Rebuttal to review of Dr. Bou-Zeid

The authors thank the reviewer for his kind words in his opening paragraph. We will address his comments point-by-point.

1. *Eq 11: It would be useful to explain what Q represents physically (phase change, radiative divergence, . . .). Also it should be included in 13 since the authors also use it to represent sources of heat unrelated to evaporation/condensation.*
   Q can be any source or sink of heat. Phase changes are excluded from Q, as the dry dynamics do not support those, and the moist dynamics are based on the liquid water potential temperature that is constant under phase changes. The reviewer is correct that sources and sinks need to be included in Eq. 13 as well, and we will do so in the revised manuscript.

2. *Page 5, first few paragraphs of the section "Gird" and many other places in the text. The authors use too many paragraphs. Some should be consolidated. E.g. the first 2 paragraphs of this section should be joined.*
   We will carefully go through the text and merge paragraphs at the suggested location and wherever appropriate.

3. *Eq 28: So I presume here the authors use j as the vertical index. That should be specified. Also maybe at some point the authors should point out that only the bottom and top boundary conditions (is it detailed sufficiently?) need a special treatment like this since the other are periodic.*
   The reviewer is correct that j is the vertical index. We will explicitly mention this in the revised manuscript. We will also include an explicit reference to the fact that only the vertical dimension needs a special treatment.

4. *Eq 28 again: At some point later in the paper I thought the authors mention that with 4th order accurate scheme 2 ghost cells are needed. If that is so, why is there a need for a biased formulation in 28 that would only use one ghost cell below the surface.*
   Many operations involve a sequential application of two operators. For instance, in the 4^{th}-order diffusion, we compute the laplacian as the divergence of a gradient. In this operation, only the gradient can make use of both ghost cells, but the divergence cannot, and therefore relies on a biased operator at the wall.

5. *Eq 36 and other places: it would be useful if for each of these options (2nd versus 4th order for example), the flag that controls it in the code input file is listed. This will make it easy for the user to see how to control these options.*
   MicroHH comes with a document that lists all the available options. We have failed to mention this in the text and will add it to the revised manuscript. We will explain as well in the revised manuscript that the model defaults to the order of generated grid.

6. *Eq 41: tilde is later used for filtering. Maybe denote the intermediate velocity with something else like an asterisk.*
   We will follow the suggestion of the reviewer to avoid confusion between filtered

variables and the intermediate velocity.

7. *The fact that the code is mainly periodic in the horizontal direction should be underlined earlier in the paper than it is now. Maybe in the abstract.*
   We agree with the reviewer that an earlier notification is necessary, because it clarifies both the grid description and the pressure solver. We will introduce it in the introduction of the revised manuscript.

8. *After Eq 47: please provide a reference to the "Thomas algorithm"*
   We will include a reference in the revised manuscript.

9. *LES equations 63 and so on are only for very high RE, i.e. wall modeled LES. Please specify that. Also it would be simple to use the code as a finite Re LES code by keeping the viscous term in 63. Why is this not pursued?*
   We will follow the reviewer's suggestion and mention that our LES is developed for very high Re. Extending our code to a finite Re LES code would be trivial, but has not been pursued yet. The reason is that most MicroHH users that run the model in LES-mode run atmospheric cases.

10. *"Surface Model" section. The authors only provide the LES surface model. This should be specified. Also better is to add a description of how the DNS wall boundary condition is treated, presumably through a viscous wall stress. Also, the language seems to suggest that the LES is only over rough walls. There is nothing that prevent the code from simulating a smooth surface using the z0 (~ v/u*) of a smooth wall. This should be clarified.*
    The description of the DNS boundary conditions is contained in 3.7, but we failed to make this clear to the reviewer. We will improve both Section 3.7 and 4.2 to clarify our implementation. The code could indeed specify the z0 of a smooth wall, but also here, it has not been implemented yet.

11. *First line after eq 73: please add "kinematic" to the description of B0.*
    Correct. We will add this.

12. *Eq 74: the application of a log law to each velocity component separately is an approximation so the equals here should be replaced by ≈. Also this is a LOCAL MOST wall model. This is not a trivial detail and should be specified and discussed briefly with references to papers that discuss the implications in more detail.*
    We will introduce the approximation symbol. Furthermore, we will discuss the results with respect to existing literature, such as the reviewer's paper in Physics of Fluids (2005).

13. *Eqs 87 and 88: why not use an explicit approach using the fluxes at the previous time step? This is commonly done and since the CFL condition is typically quite < 1 this should be ok? What are the advantages of an explicit approach?*
    Using the fluxes of the previous time step is often a good solution, but can lead to inaccuracies under, for instance, free convection, where fluxes and wind speeds can change fast at the surface, or under conditions of changing stability. Our methods

have a 100% convergence guarantee under all conditions. Furthermore, it is based on a lookup table that starts searching from the value at the previous time step, which makes it a very fast procedure.

14. *Eq 90 is confusing. For example, under steady state this almost looks like the pressure gradient is 0. Should the mean RHS <f1> be added? The fact that the pressure gradient force must balance the surface stress force under steady state should be stated.*
    With Eq. 90, we aimed to show that the forcing is just part of the total tendency (note the $F_{p;ls}$ suffix). It is a definition rather than an equality. As we have failed to explain it properly, we will clarify this in the improved manuscript.

15. *Eq 93: is the momentum balance changed when a subsidence velocity is added to scalars?*
    It is not. Solving the momentum balance in a doubly-periodic domain under subsidence conditions is a non-trivial exercise that deserves its own study. We follow the simplified treatment that is used in other codes such as DALES and UCLALES. We will add an additional explanation to the paper.

16. *Page 18 lines 9-11: please provide reference or URLs for these libraries and codes.*
    We will add URLs to the referenced libraries and tools.

17. *Figure 1: which of the blue or green is the energy conserving 4th order or the most accurate. Also, did the authors describe the 2 methods using these names in the numerics section?*
    The green line is the energy-conserving discretization, whereas the blue line is the accurate one. Surprisingly, the energy-conserving discretization is in the Taylor-Green-vortex test case also the most accurate one, but this does not apply to all test cases. We forgot to explain the abbreviations in the legend of Figure 1, and will do so in the figure caption of the revised manuscript. Furthermore, we will improve the color scheme to ensure that all cases can be easily distinguished.

18. *ALL figures look like they have problems with some axis labels (some minus signs appear) and so on, please improve quality. If all looks good on the authors computers check that the PDF appears the same on other machines.*
    Something apparently went wrong in the process of adding the GMD logos to the manuscript. In the current online version, as well as in the revised manuscript, all labels are in order.

19. *Why include RK3 in the code release at all given the results?*
    We will keep the RK3 case for testing purposes and for potential extension with implicit-in-time diffusion in the future. The reviewer is correct that our tests show that the RK4 scheme is beneficial under all conditions.

20. *Page 20 line 9, delete "for"*
    We will fix the sentence.

21. *Figure 2: slope at smallest dt looks the same for RK3 and RK4, no?*
It appears so. The lines are bumpy and the exact slopes are hard to extract. We hope nonetheless, that the reviewer is convinced about the difference in convergence and accuracy between the two methods.

22. *Figure 4: symbols not appearing in legend.*
We believe this is related to the previous problem (point 18) we had with the figure axes. In our current version, all symbols are visible.

23. *Section 8.4: give some info about MOSER code for comparison.*
The code of MOSER is spectral with Chebychev polynomials in the non-periodic dimension. We will explain this in the revised manuscript.

24. *Page 22 line 6-10: use of word "data" to describe MOSER results is not a good choice here.*
We will refer to MOSER's result as "model output data", rather than "data" in the revised manuscript.

25. *Figure 6: clearly the spectra of MOSER have some noise or aliasing issues that should be mentioned.*
The spectra of MOSER display aliasing in the pressure data, most likely related to the velocity multiplications in the Poisson equation that solves for the pressure. We will make this clear in the text.

26. *Page 24 Line 17: here the authors use the term "potential temperature flux" but previous they used "buoyancy flux". Pick one since they mean the same thing in dry cases. I would suggest potential T flux since it is a more accurate physics description.*
We distinguish between the two. The dry dynamics have potential temperature as the governing variable, therefore the bottom BC is a potential temperature flux. Our simplified thermodynamics use buoyancy as the governing variable, and therefore a kinematic buoyancy flux as the bottom BC. We will clarify the text.

27. *Figure 7: maybe use log scale for y.*
We will remake the figure with a log scale and introduce it into the revised paper.

28. *Page 25 line 8: delete "quickly"*
We will remove the word "quickly".

29. *Figure 9a: area coverage of what? Updrafts? Please clarify.*
We were referring to the area coverage of cloud and cloud-core that are contained in the legend. We will make this explicitly clear in the figure caption in the revised manuscript.

30. *Section 9.3 and in general how is the code initialized? Random perturbations are added to mean profiles? Did the author try alternative approaches to seed turbulence?*
The code is initialized with random perturbations over the mean profiles, which is

sufficient for convective cases. We have also the options of introducing large vortices that are more efficient in generating turbulence under neutral or stable conditions. We will explain these options in the revised manuscript.

31. *Section 10: please provide info about the machines in section 10.1 (interconnect speed, processors per node, memory per nodes, ...). These details are needed to understand code scaling.*
    We will introduce references to the machine specifications and introduce a brief description of each of them in the revised manuscript.

32. *Figure 11: x axis label should be "processors"*
    We will fix this in the revised manuscript and use the word "cores".

Rebuttal to anonymous reviewer

The authors thank the reviewer for his thorough review. We will first address the reviewer's high-level comments, and thereafter the detailed comments point-by-point.

High-level suggested revisions:

1. *In sections 1-2, switching between anelastic and Boussinesq should be made clearer, and with what approximations. In the rest of the paper, it should be clear what "mode" each test is run in. - Claims of conservation should state the caveat that the simplified equations are in flux-conservation form, but that they are not fully mass- or energy-conservative (for example, looking at total mass, \rho_0 + \rho', in equation 10, is not conservative). p7 l18 how is it "fully energy conserving"? - A little more discussion of why this discretization was chosen, and what its benefits/limitations are. A little extra information would be a good way to flesh out the conclusion and provide more context for the reader.*
   We will improve the revised manuscript with respect to the differences between Boussinesq and anelastic in the model implementation. In short, the implementation of the governing equations is the same under both approximations, but under Boussinesq, the reference density and potential temperature are constant with height in the momentum and mass-conservation equations.

   Based on the reviewer's comments, we have not made our claims of energy conservation sufficiently clear. In Bannon (1996)'s anelastic approximation, the governing equations are energy conserving, in the sense that there is a correct transfer between kinetic and potential energy. This, however, does not mean that the discrete implementation is energy conserving. Our spatial discretization that follows Morinishi et al. (1998), conserves mass, momentum, and kinetic energy, which we demonstrate in the paper. We will make the distinction between energy conservation in the governing equations and in the implementation clear throughout the improved manuscript.

Detailed minor revisions:

2. *p1 abstract: "code reaches speedups of more than ... conventional code" running on what processors? Generally best to express it as a % of peak FLOPS and specific to the two architectures you compare in results.*
   We will explicitly mention in the abstract that it concerns single-GPU simulations and move the detailed information to the section on the scaling.

3. *p1 "approach the synoptic scales" remove the? to clarify, maybe add LES resolution (< 1km?) at "scales of 1000km or more"?*
   We will follow the reviewer's suggestion and add some explicit numbers to the statement.

4. *p2 l3, "order codes"? Older codes?*
   "Order codes" will be changed to "other codes".

5. *p2 last intro paragraph . . . it is worth mentioning Sec 5 (output), and 7 (instructions to reproduce), to encourage others to do the same, maybe mention w/ sec 13 or even move those sections to the end?*
   We agree with the reviewer that all sections need to be mentioned. We are not sure what the reviewer means by "encouraging others to do the same" Does this refer to reproducing our test cases?

6. *p2 l18 "constant with height z" maybe restate \rho_0(z) only to support eq. (2)?*
   We will write rho_0(z) instead of rho_0, to make clear that rho_0 is a function of height.

7. *p3 Derivation of eq(4) should be either referenced or add an extra step ... eq (5) should come first, for example, to introduce the potential temperature EOS that's sub- stituted into eq(4). - p3 l20 perturbational pressure form - not conservative / does not match eq (2)?*
   The reviewer is correct. We shall swap the order of the equation of state and the momentum equation and cite the paper of Bannon (1996) earlier.

8. *p4 l16, introduced N without an equation/definition?*
   We shall introduce the definition of N2 = db_0/dz in the text, and do so for all thermodynamic modes.

9. *p6-7 I appreciate the compactness of the notation and clarity in presenting it.*
   We appreciate the kind words of the reviewer. It has been a challenge to find a suitable notation.

10. *p7 eq (40), why not use a similarly compact 4th-order 5-pt wide stencil, instead of the larger 7-pt wide one?*
    The 7-pt stencil has the advantage that it is built out of the same building blocks as the other operators, and thus uses the same ghost cells.

11. *p9 DFT solver eq (45) is not clear ... assuming periodic bc's or cosine transform for Neumann bc's on pressure? Is there a reference for this approach?*
    The DFT operator is only performed in the periodic x and y directions. Based on comments of the first reviewer, we will introduce earlier in the paper that our code is periodic in the two horizontal dimensions.

12. *p9 "hat" DFT notation conflicts with "average" notation on p6.*
    We will introduce a different symbol for the Fourier transform in the revised manuscript.

13. *p9 eq(46-47) could mention "corresponding to eq(39-40) respectively" around l17-18?*
    We agree with the reviewer's suggestion and will refer to those equations.

14. *p9 l24, Ah! That's a big assumption, periodic lateral boundaries. Should be moved up and stated prominantly, along with motivation/limitations. Now I understand why p4*

l15 "periodic with slopes" was introduced.
Following both reviewers' comments, we will introduce in the introduction that our code is doubly periodic.

15. *p11 l12-16, is the model-top pressure constant in time or modified every time step? what value is used?*
    The model top pressure is the final result of the described procedure and depends on the surface pressure and the chosen reference profiles of temperature and humidity. MicroHH has the option of a constant profile in time, as well as a reference profile that updates in time.

16. *p12 l5, is filtering actually applied in your algorithm, and if so, at what resolution? Do you do anything to prevent discrete aliasing of unresolved wavelengths?*
    We do not use explicit filtering, but rely on the grid scale as a filter, which is a common procedure with atmospheric LES. With our numerical schemes, aliasing errors are small. We will introduce a short discussion on this in the revised manuscript.

17. *p12 tilde variables conflict with tilde "intermediate velocity" in eq (41)*
    We will use a different symbol in the revised manuscript.

18. *p12 eq (67) S_{ij} subscript? and what's the definition of Sˆ2?*
    In the revised manuscript we will write the full expression in terms of S_{ij}. The reviewer is correct that we forgot the subscripts.

19. *p12 l21, Nˆ2 definition here different than above p4 l16?*
    The reviewer is correct. We have failed to make clear that depending on the chosen thermodynamics, an appropriate definition of N^2 is used. We will clarify this in the improved manuscript, as mentioned in our reply to point 8.

20. *p13-15 sec 4.2 . . . is this a new atm turbulence model? The reference Wyngaard (2010) is an entire book, and it is not clear which tests warrant which boundary conditions, etc. p15 l5 is particularly confusing . . . might be worth describing Obukhov length and its use as a stability/mixing parameter, and why a look-up table is needed.*
    We will clarify the text. The lookup table is only there for performance reasons, as it outperforms a Newton-Raphson method.

21. *p15 l11, why would you not just include a background U_f and define a perturbational velocity from that? That would be compatible with periodic bc's, guarantee mass conservation, etc.*
    By doing so, the problem will remain. If the large-scale pressure force is applied to the perturbation velocities only, it is no longer ensured that the perturbations average to zero, without applying the presented correction.

22. *p15 bottom "adveciton" should be "advection"?*
    The reviewer is correct.

23. *p17 l19. "precompiler statements"? Meaning #define of GPU CUDA code? Any thoughts or statements on maintaining the different code bases in your C++ framework?*
    The use of precompiler statements is unavoidable, as we do not want to force the non-GPU user to install CUDA and compile the GPU code as well. We have chosen for an implementation in which the GPU code based is minimized, in order to ensure maintainability. We will elaborate our description of the CUDA implementation.

24. *p18 top, MPI-IO should have a reference?*
    We will introduce a reference.

25. *p18 l9, change netCDF footnote to reference?*
    We will introduce a reference.

26. *p18, maybe sections 5-7 should be moved/merged with 13 or all in an appendix?*
    We disagree with the reviewer here. We consider the presented topics in sections 5-7 of high relevance for a model description paper. Section 13 is located at the end of the paper following the GMD guidelines.

27. *p18 l25, love the post-processing mode based on restart files!*
    We thank the reviewer for this compliment. We would be very happy if this convinces the reviewer to use our code.

28. *p19 eq (98) should "4 \pi y" be z?*
    The reviewer is correct!

29. *p19 figure 1 / p20 l1 discussion ... L1 error in 2D should asymptote to h^4, even with 3rd-order boundary errors (O(N) pts * O(h^3) boundary error vs. O(N^2) pts O(h^4) interior error). Please explain? Also adding a 2nd set of dotted lines for 3rd- and 4th-order on the bottom set if u,v 4M fields will better show the break.*
    In the 4^{th}-order scheme, the boundary condition for vertical velocity w is set for global mass conservation rather than for 4^{th}-order accuracy. We will add a second set of dotted lines to help the reader observe the convergence of the schemes.

