the second node since this is the node we use as the exchange location for the wall model. We compute the boundary layer height h with the same method used in the references. We first find the height $h_{0.05}$ at which the shear stress $\sqrt{(u'w')^2 + (v'w')^2}$ is less than 5 % of the wall shear stress and extrapolate by $h = \frac{h_{0.05}}{0.95}$.

The friction velocity and buoyancy flux are well within the range of the reference results. This indicates that our wall-modeling approach and boundary condition yield accurate results. The boundary layer height decreases with increasing resolution, as was also observed in Beare et al. (2006).

To examine the performance of the new boundary condition in more detail, we show the time evolution of friction velocity and buoyancy flux in Fig. 9. In the initial seconds of the simulation, both quantities exhibit large spikes. The friction velocity first decreases quickly as a boundary layer develops, decreasing shear in the lowest part of the domain. The surface heat flux increases in magnitude as the surface cools, and thus the temperature gradient near the surface increases. Both quantities stabilize towards the end of the simulation, indicating that the simulation has reached a state of equilibrium. This behavior is qualitatively similar to the results reported in Beare et al. (2006) and Sauer and Muñoz-Esparza (2020). Despite varying formulations, different approaches yield similar results near the equilibrium state of the boundary layer.

It is worth noting that the reference results also yield a large variation in all predicted quantities, as noted by other studies comparing to this case, e.g., van Heerwaarden et al. (2017) and Sauer and Muñoz-Esparza (2020). Generally, our results within the boundary layer fall well within the range of the results obtained by other solvers, despite the differences in approach, subgrid-scale models, etc. At the top of the boundary layer, our model requires higher resolution than most other solvers to accurately represent the capping inversion. We believe this is caused by the model not representing the temperature gradient accurately enough. One way to improve the model is to further refine the collision operator of the advection–diffusion LBM. In a comparison of dif-

Table 2. Friction velocity, boundary layer height, and buoyancy flux of the GABLS1 case.

Quantity	VirtualFluids		Beare et al. (2006)		Gadde et al. (2021)	
	$\Delta x = 2\,\mathrm{m}$	$\Delta x = 1\,\mathrm{m}$	$\Delta x = 2\,\mathrm{m}$	$\Delta x = 1\,\mathrm{m}$	Smag.	LASD
$u_*/\mathrm{m\,s^{-1}}$	0.27	0.26	0.24–0.28	–	0.265	0.253
h/m	179	162	162–197	149–164	166	166
$-\frac{g\langle w\theta\rangle}{\theta_0} \cdot 10^4/\mathrm{m^2/s^3}$	4.05	3.66	3.5–4.7	–	4.1	3.8

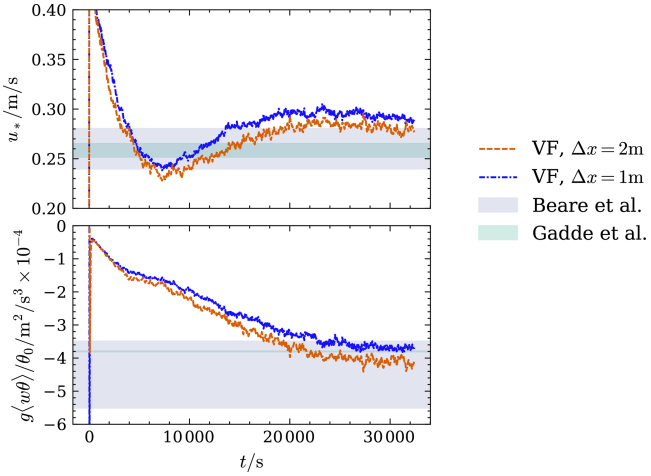

Figure 9. Time evolution of friction velocity and buoyancy flux of the GABLS1 reference case.

ferent advection–diffusion collision operators, Gruszczyński and Łaniewski-Wołłk (2022) found a two-relaxation time central moment operator to be more accurate than the central moment operator most similar to the one employed here. By introducing more relaxation times, leading-order error terms can be canceled out, and accuracy of the collision operator can be improved, as was done, for example, in Geier et al. (2017).

4 Conclusions

This paper presents a novel method for conducting large-eddy simulation of thermally stratified atmospheric boundary layers using the double-distribution function (DDF) lattice Boltzmann method (LBM) in a GPU-resident solver. Very few applications of the LBM to stratified atmospheric boundary layers have been presented in the literature so far, and this work comprises the first application of a DDF approach to such flows in conjunction with employing GPUs. We give a thorough description of our methodology for the simulation of the bulk flow, present a novel boundary condition to use a combined wall model for wall shear stress and heat flux prescribed by Monin–Obukhov similarity theory, and present other models implemented in the GPU-resident LBM solver VirtualFluids in order to simulate stratified atmospheric boundary layers, including horizontally averaged buoyancy, Coriolis force, and Rayleigh damping layer.

We test our model in simulations of conventionally neutral and stably stratified boundary layers. Simulations of the conventionally neutral boundary layer agree very well with reference data obtained with both pseudospectral and finite-volume methods. At a coarse resolution of 7 m, the inversion layer can not be represented accurately. At a finer resolution of 3.5 m, the results match closely with the reference data at twice the resolution obtained with pseudospectral and finite-volume methods. Second-order statistics also agree very well. The spectra obtained at three heights show that the LBM exhibits excellent spectral properties and has lower diffusion than the finite-volume solver at higher resolution. Damping of vertical motions near the inversion layer is also clearly present.

