

## **Authors' responses to all reviewers**

First of all we would like to thank all reviewers for their comments and suggestions for improving the manuscript. We tried to take all of them into account in the revised manuscript. Please find our responses to all referee comments below. The changes made in the text can be also traced back in the attached marked-up manuscript. Please note that we also did some minor corrections based on comments provided by personal communication. In particular, these corrections are:

- Scalability tests have been performed on a Cray-XC40 (instead of Cray-XC30)
- We decided to cite Schumann and Sweet (1988) who originally developed the FFT method used in PALM.
- We corrected Figure 7 (wrong indices in the upper left grid box)
- We added a note on recent work on wind turbine parametrizations done by the work group of Heinemann at University of Oldenburg
- We updated the section "Current and future developments" (the current revision of PALM already comes with the RRTMG radiation model and a land surface model, which was not the case at the date the manuscript was initially submitted. Moreover, a full two-way self-nesting is currently under way).

### **Reply to Referee #1 (RC C413)**

#### **Referee comment #1**

Perhaps it is a matter of semantics, but I would suggest the authors rethink the casual use of resolution to describe the numerical grid spacing. In my view, using the word "resolution" implies to users and non-experts that the model is capable of resolving features on the order of the grid scale. Spectral analysis of course shows that the effective resolution is several times larger than the grid spacing due to many factors. I understand what the authors are conveying, but I suggest that they change "resolution" to "spacing" when describing explicit values ( e.g.,5-m spacing, not 5-m resolution). Resolution in broad terms is still reasonable (e.g.we increased the model resolution).

#### **Authors' response**

We followed the suggestion of the reviewer and replaced "resolution" with "spacing" wherever possible.

#### **Changes in the text**

See attached marked-up manuscript.

## **Referee comment #2**

I do not think the authors explicitly say why, but I am curious why 5th-order advection was chosen for LES applications. The scheme, coupled with 3rd-order RK time stepping is known to be overly dissipative, even beyond the grid scale (see, for example, Gibbs and Fedorovich 2014 “Comparison of Convective Boundary Layer Velocity Spectra Retrieved from Large- Eddy-Simulation and Weather Re- search and Forecasting Model Data”). I’m not sure it needs to be justified, but several other model documents discuss why particular numerics are used. It is but one data point, but I would be interested.

## **Authors’ response**

As a matter of fact, the “older” PALM simulations have been conducted using the non-dissipative 2nd-order scheme after Piacsek and Williams (1970) in combination with the 3rd-order Runge Kutta scheme. A few years ago we implemented the 5th-order scheme, based on the finding of Wicker and Skamarock (2002) that the combination of the 3rd-order Runge-Kutta time-stepping scheme gives best results in combination with the 5th-order scheme. This was also cited in Gibbs and Fedorovich (2014). We revised the respective paragraph accordingly.

## **Changes in the text**

By default, the advection terms in Eqs. (1) - (5) are discretized using an upwind-biased 5th-order differencing scheme in combination with a 3rd-order Runge-Kutta time-stepping scheme after Williamson (1980). Wicker and Skamarock (2002) compared different time- and advection differencing schemes and found that this combination give the best results regarding accuracy and algorithmic simplicity. However, the 5th-order differencing scheme is known to be overly dissipative. It is thus also possible to use a 2nd-order scheme after Piacsek and Williams (1970). The latter scheme is non-dissipative, but it suffers from immense numerical dispersion. Time discretization can also be achieved using 2nd-order Runge–Kutta or 1st-order Euler schemes.

## **Referee comment #3**

The authors employ the ever popular Deardorff 1.5-order TKE closure. They also note that future uses of the code will almost certainly extend to stable boundary layers. Are the authors considering an update to the closure? For instance, the scheme was never really designed for stable boundary layers, per se. The formulation for  $K_h$  will almost certainly lead to values larger than  $K_m$  for realistic values of  $N$ . This can overestimate the effects of stratification (as shown by Schumann 1991). Another byproduct is the possibility of a near step-jump in  $K_h$  values between adjoining grid cells in transition periods. This patchy Prandtl pattern can modify local patterns of temperature gradients in a way that can lead to false conclusions about the flow. Do the authors use PALM much for stable conditions? If so, do they have a feeling for how well the scheme behaves in these situations?