30. *p20 l8, "diffusion off" you mean viscosity, no source terms, etc. so that total energy should be conserved? What's your equation for "energy" in this test?*
    In this case, energy is kinetic energy. The model is run without viscosity and source terms, but with pressure solver to satisfy the continuity equation. We will clarify this in the improved manuscript.

31. *p20 l10, "its energy conservation." you mean improved? It doesn't conserve energy exactly.*
    The spatial discretization does conserve energy, but it is the time discretization that

does not. We shall clarify this in the text.

32. *p20 figure 2, maybe put top figure on log |\DeltaE| scale as well to distinguish the results better?*
    We prefer to keep our axis in its current form to show that our schemes are losing energy and therefore cannot lead to a blowup of the numerical solution. This is not possible if we plot the absolute value on a log scale. We shall clarify this in the improved manuscript.

33. *p21, line 4. Isn't there a difference in maximum CFL for each as well?*
    There is. In this experiment, however, we chose to compare the accuracy that can be achieved a fixed time step, as this allows us to estimate the convergence.

34. *p22, l6, "perfect match" . . . so perfect it's hard to see any difference at all. What do you attribute that too, since you have completely different discretizations, etc. How were the Moser 1999 results so similar? Could you quantify the differences, plot them, and explain them?*
    Both codes have fully converged results and are therefore identical if sufficient samples are averaged. Direct numerical simulation has, unlike LES, an exact solution, which makes the solution independent of the numerical schemes at sufficient resolution.

35. *p24, l12 . . . ditto for "nearly perfect match" here. Fig 6 also shows a "kink" in E_pp at higher \kappa. Is it worth explaining?*
    MOSER has spectral schemes, which introduce aliasing errors in the highest wave numbers. Even though aliasing errors are removed when the nonlinear operators are applied, the solver for the pressure introduces new ones.

36. *p26 l8, Fig 9a,d - why are the vertical velocities diverging with resolution?*
    This is often observed in LES-simulations of cumulus-topped boundary layers. Individual plumes that have a radius of only a few grid cells tend to overestimate velocity.

37. *p29 bottom p30. By putting these on a single GPU, you are avoiding communication overheads for the GPU. Did you run 1 MPI rank on the GPU? Did you run "n" MPI ranks on the CPU? For the B512 run you are getting very good (90%?) strong scaling for 1-4 CPU nodes.*
    In our view, GPUs mostly deliver a benefit if simulations can be run on a single GPU. Therefore, we have taken one GPU. Furthermore, at the moment, MicroHH is only supporting single GPU simulations. We agree with the reviewer that our comparison might be unfair in the sense that the GPU simulation does not need communication, whereas the CPU simulation does. We have, however, decided to focus on simulation of sizes that are common in atmospheric LES studies. We will improve the discussion in the paper to make this clear.

38. *p30 l4, “a parameterizations . . . has been” singular?*
    The reviewer is correct!

39. *p30 section 12 . . . could add a more comprehensive summary, call out any limitations or tradeoffs.*
    We will elaborate the conclusions and highlight MicroHH’s most important features and limitations in the revised manuscript.

MicroHH 1.0: a computational fluid dynamics code for direct numerical simulation and large-eddy simulation of atmospheric boundary layer flows

Chiel C. van Heerwaarden[1,2], Bart J. H. van Stratum[1,2], Thijs Heus[3], Jeremy A. Gibbs[4], Evgeni Fedorovich[5], and ~~Juan-Pedro~~ Juan Pedro Mellado[2]

[1]Meteorology and Air Quality Group, Wageningen University, Wageningen, The Netherlands
[2]Max Planck Institute for Meteorology, Hamburg, Germany
[3]Cleveland State University, Cleveland, OH, USA
[4]Department of Mechanical Engineering, University of Utah, UT, USA
[5]University of Oklahoma, Norman, OK, USA

Correspondence to: Chiel van Heerwaarden
(chiel.vanheerwaarden@wur.nl)

Abstract. This paper describes MicroHH 1.0, a new and open source (www.microhh.org) computational fluid dynamics code for the simulation of turbulent flows in the atmosphere. It is primarily made for direct numerical simulation, but also supports large-eddy simulation (LES). The paper covers the description of the governing equations, their numerical implementation, and the parametrizations included in the code. Furthermore, the paper presents the validation of the dynamical core in the form of convergence and conservation tests, and comparison of simulations of channel flows and slope flows against well-established test cases. The full numerical model, including the associated parametrizations for LES, has been tested for a set of cases under stable and unstable conditions, under the Boussinesq and anelastic approximation, and with dry and moist convection under stationary and time-varying boundary conditions. The paper presents performance tests showing good scaling from 256 to 32,768 processes. The Graphical Processing ~~Unit-enabled~~ Unit (GPU)-enabled version of the code ~~reaches speedups~~ can reach a speedup of more than an order of magnitude ~~with respect to the conventional code for a variety of cases~~for simulations that fit in the memory of a single GPU.

1 Introduction

In this paper we present a description of MicroHH 1.0, a new Computational Fluid Dynamics code for the simulation of turbulent flows in doubly periodic domains, with a focus on those in the atmosphere. MicroHH is designed for the direct numerical simulation (DNS) technique, but also supports the large-eddy simulation (LES) technique. Its applications range from neutral channel flows to cloudy atmospheric boundary layers in large domains. MicroHH is written in C++ and the Graphical Processing Units-enabled parts of the code in CUDA. The simulation algorithms have been designed and are written from scratch with the goal to create a fast and highly parallel code that is able to run on machines with more than 10,000 cores. This is a key requirement for the code to be able to perform DNS at very high Reynolds numbers, or to do LES at very high resolution (grid spacing less than 1 m), or in domains that approach the synoptic scales (beyond 1000 km). We decided to start from scratch, in order to be able to use C++ and its extensive possibilities in object oriented- and metaprogramming. Furthermore, the implementation of a dynamical core that is fully fourth-order in space, which is very beneficial for DNS, but to retain the option to switch to second-order accuracy for LES, required a new code design.

Even though we started from scratch, many of the ideas are the results of our experiences with ~~order~~ other codes. Here, DALES (Heus et al., 2010), UCLA-LES (Stevens et al., 2005), and PALM (Maronga et al., 2015), deserve a reference as MicroHH could not have been possible without those.

This paper is built up as following: in Sect. 2, we provide a full description of the governing equations of the dynamical core, and their numerical implementation is discussed in Sect. 3. Subsequently, in Sect. 4 we present the parameterizations and their underlying assumptions. Section 5 discusses the technical details of the code, and Sections 6 and 7 explain how to run the model and which output is generated. This is followed by a series of model tests on the validity and accuracy of the dynamical core in Sect. 8, and a series of more applied atmospheric flow cases based on previous studies (Sect. 9). Hereafter, the parallel performance is evaluated (Sect. 10). ~~After~~ Then, an overview of published work with MicroHH is presented (Sect. 11), followed by the future plans (Sect. 12) and the concluding remarks (Sect. 13). Finally, there is a short description where to get MicroHH, and where to find its tutorials and a selection of visualisations (Sect. 14).

2 Dynamical core: governing equations

The dynamical core of MicroHH solves the conservation equations of mass, momentum, and energy under the anelastic approximation (Bannon, 1996). Under this approximation, the state variables density, pressure, and temperature are described as small fluctuations (denoted with a prime in this paper) from corresponding vertical reference profiles (denoted with subscript zero) that are functions of height only. This form of the approximation directly simplifies to the Boussinesq approximation if the reference density ~~ρ_0 is assumed~~ $\rho_0(z)$ is taken to be constant with height z. Consequently, MicroHH does not need separate implementations of Boussinesq and anelastic approximations. To facilitate the subsequent discussion of the conservation equations, we define the scale height for density H_ρ based on the reference density profile

$$H_\rho \equiv \left(\frac{1}{\rho_0} \frac{d\rho_0}{dz} \right)^{-1}. \tag{1}$$

2.1 Conservation of mass

The conservation of mass is formulated using Einstein summation as

$$\frac{\partial \rho_0 u_i}{\partial x_i} = \rho_0 \frac{\partial u_i}{\partial x_i} + \rho_0 w H_\rho^{-1} = 0, \tag{2}$$

where u_i is the velocity vector (u, v, w) and x_i is the position vector (x, y, z). This formulation conserves the reference mass, as density perturbations are ignored in the equation (Lilly, 1996).

Under the Boussinesq approximation ($H_\rho \to \infty$), Eq. 2 simplifies to conservation of volume

$$\frac{\partial u_i}{\partial x_i} = 0. \tag{3}$$

2.2 ~~Conservation of momentum~~ Thermodynamic relations and ~~the equation~~ conservation of ~~state~~momentum

The thermodynamic relation between the fluctuations of virtual potential temperature, pressure, and density under the anelastic approximation is (see Bannon (1996) for its derivation)

$$\frac{\theta'_v}{\theta_{v0}} \approx \frac{p'}{\rho_0 g H_\rho} - \frac{\rho'}{\rho_0}, \tag{4}$$

where θ'_v is the perturbation virtual potential temperature, θ_{v0} the reference virtual potential temperature, p' is the perturbation pressure, g is the gravity acceleration, and ρ' is the perturbation density.

The corresponding momentum equation is written in the flux form, in order to assure ~~the best possible mass and~~ momentum conservation. The hydrostatic balance $dp_0/dz = -\rho_0 g$ has been subtracted ~~to arrive at the perturbation form~~ and Eq. 4 has been used to introduce potential temperature as the buoyancy variable to formulate the conservation of momentum as

$$\begin{aligned}\frac{\partial u_i}{\partial t} &= -\frac{1}{\rho_0}\frac{\partial \rho_0 u_i u_j}{\partial x_j} - \frac{\partial}{\partial x_i}\left(\frac{p'}{\rho_0}\right) \\ &+ \delta_{i3} g \frac{\theta'_v}{\theta_{v0}} + \nu \frac{\partial^2 u_i}{\partial x_j^2} + F_i,\end{aligned} \tag{5}$$

where ~~p' is the perturbation pressure,~~ δ is the Kronecker delta, ~~g is the gravity acceleration, θ'_v is the perturbation virtual potential temperature, θ_{v0} the reference virtual potential temperature,~~ ν the kinematic viscosity, and vector F_i represents external forces resulting from parameterizations or large-scale forcings. As Bannon (1996) showed, this formulation is energy-conserving in the sense that there is a consistent transfer between kinetic and potential energy.

~~The corresponding equation of state is (see Bannon (1996) for its derivation)~~

$$\frac{\theta'_v}{\theta_{v0}} = \frac{p'}{\rho_0 g H_\rho} - \frac{\rho'}{\rho_0}.$$

Under the Boussinesq approximation, the two equations simplify to

$$\frac{\theta'_v}{\theta_{v0}} \approx -\frac{\rho'}{\rho_0}, \tag{6}$$

$$\begin{aligned}\frac{\partial u_i}{\partial t} &= -\frac{\partial u_i u_j}{\partial x_j} - \frac{1}{\rho_0}\frac{\partial p'}{\partial x_i} \\ &+ \delta_{i3} g \frac{\theta'_v}{\theta_{v0}} + \nu \frac{\partial^2 u_i}{\partial x_j^2} + F_i.\end{aligned} \tag{7}$$

2.3 Pressure equation

The equation to acquire the pressure is diagnostic, because density fluctuations are neglected in the mass conservation equation under the anelastic approximation (Eq. 2). To simplify the notation, we define a function $f(u_i)$ that contains all right-hand side terms of Eq. 5, except the pressure gradient. To arrive at the equation that allows us to solve for the pressure, we multiply the equation with the base density ρ_0 and take its divergence. Conservation of mass ensures that the tendency term vanishes, and an elliptic equation for pressure remains

$$\frac{\partial}{\partial x_i}\left[\rho_0 \frac{\partial}{\partial x_i}\left(\frac{p'}{\rho_0}\right)\right] = \frac{\partial \rho_0 f(u_i)}{\partial x_i}. \tag{8}$$

Under the Boussinesq approximation the equation simplifies to

$$\frac{\partial^2}{\partial x_i^2}\left(\frac{p'}{\rho_0}\right) = \frac{\partial f(u_i)}{\partial x_i}. \tag{9}$$

In Sect. 3 we explain how these equations are solved numerically.

2.4 Conservation of an arbitrary scalar

The conservation equation of an arbitrary scalar ϕ is written in flux form

$$\frac{\partial \phi}{\partial t} = -\frac{1}{\rho_0}\frac{\partial \rho_0 u_j \phi}{\partial x_j} + \kappa_\phi \frac{\partial^2 \phi}{\partial x_j^2} + S_\phi, \tag{10}$$

where κ_ϕ is the diffusivity of the scalar, and S_ϕ represents sources and sinks of the variable.

2.5 Conservation of energy

MicroHH provides multiple options for the energy conservation equation. The conservation equation for potential temperature for dry dynamics θ can be written as

$$\frac{\partial \theta}{\partial t} = -\frac{1}{\rho_0}\frac{\partial \rho_0 u_j \theta}{\partial x_j} + \kappa_\theta \frac{\partial^2 \theta}{\partial x_j^2} + \frac{\theta_0}{\rho_0 c_p T_0} Q, \tag{11}$$

where κ_θ is the thermal diffusivity for heat, and Q represents external sources and sinks of heat. A second option for moist dynamics is available. This has an identical conservation equation, but with liquid water potential temperature θ_l~~(moist dynamics)~~, rather than θ as the conserved variable (see Sect. 3.9 for details).

A third, more simplified mode, is available for dry dynamics under the Boussinesq approximation. Here, the equation of state (Eq. 6) can be eliminated and the conservation of momentum and energy can be written ~~using buoyancy $b \equiv -(g/\rho_0)\rho'$ as~~ in terms of buoyancy $b \equiv (g/\theta_{v0})\,\theta'_v$ as

$$\frac{\partial u_i}{\partial t} + \frac{\partial u_i u_j}{\partial x_j} = -\frac{1}{\rho_0}\frac{\partial p'}{\partial x_i} + \delta_{i3} b + \nu \frac{\partial^2 u_i}{\partial x_j^2}, \tag{12}$$

$$\frac{\partial b}{\partial t} + \frac{\partial b u_j}{\partial x_j} = \kappa_b \frac{\partial^2 b}{\partial x_j^2}\cancel{.} + Q_b, \tag{13}$$

with κ_b being the diffusivity for buoyancy~~.~~, and Q_b is an external buoyancy source. By using buoyancy, length and time remain as the only two dimensions, which proves convenient for dimensional analysis. In this formulation, θ'_v is the fluctuation of the virtual potential temperature with respect to the surface value θ_{v0}. The consequence is that the buoyancy increases with height in a stratified atmosphere, analogously to the virtual potential temperature (see Garcia and Mellado (2014), their Fig. B1 and van Heerwaarden and Mellado (2016), their Fig. 7a)

With a slight modification to the ~~previous set of equations~~definition of θ'_v, it is possible to study slope flows in periodic domains. ~~If we~~ We define θ'_v as the fluctuation with respect to a linearly stratified background profile $\theta_{v0} + (d\theta_v/dz)_0 z$. The background stratification in units of buoyancy is $N^2 \equiv (g/\theta_{v0})(d\theta_v/dz)_0$. If we work out the governing equations again and introduce a slope α (~~positive anticlockwise~~x-axis pointing upslope, see Fedorovich and Shapiro (2009), their Fig. 1) in the x-direction, ~~take the proper gravity vector, and subtract the background buoyancy profile $N^2 z$ from the buoyancy value, the set of Eqs. 12 and ?? becomes~~

$$\frac{\partial u}{\partial t} + \frac{\partial u_j u}{\partial x_j} = -\frac{1}{\rho_0}\frac{\partial p'}{\partial x} + \sin(\alpha) b + \nu \frac{\partial^2 u}{\partial x_j^2},$$

$$\frac{\partial w}{\partial t} + \frac{\partial u_j w}{\partial x_j} = -\frac{1}{\rho_0}\frac{\partial p'}{\partial z} + \cos(\alpha) b + \nu \frac{\partial^2 w}{\partial x_j^2},$$

$$\frac{\partial b}{\partial t} + \frac{\partial b u_j}{\partial x_j} = \kappa_b \frac{\partial^2 b}{\partial x_j^2} - (u \sin(\alpha) + w \cos(\alpha)) N^2$$

we find

$$\frac{\partial u}{\partial t} + \frac{\partial u_j u}{\partial x_j} = -\frac{1}{\rho_0}\frac{\partial p'}{\partial x} + \sin(\alpha) b + \nu \frac{\partial^2 u}{\partial x_j^2}, \tag{14}$$

$$\frac{\partial w}{\partial t} + \frac{\partial u_j w}{\partial x_j} = -\frac{1}{\rho_0}\frac{\partial p'}{\partial z} + \cos(\alpha) b + \nu \frac{\partial^2 w}{\partial x_j^2}, \tag{15}$$

$$\frac{\partial b}{\partial t} + \frac{\partial b u_j}{\partial x_j} = \kappa_b \frac{\partial^2 b}{\partial x_j^2} - (u \sin(\alpha) + w \cos(\alpha)) N^2 + Q_b, \tag{16}$$

where the evolution equation of ~~v is omitted as it contains no changes.~~ v is omitted as it contains no changes.