Simulations of the stably stratified GABLS1 reference case also yield satisfactory results. The proposed boundary condition is able to properly reproduce the friction velocity and buoyancy flux at the wall. The boundary layer height agrees with results obtained at the same resolution.

A general shortcoming of the model is its inability to correctly reproduce the direction of the geostrophic wind, and this discrepancy grows with increasing resolution. However, we find that this does not affect the results in the boundary layer, and the misalignment is small. Overall, we achieve satisfactory accuracy for both cases. The method exhibits the expected behavior, generally being more accurate than a second-order finite-volume method, but not as accurate as a pseudospectral approach.

The present model exhibits excellent computational efficiency. All simulations, even the highly resolved neutral boundary layer with ≈ 140 million nodes, are carried out on a single graphics card. At a coarse resolution of $\Delta x = 7\,\mathrm{m}$, the simulation is carried out 4.5 times faster than real time. Increasing resolution to $\Delta x = 3.5\,\mathrm{m}$ results in a simulation at 0.28 real time.

This article comprises a model for an empty boundary layer with flat terrain. Future work will focus on implementing a precursor–successor setup to simulate wind farms. One of the limitations of our model is that the surface layer boundary condition is only formulated for straight walls. As noted by Asmuth et al. (2021), an extension of the inverse momentum exchange method is possible but not available as

of yet. Nevertheless, this work represents an important step for the lattice Boltzmann method and CFD in general towards simulations of stratified boundary layers while fully leveraging the computational efficiency of GPUs. It represents one of the most cost-effective and fastest methods available to conduct LES of such complexity. The reduction in computational cost benefits researchers in many ways, from reducing development time to enabling larger parameter studies. Furthermore, a reduction in computational cost is crucial for enabling industrial application of LES, for example in the wind energy sector. That way, manufacturers, developers, and operators can take into account the complex behavior of the atmospheric boundary layer and reduce model uncertainties, ultimately reducing the cost of electricity.

Appendix A: Unsuccessful preliminary studies

The development of this model was rich with paths that led us nowhere, as is often the case when developing a new model. We want to record some of those paths in the hope that others might learn to either avoid those paths or will see where we went wrong and will be able to tell us what we should have done instead.

A1 Hybrid finite-difference scheme

We first tried to implement a hybrid solver by using finite differences for the advection–diffusion problem. The hybrid method has a number of benefits. The finite-difference approach is much simpler and much more well known; hence there is also more literature on the topic. Furthermore, it requires significantly less memory while also yielding higher computational performance. Hence it is used in a number of other thermal LBM models, for example Onodera et al. (2021) and Feng et al. (2021). Results for the canonical test cases looked very promising, so we decided to pursue this direction further. However, when we simulated the atmospheric boundary layer, we could never avoid spurious oscillations that ultimately degraded the simulation, particularly at the top of the inversion layer. We implemented a variety of approaches, beginning with central differences and Euler forward time stepping. We refined our approach, using a variety of different finite-difference schemes, such as the second-order upwind scheme, the MUSCL scheme (Van Leer, 1977), the QUICK and QUICKEST schemes (Leonard, 1979), and the mixed fourth-order central difference and QUICK schemes. We also tried a second-order Adam–Bashforth time integration but all to no avail. At this point we pivoted to a DDF approach that was already implemented in VirtualFluids since adding even higher-order approaches did not seem promising. As of yet it is unclear what the differences were in our approach compared to, for example, the model implemented in ProLB (Feng et al., 2021). On the one hand, we use a much less diffusive collision operator; thus oscillations are not damped as much. On the other hand, no other study actually simulates a capping inversion, where the oscillations originated from our simulations. Furthermore, there is very little literature concerning this issue. We speculate that the instabilities occur due to the differences in stencil/lattice. The lattice Boltzmann method only accesses the direct neighbors, while all the higher-order methods require information from the second neighbor as well; thus information can travel at different speeds. As this was not the aim of this work, we did not explore this direction further.

A2 Reference temperature

To improve the numerical precision, the populations $g_{\iota\kappa\lambda}$ are set so that $\theta - \theta_\mathrm{m} = \sum g_{\iota\kappa\lambda}$. The choice of the reference temperature θ_m is crucial to improve the accuracy of the simulation. A naïve choice would be to either set it to the reference temperature θ_r or the surface temperature θ_0; however we found that oscillations tended to originate from areas where the temperature is far away from θ_m. We found that the best choice was usually to set θ_m to a value of the temperature in the inversion layer.