## **Authors’ response**

The reviewer is right that the SGS model is generally better suited for simulating convective boundary layers. However, as far as the grid spacing of the model is chosen fine enough to capture the bulk of the turbulence scales, the formulation of the SGS is rather unimportant. This was also shown in the LES intercomparison study published by Beare et al. (2006). Our philosophy has always

followed the concept of LES to resolve the bulk of the turbulence. Hence, we would rather decrease the model grid spacing when simulating stably-stratified boundary layers than to improve the SGS model. However, there are special cases where this concept cannot be pursued, e.g. when there is a lack of computational resources. In this case, one might opt for a more sophisticated SGS model. As a matter of fact we have plans to implement the dynamic subfilter closure model after Esau (2004). We added this plan to the outlook in the revised manuscript

### **Changes in the text**

In order to allow for a sufficient representation of SGS turbulence when using relatively coarse meshes, we intend to implement the dynamic Smagorinsky turbulence closure model after Esau (2004).

### **Referee comment #4**

On page 1552 line 12, the authors state: "The model is initialized by horizontally homogeneous vertical profiles of potential temperature, specific humidity (or a passive scalar), and the horizontal wind velocities." I am wondering if the code uses a specific method to generate turbulence initially. For instance by randomly perturbing the first model level values of potential temperature.

### **Authors' response**

The default approach in PALM is to apply random perturbations on the fields of the horizontal velocity components. We added a note to the manuscript.

### **Changes in the text**

Normally distributed random perturbations with a user-defined amplitude can be imposed to the fields of the horizontal velocities components to initiate turbulence.

### **Referee comment #5**

The document describes a very robust system of available components (terrain, oceans, canopy, etc). I wonder if the authors have any test/ideal cases that might demonstrate the code's performance on famous general cases and those that might benefit from the new additions? It might be reassuring to readers to see visual proof of physical capabilities of the model.

### **Authors' response**

We added some visual proof that shows the specific features of PALM. (see Figs. 5, 6, 8, and 9 in the revised manuscript). Moreover we added citations to already published animations.

### **Changes in the text**

See attached marked-up manuscript.

**Referee comment #6**

Perhaps it is my own stylistic preference, but I do not like the inclusion of code samples. I am not sure if it is required with this journal, but if not I would suggest removing them. The authors are talented in describing the procedures using words. In my estimation, a general new user might not care about the literal outlay of code that was used to make efficient loops. I suggest using an approach like that in the WRF technical documentation (see Skamarock 2008, "A Description of the Advanced Research WRF Version 3"), where procedures are described in text (and charts when necessary).

**Authors' response**

We do think that most of the code fragments are important for the manuscript when it comes to the description of the technical realization, particularly regarding optimization of the code. However, we decided to remove the detailed description of the loop structure for tendencies and prognostic equations, also based on the comment of another referee.

**Changes in the text**

See reply to comment #2 of Referee #2 below.

**Referee comment #7**

On Page 1540, line 16, I suggest changing the sentence to read, "the first key features of LES were studied by Lilly (1967) and Deardorff (1973, 1974)"

**Authors' response**

The sentence has been revised due to a comment from another referee

**Changes in the text**

First investigations using LES were performed by Lilly (1967) and Deardorff (1973, 1974).

**Referee comment #8**

Page 1540, line 24: I suggest changing Turbulent to Turbulence

**Authors' response**

Corrected.

**Referee comment #9**

Page 1541, lines 19-20: I suggest changing the sentence to read, "Thus, Raasch and Schroter (2001) can no longer ..."

**Authors' response**

Corrected.