3 Dynamical core: numerical implementation

3.1 Grid

MicroHH is discretized on a staggered Arakawa C-grid, where the scalars are located in the center of a grid cell and the three velocity components at the faces. The code can work with stretched grids in the vertical dimension. The grid is initialized from a vertical profile that contains the heights of the cell centres. The locations of the faces are determined consistently with the spatial order of the interpolations that are described in Sect. 3.4. All spatial operators in the model, such as the advection and diffusion, default to the same order as the grid, and can be overriden according to the user's wishes (see Sect. 6).

There is the option to apply a uniform translation velocity to the grid, thus to let the grid move with the flow. This so-called Galilean transformation is allowed as the Navier-Stokes equations are invariant under translation. It has the potential to allow for larger time steps and to increase the accuracy of simulations.

3.2 Three-dimensional fields

In order to solve the governing equations, MicroHH generates at initialization three-dimensional fields of the prognostic variables. These are the three velocity components (Eqs. 5 or 7), and the thermodynamic variables (Eqs. 11, 13, or 16). Furthermore, the user has the option to define additional passive scalars (Eq. 10). Each of the prognostic fields has an additional three-dimensional field assigned to store its tendency (see Sect. 3.3). Furthermore, a diagnostic field is assigned for the pressure, as well as three or four additional ones for intermediate computations. Newly implemented physical parameterizations have the option to request additional three-dimensional fields at initialization of the specific parameterization.

The generation of turbulence requires perturbations to the initial fields. MicroHH has two option to superimpose perturbations on any of the prognostic variables. These perturbations can be random noise of which the amplitude and location can be controlled, as well as two-dimensional rotating vortices with an axis aligned with the x- or y-dimension. The former option is the most commonly used method to start convective turbulence, whereas the latter is the default for neutral or stably-stratified flows, which develop turbulence more easily from larger perturbations.

3.3 Time integration

The prognostic equations are solved using low-storage Runge-Kutta time integration schemes. Such schemes require two fields per variable: one that contains the actual value, which we denote with ϕ in this section, and one that represents the tendencies, denoted with $\delta\phi$. The code provides two options: a three-stage third-order scheme (Williamson, 1980) and a five-stage fourth-order scheme (Carpenter and Kennedy, 1994). Both can be written in the same generic form in semi-discrete formulation as

$$(\delta\phi)_n = f(\phi_n) + a_n (\delta\phi)_{n-1} \tag{17}$$

$$\phi_{n+1} = \phi_n + b_n \Delta t (\delta\phi)_n, \tag{18}$$

where f is a function that represents the computation of all right-hand side terms, a_n and b_n are the coefficients for the Runge-Kutta method at stage n, and Δt is the time step. Expression $f(\phi_n)$ represents thus the actual tendency calculated using, for instance, Eqs. 5 or 10, whereas $(\delta\phi)_n$ is a composite of the actual tendency and those from the previous stages. In low-storage form, the tendencies of the previous stage $(\delta\phi)_{n-1}$ are retained and multiplied with a_n at the beginning of a stage, except for the first stage, where $a_1 = 0$.

For the third-order scheme the vectors a_n and b_n are

$$a_n = \left\{0, -\frac{5}{9}, -\frac{153}{128}\right\}, \tag{19}$$

$$b_n = \left\{\frac{1}{3}, \frac{15}{16}, \frac{8}{15}\right\}. \tag{20}$$

For the fourth-order scheme the vectors a and b are

$$\begin{aligned} a_n &= \left\{0, -\frac{567301805773}{1357537059087}, -\frac{2404267990393}{2016746695238}, \right. \\ & \left. -\frac{3550918686646}{2091501179385}, -\frac{1275806237668}{842570457699} \right\} \end{aligned} \tag{21}$$

$$\begin{aligned} b_n &= \left\{\frac{1432997174477}{9575080441755}, \frac{5161836677717}{13612068292357}, \frac{1720146321549}{2090206949498}, \right. \\ & \left. \frac{3134564353537}{4481467310338}, \frac{2277821191437}{14882151754819} \right\} \end{aligned} \tag{22}$$

The reduced truncation error of the fourth-order scheme makes the scheme preferable over the third-order scheme under many conditions (see Sect. 8.2). The code can be run with a fixed Δt, as well as an adaptive time step based on the local flow velocities.

3.4 ~~Grid~~

~~MicroHH is discretized on a staggered Arakawa C-grid, where the scalars are located in the center of a grid cell and the three velocity components at the faces.~~

~~The code can work with stretched grids in the wall-bounded dimension. The grid is initialized from a vertical profile that contains the heights of the cell centres. The locations of the faces are determined consistently with the spatial order of the interpolations that are described in the next section.~~

~~There is the option to apply a uniform translation velocity to the grid, thus to let the grid move with the flow. This so-called Galilean transformation is allowed as the Navier-Stokes equations are invariant under translation. It has the potential to allow for larger time steps and to increase the accuracy of simulations.~~

3.4 Building blocks of the spatial discretization

The spatial operators are based on finite differences. The code supports second-order and fourth-order accurate discretizations following Morinishi et al. (1998); Vasilyev (2000). From Taylor series, spatial operators can be derived that constitute the building blocks of more advanced operators, such as the advection and diffusion operators. In the following subsections we describe the elementary operators and the composite operators that can be derived from them. We ~~use only two dimensions for brevity~~have selected a set of examples that cover the relevant operators.

We define two second-order interpolation operators, one with a small stencil and one with a wide stencil, as

$$\phi_{i,j\,i,j,k} \approx \overline{\phi}^{2x}_{i,j\,i,j,k} \equiv \frac{\phi_{i-\frac{1}{2},j,k} + \phi_{i+\frac{1}{2},j,k}}{2}, \tag{23}$$

$$\phi_{i,j\,i,j,k} \approx \overline{\phi}^{2xL}_{i,j\,i,j,k} \equiv \frac{\phi_{i-\frac{3}{2},j,k} + \phi_{i+\frac{3}{2},j,k}}{2}, \tag{24}$$

Interpolations are marked with a ~~hat~~bar. The superscript indicates the spatial order (2), and the direction (x) and has an extra qualifier L when it is taken using the wide stencil. The subscript indicates the position on the grid (i,j).

The gradient operators, denoted with letter δ, are defined in a similar way

$$\left.\frac{\partial\phi}{\partial x}\right|_{i,j\,i,j,k} \approx \delta^{2x}\left(\phi\right)_{i,j\,i,j,k} \equiv \frac{\phi_{i+\frac{1}{2},j,k}-\phi_{i-\frac{1}{2},j,k}}{x_{i+\frac{1}{2}}-x_{i-\frac{1}{2}}} \tag{25}$$

$$\left.\frac{\partial\phi}{\partial x}\right|_{i,j\,i,j,k} \approx \delta^{2xL}\left(\phi\right)_{i,j\,i,j,k} \equiv \frac{\phi_{i+\frac{3}{2},j,k}-\phi_{i-\frac{3}{2},j,k}}{x_{i+\frac{3}{2}}-x_{i-\frac{3}{2}}} \tag{26}$$

We use the Einstein summation in the operators. For instance, the divergence of vector $\cancel{u_i|_{i,j}}$ $u_i|_{i,j,k}$ can be written as $\cancel{\delta^{2x_i}\left(u_i\right)_{i,j}}$ $\delta^{2x_i}\left(u_i\right)_{i,j,k}$.

The fourth-order operators, written down in the same notation, are defined as

$$\phi_{i,j\,i,j,k} \approx \overline{\phi}^{4x}{}_{i,j\,i,j,k} \equiv \frac{-\phi_{i-\frac{3}{2},j,k}+9\phi_{i-\frac{1}{2},j,k}+9\phi_{i+\frac{1}{2},j,k}-\phi_{i+\frac{3}{2},j,k}}{16}. \tag{27}$$

The biased version of this operator (subscript b) can be applied in the vicinity of the boundaries at the bottom and top. Here, we show the biased stencil that can be applied for vertical interpolation near the bottom

$$\phi_{i,j\,i,j,k} \approx {}^{4xb}_{i,j}\overline{\phi}^{4zb}_{i,j,k} \equiv \frac{5\phi_{i,j,k-\frac{1}{2}}+15\phi_{i,j,k+\frac{1}{2}}-5\phi_{i,j,k+\frac{3}{2}}+\phi_{i,j,k+\frac{5}{2}}}{16}. \tag{28}$$

Note that we only write down the bottom boundary for brevity.

The centered and biased fourth-order gradient operators are

$$\begin{aligned}\left.\frac{\partial\phi}{\partial x}\right|_{i,j\,i,j,k} &\approx \delta^{4x}\left(\phi\right)_{i,j\,i,j,k}\\ &\equiv \frac{\phi_{i-\frac{3}{2},j,k}-27\phi_{i-\frac{1}{2},j,k}+27\phi_{i+\frac{1}{2},j,k}-\phi_{i+\frac{3}{2},j,k}}{x_{i-\frac{3}{2}}-27x_{i-\frac{1}{2}}+27x_{i+\frac{1}{2}}-x_{i+\frac{3}{2}}},\end{aligned} \tag{29}$$

and

$$\begin{aligned}\left.\frac{\partial\phi}{\partial z}\right|_{i,j\,i,j,k} &\approx \delta^{4xb\,4zb}\left(\phi\right)_{i,j\,i,j,k}\\ &\equiv \frac{-23\phi_{i,j,k-\frac{1}{2}}+21\phi_{i,j,k+\frac{1}{2}}+3\phi_{i,j,k+\frac{3}{2}}-\phi_{i,j,k+\frac{5}{2}}}{-23x_{i-\frac{1}{2}}z_{k-\frac{1}{2}}+21x_{i+\frac{1}{2}}z_{k+\frac{1}{2}}+3x_{i+\frac{3}{2}}z_{k+\frac{3}{2}}-x_{i+\frac{5}{2}}z_{k+\frac{5}{2}}}\end{aligned} \tag{30}$$

3.5 Boundary conditions

The lateral boundaries in MicroHH are periodic. The bottom and top boundary conditions can be formulated in their most general form as the Robin boundary condition

$$a\phi_s + b\left.\frac{\partial\phi}{\partial z}\right|_s = c, \tag{31}$$

with a, b and c as constants. This gives the Dirichlet boundary condition when $a=1$, $b=0$, and the Neumann boundary condition when $a=0$, $b=1$.

MicroHH makes use of ghost cells in order to avoid the need of biased schemes for single interpolation or gradient operators near the wall. The values at the ghost cells are derived making use of the boundary conditions following Morinishi et al. (1998). The ghost cells for the Dirichlet boundary conditions in the second-order accurate discretization are

$$\phi_{-\frac{1}{2}} = 2c - \phi_{\frac{1}{2}}, \tag{32}$$

whereas those for the Neumann boundary condition are

$$\phi_{-\frac{1}{2}} = -c\left(-z_{-\frac{1}{2}} + z_{\frac{1}{2}}\right) + \phi_{\frac{1}{2}}. \tag{33}$$

In case of the fourth-order scheme, we have two ghost cells, and therefore a second boundary condition is required. Here, we set the third derivative equal to zero following (Morinishi et al., 1998). For the Dirichlet boundary condition we then acquire the following expressions for the ghost cells

$$\phi_{-\frac{1}{2}} = \frac{8c - 6\phi_{\frac{1}{2}} + \phi_{\frac{3}{2}}}{3}, \tag{34}$$

$$\phi_{-\frac{3}{2}} = 8c - 6\phi_{\frac{1}{2}} + \phi_{\frac{3}{2}}, \tag{35}$$

whereas in case of a Neumann boundary condition we find

$$\phi_{-\frac{1}{2}} = -c\frac{z_{-\frac{3}{2}} - 27z_{-\frac{1}{2}} + 27z_{\frac{1}{2}} - z_{\frac{3}{2}}}{24} + \phi_{\frac{1}{2}}, \tag{36}$$

$$\phi_{-\frac{3}{2}} = -3c\frac{z_{-\frac{3}{2}} - 27z_{-\frac{1}{2}} + 27z_{\frac{1}{2}} - z_{\frac{3}{2}}}{24} + \phi_{\frac{3}{2}}. \tag{37}$$

3.6 Advection

We use the previously introduced notation to describe the more complex operators and expand them for illustration. The advection term is discretized in the flux form, where ϕ is an arbitrary scalar located in the center of the grid cell. In the second-order case, this gives the following discretization:

$$\begin{aligned}
\left.\frac{\partial u\phi}{\partial x}\right|_{i,j\,i,j,k} + \left.\frac{\partial v\phi}{\partial y}\right|_{i,j\,i,j,k} &\approx \delta^{2x}\left(u\overline{\phi}^{2x}\right)_{i,j\,i,j,k} + \delta^{2y}\left(v\overline{\phi}^{2y}\right)_{i,j\,i,j,k} \\
&= \frac{u_{i+\frac{1}{2},j,k}\overline{\phi}^{2x}_{i+\frac{1}{2},j,k} - u_{i-\frac{1}{2},j,k}\overline{\phi}^{2x}_{i-\frac{1}{2},j,k}}{} x_{i+\frac{1}{2}} - x_{i-\frac{1}{2}} \\
&+ \frac{v_{i,j+\frac{1}{2},k}\overline{\phi}^{2y}_{i,j+\frac{1}{2},k} - v_{i,j-\frac{1}{2},k}\overline{\phi}^{2y}_{i,j-\frac{1}{2},k}}{} y_{j+\frac{1}{2}} - y_{j-\frac{1}{2}}.
\end{aligned} \tag{38}$$

The discretization of the advection of the velocity components (see Eqs. 5 and 7) involves extra interpolations as the following example illustrates:

$$\begin{aligned}
\left.\frac{\partial vu}{\partial x}\right|_{i,j\,i,j,k} &= \delta^{2x}\left(\overline{v}^{2y}\overline{u}^{2x}\right)_{i,j\,i,j,k} \\
&= \frac{\overline{v}^{2y}_{i+\frac{1}{2},j,k}\overline{u}^{2x}_{i+\frac{1}{2},j,k} - \overline{v}^{2y}_{i-\frac{1}{2},j,k}\overline{u}^{2x}_{i-\frac{1}{2},j,k}}{} x_{i+\frac{1}{2}} - x_{i-\frac{1}{2}}.
\end{aligned} \tag{39}$$

In the standard fourth-order scheme, the scalar advection in flux form is represented by

$$
\begin{aligned}
\left.\frac{\partial u\phi}{\partial x}\right|_{i,j\,i,j,k} &\approx \delta^{4x}\left(u\overline{\phi}^{4x}\right)_{i,j\,i,j,k} \\
&= \Bigg(u_{i-\frac{3}{2},j\,i-\frac{3}{2},j,k}\overline{\phi}^{4x}_{i-\frac{3}{2},j\,i-\frac{3}{2},j,k} - 27u_{i-\frac{1}{2},j\,i-\frac{1}{2},j,k}\overline{\phi}^{4x}_{i-\frac{1}{2},j\,i-\frac{1}{2},j,k} \\
&\quad +27u_{i+\frac{1}{2},j\,i+\frac{1}{2},j,k}\overline{\phi}^{4x}_{i+\frac{1}{2},j\,i+\frac{1}{2},j,k} - u_{i+\frac{3}{2},j\,i+\frac{3}{2},j,k}\overline{\phi}^{4x}_{i+\frac{3}{2},j\,i+\frac{3}{2},j,k}\Bigg) \\
&\quad / \left(x_{i-\frac{3}{2}} - 27x_{i-\frac{1}{2}} + 27x_{i+\frac{1}{2}} - x_{i+\frac{3}{2}}\right).
\end{aligned} \tag{40}
$$

Hereafter, we assume that operator notation is clear and only expand it where necessary.

MicroHH has a fully kinetic energy-conserving fourth-order advection scheme (Morinishi et al., 1998) available. ~~This~~ The scheme is constructed by interpolation of two kinetic energy-conserving second-order ~~schemes~~ discretizations to eliminate the second-order error as illustrated below

$$
\left.\frac{\partial u\phi}{\partial x}\right|_{i,j\,i,j,k} \approx \frac{9}{8}\delta^{2x}\left(u\overline{\phi}^{2x}\right)_{i,j\,i,j,k} - \frac{1}{8}\delta^{2xL}\left(u\overline{\phi}^{2xL}\right)_{i,j\,i,j,k} \tag{41}
$$

to ensure that velocity variances are conserved under advection.

Velocity interpolations, such as those in Eq. 39, still need to be performed with fourth-order accuracy (Eq. 27) in order to be fourth-order accurate (see Morinishi et al. (1998) for details). The expression

$$
\left.\frac{\partial vu}{\partial x}\right|_{i,j\,i,j,k} \approx \frac{9}{8}\delta^{2x}\left(\overline{v}^{4y}\overline{u}^{2x}\right)_{i,j\,i,j,k} - \frac{1}{8}\delta^{2xL}\left(\overline{v}^{4y}\overline{u}^{2xL}\right)_{i,j\,i,j,k} \tag{42}
$$

includes, for instance, a combination of second- and fourth-order interpolations.