Appendix B: Boundary condition fluid

We set a slip boundary condition using a similar method to the one used to set the flux boundary condition. At the uppermost fluid node, we compute the velocity from Eq. (7) and then compute the tangential velocity with

$$u_i^\mathrm{t} = u_i - (u_j n_j) n_i. \tag{B1}$$

Then we apply the bounce-back rule:

$$f_{\overline{\iota\kappa\lambda}} = f_{\iota\kappa\lambda} - 2\rho w_{\iota\kappa\lambda} \frac{u_i c_{\iota\kappa\lambda,i}}{c_\mathrm{s}^2}. \tag{B2}$$

At the bottom boundary we employ the iMEM approach from Asmuth et al. (2021). We want to give a few clarifications and correct some misprints in the original publication. Recall that the momentum transferred from *the fluid to the wall* is

$$\Delta p_{\iota\kappa\lambda,i} = (f_{\iota\kappa\lambda} + f_{\overline{\iota\kappa\lambda}}) c_{\iota\kappa\lambda,i}. \tag{B3}$$

Hence, the total force exerted onto the wall by the fluid is

$$F_i = \frac{\Delta x^3}{\Delta t} \sum_{\iota\kappa\lambda\in\Gamma} \Delta p_{\iota\kappa\lambda,i}, \tag{B4}$$

where Γ is the set of all links cutting the wall. From the wall model we compute the force acting on the wall from the wall shear stress

$$F_i = \tau_i^\mathrm{w} \Delta x^2. \tag{B5}$$

We now seek a wall velocity u^w such that the bounce-back rule in Eq. (B2) results in the correct force. To that end, we

split the force into two components, the force due to the population and due to the wall velocity, F^{f} and $F^{u_{\mathrm{w}}}$:

$$F_i^{\mathrm{f}} = \frac{\Delta x^3}{\Delta t} \sum_{\iota\kappa\lambda\in\Gamma} c_{\iota\kappa\lambda,i}(f_{\iota\kappa\lambda} + f_{\overline{\iota\kappa\lambda}}), \tag{B6}$$

$$F_i^{u_{\mathrm{w}}} = -\frac{\Delta x^3}{\Delta t} \sum_{\iota\kappa\lambda\in\Gamma} 2\rho w_{\iota\kappa\lambda} c_{\iota\kappa\lambda,i} \frac{u_j c_{\iota\kappa\lambda,j}}{c_{\mathrm{s}}^2}. \tag{B7}$$

Thus, $F_i^{u_{\mathrm{w}}} = F_i - F_i^{\mathrm{f}}$, resulting in a system of equations that needs to be solved for u^{w}. For the case of a straight wall at the bottom, the solution is

$$\boldsymbol{u}^{\mathrm{w}} = -\frac{3c_{\mathrm{s}}^2}{\rho}\frac{\Delta t^2}{\Delta x^4}\left(3F_1^{u_{\mathrm{w}}}, 3F_2^{u_{\mathrm{w}}}, F_3^{u_{\mathrm{w}}}\right)^{\mathrm{T}}. \tag{B8}$$

Appendix C: Convergence study

To determine the order of convergence of the advection–diffusion equation numerically, we simulate the advection–diffusion of a Gaussian hill of concentration. Note that in this case, θ is a passive scalar. The initial field of concentration θ is described by Krüger et al. (2017, p. 322):

$$\theta(\boldsymbol{x}, t=0) = \theta_0 \exp -\frac{|\boldsymbol{x}-\boldsymbol{x}_0|^2}{2\sigma_0}. \tag{C1}$$

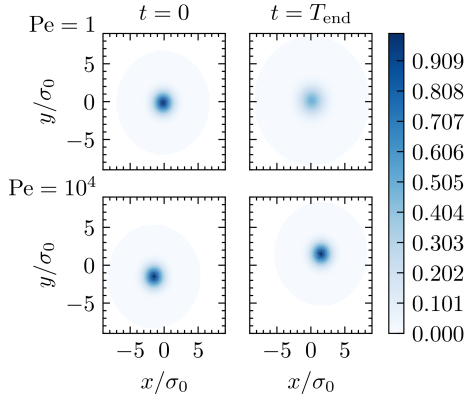

Figure C1. Analytical solution for the concentration in the test case of a Gaussian hill of concentration, projected onto the x–y plane. In the white region concentration $\theta \leq 1 \times 10^{-10}$.

Under a constant advection velocity $\boldsymbol{u}$,

$$\theta(\boldsymbol{x}, t) = \frac{\sigma_0^2}{\sigma_0^2 + \sigma_D^2}\theta_0 \exp -\frac{|\boldsymbol{x} - \boldsymbol{x}_0 - \boldsymbol{u}t|^2}{2(\sigma_0^2 + \sigma_D^2)} \tag{C2}$$

is a solution for the advection–diffusion equation

$$\frac{\partial\theta}{\partial t} + \boldsymbol{u}\cdot\frac{\partial\theta}{\partial\boldsymbol{x}} = \frac{1}{Pe}\frac{\partial^2\theta}{\partial\boldsymbol{x}^2}, \tag{C3}$$

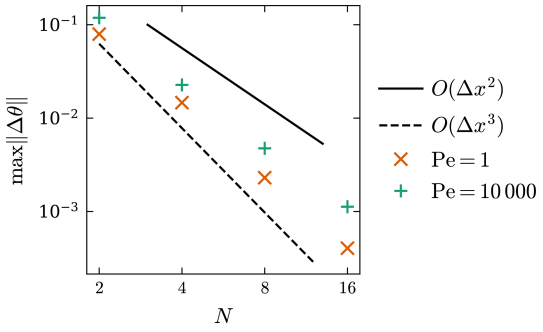

Figure C2. Maximum error $\Delta\theta$ of the numerical results of simulating the advection–diffusion of a Gaussian hill of concentration.