**Referee comment #10**

In several instance (for example Equation 8) the authors use multiple parentheses in a single expression. I find it more readable if the interior parenthesis are brackets, or some other more easily differentiated brace.

**Authors' response**

We tried to improve the readability of Eqs. 8, 28, 30, 67, 77, 94, and 126.

**Changes in the text**

See attached marked-up manuscript.

## Reply to Referee #2 (RC C627)

### Referee comment #1

It would be useful to include at least one or two simple examples cases of LES with PALM, e.g., some of the typical canonical ABL cases. The examples could include description of the setup as well as some basic results (mean profiles of wind speed and potential temperature), and perhaps a three-dimensional iso-surface level of some quantity.

### Authors' response

Due to the length of the manuscript and the fact that manifold papers have been published, where PALM was successfully applied, we believe that showing standard cases would not be of much use. However, we acknowledge (see also comment from referee #1) that giving visual proof will be a useful extension of the manuscript. We thus decided to show exemplary figures of PALM simulations where the specific features of PALM were applied (see Figs. 5, 6, 8, and 9 in the revised manuscript). Moreover we tried to state former validation and intercomparison studies more precisely in the text wherever possible (e.g. BOMEX, DYCOMS-II, wind tunnel data of Martinuzzi and Tropea (2003), LES intercomparison of Beare et al. 2006, etc.).

### Changes in the text

See attached marked-up manuscript.

### Referee comment #2

When implementation of the advection algorithm is discussed it is not necessary to include the actual code, but it would be sufficient to describe two different implementation of do loops.

### Authors' response

We agree with the referee, the actual code takes far too much space for relatively few information. We have hence revised the respective paragraph by excluding the previous loop design and shortening the new loop layout.

### Changes in the text

The original PALM code calculated the different contributions to the tendency terms (i.e., advection, buoyancy, diffusion, etc.) and the final prognostic equation for each prognostic quantity in separate 3-D-loops over the three spatial directions. In case of large 3-D-arrays that do not fit into the cache of cache based processors like Intel-Xeon or AMD-Athlon, the array data has to be reloaded from the main memory for each 3-D-loop, which is extremely time consuming. For this reason, the outer loops over  $i$  and  $j$  have been extracted from each 3-D-loop, now forming a 2-D-loop over all tendencies and prognostic equations, e.g.:

```
DO i = nx1, nxr
  DO j = nys, nyn
```

```

        DO k = nzb+1, nzt
!--      advection term
          tend(k,j,i) =...
        ENDDO

        DO k = nzb+1, nzt
!--      diffusion term
          tend(k,j,i) = tend(k,j,i) +...
        ENDDO

...    ! further tendencies

      ENDDO
ENDDO

```

### **Referee comment #3**

Page 1540, line 15 - The statement “The idea of LES goes back to Smagorinsky (1963)” is not accurate. Smagorinsky (1963) discussed global circulation modeling and not simulations of fully developed, three-dimensional atmospheric boundary layer turbulence. Smagorinsky (1963) can be given credit for relating eddy viscosity to the magnitude of the local, horizontal rate of strain following previous work of von Neumann and Richtmyer on shock capturing numerical simulations (Smagorinsky, Large Eddy Simulation - Where Do We Stand?, International Workshop, December 19-21, 1990, St. Petersburg Beach Florida). Instead Lilly (1966, 1967; <http://opensky.library.ucar.edu>) and Deardorff can be credited by developing LES.

### **Authors’ response**

The referee is right. Despite the importance of Smagorinsky’s work regarding SGS modeling, his work was not directly related to LES modeling. We thus follow the suggestion of the reviewer and rewrote the sentence.

### **Changes in the text**

First investigations using LES were performed by Lilly (1967) and Deardorff (1973, 1974).

### **Referee comment #4**

Page 1540, line 20 – Some references to review papers on LES of ABLs should be included (e.g. Mason 1994).