To increase the overall accuracy of the second-order advection operator, there is an option available to only increase the interpolation part to fourth order

$$
\left.\frac{\partial u\phi}{\partial x}\right|_{i,j\,i,j,k} \approx \delta^{2x}\left(u\overline{\phi}^{4x}\right)_{i,j\,i,j,k}. \tag{43}
$$

3.7 Diffusion

We apply a discretization for diffusion that can be written as the divergence of a gradient, using the building blocks defined earlier in this section. As this operator is identical in all directions, we present it in one direction only

$$
\kappa_\phi \left.\frac{\partial^2\phi}{\partial x^2}\right|_{i,j\,i,j,k} \approx \kappa_\phi\delta^{2x}\left(\delta^{2x}\left(\phi\right)\right)_{i,j\,i,j,k}, \tag{44}
$$

$$
\kappa_\phi \left.\frac{\partial^2\phi}{\partial x^2}\right|_{i,j\,i,j,k} \approx \kappa_\phi\delta^{4x}\left(\delta^{4x}\left(\phi\right)\right)_{i,j\,i,j,k}. \tag{45}
$$

On an equidistant grid, this provides the well-known second-order accurate operator for the second derivative

$$
\kappa_\phi\delta^{2x}\left(\delta^{2x}\left(\phi\right)\right)_{i,j\,i,j,k} = \kappa_\phi\frac{\phi_{i-1,j,k} - 2\phi_{i,j,k} + \phi_{i+1,j,k}}{\left(\Delta x\right)^2}, \tag{46}
$$

where Δx is the uniform grid spacing.

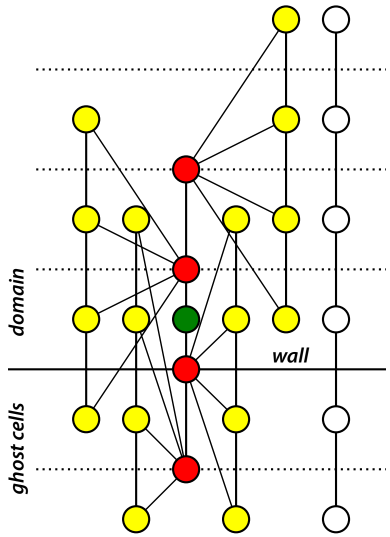

Figure 1. Schematic of the diffusion discretization near the wall. The green node is the evaluation point at the center of the first cell above the wall, the red node is the stencil of the divergence operator, and yellow nodes show the stencils of the four gradient operators over which the divergence is evaluated. White nodes indicate the extent of the stencil.

For the fourth-order accurate operator, a seven-point stencil is used:

$$
\begin{aligned}
&\kappa_\phi \delta^{4x}\left(\delta^{4x}\left(\phi\right)\right)_{i,j\;i,j,k} \\
&= \frac{\kappa_\phi}{576\left(\Delta x\right)^2}\Big(\phi_{i-3,j\;i-3,j,k} - 54\phi_{i-2,j\;i-2,j,k} + 783\phi_{i-1,j\;i-1,j,k} \\
&-1460\phi_{i,j\;i,j,k} + 783\phi_{i+1,j\;i+1,j,k} - 54\phi_{i+2,j\;i+2,j,k} + \phi_{i+3,j\;i+3,j,k}\Big).
\end{aligned}
\tag{47}
$$

~~The seven point wide stencil and its properties has been discussed in detail in Castillo et al. (1995).~~

Whereas diffusion can be computed with fourth-order accuracy using a five-point stencil, we use a seven-point stencil as it extends naturally to non-uniform grids as explained in Castillo et al. (1995). The usage of a seven-point stencil requires special care near the walls. In Fig. 1 we show an example of how the second derivative in the vertical direction is computed for a scalar at the first model level (green node in Fig. 1). The calculation of the divergence (Fig. 1, red stencil) requires the gradient located at the first face below the wall (lowest red node in Fig. 1), which can only be acquired using the biased gradient operator (Eq. 30 and yellow stencil connected to lowest red node in Fig. 1). The extent of the complete stencil near the wall (white nodes, Fig. 1) is thus six points, rather than seven.

3.8 Pressure

Eqs. 8 and 9 are solved following the method of Chorin (1968). This is a fractional step method that first computes intermediate values of the velocity components for the next time step, based on all right hand side terms of the momentum conservation equation Eq. 5

$$u_i^*\Big|_{i,j,k}^{t+1} = u_i|_{i,j,k}^{t} + \Delta t \, f_i|_{i,j,k}^{t} \,, \tag{48}$$

with the intermediate velocity components denoted with ~~a tilde~~an asterix.

The velocity values at the next time step can be computed as soon as the pressure is known, using

$$u_i|_{i,j,k}^{t+1} = u_i^*\Big|_{i,j,k}^{t+1} - \Delta t \, \delta^{nx_i} \left(\frac{p}{\rho_0} \right) \Big|_{i,j,k}^{t} . \tag{49}$$

In order to compute the pressure, we multiply the previous equation with the reference density and take its gradient, arriving at

$$\begin{aligned} \delta^{nx_i} \left(\rho_0 u_i \right)|_{i,j,k}^{t+1} &= \delta^{nx_i} \left(\rho_0 u_i^* \right) \Big|_{i,j,k}^{t+1} \\ &- \Delta t \, \delta^{nx_i} \left[\rho_0 \delta^{nx_i} \left(\frac{p}{\rho_0} \right) \right] \Big|_{i,j,k}^{t} , \end{aligned} \tag{50}$$

where n indicates the spatial order, and the subscript i in superscript x_i indicates that δ^{nx_i} is a divergence operator. The left hand side equals zero due to mass conservation at the next time step (Eq. 2). The resulting equation is the Poisson equation that is the discrete equivalent of Eq. 8. Rewriting this equation leads to

$$\frac{\delta^{nx_i} \left(\rho_0 u_i^* \right)|_{i,j,k}^{t+1}}{\Delta t} = \delta^{nx_i} \left[\rho_0 \delta^{nx_i} \left(\frac{p}{\rho_0} \right) \right] \Big|_{i,j,k}^{t} . \tag{51}$$

To simplify the notation, we denote the left-hand side term as ψ and the p/ρ_0 term on the right hand side as π. Solving a Poisson equation is a global operation. Because the computed fields are periodic in the horizontal directions on an equidistant grid, and a Poisson equation is linear, we can perform a Fourier transform in the two horizontal directions

$$\widehat{\psi}_{l,m,k} = -k_{*n}^2 \widehat{\pi}_{l,m,k} - l_{*n}^2 \widehat{\pi}_{l,m,k} + \delta^{nz} \left[\rho_0 \delta^{nz} \left(\widehat{\pi} \right) \right]_{l,m,k} , \tag{52}$$

where Fourier transformed variables are denoted with a hat, the spatial order of the operation with n, and the wave numbers in the two horizontal dimensions x and y are l and m respectively. Variables k_*^2 and l_*^2 are the squares of the modified wave numbers

$$-k_{*2}^2 \equiv 2 \frac{\cos(k \Delta x)}{\left(\Delta x\right)^2} - \frac{2}{\left(\Delta x\right)^2} \tag{53}$$

and

$$\begin{aligned} -k_{*4}^2 &\equiv 2 \frac{\cos(3k\Delta x) - 54 \cos(2k\Delta x) + 783 \cos(k\Delta x)}{576 \left(\Delta x\right)^2} \\ &- \frac{1460}{576 \left(\Delta x\right)^2} , \end{aligned} \tag{54}$$

where the former is the modified wave number for the second-order accurate solver and the latter is the wave number for the fourth-order one. Note that the coeffecients correspond to those in Eqs. 46 and 47. Both expressions satisfy the limit $\lim_{\Delta x \to 0} k_{*n}^2 = k^2$, where n is the order of the scheme.

Solving Eq. 52 for $\widehat{\pi}$ requires solving a banded matrix ~~, which~~ for the vertical direction in which the walls are located. This matrix is tridiagonal for the second-order solver and hepta-diagonal for the fourth-order solver. For this, a standard Thomas algorithm (Thomas, 1949) is used. After the pressure is acquired, inverse Fourier transforms are applied and subsequently the pressure gradient term (see Eqs. 5 and 7) is computed for all three components of the velocity tendency. Note that the computation of the corrected velocity components does not require a boundary condition for pressure (see Vreman (2014) for details).

3.9 ~~Boundary conditions~~

~~The lateral boundaries in MicroHH are periodic. The bottom and top boundary conditions can be formulated in their most general form as the Robin boundary condition~~

$$a\phi_s + b\left.\frac{\partial \phi}{\partial z}\right|_s = c,$$

~~with a, b and c as constants. This gives the Dirichlet boundary condition when $a = 1$, $b = 0$, and the Neumann boundary condition when $a = 0$, $b = 1$.~~

~~MicroHH makes use of ghost cells in order to avoid the need of biased schemes for single interpolation or gradient operators near the wall. The values at the ghost cells are derived making use of the boundary conditions following Morinishi et al. (1998). The ghost cells for the Dirichlet boundary conditions in the second-order accurate discretization are~~

$$\phi_{-\frac{1}{2}} = 2c - \phi_{\frac{1}{2}},$$

~~whereas those for the Neumann boundary condition are~~

$$\phi_{-\frac{1}{2}} = -c\left(-z_{-\frac{1}{2}} + z_{\frac{1}{2}}\right) + \phi_{\frac{1}{2}}.$$

~~In case of the fourth-order scheme, we have two ghost cells, and therefore a second boundary condition is required. Here, we set the third derivative equal to zero following (Morinishi et al., 1998). For the Dirichlet boundary condition we then acquire the following expressions for the ghost cells~~

$$\phi_{-\frac{1}{2}} = \frac{8c - 6\phi_{\frac{1}{2}} + \phi_{\frac{3}{2}}}{3},$$

$$\phi_{-\frac{3}{2}} = 8c - 6\phi_{\frac{1}{2}} + \phi_{\frac{3}{2}},$$

~~whereas in case of a Neumann boundary condition we find~~

$$\cancel{\phi_{-\frac{1}{2}} \equiv -c\frac{z_{i-\frac{3}{2}} - 27z_{i-\frac{1}{2}} + 27z_{i+\frac{1}{2}} - z_{i+\frac{3}{2}}}{24} + \phi_{\frac{1}{2}},}$$

$$\cancel{\phi_{-\frac{3}{2}} \equiv -3c\frac{z_{i-\frac{3}{2}} - 27z_{i-\frac{1}{2}} + 27z_{i+\frac{1}{2}} - z_{i+\frac{3}{2}}}{24} + \phi_{\frac{3}{2}}.}$$

3.9 Thermodynamics

MicroHH supports the potential (θ) and liquid water potential (θ_l) temperature as thermodynamic variables (Sect. 2.5). The dry (θ) and moist (θ_l) thermodynamics are related through the use of a total specific humidity q_t, which is defined as the sum of the water vapour specific humidity (q_v) and the cloud liquid water specific humidity (q_l). In the absence of liquid water, $\theta_\mathrm{l} = \theta$, in the presence of liquid water, the liquid water potential temperature is approximated as (Betts, 1973)

$$\theta_\mathrm{l} \approx \theta - \frac{L_\mathrm{v}}{c_\mathrm{p}\,\Pi} q_\mathrm{l}, \tag{55}$$

where L_v is the latent heat of vaporization, c_p the specific heat of dry air at constant pressure, and Π is the Exner function

$$\Pi = \left(\frac{p}{p_{00}}\right)^{R_\mathrm{d}/c_\mathrm{p}}, \tag{56}$$

where p is the hydrostatic pressure, p_{00} a constant reference pressure, and R_d the gas constant for dry air. The cloud liquid water content is calculated as

$$q_\mathrm{l} = \max(0,\ q_\mathrm{t} - q_\mathrm{s}), \tag{57}$$

where q_s is the saturation specific humidity

$$q_\mathrm{s} = \frac{\epsilon\, e_\mathrm{s}}{p - (1-\epsilon)\, e_\mathrm{s}}, \tag{58}$$

with ϵ the ratio between the gas constant for dry air and the gas constant for water vapour ($R_\mathrm{d}/R_\mathrm{v}$), and e_s the saturation vapor pressure. The latter is approximated using a 10^{th} order Taylor expansion at $T = 0$ degree Celsius of the Arden Buck equation (Buck, 1981). q_l is adjusted iteratively to arrive at a consistent state where $q_\mathrm{v} = q_\mathrm{s}$. Finally, the virtual potential temperature (Eq. 5) is defined in MicroHH as

$$\theta_\mathrm{v} \equiv \theta\left(1 - \left[1 - \frac{R_\mathrm{v}}{R_\mathrm{d}}\right] q_\mathrm{t} - \frac{R_\mathrm{v}}{R_\mathrm{d}} q_\mathrm{l}\right) \tag{59}$$

The base state pressure and density are calculated assuming a hydrostatic equilibrium: $\mathrm{d}p_0 = -\rho_0 g \mathrm{d}z$, with the density defined as $\rho_0 = p_0 / (R_\mathrm{d}\, \Pi\, \theta_{\mathrm{v}0})$. Integration with height results in

$$p_{0;k+1} = p_{0;k} \exp\left(\frac{-g(z_{k+1} - z_k)}{R_\mathrm{d}\, \Pi\, \theta_{\mathrm{v}0}} \right) \tag{60}$$

where $\theta_{\mathrm{v}0}$ is the average virtual potential temperature between z_k and z_{k+1}. This equation is applied from a given surface pressure to the model top, alternating the calculations at the full and half model levels. That is, given the full thermodynamic state (pressure and density) at a full level k, the thermodynamic state can be advanced from the half level $k - \frac{1}{2}$ to $k + \frac{1}{2}$. Using the newly calculated state at $k + \frac{1}{2}$, pressure and density at $k+1$ can be calculated.

The base state density ρ_0 that is used in the dynamical core (Sect. 2) is calculated using the initial virtual potential temperature profile, and is not updated during the experiment. The density and hydrostatic pressure used in the moist thermodynamics can optionally be updated every time step, following the same procedure as explained in Boing (2014).

3.10 Rotation

The effects of a rotating reference frame on an f-plane can be included through the Coriolis force. ~~MicroHH can run on an f-plane, where the related tendencies of~~ The acceleration due to the Coriolis force $F_{i,\mathrm{cor}}$ is computed for the two horizontal velocity components ~~are calculated as~~ (index 1 and 2 in Eqs. 5 and 7) as

$$F_{1,\mathrm{cor}}\Big|_{\cancel{i,j,k,F_\mathrm{cor}}\,i,j,k} = f_0 v_{i,j,k}, \tag{61}$$

$$F_{2,\mathrm{cor}}\Big|_{\cancel{i,j,k,F_\mathrm{cor}}\,i,j,k} = -f_0 u_{i,j,k}, \tag{62}$$

with f_0 as Coriolis parameter specified by the user.

4 Physical parameterizations

4.1 Subfilter-scale model for large-eddy simulation

With the governing equations described in Sect. 2 it is possible to resolve the flow down to the scales where molecular viscosity acts. In many applications, however, such simulations are too costly. In that case, one may opt for large-eddy simulation (LES), where filtered equations are used to describe the largest scales of the flow, and the subfilter-scale motions are modeled. The LES implementation in MicroHH assumes very high Reynolds numbers in which the molecular viscosity is neglected. Filtering of the anelastic conservation of momentum equation (Eq. 5), with a tilde applied to denote filtered variables, leads to

$$\begin{aligned} \frac{\partial \widetilde{u}_i}{\partial t} &= -\frac{1}{\rho_0} \frac{\partial \rho_0 \widetilde{u}_i \widetilde{u}_j}{\partial x_j} - \frac{\partial \pi}{\partial x_i} - \frac{1}{\rho_0} \frac{\partial \rho_0 \tau_{ij}}{\partial x_j} \\ &+ \delta_{i3} g \frac{\widetilde{\theta}'_v}{\theta_{v0}} + F_i. \end{aligned} \tag{63}$$

In this equation, a tensor τ_{ij} is defined as

$$\tau_{ij} \quad \equiv \quad \widetilde{u_i u_j} - \widetilde{u}_i \widetilde{u}_j - \frac{1}{3}\left(\widetilde{u_i u_i} - \widetilde{u}_i \widetilde{u}_i\right). \tag{64}$$

This is the anisotropic subfilter-scale kinematic momentum flux tensor. The isotropic part of the full momentum flux tensor has been added to the pressure, providing the modified pressure

$$\pi \quad \equiv \quad \frac{\widetilde{p}'}{\rho_0} + \frac{1}{3}\left(\widetilde{u_i u_i} - \widetilde{u}_i \widetilde{u}_i\right). \tag{65}$$

As τ_{ij} contains the filtered product of unfiltered velocity components, this quantity needs to be parameterized. MicroHH uses the Smagorinsky-Lilly (Lilly, 1968) model, in which τ_{ij} is modeled as

$$\tau_{ij} \quad = \quad -K_m \left(\frac{\partial \widetilde{u}_i}{\partial x_j} + \frac{\partial \widetilde{u}_j}{\partial x_i} \right), \tag{66}$$

with K_m interpreted as the subfilter eddy-diffusivity. This quantity is modeled as

$$K_m \quad = \quad \lambda^2 2S_{ij}Sij^{\frac{1}{2}} \left(1 - \frac{\dfrac{g}{\theta_{v0}} \dfrac{\partial \widetilde{\theta}_v}{\partial z}}{Pr_t S^2} \right)^{\frac{1}{2}}, \tag{67}$$

and is proportional to the magnitude ~~of S of~~ $S \equiv (2S_{ij}S_{ij})^{\frac{1}{2}}$ of the strain tensor S_{ij}, which is defined as

$$S_{ij} \quad \equiv \quad \frac{1}{2}\left(\frac{\partial \widetilde{u}_i}{\partial x_j} + \frac{\partial \widetilde{u}_j}{\partial x_i} \right). \tag{68}$$

The subfilter eddy diffusivity thus takes into account the local stratification ~~$N^2 \equiv (g/\theta_{v0})/(\partial\widetilde{\theta}_v/\partial z)$~~ and the turbulent Prandtl number Pr_t. The latter is set to $\frac{1}{3}$ by default, but can be overridden in the settings. The length scale λ is the mixing length defined following Mason and Thomson (1992), as

$$\frac{1}{\lambda^n} = \frac{1}{\left[\kappa\left(z+z_0\right)\right]^n} + \frac{1}{\left(c_s \Delta\right)^n}, \tag{69}$$

which is an arbitrary matching function (n is a free parameter, set to 2 in MicroHH) ~~matching function~~ between the mixing length following wall scaling to the subfilter length scale (filter size) $\Delta \equiv (\Delta x \Delta y \Delta z)^{\frac{1}{3}}$, related to the grid spacing. The grid scale is used as an implicit filter, thus no explicit filtering is applied. In case of a high-Reynolds number atmospheric LES with an unresolved near-wall flow, the vertical gradients of the horizontal velocity components $\partial \widetilde{u}_{i,j}/\partial z$ in the strain tensor are replaced with the theoretical gradients predicted from Monin-Obukhov similarity theory. Evaluation of these gradients is explained in detail in Section 4.2.