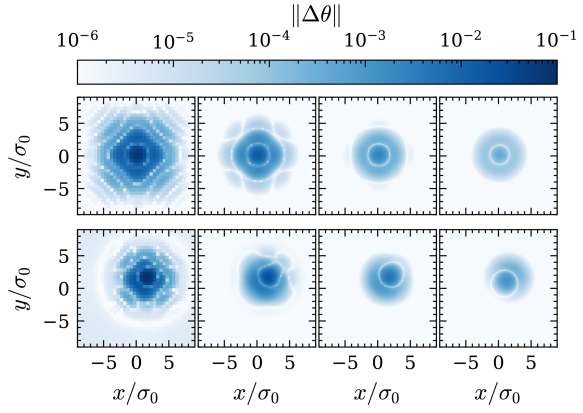

Figure C3. Absolute difference between analytical and numerical solution of the Gaussian hill of concentration at $t = T_{\mathrm{end}}$. The top row shows the case with $Pe = 1$ and the bottom row the case with $Pe = 10^4$. Resolution increases from left to right.

with $\sigma_D = \sqrt{2Dt}$, the Peclét number $Pe = \frac{\sigma_0 U}{D}$, and $\boldsymbol{u} = [1,1,1]^{\mathrm{T}}U$. We conduct a convergence study for a diffusion-dominated problem ($Pe = 1$) and an advection-dominated problem ($Pe = 10^5$). We vary $N = \sigma_0/\Delta x = 2, 4, 8, 16$, while we keep $\sigma_0 = 1$ and $\Delta t = 1$ constant. In the case of $Pe = 1$, we set the diffusivity in lattice units $\tilde{D} = D\frac{\Delta t}{\Delta x^2} = 0.01$, and we simulate until $T_{\mathrm{end}} = 0.3\frac{\sigma_0}{U}$ is reached. In the case of $Pe = 10^5$, $\tilde{D} = 1 \times 10^{-5}$ and $T_{\mathrm{end}} = 3\frac{\sigma_0}{U}$. In both cases the domain has a size of $18\sigma_0 \times 18\sigma_0 \times 18\sigma_0$. To optimally utilize the simulation domain, we set $\boldsymbol{x}_0 = -T_{\mathrm{end}}U/2$. The analytical solutions for both Peclét numbers at $t = 0$ and $t = T_{\mathrm{end}}$ are shown in Fig. C1. The results of the convergence test for both Peclét numbers can be found in Fig. C2. The results clearly show a convergence rate slightly above second order in both cases, as expected. More precisely, we compute a convergence rate of 2.5 and 2.2 for $Pe = 1$ and $Pe = 10^5$, respectively. A more detailed view of the error can be found in Fig. C3, where we show the absolute difference between the analytical and numerical solution. We see that in both cases, errors decrease with higher resolution and that the error is symmetric in the diffusion-dominated case, while the advec-

tion case exhibits errors aligned with the direction of advection. We also see that errors become negligibly small towards the boundaries of the domain, so the domain was chosen to be large enough to not affect the results.

Appendix D: Scaling study

Table D1. $\frac{T_{\text{wall}}}{T_{\text{sim}}}$ for all cases of the scaling study.

Number of GPUs	1	2	4	8
Weak scaling	1.81	2.03	2.12	2.12
Strong scaling	1.81	1.09	0.63	0.55
Strong scaling reference	1.81	0.92	0.45	0.23

We conduct both a weak and a strong scaling study of the model based on the neutrally stratified test case with grid C ($\Delta x = 3.5\,\text{m}$). However, we want to emphasize that the model in its current state has not yet been optimized for performance and not at all for multi-GPU performance. These improvements will be a topic of future studies.

The study is conducted on the Dardel supercomputer from 1 to 8 GPUs. Each node has 4 Nvidia GH200 Grace Hopper superchips with 120 GB memory each. The domain is partitioned into equally sized subdomains along the streamwise direction. This was found to be the optimal partitioning strategy for this case. Each case is run for 100 s of simulation time, and the wall time T_{wall} of the longest-running process is recorded. On a single GPU, the domain has a size of $3060\,\text{m} \times 3600\,\text{m}$ to fully use the available memory of the card.

For the weak scaling study we simply extended the domain to $7200\,\text{m} \times 3600\,\text{m}$, $7200\,\text{m} \times 7200\,\text{m}$, and $14\,400\,\text{m} \times 7200\,\text{m}$ for 2, 4, and 8 GPUs, respectively. The reference wall time $T_{\text{wall,ref}}$ for all simulations is the wall time of the single-GPU case.

In the strong scaling study, we partition the same domain into smaller and smaller subdomains. However, if the subdomain becomes too small, the GPU is not saturated any more, and the parallel efficiency decreases. We therefore ran a reference case for each case of the scaling study where the domain had the same size as one subdomain and use the wall time of that case as $T_{\text{wall,ref}}$. By normalizing in this manner, ideal scaling corresponds to a constant ratio of $T_{\text{wall,ref}}/T_{\text{wall}}$.

The wall time for each case can be found in Table D1, and the results of both scaling studies are shown in Fig. D1.