**Authors' response**

We added the review of Mason (1994) as well as the LES intercomparison of Beare et al. (2006) to the text

**Changes in the text**

Nowadays, thanks to increasing power of modern supercomputers, the technique is well-known and widely spread within the boundary-layer meteorology community (see review in Mason, 1994). [...] A detailed intercomparison of different LES codes can be found in Beare et al. (2006).

**Referee comment #5**

Page 1551, equations 31, 32, 33, etc. - Symbol Psi is usually used for the integral of the stability function which is usually denoted with Phi.

**Authors' response**

We follow the suggestion of the referee and named the similarity functions with  $\Phi$  (as  $\Phi$  was used for Source/Sink terms, these are now labeled with  $\Psi$  instead).

**Changes in the text**

See attached marked-up manuscript.

**Referee comment #6**

Page 1558, line 10 - Acronym "PSU" should be defined.

**Authors' response**

Practical salinity unit (PSU) was added to the text.

**Changes in the text**

See attached marked-up manuscript.

**Referee comment #7**

Page 1584, line 14 - Instead of "immense" at least an order of magnitude estimate should be given.

**Authors' response**

We replaced "immense numbers" with "billions of".

**Referee comment #8**

Page 1596, line 29 - Instead of "nonlinear" it should be "nonlinearly."



**Authors' response**

Corrected.

**Referee comment #9**

Page 1597, line 1 - Instead of "nonlinear" it should be "nonlinearly."

**Authors' response**

Corrected.

**Referee comment #10**

Page 1597, line 2 - Instead of "speed up" it should be either "sped up" or "speeded up."

**Authors' response**

Corrected.

**Referee comment #11**

Page 1597, line 29 - Instead of "speed up" it should be either "sped up" or "speeded up."

**Authors' response**

Corrected.

**Referee comment #12**

Page 1607, line 3 - I thought that it was mentioned at the beginning that the code is already "anelastic."

**Authors' response**

It was stated on page 1543, line 5-6 that the code is currently using the Boussinesq-approximation, but it was not stated that the anelastic approximation is already available.

**Referee comment #13**

Page 1609, line 16-24 - The discussion lacks depth and adds little value to the manuscript either it should be omitted or expanded.

**Authors' response**

We are afraid that we do not agree with the opinion of the reviewer in this particular case. From our point of view, the paragraph - as it is - points out clearly that the increasing computational power of today's supercomputers finally will allow for more rigorous investigations of the stable boundary layer, which has previously not been possible (or very limited) due to the high computational demands (owing to fine grid spacing and small time steps). Our future developments (e.g. vertical self-nesting) aim at decreasing the computational load for such applications to up to 90% compared to non-nested grids. In summary, the paragraph gives comprehensible information on future applications of PALM and we think it is thus well-suited to be placed under "Future perspectives". Also, we believe that it would not be of much help for the reader to go more into detail regarding possible research questions and stable boundary layer phenomena.

## **Reply to SC C254**

### **Short comment**

In particular, please note that for your paper, the following requirements have not been met in the Discussions paper – please correct this in your revised submission to GMD (I do realise you mention code availability in the final paragraph, but this should be in a clearly labelled separate section)

### **Authors' response**

The section “Code management and regulations” has been moved to the end of the manuscript and was renamed to “Code availability”.

### **Changes in the text**

See attached marked-up manuscript.

## **Reply to SC C307**

### **Short comment**

I guess on the left side of equation (17) (...) has to be deleted.

### **Authors' response**

Corrected.