The same approach is followed for all scalars, including the thermodynamic variables discussed in Sect. 2.5:

$$\frac{\partial \widetilde{\phi}}{\partial t} \quad = \quad -\frac{1}{\rho_0}\frac{\partial \rho_0 \widetilde{u}_j \widetilde{\phi}}{\partial x_j} - \frac{1}{\rho_0}\frac{\partial \rho_0 R_{\phi,j}}{\partial x_j} + \widetilde{S}_\phi. \tag{70}$$

The term $R_{\phi,j}$ refers to the subfilter flux of $\widetilde{\phi}$ and is defined as

$$R_{\phi,j} = \widetilde{u_j \phi} - \widetilde{u}_j \widetilde{\phi}. \tag{71}$$

The subfilter-scale flux is parameterized in terms of the gradient

$$R_{\phi,j} = -\frac{K_m}{Pr_t}\frac{\partial\widetilde{\phi}}{\partial x_j}. \tag{72}$$

4.2 Surface model

The LES implementation of MicroHH uses a surface model ~~to compute~~ that is constrained to rough surfaces and high Reynold numbers, which is a typical configuration for atmospheric flows. This model computes the surface fluxes of the horizontal momentum components and the scalars (including thermodynamic variables) ~~in flows over rough surfaces at high Reynolds numbers. This is a typical configuration for atmospheric flows. The surface model is entirely built on~~ using Monin-Obukhov Similarity Theory (MOST) (~~see Wyngaard (2010))that~~ Wyngaard (2010, his Sect. 10.2)). MOST relates surface fluxes of variables to their near-surface gradients using empirical functions that depend on the height of the first model level z_1 divided by the Obukhov length L as an argument. Length L is defined as

$$L \equiv -\frac{u_*^3}{\kappa B_0}, \tag{73}$$

where u_* is the friction velocity, κ is the Von Karman constant and B_0 is the surface kinematic buoyancy flux. L represents the height at which the buoyancy production/destruction of turbulence kinetic energy equals the shear production. In MicroHH, we use a local implementation of MOST, i.e., each grid point has its own value of L. This choice can lead to a overestimation of near-surface wind due to violation of the MOST assumption of horizontal homogeneity (Bou-Zeid et al., 2005, their Fig. 18), but it allows for a more straightforward extension to heterogeneous land surfaces.

Following MOST, the friction velocity u_* and the momentum fluxes may be related to the near-surface wind gradient as

$$\frac{\kappa z_1}{u_*}\frac{\partial U}{\partial z} \cancel{\equiv}\approx -\frac{\kappa z_1 u_*}{\overline{u'w'}}\frac{\partial\widetilde{u}}{\partial z} \cancel{\equiv}\approx -\frac{\kappa z_1 u_*}{\overline{v'w'}}\frac{\partial\widetilde{v}}{\partial z} \cancel{\equiv}\approx \phi_m\left(\frac{z_1}{L}\right), \tag{74}$$

where U is defined as $\sqrt{\widetilde{u}^2+\widetilde{v}^2}$, and $\overline{u'w'}$ and $\overline{v'w'}$ as the surface momentum fluxes for the two wind components. These relationships can be integrated from the roughness length z_{0m} to z_1 resulting in

$$u_* = f_m\left(U_1 - U_0\right), \tag{75}$$

$$\overline{u'w'} = -u_* f_m\left(\widetilde{u}_1 - \widetilde{u}_0\right), \tag{76}$$

$$\overline{v'w'} = -u_* f_m\left(\widetilde{v}_1 - \widetilde{v}_0\right), \tag{77}$$

with f_m defined as:

$$f_m \equiv \frac{\kappa}{\ln\left(\frac{z_1}{z_{0m}}\right) - \Psi_m\left(\frac{z_1}{L}\right) + \Psi_m\left(\frac{z_{0m}}{L}\right)}, \tag{78}$$

with Ψ_m desribed in Eqs. 83 and 85.

The same procedure for scalars is followed, with

$$\frac{\kappa z_1 u_*}{\overline{\phi'w'}}\frac{\partial\widetilde{\phi}}{\partial z} = \phi_h\left(\frac{z_1}{L}\right), \tag{79}$$

and in integrated form

$$\overline{\phi' w'} = u_* f_h \left(\widetilde{\phi}_1 - \widetilde{\phi}_0 \right), \tag{80}$$

with

$$f_h \equiv \frac{\kappa}{\ln\left(\frac{z_1}{z_{0h}}\right) - \Psi_h\left(\frac{z_1}{L}\right) + \Psi_h\left(\frac{z_{0h}}{L}\right)}, \tag{81}$$

with Ψ_h desribed in Eqs. 83 and 85.

The functions ϕ_m, ϕ_h, Ψ_m, and Ψ_h are empirical and depend on the static stability of the atmosphere. Under unstable conditions we follow (Wilson, 2001; Wyngaard, 2010)

$$\phi_{m,h} = \left(1 + \gamma_{m,h} \left|\zeta\right|^{2/3}\right)^{-1/2}, \tag{82}$$

$$\Psi_{m,h} = 3 \ln\left(\frac{1 + \phi_{m,h}^{-1}}{2}\right), \tag{83}$$

where ζ is the ratio of a height and the Obukhov length L, $\gamma_m = 3.6$ and $\gamma_h = 7.9$. Under stable conditions we use (Högström, 1988; Wyngaard, 2010)

$$\phi_{m,h} = 1 + \lambda_{m,h}\zeta, \tag{84}$$

$$\Psi_{m,h} = -\lambda_{m,h}\zeta, \tag{85}$$

where $\lambda_m = 4.8$ and $\lambda_h = 7.8$.

With the equations above, the surface fluxes, surface values and near-surface gradients can be computed, but only if the Obukhov length L is known. The surface model calculates the Obukhov length by relating the dimensionless parameter z_1/L to a Richardson number. The employed formulation of the Richardson number depends on the chosen boundary condition in the model. Three possible options are available:

- fixed momentum fluxes and a fixed surface buoyancy flux. Both the friction velocity u_* and the surface buoyancy flux B_0 are specified. Under these conditions we define the Richardson number Ri_a equal to z_1/L; L can be computed directly from the expression

$$Ri_a \equiv \frac{z_1}{L} = -\frac{\kappa z_1 B_0}{u_*^3}. \tag{86}$$

- a fixed horizontal velocity U_0 at the surface and a fixed surface buoyancy flux B_0. The friction velocity u_* is unknown. Now, L needs to be retrieved from the implicit relationship

$$Ri_b \equiv \frac{z_1}{L} f_m^3 = -\frac{\kappa z_1 B_0}{\left(U_1 - U_0\right)^3}. \tag{87}$$

- a fixed surface velocity U_0 and a fixed surface buoyancy b_0. With this boundary condition, the surface values of the horizontal velocities and the buoyancy are given, and both u_* and the surface buoyancy flux B_0 are unknown. L is then retrieved from

$$Ri_c \equiv \frac{z_1}{L}\frac{f_m^2}{f_h} = \frac{\kappa z_1\left(\widetilde{b}_1 - \widetilde{b}_0\right)}{\left(U_1 - U_0\right)^2}. \tag{88}$$

In cases of the latter two options, a ~~lookup table is created with~~ solver is required to find the value of L ~~as a function of~~ that satisfies the equation, as f_m (Eq. 78) and f_h (Eq. 81) both depend on L as well. For performance reasons, we have created a lookup table-based approach that relates L to the Richardson number. The lookup table has 10^4 entries, of which 90 percent is spaced uniformly between $z_1/L = -5$ to 5. The remaining 10 percent are used to stretch the negative range up to $z_1/L = -10^4$ to allow for the correct free convection limit.

4.3 Large-scale forcings

4.3.1 Pressure force

MicroHH provides two options to introduce a large-scale pressure force into the model. The first is to enforce a constant massflux through the domain in the streamwise direction. In this approach the desired large-scale velocity U_f is set, and the corresponding pressure gradient is computed. We follow here the approach of van Reeuwijk (2007). In this approach, the u-component of the horizontal momentum equation (Eq. 5) is volume-averaged to acquire

$$\frac{\langle u\rangle^{n+1} - \langle u\rangle^{n}}{\Delta t} = \langle f_1\rangle - \left\langle \frac{\partial}{\partial x}\left(\frac{p}{\rho_0}\right)\right\rangle + F_{\cancel{p;ls}\,p;\mathrm{ls}}, \tag{89}$$

where brackets indicate a volume average, f_1 contains all the righthand side terms of the u-component of the conservation of momentum, except for the dynamic pressure, which is contained in the second term. The large-scale pressure force ~~$F_{p;ls}$~~ $F_{p;\mathrm{ls}}$, which is to be computed, is the last term. We can now set $\langle u\rangle^{n+1} = U_f$ to explicitly set the volume-averaged velocity in the next time step. Furthermore, the volume-averaged horizontal pressure gradient vanishes, because of the periodic boundary condition, which makes ~~$F_{p;ls}$~~ $F_{p;\mathrm{ls}}$ the only unknown. The acquired pressure force $F_{p;\mathrm{ls}}$ will be added ~~following~~

$$\left.\frac{\partial u}{\partial t}\right|_{i,j,k,F_{p;ls}} \equiv F_{p;ls}$$

as an external force in the equation of zonal velocity (F_1 in Eqs 5 and 7).

The second option is to enforce a large-scale pressure force through the geostrophic wind components u_g and v_g, in combination with rotation, with the ~~tendencies~~ accelerations of the two horizontal velocity components ~~calculated as~~ $F_{i,p;\mathrm{ls}}$ calculated as

$$\left.F_{1,p;\mathrm{ls}}\right|_{\cancel{i,j,k,F_{p;ls}}\,i,j,k} = -f_0 v_{g;k}, \tag{90}$$

$$\left.F_{2,p;\mathrm{ls}}\right|_{\cancel{i,j,k,F_{p;ls}}\,i,j,k} = f_0 u_{g;k}, \tag{91}$$

where $u_{g;k}$ and $v_{g;k}$ ~~as~~ are user-specified vertical profiles of geostrophic wind components.

4.3.2 Large-scale sources and sinks

Large-scale sources and sinks, representing for instance large-scale ~~adveciton~~ advection or radiative cooling, can be prescribed for each variable separately. The user has to provide vertical profiles of large-scale ~~tendencies $S_{\phi;ls}$~~ sources and sinks $S_{\phi;\mathrm{ls}}$ that are added to the total tendencies.

4.3.3 Large-scale vertical velocity

A second method of introducing large-scale thermodynamic effects is through the inclusion of a large-scale vertical velocity. In this case, each scalar gets an additional ~~tendency term~~ source term $S_{\phi,\mathrm{w,ls}}$ of the form

$$S_{\phi,\mathrm{w,ls}}\Big|_{i,j,k,ls\,i,j,k} = -w_{ls;k\,\mathrm{ls};k}\,\delta^{2x}\left(\langle\phi\rangle_k\right), \tag{92}$$

where $w_{\mathrm{ls};k}$ is a user-specified vertical profile of large-scale vertical velocity, $\langle\phi\rangle_k$ is the horizontally-averaged vertical profile at height z_k for scalar ϕ. The tendency term is not applied to the momentum variables.

4.4 Buffer layer

MicroHH has the option to damp gravity waves in the top of the simulation domain in a so-called buffer layer. The ~~tendency~~ source term $S_{\phi,\mathrm{buf}}$ associated with the damping at grid cell i,j,k is calculated for an arbitrary variable ϕ as

$$S_{\phi,\mathrm{buf}}\Big|_{i,j,k,\mathrm{buf}\,i,j,k} = \frac{\phi_{i,j,k} - \phi_{0;k}}{\tau_{d;k}} \tag{93}$$

where ϕ_0 is taken from a user-specified vertical reference profile, and time scale τ_d is a measure for the strength of the damping. It varies with height and is calculated at height z_k following

$$\tau_{d;k}^{-1} = \sigma\left(\frac{z_k - z_{b;bot}}{z_{b;top} - z_{b;bot}}\right)^{\beta}, \tag{94}$$

where σ is the damping frequency chosen by the user and β an exponent that describes the shape of the vertical profile of the damping frequency, which is always zero at the bottom ($z_{b;bot}$) and σ at the top ($z_{b;top}$) of the layer.

5 ~~Model output~~

4.1 ~~Statistics~~

~~A large set of output statistics can be computed during runtime at user-specified time intervals. The statistics module provides vertical profiles of means, second-, third- and fourth-order moments of all prognostic variables, as well as advective and diffusive fluxes. Furthermore, there are multiple diagnostic variables, such as the pressure, the pressure variance and its vertical flux. The thermodynamics generate their own statistics based on the chosen option. The moist thermodynamics provides a large set of cloud statistics.~~

~~There is a separate module for budget statistics, which provides the budgets of all components of the Reynolds stress tensor, and those of the variance and vertical flux of the thermodynamic variables.~~

~~One of the key features of the MicroHH statistics routine is that an arbitrary mask can be passed into the routine over which the statistics are calculated. This allows, for instance, for a very simple way of computing conditional statistics in updrafts or clouds, which is demonstrated later in Section 9.2.~~

4.1 ~~Two- and three-dimensional output~~

~~It is possible to save two-dimensional cross sections and three-dimensional fields of any of the prognostic and diagnostic variables of the model, as well as of derived variables. This output can be made at user-specified time intervals. Cross sections can be made in any chosen $xy-$, $xz-$, and $yz-$plane. Derived variables (any arbitrary function of existing model variables), can be easily added to the code by the user.~~

5 Technical details of the code

5.1 Code structure

MicroHH is written in C++ and uses elements of object-oriented programming and template metaprogramming. The code components are written in classes that define the interface. Inheritance is used to allow for specializations of classes. This way of organizing the code has two advantages: it minimizes switches and it allows code components and their extensions to reside in their own file, which increases code clarity and facilitates the merging of new code extensions. High performance of computational kernels is achieved by executing kernels in their own function, with explicit inclusions of the `restrict` keyword to notify the compiler that fields do not overlap in memory. Furthermore, compiler-specific pragmas are used to aid vectorization on Intel compilers.

5.2 GPU

MicroHH is enabled to run on fast and energy-efficient Graphical Processing Units (GPU). This promising technique has been pioneered in atmospheric flows by Schalkwijk et al. (2012) and has shown its potential in weather forecasting (Schalkwijk et al., 2015). The implementation is based on NVIDIA's CUDA. The CPU and GPU version reside in the same code base, where the GPU implementation is activated with the help of precompiler statements. The philosophy is that the entire model is initialized on the CPU and that the GPU implementation is only activated right before starting the main time loop. At that ~~stage~~ moment, the required fields are copied in double precision accuracy to the GPU, and the entire time integration is done there. Synchronization only happens when statistics have to be computed or when restart files or cross sections of flow fields are saved to disk, to ensure high performance. The design choice to do the entire initialization on the CPU minimizes the amount of GPU code, and therefore allows for maintaining a single code base for the CPU and GPU code.

5.3 Parallelization

The code uses the Message Passing Interface (MPI) in order to run on a large number of cores. The three-dimensional simulation domain is split into vertically-oriented columns standing on a two-dimensional grid.

The code assigns one MPI task to each grid column using the MPI_Cart_create function, and uses this grid to detect the IDs of neighboring processes. In order to avoid complex packing routines, we make use of MPI datatypes wherever possible. The MPI calls are hidden in an interface to avoid the need to manually write MPI calls.

The input/output (IO) is entirely based on MPI-IO, the parallel IO framework that comes with MPI, to ensure that three-dimensional fields and two-dimensional cross sections are stored as single files. We have chosen MPI-IO in order to limit the number of files resulting from simulations on a large number of processes and to allow for restarts on a different number of processes. In order to keep complexity of the IO as low as possible, we make use of the MPI_Sub_array function in combination with MPI_File_write_all in order to write the fields.