The weak scaling efficiency of the model is already good. Even at 8 GPUs the efficiency is still at 85 %, indicating that the model is suitable for large-scale computation. We also find that the efficiency does not noticeably decrease from 4 to 8 GPUs. We observe that the model does not scale as well in the strong scaling study. On the one hand, this can be explained simply by the square-cube law; i.e., the volume of the subdomains shrinks faster than the surface. The

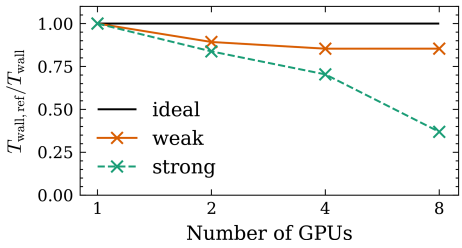

Figure D1. Results of the strong and weak scaling study from 1 to 8 GPUs.

communication hiding implemented in VirtualFluids becomes less effective for smaller subdomains; see Geier et al. (2025). This problem is exacerbated due to the planar averaging in Eq. (5), which also requires inter-GPU communication and becomes less efficient when partitioned along the streamwise direction. Nevertheless, the efficiency is still around 35 %. Comparing these results with Min et al. (2024), we find that our model has comparable scaling efficiency in the weak scaling study, although we use significantly fewer GPUs.

Appendix E: Unit conversion in LBM and its effect on small forces

The LBM relies on non-dimensionalizing all quantities with grid size Δx, time step size Δt, and reference density ρ_0. Furthermore, recall that $\frac{\Delta x}{\sqrt{3}\Delta t} = c_s = \frac{V_0}{\text{Ma}}$, which we have kept constant. Thus, velocities in SI units are scaled as $\tilde{u} = \frac{u}{C_u}$, where $C_u = \frac{\Delta x}{\Delta t} = \text{const}$, and $\tilde{u}$ is the velocity in LB units. Forces, however, are scaled as $\tilde{f} = \frac{f}{C_f}$, with $C_f = \rho_0 \frac{\Delta x}{\Delta t^2} = \rho_0 \frac{3c_s^2}{\Delta x}$ (Krüger et al., 2017, Chap. 7). Therefore, non-dimensionalized forces decrease with grid spacing. In the cumulant LBM, forces are only applied via Eq. (7) (Geier et al., 2015). Therefore, the result of the addition becomes less accurate with increasing resolution, since the contribution of the force can not be represented in finite-precision floating point numbers. This becomes particularly important for small forces, such as the Coriolis force, and when using single-precision floating point numbers. We have tried the Kahan summation algorithm (Kahan, 1965) as a remedy but that did not improve the results.

Code and data availability. VirtualFluids is available as open-source code at https://git.rz.tu-bs.de/irmb/VirtualFluids (last access: 15 July 2026). The model described in this paper is published in version 0.3.0 (Kutscher et al., 2026, https://doi.org/10.5281/zenodo.20681486). The data for creating the plots in Sect. 3 and the corresponding postprocessing scripts are available at https://source.coderefinery.org/wind_energy_uu/stratificationinlbm (last access: 15 July 2026). A permanent record is kept at https://doi.org/10.5281/zenodo.20512678 (Korb, 2026).

Author contributions. HK, HA, and SI conceptualized the project. HK, HA, MG, and MS developed the methodology and implemented the code. HK conducted the simulations and the postprocessing under the guidance of HA and SI. HK prepared the original draft, and all authors reviewed and edited the paper. SI acquired the resources and supervised the project.

Competing interests. The contact author has declared that none of the authors has any competing interests.

Disclaimer. Publisher's note: Copernicus Publications remains neutral with regard to jurisdictional claims made in the text, published maps, institutional affiliations, or any other geographical representation in this paper. The authors bear the ultimate responsibility for providing appropriate place names. Views expressed in the text are those of the authors and do not necessarily reflect the views of the publisher.

Acknowledgements. HK would like to thank Antonio Salvini, Luca Lanzilao, and Hugo Olivares-Espinosa for their helpful discussions and Lawrence Cheung and Gopal Yalla for their help with the results from the Exawind benchmark. Some of the computations were enabled by resources provided by the National Academic Infrastructure for Supercomputing in Sweden (NAISS), partially funded by the Swedish Research Council through grant agreement no. 2022-06725.

Financial support. The publication of this article was funded by the Swedish Research Council, Forte, Formas, and Vinnova.

Review statement. This paper was edited by Sukanta Basu and reviewed by three anonymous referees.

References

Abkar, M. and Porté-Agel, F.: Influence of Atmospheric Stability on Wind-Turbine Wakes: A Large-Eddy Simulation Study, Phys. Fluids, 27, 035104, https://doi.org/10.1063/1.4913695, 2015.

Abkar, M., Bae, H. J., and Moin, P.: Minimum-Dissipation Scalar Transport Model for Large-Eddy Simulation of Turbulent Flows, Phys. Rev. Fluids, 1, 041701, https://doi.org/10.1103/PhysRevFluids.1.041701, 2016.

Alihussein, H., Geier, M., and Krafczyk, M.: A Parallel Coupled Lattice Boltzmann-Volume of Fluid Framework for Modeling Porous Media Evolution, Materials, 14, 2510, https://doi.org/10.3390/ma14102510, 2021.

Allaerts, D. and Meyers, J.: Boundary-Layer Development and Gravity Waves in Conventionally Neutral Wind Farms, J. Fluid Mech., 814, 95–130, https://doi.org/10.1017/jfm.2017.11, 2017.