# The Parallelized Large-Eddy Simulation Model (PALM) version 4.0 for atmospheric and oceanic flows: model formulation, recent developments, and future perspectives

**B. Maronga<sup>1</sup>, M. Gryschka<sup>1</sup>, R. Heinze<sup>1</sup>, F. Hoffmann<sup>1</sup>, F. Kanani-Sühring<sup>1</sup>, M. Keck<sup>1</sup>, K. Ketelsen<sup>2</sup>, M. O. Letzel<sup>3</sup>, M. Sühring<sup>1</sup>, and S. Raasch<sup>1</sup>**

<sup>1</sup>Institute of Meteorology and Climatology, Leibniz Universität Hannover, Hannover, Germany

<sup>2</sup>Software Consultant, Berlin, Germany

<sup>3</sup>Ingenieurbüro Lohmeyer GmbH & Co. KG, Karlsruhe, Germany

Correspondence to: B. Maronga ([maronga@muk.uni-hannover.de](mailto:maronga@muk.uni-hannover.de))

## Abstract

In this paper we present the current version of the Parallelized Large-Eddy Simulation Model (PALM) whose core has been developed at the Institute of Meteorology and Climatology at Leibniz Universität Hannover (Germany). PALM is a Fortran 95-based code with some Fortran 2003 extensions and has been applied for the simulation of a variety of atmospheric and oceanic boundary layers for more than 15 years. PALM is optimized for use on massively parallel computer architectures and was recently ported to general-purpose graphics processing units. In the present paper we give a detailed description of the current version of the model and its features, such as an embedded Lagrangian cloud model and the possibility to use Cartesian topography. Moreover, we discuss recent model developments and future perspectives for LES applications.

## 1 Introduction

In meteorology, Large-eddy simulation (LES) has been used since the early 1970s for various research topics on turbulent flows at large Reynolds numbers in the atmospheric boundary layer (ABL). ~~The idea of LES goes back to Smagorinsky (1963) and first key features of LES have been studied~~ First investigations using LES were performed by Lilly (1967) and Deardorff (1973, 1974). Nowadays, thanks to increasing power of modern supercomputers, the technique is well-known and widely spread within the boundary-layer meteorology community (see review in Mason, 1994). Numerous studies in boundary-layer research that made use of LES have been published since then, with gradually increasing model resolution over the years (Moeng, 1984; Mason, 1989; Wyngaard et al., 1998; Sullivan et al., 1998; Sorbjan, 2007; Maronga, 2014, among many others). ~~LES~~ A detailed intercomparison of different LES codes can be found in Beare et al. (2006). LES models solve the three-dimensional (3-D) prognostic equations for momentum, temperature, humidity, and other scalar quantities. The principle of LES is based on the separation of scales. ~~Turbulent~~ Turbulence scales that are larger than a certain filter width are directly resolved,

whereas the effect of smaller scales is parametrized by a subgrid-scale (SGS) turbulence model. As the bulk part of the energy is contained in the large eddies, about 90 % of the turbulence energy can be resolved by means of LES (e.g. Heus et al., 2010). In practice, the filter width often depends on the grid resolution and therefore on the phenomenon that is studied. Typical filter widths can thus range from 50–100 m for phenomena on a regional scale like arctic cold-air outbreaks (e.g. Gryschka and Raasch, 2005) down to 0.5–2 m for LES of the urban boundary layer with very narrow streets (e.g. Kanda et al., 2013), or for simulations of the stable boundary layer (e.g. Beare et al., 2006).

In this overview paper we describe the Parallelized LES Model (PALM) whose core has been developed at the Institute of Meteorology and Climatology (IMUK) at Leibniz Universität Hannover (Germany). The model is based on the non-parallelized LES code described by Raasch and Etling (1991). The parallelized version was developed about 6 years later and its first formulation can be found in Raasch and Schröter (2001). Therewith, PALM was one of the first parallelized LES models for atmospheric research at all. Many people have helped developing the code further over the past 15 years, and large parts of the code have been added, optimized and improved since then. For example, embedded models such as a Lagrangian cloud model (LCM) as part of a Lagrangian particle model (LPM), and a canopy model have been implemented. Also, an option for Cartesian topography is available. Moreover, the original purpose of the model to study atmospheric turbulence was extended by an option for oceanic flows. ~~It thus appears plausible that the paper of~~ Thus, Raasch and Schröter (2001) can no longer be considered an adequate reference for current and future research articles.