5.4 External dependencies

MicroHH depends on several external software tools or libraries. It uses the CMake (https://cmake.org) build system for the generation of Makefiles. CMake allows for parallel builds, which minimizes the compilation time, and it is easy to add configurations for different machines. Furthermore, the FFTW3 library (Frigo and Johnson, 2005) is used for the computation of fast-Fourier transforms. The statistical routines make use of netCDF software developed by UCAR/Unidata [1](http://doi.org/10.5065/D6H70CW6). In order to run the provided test cases and their output scripts, a Python (https://www.python.org) installation including the NumPy ~~and Matplotlib~~ (van der Walt et al. (2011), http://www.numpy.org) and Matplotlib (Hunter (2007), https://matplotlib.org) modules is required. Automatic documentation generation can be done using Doxygen (http://doxygen.org), but this is optional.

6 Running simulations

In order to run a simulation with MicroHH, a sequence of steps needs to be taken. Each simulation has an `.ini` file that contains the settings of the simulation, a `.prof` file that contains the (initial) vertical profiles of all required variables, and, if time-varying boundary conditions are desired, a file with the prescribed time evolution for all time-varying boundary conditions. MicroHH provides a document (`doc/input.pdf`) that contains an overview of all possible options that can be specified in the `.ini` file.

To prepare a simulation with the name `test_simulation`, MicroHH needs to be run in the following way

```
./microhh init test_simulation
```

[1]

where it is assumed that `test_simulation.ini` and `test_simulation.prof` are available in the directory where the command is triggered. This procedure will create the initial fields of all prognostic variables and save the required fields for those model components that need to save their state to guarantee bitwise identical restarts.

After the previously described initialization phase, the execution of

```
./microhh run test_simulation
```

will start the actual simulation. Depending on the chosen output intervals, the simulation will create restart files, statistics, cross sections, and field dumps. MicroHH can restart from any time at which the restart files are saved.

The last mode in which the code can run is the post-processing mode. By running

```
./microhh post test_simulation
```

the code will generate statistics from saved restart files at a specified time interval. This mode allows the user to create new statistics and calculate those from saved data, without having to rerun the simulation.

7 Model output

7.1 Statistics

A large set of output statistics can be computed during runtime at user-specified time intervals. The statistics module provides vertical profiles of means, second-, third- and fourth-order moments of all prognostic variables, as well as advective and diffusive fluxes. Furthermore, there are multiple diagnostic variables, such as the pressure, the pressure variance and its vertical flux. The thermodynamics generate their own statistics based on the chosen option. The moist thermodynamics provides a large set of cloud statistics. There is a separate module for budget statistics, which provides the budgets of all components of the Reynolds stress tensor, and those of the variance and vertical flux of the thermodynamic variables.

One of the key features of the MicroHH statistics routine is that an arbitrary mask can be passed into the routine over which the statistics are calculated. This allows, for instance, for a very simple way of computing conditional statistics in updrafts or clouds, which is demonstrated later in Section 9.2.

7.2 Two- and three-dimensional output

It is possible to save two-dimensional cross sections and three-dimensional fields of any of the prognostic and diagnostic variables of the model, as well as of derived variables. This output can be made at user-specified time intervals. Cross sections can be made in any chosen $xy-$, $xz-$, and $yz-$plane. Derived variables (any arbitrary function of existing model variables), can be easily added to the code by the user.

8 Validation of the dynamical core

In this section, we present a series of cases intended to validate MicroHH under a wide range of settings. Each of these test cases is available in the `cases/` directory in the MicroHH repository, where all detailed settings can be found (see Sect. 14). Below, we present only the most relevant information per case.

8.1 Taylor-Green vortex

The two-dimensional Taylor-Green vortex (`cases/taylorgreen`) presents an ideal test case for a dynamical core as it has an analytical solution, even though it is nonlinear. This flow is composed of two rotating vortices whose evolution in a domain $[0,1;0,0.5]$ is described with

$$u(x,z,t) = \sin(2\pi x)\cos(\pi z)f(t), \tag{95}$$

$$w(x,z,t) = \cos(2\pi x)\sin(\pi z)f(t), \tag{96}$$

$$p(x,z,t) = \frac{1}{4}\left(\sin(4\pi x) + \sin(4\pi \cancel{y}z)\right)f(t)^2, \tag{97}$$

where $f(t) = 8\pi^2\nu t$.

We use the analytical form at $t = 0$ as the initial condition and run this case for one vortex rotation ($t = 1$), with $\nu = (800\pi^2)^{-1}$. We compare the result against the analytical solution for a set of grid spacings and with the second-order and fourth-order dynamical core; for the latter we compare the most accurate advection scheme and the fully energy-conserving one.

Figure 2 shows the error convergence of the simulations. The error for a variable ϕ is computed as $\sum \Delta x \Delta z \, |\phi_{i,k} - \phi_{\mathrm{ref},i,k},|$ over the two-dimensional domain, where Δx and Δz are the uniform grid spacings used in this case and ϕ_{ref} is the analytical solution. All variables converge according to the order of the numerical scheme. The fourth-order dynamical core loses accuracy at fine grid spacings. This is due to the boundary condition for the vertical velocity that sacrifies an order of accuracy to ensure global mass conservation (Morinishi et al., 1998).

8.2 ~~Energy~~ Kinetic energy conservation and time accuracy

The second test of the dynamical core consists of combined evaluation of ~~energy~~ kinetic energy ($\mathrm{KE} \equiv \frac{1}{2}\left(u^2 + v^2 + w^2\right)$) conservation and time accuracy (`cases/conservation`). In this experiment, we ~~switch the diffusion off and~~ run the model with only the advection and pressure solver enabled and advect random noise through the domain for 1000 seconds. These tests have been conducted with the third- and fourth-order Runge-Kutta schemes. We have chosen ~~for~~ the fourth-order spatial discretization in order to demonstrate its energy conservation.

The loss of kinetic energy as a function of time is shown in Fig. 3a. The fourth-order scheme results in a smaller energy loss for the same time step and ~~an~~ a faster convergence. The error-convergence plot (Fig. 3b) shows that the error convergence is in accordance with the order of the respective scheme. Furthermore, it illustrates the fact that, if high accuracy in time is desired, the five-stage fourth-order scheme is less expensive than the three-stage third-order scheme. For instance, at a Δt of

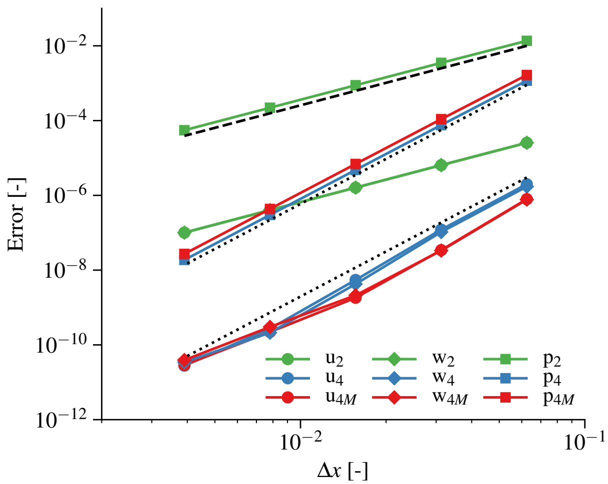

Figure 2. Convergence of the spatial discretization error in the two-dimensional Taylor-Green vortex. Subscript 2 indicates the second-order scheme, 4 the most accurate fourth-order scheme, and $4M$ the fully energy-conserving fourth-order scheme. The dashed black line is the reference for second-order convergence, the dotted black ~~line for~~ lines indicate fourth-order convergence.

10, the error of the fourth-order scheme is approximately equal to the error of the third-order scheme at a Δt of 2.5. To reach this accuracy, the fourth-order scheme needs only 5 steps per 10 time units, whereas the third-order scheme needs 12 steps.

8.3 Laminar ~~katabatic~~ anabatic flow

To test the buoyancy routine and the option to put the domain on a slope, a ~~laminar katabatic flow~~ Prandl-type anabatic slope flow (Prandtl, 1942) has been simulated ~~, based on the test case of Prandtl (1942) (~~(`cases/`~~`prandtl`~~`prandtlslope`). In this test case, the surface is inclined at an angle of 30° and a linearly stratified atmosphere ($N = 1$ s^{-1}) is ~~cooled~~ heated from below with a fixed surface buoyancy flux of ~~-0.005~~0.005 m^2 s^{-3}.

The fluid, which was initially at rest, goes through a series of decaying oscillations after the ~~negative~~ buoyancy flux is applied at the surface. Eventually, it reaches the steady state corresponding to the Prandtl model solution. Numerical integration was performed sufficiently long for the oscillation amplitude to become a small fraction of the amplitude of the first oscillation. Comparison of horizontal wind u and buoyancy b of analytical and numerical solutions ~~, which very closely agree with each other, is presented~~ is shown in Fig. 4. For both variables the solutions closely agree with each other.

8.4 Turbulent channel flow

For fully turbulent flows, the numerical solutions cannot be compared against any analytical testcases. Therefore, we validate our results against a channel flow at a Reynolds-τ number of 590 (Moser et al., 1999) for means, variances, spectra, and

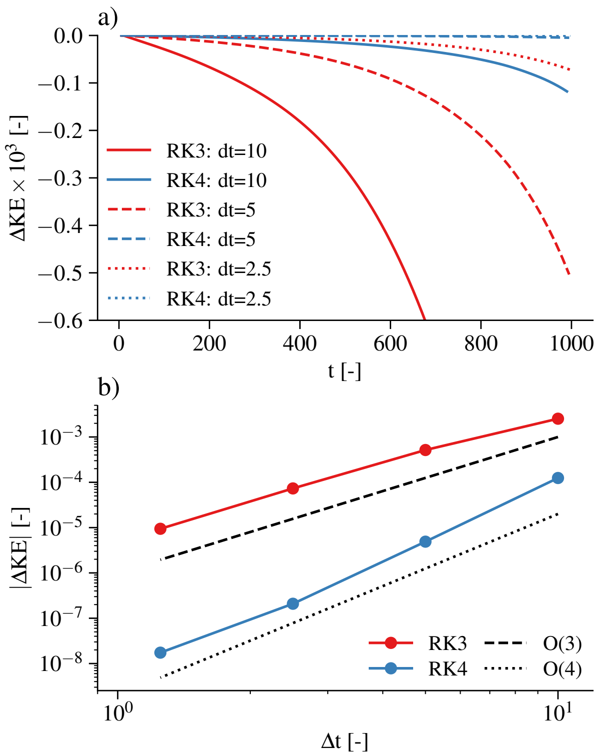

Figure 3. Time evolution of the kinetic energy ~~loss~~ change ΔKE during 1000 time units of random noise advection for the RK3 and RK4 time integration schemes with three different time steps (a). ~~Error~~ Kinetic energy change convergence of the temporal discretization for the RK3 and RK4 scheme (b).

second-order budget statistics (`cases/moser590`). The case is run at a resolution of $768 \times 384 \times 256$ grid points. The original numerical simulation data of Moser et al. (1999) has been produced on a $384 \times 384 \times 256$ grid, with spectral schemes in the horizontal dimensions and Chebyshev polynomials in the vertical.

Figure 5a shows the normalized horizontally-averaged streamwise velocity. The normalized rms of all three velocity components are presented in Fig. 5b. All plotted variables show a perfect match with the data and are indistinguishable from Moser's data. In order to further assess the accuracy of the data, we show the second-order budgets of the variances in Fig. 6. Also here, the match with the reference data is excellent, which indicates that the whole range of spatial scales in the flow is represented well and that the fourth-order scheme is well able to pick up the ~~small scale~~ small-scale details of the flow.

The findings in the previous paragraph are further corroborated by the spectra shown in Fig. 7. Over the whole range of scales, the match between our simulation and that of ~~Moser~~ Moser et al. (1999) is excellent. Note that the spectra from Moser's simulation display an increase in pressure variance at the highest wave numbers. This increase is the result of aliasing errors at high wave numbers that are typical for the spectral schemes that Moser et al. (1999) used.

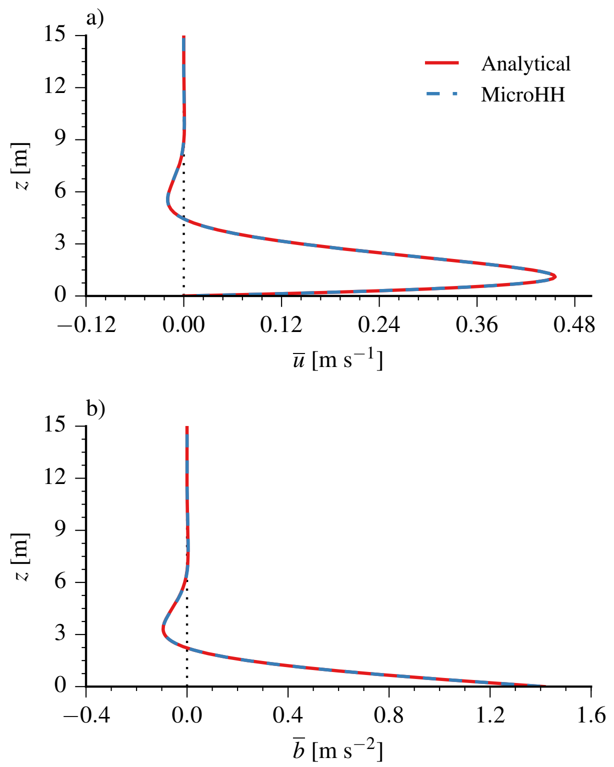

Figure 4. Normalized numerical Prandtl-model solutions for velocity u (left) and buoyancy b (right) compared to their analytical counterparts.

8.5 Turbulent katabatic flow

The final evaluation of the dynamical core without parametrizations enabled is based on the direct numerical simulation of a turbulent katabatic flow. Here, a buoyancy driven slope flow is simulated following the setup of Fedorovich and Shapiro (2009) (`cases/drycblslope`).

A flow over a slope inclined at an angle α of 60° is simulated with a fixed uniform surface buoyancy flux of -0.5 m^2 s^{-3}. The simulation is performed in a domain of 0.64 m $\times$ 0.64 m $\times$ 1.6 m using a uniform grid of 256 $\times$ 256 $\times$ 640 points. The initial state is a fluid at rest with a linear buoyancy stratification N of 1 s^{-1}. No-slip boundary conditions are applied at the bottom, free-slip at the top.

Turbulent motion starts quickly after the buoyancy flux is applied at the surface. It is characterized by random, large-amplitude fluctuations of velocity and buoyancy fields in the near-slope core region, and shows a quasi-periodic oscillatory behavior at larger distances from the slope. Mean profiles of along-slope velocity component and buoyancy, as well as profiles of second-order turbulence statistics, such as kinematic turbulent fluxes of momentum and buoyancy, and velocity-component

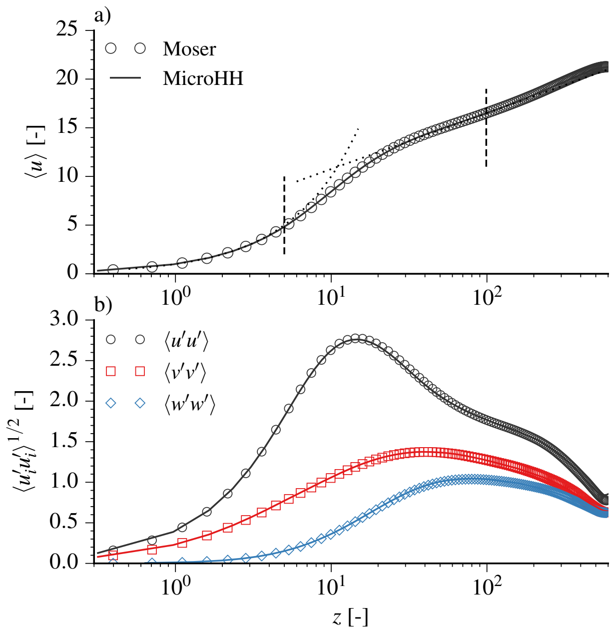

Figure 5. Velocity means and variances for Moser et al. (1999) channel flow case at Reynolds-τ 590. The dashed vertical lines marks the spectra locations. Height z is normalized with u_τ/ν, velocities with u_τ^{-1}.

and buoyancy fluctuation variances, were evaluated by averaging the simulated flow fields spatially over the along-slope planes and temporally over five oscillation periods beyond the transition stage.

For comparison, the same katabatic flow case was reproduced using the numerical code (hereafter referred to as FS09) that was employed to simulate turbulent slope flows in Shapiro and Fedorovich (2008) and Fedorovich and Shapiro (2009). In that code, the time advancement was performed with an Asselin-filtered second-order leapfrog scheme (Durran, 2013). The spatial discretizations are identical to the second-order accurate ones of MicroHH.

Numerical results obtained with both numerical codes testify that stable environmental stratification in combination with negative surface buoyancy forcing in the katabatic flow leads to an effective suppression of vertical turbulent exchange in the flow region adjacent to the slope. This suppression results in a shallow near-surface sublayer with strong buoyancy gradients (Fig. 8a) capped by a narrow jet with peak velocity located very close to the ground (Fig. 8b). Further comparison has been performed on the slope-normal fluxes of momenum and buoyancy (not shown), where a nearly perfect match has been reproduced as well.