Arya, S. P.: Introduction to Micrometeorology, 2nd edn., vol. 79 of International Geophysics Series, Academic Press, San Diego, ISBN 0-12-059354-8, 2001.

Asmuth, H., Janßen, C. F., Olivares-Espinosa, H., and Ivanell, S.: Wall-Modeled Lattice Boltzmann Large-Eddy Simulation of Neutral Atmospheric Boundary Layers, Phys. Fluids, 33, 105111, https://doi.org/10.1063/5.0065701, 2021.

Basu, S., Holtslag, A. A. M., Van De Wiel, B. J. H., Moene, A. F., and Steeneveld, G.-J.: An Inconvenient "Truth" about Using Sensible Heat Flux as a Surface Boundary Condition in Models under Stably Stratified Regimes, Acta Geophys., 56, 88–99, https://doi.org/10.2478/s11600-007-0038-y, 2008.

Beare, R. J., Macvean, M. K., Holtslag, A. A. M., Cuxart, J., Esau, I., Golaz, J.-C., Jimenez, M. A., Khairoutdinov, M., Kosovic, B., Lewellen, D., Lund, T. S., Lundquist, J. K., Mccabe, A., Moene, A. F., Noh, Y., Raasch, S., and Sullivan, P.: An Intercomparison of Large-Eddy Simulations of the Stable Boundary Layer, Bound.-Lay. Meteorol., 118, 247–272, https://doi.org/10.1007/s10546-004-2820-6, 2006.

Berg, J., Patton, E. G., and Sullivan, P. P.: Large-Eddy Simulation of Conditionally Neutral Boundary Layers: A Mesh Resolution Sensitivity Study, J. Atmos. Sci., 77, 1969–1991, https://doi.org/10.1175/JAS-D-19-0252.1, 2020.

Bou-Zeid, E., Meneveau, C., and Parlange, M.: A Scale-Dependent Lagrangian Dynamic Model for Large Eddy Simulation of Complex Turbulent Flows, Phys. Fluids, 17, 025105, https://doi.org/10.1063/1.1839152, 2005.

Businger, J. A., Wyngaard, J. C., Izumi, Y., and Bradley, E. F.: Flux-Profile Relationships in the Atmospheric Surface Layer, J. Atmos. Sci., 28, 181–189, https://doi.org/10.1175/1520-0469(1971)028<0181:FPRITA>2.0.CO;2, 1971.

Deardorff, J.: A Three-dimensional Numerical Investigation of the Idealized Planetary Boundary Layer, Geophysical Fluid Dynamics, 1, 377–410, https://doi.org/10.1080/03091927009365780, 1970.

d'Humières, D.: Generalized Lattice-Boltzmann Equations, in: Rarefied Gas Dynamics: Theory and Simulations, Progress in Astronautics and Aeronautics, American Institute of Aeronautics and Astronautics, 450–458, https://arc.aiaa.org/doi/10.2514/5.9781600866319.0450.0458 (last access: 15 July 2026), 1994.

Feng, Y., Miranda-Fuentes, J., Guo, S., Jacob, J., and Sagaut, P.: ProLB: A Lattice Boltzmann Solver of Large-Eddy Simulation for Atmospheric Boundary Layer Flows, J. Adv. Model. Earth Sy., 13, e2020MS002107, https://doi.org/10.1029/2020MS002107, 2021.

Gadde, S. N., Stieren, A., and Stevens, R. J. A. M.: Large-Eddy Simulations of Stratified Atmospheric Boundary Layers: Comparison of Different Subgrid Models, Bound.-Lay. Meteorol., 178, 363–382, https://doi.org/10.1007/s10546-020-00570-5, 2021.

Gehrke, M. and Rung, T.: Periodic Hill Flow Simulations with a Parameterized Cumulant Lattice Boltzmann Method, Int. J. Numer. Meth. Fl., 94, 1111–1154, https://doi.org/10.1002/fld.5085, 2022.

Geier, M., Schönherr, M., Pasquali, A., and Krafczyk, M.: The Cumulant Lattice Boltzmann Equation in Three Dimensions: Theory and Validation, Comput. Math. Appl., 70, 507–547, https://doi.org/10.1016/j.camwa.2015.05.001, 2015.

Geier, M., Pasquali, A., and Schönherr, M.: Parametrization of the Cumulant Lattice Boltzmann Method for Fourth Order Accurate Diffusion Part II: Application to Flow around

a Sphere at Drag Crisis, J. Comput. Phys., 348, 889–898, https://doi.org/10.1016/j.jcp.2017.07.004, 2017.

Geier, M., Lenz, S., Schönherr, M., and Krafczyk, M.: Under-Resolved and Large Eddy Simulations of a Decaying Taylor–Green Vortex with the Cumulant Lattice Boltzmann Method, Theor. Comp. Fluid Dyn., https://doi.org/10.1007/s00162-020-00555-7, 2020.

Geier, M., Kutscher, K., Schönherr, M., Wellmann, A., Peters, S., Alihussein, H., Linxweiler, J., and Krafczyk, M.: VirtualFluids – Open Source Parallel LBM Solver, Comput. Phys. Commun., 109810, https://doi.org/10.1016/j.cpc.2025.109810, 2025.