In the present paper we will provide a comprehensive description of the current version 4.0 of PALM. The idea for this overview paper was also partly inspired by Heus et al. (2010), who gave a detailed description of the Dutch Atmospheric Large-Eddy Simulation (DALES) model.

In the course of the release of PALM 4.0 a logo was designed, showing a palm tree – a reference to the acronym PALM (see Fig. 1).

Over the last 15 years, PALM has been applied for the simulation of a variety of boundary layers, ranging from heterogeneously-heated convective boundary layers (e.g. Raasch and Harbusch, 2001; Letzel and Raasch, 2003; Maronga and Raasch, 2013), urban canopy flows (e.g. Park et al., 2012; Kanda et al., 2013), and cloudy boundary layers (e.g. Riechelmann et al., 2012; Hoffmann et al., 2015; Heinze et al., 2015) (e.g. Riechelmann et al., 2012; Hoffmann et al., 2015; Heinze et al., 2015). Moreover, it has been used for studies of the oceanic mixed layer (OML, e.g., Noh et al., 2010, 2011) and recently for studying the feedback between atmosphere and ocean by Esau (2014). PALM also participated in the first intercomparison of LES models for the stable boundary layer, as part of the Global Energy and Water Cycle Experiment Atmospheric Boundary Layer Study initiative (GABLS, Beare et al., 2006). In this experiment, PALM was for the first time successfully used with an extremely ~~high-grid-resolution~~ fine grid spacings of down to 1 m. From the very beginning, PALM was designed and optimized to run very high resolution setups and large model domains efficiently on the world's biggest supercomputers.

The paper is organized as follows: Sect. 2 deals with the description of the model equations, numerical methods and parallelization principles. Section 3 describes the embedded models such as cloud physics, canopy model and LPM, followed by an overview of the technical realization (Sect. 4). In Sect. 5 we will outline topics of past applications of PALM and discuss both upcoming code developments and future perspectives of LES applications in general. Section 7 gives a summary.

## 2 Model formulation

In this section we will give a detailed description of the model. We will confine ourselves to the atmospheric formulation and devote a separate section (see Sect. 2.7) to the ocean option. By default, PALM has six prognostic quantities: the velocity components  $u, v, w$  on a Cartesian grid, the potential temperature  $\theta$ , specific humidity  $q_v$  or a passive scalar  $s$ , and the SGS turbulent kinetic energy (SGS-TKE)  $e$ . The separation of resolved scales and SGS is implicitly achieved by averaging the governing equations (see Sect. 2.1) over discrete

Cartesian grid volumes as proposed by Schumann (1975). Moreover, it is possible to run PALM in a direct numerical simulation mode by switching off the prognostic equation for the SGS-TKE and setting a constant eddy diffusivity. For a list of all symbols and parameters, that we will introduce in Sect. 2.1, see Tables 1 and 2.

## 2.1 Governing equations

The model is based on the non-hydrostatic, filtered, incompressible Navier–Stokes equations in Boussinesq-approximated form. In the following set of equations, angle brackets denote a horizontal domain average. A subscript 0 indicates a surface value. Note that the variables in the equations are implicitly filtered by the discretization (see above), but that the continuous form of the equations is used here for convenience. A double prime indicates SGS variables. The overbar indicating filtered quantities is omitted for readability, except for the SGS flux terms. The equations for the conservation of mass, energy and moisture,



filtered over a grid volume on a Cartesian grid, then read as

$$\frac{\partial u_i}{\partial t} = -\frac{\partial u_i u_j}{\partial x_j} - \varepsilon_{ijk} f_j u_k + \varepsilon_{i3j} f_3 u_{g,j} - \frac{1}{\rho_0} \frac{\partial \pi^*}{\partial x_i} + g \frac{\theta_v - \langle \theta_v \rangle}{\langle \theta_v \rangle} \delta_{i3} - \frac{\partial}{\partial x_j} \left( \overline{u_i'' u_j''} - \frac{2}{3} e \delta_{ij} \right), \quad (1)$$