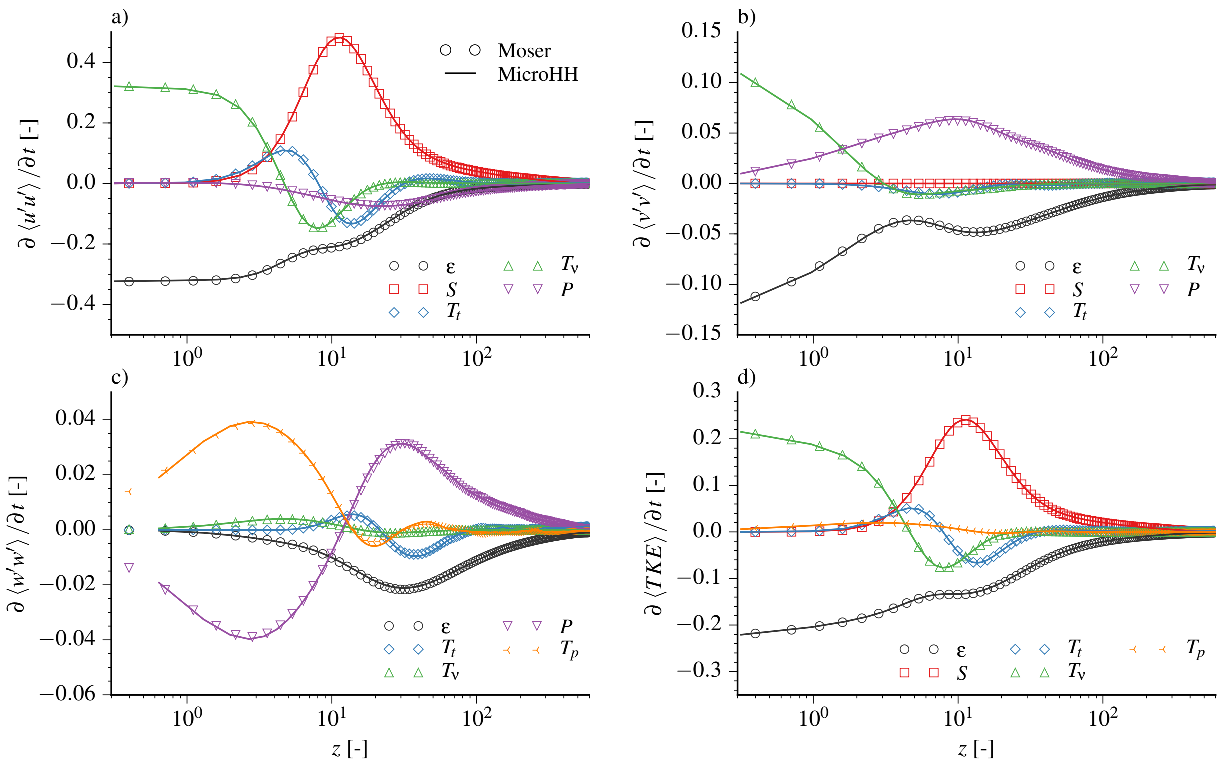

Figure 6. Budgets of variances and TKE compared against Moser et al. (1999)'s reference data at Reynolds-τ of 590. Height z is normalized with u_τ/ν, the variances and TKE budget with ν/u_τ^4.

9 Validation of atmospheric large-eddy simulations

9.1 Dry convective boundary layer with strong inversion

As a first test case of MicroHH in LES mode, we present that of Sullivan and Patton (2011) (`cases/sullivan2011`). This is a dry clear convective boundary layer that grows into a linearly stratified atmosphere with a very strong capping inversion (see Fig. 9a). The system is heated from the bottom by applying a constant kinematic ~~potential temperature~~ heat flux of 0.24 K m s^{-1}. The domain size is 5120 m $\times$ 5120 m $\times$ 2048 m. Gravity wave damping has been applied in the top 25 percent of the domain. Simulations have been run for three hours with three spatial resolutions. The time-averaged profiles have been calculated over the last hour.

The results show the formation of a well-mixed layer with an overlying capping inversion (see Fig. 9a) and the associated linear heat-flux profile with negative flux values in the entrainment zone (see Fig. 9b). The change of both quantities with resolution highlights the intrinsic challenge in resolving a boundary layer with an inversion layer that is stronger than the numerical schemes can accurately resolve. At coarse resolution, the strong inversion leads to an unphysical overshoot of the

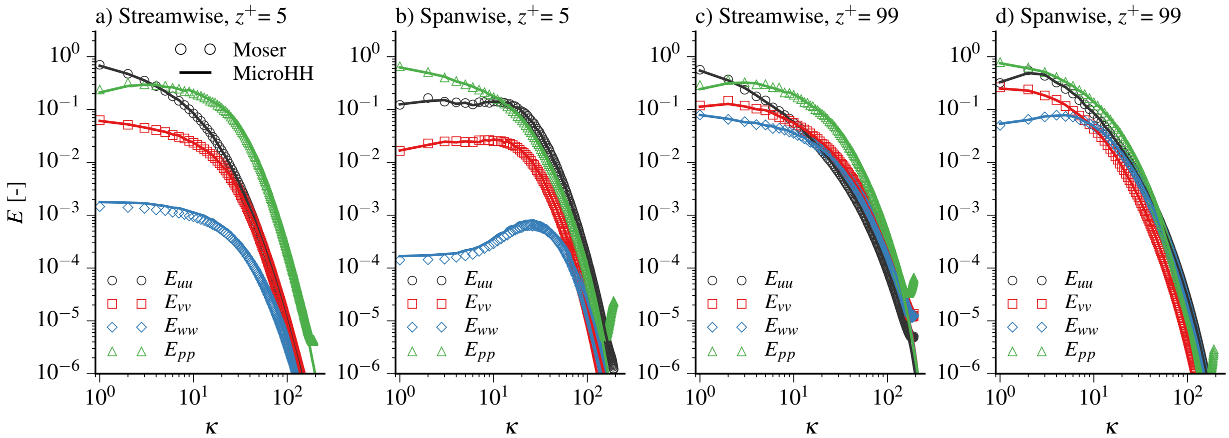

Figure 7. Spectra of all velocity components and pressure compared against Moser et al. (1999)'s reference data at Reynolds-τ of 590. The velocity spectra are normalized with u_τ^{-2}, the pressure spectra with u_τ^{-4}

potential temperature flux above the boundary layer top (see Fig. 9b). This overshoot leads to a numerical mixed layer on top of the entrainment zone that ~~disappears quickly~~ vanishes with increasing resolution.

9.2 BOMEX

The BOMEX shallow cumulus case (Siebesma et al., 2003) (`cases/bomex`), S03 hereafter, provides the opportunity to evaluate the moist thermodynamics (see Sect. 3.9) and large-scale forcings. We have repeated the case as described in S03 at the original resolution of the study (100 m $\times$ 100 m $\times$ 40 m) and at a higher resolution (10 m $\times$ 10 m $\times$ 9.375 m).

This case produces non-precipitating shallow cumulus. It has a large-scale cooling applied that represents radiation, as well as a large-scale drying to allow the atmosphere to relax to a steady state. In addition, a large-scale vertical velocity is applied over a certain height range to reproduce the appropriate synoptic conditions.

The simulation is run for 6 hours. Statistics are recorded during the final hour, including conditional statistics for the cloud-covered fields ($q_l > 0$) and for the cloud cores ($q_l > 0$ and $\theta_v > 0$). The vertical profile of area coverage and profiles of horizontally-average liquid water potential temperature θ_l, total water q_t, and vertical velocity w are shown in Fig. 10. All mean and conditionally sampled statistics at the original resolution are predominantly within one standard deviation of the ensemble mean of data from all models that participated in S03. The horizontally-averaged vertical velocities in the cloud and cloud core decrease considerably with an increase in resolution.

9.3 GABLS1

To evaluate the LES mode for stable atmospheric conditions, the GABLS1 LES intercomparison case (Beare et al., 2006) (`cases/gabls1`) was reproduced. The boundary layer develops in this case from a shallow well-mixed layer into a weakly

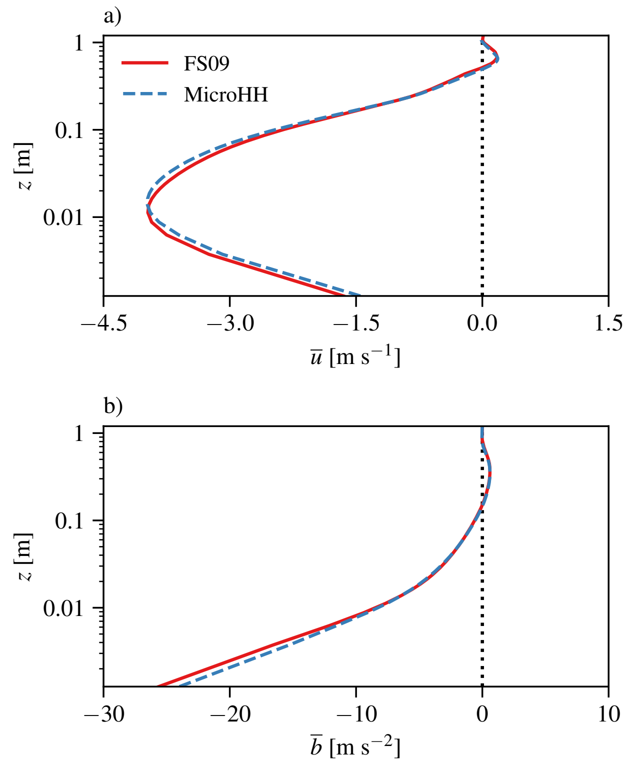

Figure 8. Profile of the mean along-slope velocity (a) and buoyancy (b) as predicted by MicroHH and FS09.

stable boundary layer, driven by a prescribed negative tendency of the surface temperature over a total integration time of 9 hours. The Boussinesq approximation is used, the advection scheme uses fourth-order accurate interpolations (Eq. 27), and the Smagorinsky subgrid turbulence scheme is set up with a Smagorinsky constant equal to 0.12, and a subgrid turbulent Prandtl number of unity. The experiments are performed at two different resolutions with grid cells of 2 m and 6.25 m, and compared to the models which participated in the study of Beare et al. (2006).

Figure 11 shows the domain and time-averaged (over a period from 28800 to 32400 s) vertical profiles of potential temperature ($\langle\theta\rangle$) and the velocity component ($\langle u\rangle$), and also time series of the boundary layer depth (z_{ABL}) and friction velocity (u_*). At the largest grid spacing of 6.25 m, it takes approximately 2 hours for the flow to become turbulent, as is evident from the delayed boundary layer growth and abrupt changes in u_*. Nonetheless, typical features like the low-level jet (Fig. 11b) are well reproduced, and all statistics are predominantly within the range of results from Beare et al. (2006). With the grid spacing reduced to 2 m, the flow becomes turbulent nearly instantaneously, but the resulting boundary layer depth and surface friction velocity are on the low side compared to the 5 models from Beare et al. (2006) which were run at this resolution.

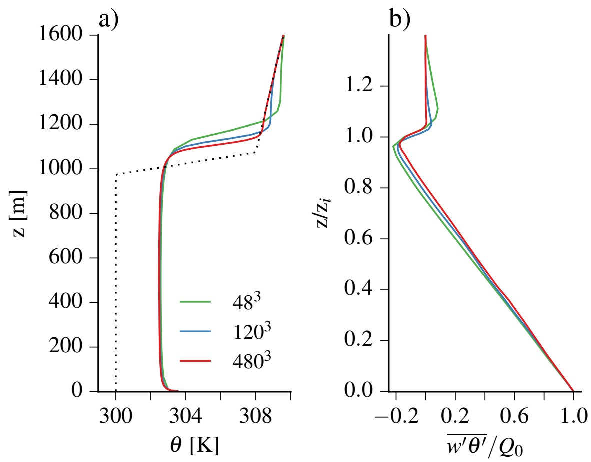

Figure 9. Vertical profiles of horizontally-averaged potential temperature (a) and normalized kinematic heat flux (b). The boundary layer depth z_i is the location of the maximum vertical gradient in the potential temperature profile shown in (a).

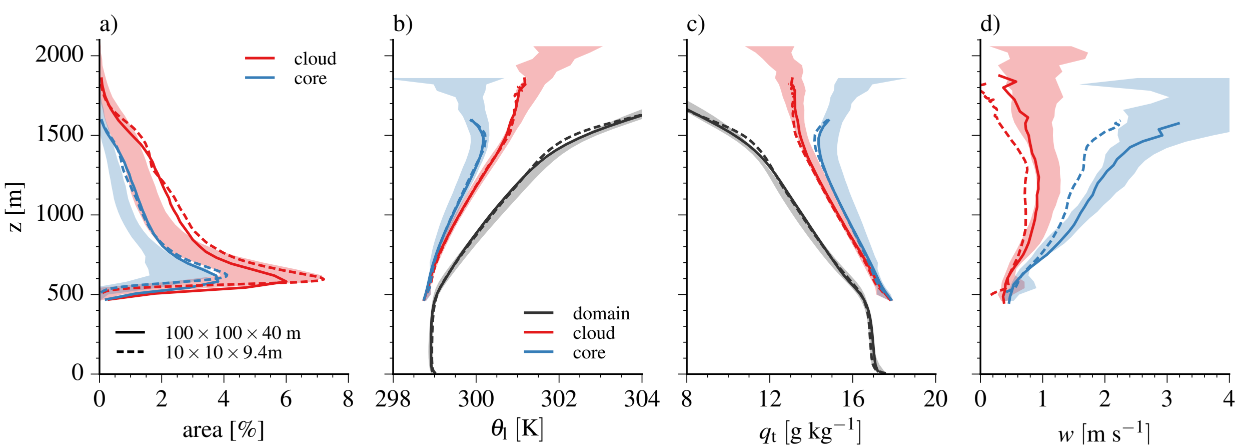

Figure 10. BOMEX LES intercomparison (S03). Shown are the domain mean, and conditionally sampled cloud ($q_l > 0$) and cloud core ($q_l > 0$ and $b - \langle b \rangle > 0$) vertical profiles of (a) area coverage of cloud and cloud core, (b) liquid water potential temperature, (c) total specific humidity and (d) vertical velocity. The results are averaged over t = 18000 s – 21600 s. The shaded area denotes the mean $\pm$ one standard deviation of the participating models from S03, the solid and dashed lines the results from MicroHH, using the original (solid) and a higher resolution (dashed) setup.

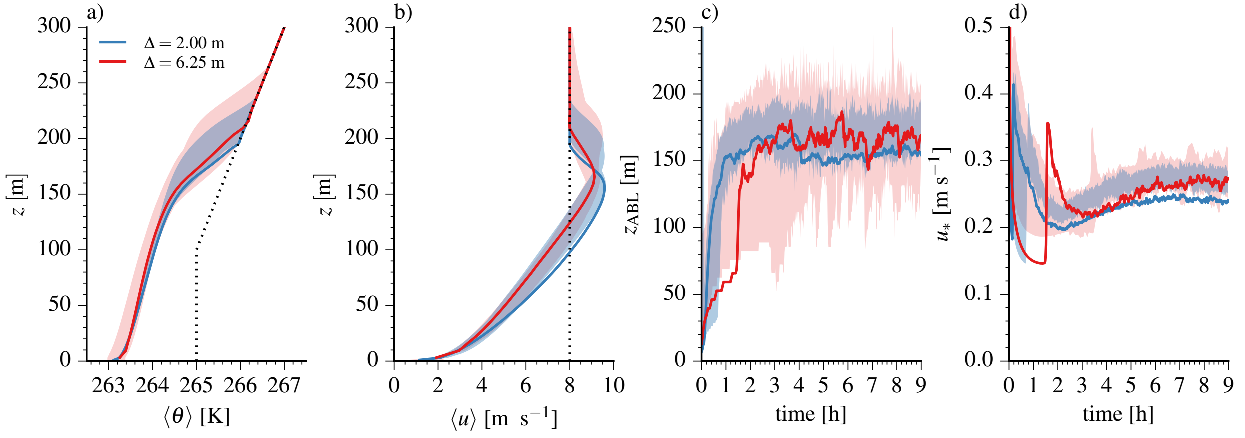

Figure 11. GABLS1 LES intercomparison (Beare et al., 2006). Shown are the vertical profiles of (a) potential temperature and (b) u-component of the velocity, and time series of the (c) boundary layer depth and (d) surface friction velocity. The shaded areas show the range in the results from the models that participated in the Beare et al. (2006) study. The dotted black lines show the initial conditions.

10 Performance

10.1 CPU

The parallel performance of MicroHH has been evaluated in strong- (`cases/strongscaling`) and weak-scaling (`cases/weakscaling`) experiments. The case used is direct numerical simulation of a buoyancy driven atmospheric boundary layer based on van Heerwaarden et al. (2014). For each simulation in the scaling experiments, a series of time steps is performed, and the mean cost per step is calculated over the series. The strong-scaling experiment has been performed on LRZ's SuperMUC[1] machine (Phase 1 Thin Node 8-core Sandy Bridge-EP Xeon E5-2680 8C, 2 processors per node, Infiniband FDR10 interconnect). In this experiment, simulations were performed on $1024 \times 1024 \times 1024$ and $2048 \times 2048 \times 1024$ grid points, with the number of processes increased throughout the scaling experiment. The weak-scaling experiment has been performed on Forschungszentrum Jülich's Juqueen[2] machine (IBM PowerPC A2, 1.6 GHz, 16 cores per node, 5D Torus network, 40 GBps). In this experiment, a fixed $64 \times 32 \times 1024$ grid is assigned to each processor and throughout the experiment the domain size is increased. The results of both experiments are shown in Figure 12.