Gruszczyński, G. and Łaniewski-Wołłk, Ł.: A Comparative Study of 3D Cumulant and Central Moments Lattice Boltzmann Schemes with Interpolated Boundary Conditions for the Simulation of Thermal Flows in High Prandtl Number Regime, I. J. Heat Mass Tran., 197, 123259, https://doi.org/10.1016/j.ijheatmasstransfer.2022.123259, 2022.

Jacob, J., Malaspinas, O., and Sagaut, P.: A New Hybrid Recursive Regularised Bhatnagar–Gross–Krook Collision Model for Lattice Boltzmann Method-Based Large Eddy Simulation, J. Turbul., 19, 1051–1076, https://doi.org/10.1080/14685248.2018.1540879, 2018.

Kahan, W.: Pracniques: Further Remarks on Reducing Truncation Errors, Commun. ACM, 8, 40, https://doi.org/10.1145/363707.363723, 1965.

Khan, M. A., Allaerts, D., Watson, S. J., and Churchfield, M. J.: Investigating the relationship between simulation parameters and flow variables in simulating atmospheric gravity waves for wind energy applications, Wind Energ. Sci., 10, 1167–1185, https://doi.org/10.5194/wes-10-1167-2025, 2025.

King, M.-F., Khan, A., Delbosc, N., Gough, H. L., Halios, C., Barlow, J. F., and Noakes, C. J.: Modelling Urban Airflow and Natural Ventilation Using a GPU-based Lattice-Boltzmann Method, Build. Environ., 125, 273–284, https://doi.org/10.1016/j.buildenv.2017.08.048, 2017.

Korb, H.: Plot data for "Large Eddy Simulation of Thermally Stratified Atmospheric Boundary Layers with a Lattice Boltzmann Method", Zenodo [data set], https://doi.org/10.5281/zenodo.20512678, 2026.

Korb, H., Bastin, J., Asmuth, H., and Ivanell, S.: The lattice Boltzmann method for wind farm simulations: a review, Wind Energ. Sci., 11, 983–1012, https://doi.org/10.5194/wes-11-983-2026, 2026.

Krüger, T., Kusumaatmaja, H., Kuzmin, A., Shardt, O., Silva, G., and Viggen, E. M.: The Lattice Boltzmann Method: Principles and Practice, Graduate Texts in Physics, 1 edn., Springer International Publishing, http://link.springer.com/10.1007/978-3-319-44649-3 (last access: 15 July 2026), 2017.

Kuhn, M. B., Cheung, L., Yalla, G., Henry de Frahan, M. T., and Mohan, P.: Exawind Benchmarks, GitHub, https://github.com/Exawind/exawind-benchmarks/tree/main/amr-wind/atmospheric_boundary_layer/neutral (last access: 15 July 2026), 2025a.

Kuhn, M. B., Henry de Frahan, M. T., Mohan, P., Deskos, G., Churchfield, M., Cheung, L., Sharma, A., Almgren, A., Ananthan, S., Brazell, M. J., Martínez-Tossas, L. A., Thedin, R., Rood, J., Sakievich, P., Vijayakumar, G., Zhang, W., and Sprague, M.: AMR-Wind: A Performance-Portable, High-Fidelity Flow Solver for Wind Farm Simulations, Wind Energy, 28, e70010, https://doi.org/10.1002/we.70010, 2025b.

Kutscher, K., Schönherr, M., Geier, M., Alihussein, H., Wellmann, A., Korb, H., Sukhman, D., Horneff, N., Yescas Frangos, P., Bastin, J., Lopez Hermida, D., Mohammadi, M. M., and Qin, Z.: VirtualFluids, Zenodo [code], https://doi.org/10.5281/zenodo.20681486, 2026.

Lanzilao, L. and Meyers, J.: A Parametric Large-Eddy Simulation Study of Wind-Farm Blockage and Gravity Waves in Conventionally Neutral Boundary Layers, J. Fluid Mech., 979, A54, https://doi.org/10.1017/jfm.2023.1088, 2024.

Lenz, S., Schönherr, M., Geier, M., Krafczyk, M., Pasquali, A., Christen, A., and 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. Aerod., 189, 151–162, https://doi.org/10.1016/j.jweia.2019.03.012, 2019.

Leonard, B. P.: A Stable and Accurate Convective Modelling Procedure Based on Quadratic Upstream Interpolation, Comput. Method. Appl. M., 19, 59–98, https://doi.org/10.1016/0045-7825(79)90034-3, 1979.

Lilly, D.: The Representation of Small-Scale Turbulence in Numerical Simulation Experiments, University Corporation for Atmospheric Research, https://doi.org/10.5065/D62R3PMM, 1966.

Malaspinas, O. and Sagaut, P.: Wall Model for Large-Eddy Simulation Based on the Lattice Boltzmann Method, J. Comput. Phys., 275, 25–40, https://doi.org/10.1016/j.jcp.2014.06.020, 2014.

Min, M., Brazell, M., Tomboulides, A., Churchfield, M., Fischer, P., and Sprague, M.: Towards Exascale for Wind Energy Simulations, Int. J. High Perform. C., 38, 337–355, https://doi.org/10.1177/10943420241252511, 2024.

Moeng, C.-H.: A Large-Eddy-Simulation Model for the Study of Planetary Boundary-Layer Turbulence, J. Atmos. Sci., 41, 2052–2062, https://doi.org/10.1175/1520-0469(1984)041<2052:ALESMF>2.0.CO;2, 1984.