$$\frac{\partial u_j}{\partial x_j} = 0 \quad (2)$$

$$\frac{\partial \theta}{\partial t} = -\frac{\partial u_j \theta}{\partial x_j} - \frac{\partial}{\partial x_j} \left( \overline{u_j'' \theta''} \right) - \frac{L_V}{c_p \Pi} \Phi \Psi_{q_v} \quad (3)$$

$$\frac{\partial q_v}{\partial t} = -\frac{\partial u_j q_v}{\partial x_j} - \frac{\partial}{\partial x_j} \left( \overline{u_j'' q_v''} \right) + \Phi \Psi_{q_v} \quad (4)$$

$$\frac{\partial s}{\partial t} = -\frac{\partial u_j s}{\partial x_j} - \frac{\partial}{\partial x_j} \left( \overline{u_j'' s''} \right) + \Phi \Psi_s. \quad (5)$$

Here,  $i, j, k \in \{1, 2, 3\}$ .  $u_i$  are the velocity components ( $u_1 = u, u_2 = v, u_3 = w$ ) with location  $x_i$  ( $x_1 = x, x_2 = y, x_3 = z$ ),  $t$  is time,  $f_i = (0, 2\Omega \cos(\phi), 2\Omega \sin(\phi))$  is the Coriolis parameter with  $\Omega$  being the Earth's angular velocity and  $\phi$  being the geographical latitude.  $u_{g,k}$  are the geostrophic wind speed components,  $\rho_0$  is the density of dry air,  $\pi^* = p^* + \frac{2}{3} \rho_0 e$  is the modified perturbation pressure with  $p^*$  being the perturbation pressure and the SGS-TKE  $e = \frac{1}{2} \overline{u_i'' u_i''}$ ; and  $g$  is the gravitational acceleration. The potential temperature is defined as

$$\theta = T / \Pi, \quad (6)$$

with the current absolute temperature  $T$  and the Exner function

$$\Pi = \left( \frac{p}{p_0} \right)^{R_d/c_p} \quad (7)$$

with  $p$  being the hydrostatic air pressure,  $p_0 = 1000$  hPa a reference pressure,  $R_d$  the specific gas constant for dry air, and  $c_p$  the specific heat of dry air at constant pressure. The

virtual potential temperature is defined as

$$\theta_v = \theta \left[ 1 + \left( \frac{R_v}{R_d} - 1 \right) q_v - q_l \right] \quad (8)$$

with the specific gas constant for water vapor  $R_v$ , and the liquid water specific humidity  $q_l$ . For the computation of  $q_l$ , see the descriptions of the embedded cloud microphysical models in Sects. 3.1 and 3.3. Furthermore,  $L_v$  is the latent heat of vaporization, and  $\Phi_{q_v}$  and  $\Phi_s - \Psi_{q_v}$  and  $\Psi_s$  are source/sink terms of  $q_v$  and  $s$ , respectively.

## 2.2 Turbulence closure

One of the main challenges in LES modeling is the turbulence closure. The filtering process yields four SGS covariance terms (see Eqs. 1–5) that cannot be explicitly calculated. In PALM, these SGS terms are parametrized using a 1.5-order closure after Deardorff (1980). PALM uses the modified version of Moeng and Wyngaard (1988) and Saiki et al. (2000). The closure is based on the assumption that the energy transport by SGS eddies is proportional

to the local gradients of the mean quantities and reads

$$\overline{u_i'' u_j''} - \frac{2}{3} e \delta_{ij} = -K_m \left( \frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) \quad (9)$$

$$\overline{u_i'' \theta''} = -K_h \frac{\partial \theta}{\partial x_i} \quad (10)$$

$$\overline{u_i'' q_v''} = -K_h \frac{\partial q_v}{\partial x_i} \quad (11)$$

$$\overline{u_i'' s''} = -K_h \frac{\partial s}{\partial x_i} \quad (12)$$

where  $K_m$  and  $K_h$  are the local SGS eddy diffusivities of momentum and heat, respectively. They are related to the SGS-TKE as follows