The strong-scaling experiment shows that increasing the number of processors leads to faster simulations. The speedup is initially close to linear, but each consecutive increase in the number of cores makes the model less efficient. Based on these results, we conclude that for the chosen test case and for the used supercomputers, a work load of approximately 2×10^6 grid points per core is the best balance between speed and computational efficiency.

[1] https://www.lrz.de/services/compute/supermuc/

[2] http://www.fz-juelich.de/ias/jsc/EN/Expertise/Supercomputers/JUQUEEN/JUQUEEN_node.html

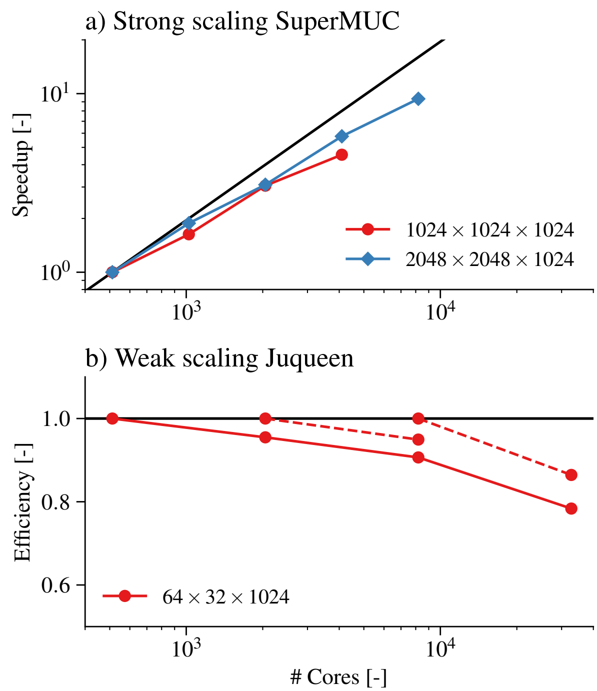

Figure 12. Speed-up from a strong-scaling experiment (a). Efficiency from a weak-scaling experiments (b). Black lines indicate perfect speed-up and efficiency. The dashed red lines show the efficiency change relative to the previous measurement.

The weak scaling shows almost 90 percent efficiency going from 512 to 8192 cores, beyond that the scaling falls off to 80 percent. This can be explained by physical properties of the machine; beyond 8192 cores a simulation no longer fits on one midplane (a physical unit consisting of 8192 cores), leading to slower communication.

10.2 Performance GPU (CUDA) implementation

The GPU implementation of MicroHH allows for fast simulations on a single device. The current state-of-the-art GPUs feature 12 GB of memory, thus simulations of maximally $512 \times 512 \times 512$ grid points of a flow with three velocity components, pressure, two scratch fields for temporary storage, and a single scalar fit in memory. Within this experiment, we compare thus GPU simulations that do not need communication against CPU simulations that require communication between cores and nodes. The reason for doing so, is that nearly all of the simulations of the presented results in Sections 8 and 9 fit within the memory of a single GPU.

To test the performance of such simulations, the performance of MicroHH on an NVIDIA Quadro K6000 (using CUDA 6.5) has been compared against the Max Planck Institute for Meteorology's cluster Thunder (2 Intel Xeon E5-2670 CPU's per node, 16 cores per node~~, Intel 15.01 with OpenMPI 1.8.4~~). Three benchmark cases have been chosen: the BOMEX moist convection

Table 1. Speedup of GPU simulation compared to respective CPU simulation performed on n cores.

case	n=1	n=16	n=32	n=64
B64	18.49	1.93	1.14	0.95
B128	28.01	2.98	1.51	0.92
B256	27.76	3.02	1.59	0.91
B512	29.88	3.03	1.56	0.86
M180	21.57	2.17	1.13	0.69
M600	22.55	2.25	1.06	0.60

case on grids of 64^3, 128^3, 256^3 and $512^2 \times 384$, and the channel flow cases of Moser et al. (1999) at a Reynolds-τ number of 180 and 590.

The results shown in Table 1 point to the great potential of GPU computing. For the considered cases, which all fit on a single GPU, it takes at least 32 cores to reach equal performance. Only at 64 cores, the CPU simulations are notably faster. Therefore, for simulations that fit into its memory, the GPU provides an excellent alternative for the CPU, especially ~~as~~ because the GPU is very energy efficient.

11 Published studies

To date, several studies have been published that make use of MicroHH or data generated with MicroHH. Van Heerwaarden et al. (2014) studied the scaling of flow over heterogeneously heated land surfaces using DNS and LES, Gentine et al. (2015) used LES to study the structure of the inversion of a convective boundary layer, van Heerwaarden and Mellado (2016) developed scaling laws for the convective boundary layer over a surface with a constant temperature from DNS data, McColl et al. (2017) improved surface-layer similarity under mildly convective conditions with the help of DNS data, and Umphrey et al. (2017) used DNS data produced with MicroHH as a reference for their simulations of slope flow.

12 Future plans

There are several ongoing projects to extend the model. Currently, a ~~parameterizations~~ parameterization for microphysics has been developed, and an interactive land surface model is under development. In addition, the immersed boundary method following Tseng and Ferziger (2003) is being implemented to allow for simulations of flow over obstacles and hills.

Furthermore, preliminary experiments have been performed to include a Domain-Specific Language (DSL) to enable the expression of complex finite difference operators in simple and compact syntax (https://github.com/Chiil/stencilbuilder/). This development has shown great potential, for two reasons. First, the DSL prevents implementation errors, as the explicit indexing

Table 2. Overview of used constants.

Symbol	Description	Value	Units
κ	Von Karman constant	0.4	-
g	Gravitational acceleration	9.81	$\mathrm{m\ s^{-2}}$
c_p	Specific heat of dry air at constant pressure	1005	$\mathrm{J\ kg^{-1}\ K^{-1}}$
p_{00}	Reference pressure	$1 \cdot 10^5$	Pa
R_d	Gas constant for dry air	287.04	$\mathrm{J\ K^{-1}\ kg^{-1}}$
R_v	Gas constant for water vapor	461.5	$\mathrm{J\ K^{-1}\ kg^{-1}}$
L_v	Latent heat of vaporization	$2.5 \cdot 10^6$	$\mathrm{J\ kg^{-1}}$

in computational kernels with spatial operators can be omitted. Second, the DSL allows for simple implementation of system-specific tuning, such as loop tiling or OpenMP.

13 Conclusions

This paper has presented a full description of MicroHH, a new computational fluid dynamics code for simulations of turbulent flows in the atmospheric boundary layer. The governing equations and their implementation has been presented, and a broad validation under a wide range of settings has been shown. MicroHH delivers the expected error convergence of the spatial and temporal schemes, and has proven to be mass, momentum, and energy conserving. The code delivers good performance in weak and strong scaling experiments. Its current limitations are the absense of horizontal boundary conditions other than periodic, and the limited set of available physical parameterizations. Both limitations will be addressed in future versions of the code.

14 Availability of code and resources

MicroHH has its own website at http://microhh.org. The code is hosted at GitHub and can be accessed either via the website, or directly from https://github.com/microhh/microhh. The GitHub website includes a wiki with several tutorials, including one to compile and run the code. The GitHub repository is coupled to Zenodo, which provides DOIs for released software. The release on which the reference paper is based is found at https://zenodo.org/badge/latestdoi/14754940. A selection of visualizations can be viewed at the MicroHH channel on Vimeo https://vimeo.com/channels/microhh/.

Appendix A: ~~Appendix~~Physical constants

Table 2 presents an overview of the chosen values for physical constants in the code.

Acknowledgements. Finishing the 1.0 version of MicroHH has only become possible thanks to extensive discussions and exchange of code with many of the developers of DALES (Huug Ouwersloot, Stephan de Roode, Arnold Moene and Jordi Vilà-Guerau de Arellano), PALM (Björn Maronga and Siegfried Raasch), UCLA-LES (Linda Schlemmer and Bjorn Stevens) and ICON (Leonidas Linardakis). We thank Alan Shapiro for the discussions on the bottom boundary condition for pressure for buoyancy driven flows and for his comments on the manuscript. We thank Elie Bou-Zeid and an anonymous reviewer for their constructive comments on the manuscript. The authors gratefully acknowledge the Gauss Centre for Supercomputing (GCS) for providing computing time for a GCS Large-Scale Project on the GCS share of the supercomputer JUQUEEN at Jülich Supercomputing Centre (JSC).

References

Bannon, P. R.: On the anelastic approximation for a compressible atmosphere, J. Atmos. Sci., 53, 3618–3628, 1996.

Beare, R. J., Macvean, M. K., Holtslag, A. A., Cuxart, J., Esau, I., Golaz, J.-C., Jimenez, M. A., Khairoutdinov, M., Kosovic, B., Lewellen, D., et al.: An intercomparison of large-eddy simulations of the stable boundary layer, Bound.-Layer Meteor., 118, 247–272, 2006.

Betts, A. K.: Non-precipitating cumulus convection and its parameterization, Quart. J. Roy. Meteor. Soc., 99, 178–196, 1973.

Boing, S. J.: The interaction of deep convective clouds and their environment, TU Delft, Delft University of Technology, 2014.

Bou-Zeid, E., Meneveau, C., and Parlange, M.: A scale-dependent Lagrangian dynamic model for large eddy simulation of complex turbulent flows, Phys. of Fluids, 17, 025 105, doi:10.1063/1.1839152, 2005.

Buck, A. L.: New equations for computing vapor pressure and enhancement factor, J. Applied Met., 20, 1527–1532, 1981.

Carpenter, M. H. and Kennedy, C. A.: Fourth-order 2N-storage Runge-Kutta schemes, Tech. Rep. TM-109112, NASA Langley Research Center, 1994.

Castillo, J. E., Hyman, J. M., Shashkov, M. J., and Steinberg, S.: The sensitivity and accuracy of fourth order finite-difference schemes on nonuniform grids in one dimension, Computers Math. Applic., 30, 41–55, 1995.

Chorin, A. J.: Numerical solution of the Navier-Stokes equations, Math. Comput., 22, 745–762, 1968.

Durran, D. R.: Numerical methods for wave equations in geophysical fluid dynamics, vol. 32, Springer Science & Business Media, 2013.

Fedorovich, E. and Shapiro, A.: Structure of numerically simulated katabatic and anabatic flows along steep slopes, Acta Geophysica, 57, 981–1010, 2009.

Frigo, M. and Johnson, S. G.: The design and implementation of FFTW3, Proc. IEEE, 93, 216–231, 2005.

Gentine, P., Bellon, G., and van Heerwaarden, C. C.: A Closer Look at Boundary Layer Inversion in Large-Eddy Simulations and Bulk Models: Buoyancy-Driven Case, J. Atmos. Sci., 72, 728–749, doi:10.1175/JAS-D-13-0377.1, 2015.

Heus, T., van Heerwaarden, C. C., Jonker, H. J. J., Siebesma, A. P., Axelsen, S., van den Dries, K., Geoffroy, O., Moene, A. F., Pino, D., de Roode, S. R., and Vilà Guerau de Arellano, J.: Formulation of the Dutch Atmospheric Large-Eddy Simulation (DALES) and an overview of its applications, Geosci. Model. Dev., 3, 415–444, doi:10.5194/gmd-3-415-2010, 2010.

Högström, U.: Non-dimensional wind and temperature profiles in the atmospheric surface layer: a re-evaluation, Bound.-Layer Meteor., 42, 55–78, 1988.

Hunter, J. D.: Matplotlib: A 2D Graphics Environment, Computing in Science & Engineering, 9, 90–95, doi:10.1109/MCSE.2007.55, 2007.

Lilly, D.: A comparison of incompressible, anelastic and Boussinesq dynamics, Atmospheric research, 40, 143–151, 1996.

Lilly, D. K.: Models of cloud-topped mixed layers under a strong inversion, Quart. J. Roy. Meteor. Soc., 94, 292–309, 1968.

Maronga, B., Gryschka, M., Heinze, R., Hoffmann, F., Kanani-Sühring, F., Keck, M., Ketelsen, K., Letzel, M. O., Sühring, M., and Raasch, S.: The Parallelized Large-Eddy Simulation Model (PALM) version 4.0 for atmospheric and oceanic flows: model formulation, recent developments, and future perspectives, Geoscientific Model Development Discussions, pp. 1539–1637, 2015.

Mason, P. J. and Thomson, D. J.: Stochastic backscatter in large-eddy simulation of boundary layers, J. Fluid Mech., 242, 51–78, 1992.

McColl, K. A., van Heerwaarden, C. C., Katul, G. G., Gentine, P., and Entekhabi, D.: Role of large eddies in the breakdown of the Reynolds analogy in an idealized mildly unstable atmospheric surface layer, Quart. J. Roy. Meteor. Soc., pp. n/a–n/a, doi:10.1002/qj.3077, qJ-16-0283.R2, 2017.

Morinishi, Y., Lund, T. S., Vasilyev, O. V., and Moin, P.: Fully conservative higher order finite difference schemes for incompressible flow, J. Comput. Phys., 143, 90–124, 1998.

Moser, R. D., Kim, J., and Mansour, N. N.: Direct numerical simulation of turbulent channel flow up to Re= 590, Phys. Fluids, 11, 943–945, 1999.

Prandtl, L.: Führer durch die Strömungslehre, Vieweg and Sohn, 1942.

Schalkwijk, J., Griffith, E. J., Post, F. H., and Jonker, H. J.: High-performance simulations of turbulent clouds on a desktop PC: Exploiting the GPU, Bull. Amer. Meteor. Soc., 93, 307–314, 2012.

Schalkwijk, J., Jonker, H. J., Siebesma, A. P., and Van Meijgaard, E.: Weather forecasting using GPU-based large-eddy simulations, Bull. Amer. Meteor. Soc., 96, 715–723, 2015.

Shapiro, A. and Fedorovich, E.: Coriolis effects in homogeneous and inhomogeneous katabatic flows, Quart. J. Roy. Meteor. Soc., 134, 353–370, 2008.

Siebesma, A. P., Bretherton, C. S., Brown, A., Chlond, A., Cuxart, J., Duynkerke, P. G., Jiang, H., Khairoutdinov, M., Lewellen, D., Moeng, C.-H., et al.: A large eddy simulation intercomparison study of shallow cumulus convection, J. Atmos. Sci., 60, 1201–1219, 2003.

Stevens, B., Moeng, C.-H., Ackerman, A. S., Bretherton, C. S., Chlond, A., de Roode, S., Edwards, J., Golaz, J.-C., Jiang, H., Khairoutdinov, M., Kirkpatrick, M. P., Lewellen, D. C., Lock, A., Müller, F., Stevens, D. E., Whelan, E., and Zhu, P.: Evaluation of Large-Eddy Simulations via Observations of Nocturnal Marine Stratocumulus, Mon. Wea. Rev., 133, 1443–1462, 2005.

Sullivan, P. P. and Patton, E. G.: The effect of mesh resolution on convective boundary layer statistics and structures generated by large-eddy simulation, J. Atmos. Sci., 68, 2011.

Thomas, L. H.: Elliptic problems in linear difference equations over a network, Watson Sci. Comput. Lab. Rept., Columbia University, New York, 1, 1949.

Tseng, Y.-H. and Ferziger, J. H.: A ghost-cell immersed boundary method for flow in complex geometry, J. Comput. Phys., 192, 593–623, 2003.

Umphrey, C., DeLeon, R., and Senocak, I.: Direct Numerical Simulation of Turbulent Katabatic Slope Flows with an Immersed-Boundary Method, Bound.-Layer Meteor., pp. 1–16, doi:10.1007/s10546-017-0252-3, 2017.

van der Walt, S., Colbert, S. C., and Varoquaux, G.: The NumPy Array: A Structure for Efficient Numerical Computation, Computing in Science & Engineering, 13, 22–30, doi:10.1109/MCSE.2011.37, 2011.

van Heerwaarden, C. C. and Mellado, J. P.: Growth and decay of a convective boundary layer over a surface with a constant temperature, Journal of the Atmospheric Sciences, 73, 2165–2177, 2016.

van Heerwaarden, C. C., Mellado, J. P., and De Lozar, A.: Scaling laws for the heterogeneously heated free convective boundary layer, J. Atmos. Sci., 71, 3975–4000, 2014.

van Reeuwijk, M.: Direct simulation and regularization modeling of turbulent thermal convection, TU Delft, Delft University of Technology, 2007.

Vasilyev, O. V.: High order finite difference schemes on non-uniform meshes with good conservation properties, J. Comput. Phys., 157, 746–761, 2000.

Vreman, A. W.: The projection method for the incompressible Navier–Stokes equations: The pressure near a no-slip wall, J. Comput. Phys., 263, 353–374, 2014.

Williamson, J. H.: Low-Storage Runge-Kutta schemes, J. Comput. Phys., 35, 48–56, 1980.

Wilson, D. K.: An alternative function for the wind and temperature gradients in unstable surface layers, Bound.-Layer Meteor., 99, 151–158, 2001.

Wyngaard, J. C.: Turbulence in the Atmosphere, Cambridge University Press, 2010.