Obrecht, C., Kuznik, F., Tourancheau, B., and Roux, J.-J.: The TheLMA Project: A Thermal Lattice Boltzmann Solver for the GPU, Comput. Fluids, 54, 118–126, https://doi.org/10.1016/j.compfluid.2011.10.011, 2012.

Obrecht, C., Kuznik, F., Tourancheau, B., and Roux, J.-J.: Multi-GPU Implementation of the Lattice Boltzmann Method, Comput. Math. Appl., 65, 252–261, https://doi.org/10.1016/j.camwa.2011.02.020, 2013.

Obrecht, C., Kuznik, F., Merlier, L., Roux, J.-J., and Tourancheau, B.: Towards Aeraulic Simulations at Urban Scale Using the Lattice Boltzmann Method, Environ. Fluid Mech., 15, 753–770, https://doi.org/10.1007/s10652-014-9381-0, 2015.

Onodera, N., Aoki, T., Shimokawabe, T., and Kobayashi, H.: Large-Scale LES Wind Simulation Using Lattice Boltzmann Method for a 10 Km × 10 Km Area in Metropolitan Tokyo, TSUBAME ESJ, 9, 2–8, 2013.

Onodera, N., Idomura, Y., Hasegawa, Y., Nakayama, H., Shimokawabe, T., and Aoki, T.: Real-Time Tracer Dispersion Simulations in Oklahoma City Using the Locally Mesh-Refined Lattice Boltzmann Method, Bound.-Lay. Meteorol., https://doi.org/10.1007/s10546-020-00594-x, 2021.

Platis, A., Hundhausen, M., Lampert, A., Emeis, S., and Bange, J.: The Role of Atmospheric Stability and Turbulence in Offshore Wind-Farm Wakes in the German Bight, Bound.-Lay. Meteo-

rol., 182, 441–469, https://doi.org/10.1007/s10546-021-00668-4, 2022.

Rozema, W., Bae, H. J., Moin, P., and Verstappen, R.: Minimum-Dissipation Models for Large-Eddy Simulation, Phys. Fluids, 27, 085107, https://doi.org/10.1063/1.4928700, 2015.

Sauer, J. A. and Muñoz-Esparza, D.: The FastEddy® Resident-GPU Accelerated Large-Eddy Simulation Framework: Model Formulation, Dynamical-Core Validation and Performance Benchmarks, J. Adv. Model. Earth Sy., 12, e2020MS002100, https://doi.org/10.1029/2020MS002100, 2020.

Smagorinsky, J.: GENERAL CIRCULATION EXPERIMENTS WITH THE PRIMITIVE EQUATIONS: I. THE BASIC EXPERIMENT, Mon. Weather Rev., 91, 99–164, https://doi.org/10.1175/1520-0493(1963)091<0099:GCEWTP>2.3.CO;2, 1963.

Stoll, R., Gibbs, J. A., Salesky, S. T., Anderson, W., and Calaf, M.: Large-Eddy Simulation of the Atmospheric Boundary Layer, Bound.-Lay. Meteorol., 177, 541–581, https://doi.org/10.1007/s10546-020-00556-3, 2020.

van Heerwaarden, C. C., van Stratum, B. J. H., Heus, T., Gibbs, J. A., Fedorovich, E., and Mellado, J. P.: MicroHH 1.0: a computational fluid dynamics code for direct numerical simulation and large-eddy simulation of atmospheric boundary layer flows, Geosci. Model Dev., 10, 3145–3165, https://doi.org/10.5194/gmd-10-3145-2017, 2017.

Van Leer, B.: Towards the Ultimate Conservative Difference Scheme. IV. A New Approach to Numerical Convection, J. Comput. Phys., 23, 276–299, https://doi.org/10.1016/0021-9991(77)90095-X, 1977.

Verstappen, R.: When Does Eddy Viscosity Damp Subfilter Scales Sufficiently?, J. Sci. Comput., 49, 94, https://doi.org/10.1007/s10915-011-9504-4, 2011.

Wang, Y., MacCall, B. T., Hocut, C. M., Zeng, X., and Fernando, H. J. S.: Simulation of Stratified Flows over a Ridge Using a Lattice Boltzmann Model, Environ. Fluid Mech., 20, 1333–1355, https://doi.org/10.1007/s10652-018-9599-3, 2020.

Yang, X., Mehmani, Y., Perkins, W. A., Pasquali, A., Schönherr, M., Kim, K., Perego, M., Parks, M. L., Trask, N., Balhoff, M. T., Richmond, M. C., Geier, M., Krafczyk, M., Luo, L.-S., Tartakovsky, A. M., and Scheibe, T. D.: Intercomparison of 3D Pore-Scale Flow and Solute Transport Simulation Methods, Adv. Water Resour., 95, 176–189, https://doi.org/10.1016/j.advwatres.2015.09.015, 2016.

Yang, X. I. A., Park, G. I., and Moin, P.: Log-Layer Mismatch and Modeling of the Fluctuating Wall Stress in Wall-Modeled Large-Eddy Simulations, Phys. Rev. Fluids, 2, 104601, https://doi.org/10.1103/PhysRevFluids.2.104601, 2017.