$$K_m = c_m l \sqrt{e}, \quad (13)$$

$$K_h = \left( 1 + \frac{2l}{\Delta} \right) K_m. \quad (14)$$

Here,  $c_m = 0.1$  is a model constant and  $\Delta = \sqrt[3]{\Delta x \Delta y \Delta z}$  with  $\Delta x$ ,  $\Delta y$ ,  $\Delta z$  being the grid [resolutions](#) [spacings](#) in  $x$ ,  $y$  and  $z$  direction, respectively. The SGS mixing length  $l$  depends on height  $z$  (distance from the wall when topography is used),  $\Delta$ , and stratification and is

calculated as

$$l = \begin{cases} \min \left( 1.8z, \Delta, 0.76\sqrt{e} \left( \frac{g}{\theta_{v,0}} \frac{\partial \theta_v}{\partial z} \right)^{-\frac{1}{2}} \right) & \text{for } \frac{\partial \theta_v}{\partial z} > 0, \\ \min(1.8z, \Delta) & \text{for } \frac{\partial \theta_v}{\partial z} \leq 0. \end{cases} \quad (15)$$

Moreover, the closure includes a prognostic equation for the SGS-TKE:

$$\frac{\partial e}{\partial t} = -u_j \frac{\partial e}{\partial x_j} - \left( \overline{u_i'' u_j''} \right) \frac{\partial w_i}{\partial x_j} + \frac{g}{\theta_{v,0}} \overline{u_3'' \theta_v''} - \frac{\partial}{\partial x_j} \left[ \overline{u_j'' \left( e + \frac{p''}{\rho_0} \right)} \right] - \epsilon. \quad (16)$$

The pressure term in Eq. (16) is parametrized as

$$-\frac{\partial}{\partial x_j} \left[ \overline{u_j'' \left( e + \frac{p''}{\rho_0} \right)} \right] = -2K_m \frac{\partial e}{\partial x_j} \quad (17)$$

and  $\epsilon$  is the SGS dissipation rate within a grid volume, given by

$$\epsilon = \left( 0.19 + 0.74 \frac{l}{\Delta} \right) \frac{e^{\frac{3}{2}}}{l}. \quad (18)$$

Since  $\theta_v$  depends on  $\theta$ ,  $q_v$ , and  $q_l$  (see Eq. 8), the vertical SGS buoyancy flux  $\overline{w'' \theta_v''}$  depends on the respective SGS fluxes (Stull, 1988, Chap. 4.4.5):

$$\overline{w'' \theta_v''} = K_1 \cdot \overline{w'' \theta''} + K_2 \cdot \overline{w'' q_v''} - \theta \cdot \overline{w'' q_l''}, \quad (19)$$

with

$$K_1 = 1 + \left( \frac{R_v}{R_d} - 1 \right) q_v - q_l, \quad (20)$$

$$K_2 = \left( \frac{R_v}{R_d} - 1 \right) \theta, \quad (21)$$

and the vertical SGS flux of liquid water, calculated as

$$\overline{w'' q_l''} = -K_h \frac{\partial q_l}{\partial z}. \quad (22)$$

Note that this parametrization of the SGS buoyancy flux (Eq. 19) differs from that used with bulk cloud microphysics (see Sect. 3.1.8).

## 2.3 Discretization

The model domain in PALM is discretized in space using finite differences and equidistant horizontal grid spacings ( $\Delta x$ ,  $\Delta y$ ). The grid can be stretched in the vertical direction well above the ABL to save computational time in the free atmosphere. The Arakawa staggered C-grid (Harlow and Welch, 1965; Arakawa and Lamb, 1977) is used, where scalar quantities are defined at the center of each grid volume, whereas velocity components are shifted by half a grid width in their respective direction so that they are defined at the edges of the grid volumes (see Fig. 2). It is thus possible to calculate the derivatives of the velocity components at the center of the volumes (same location as the scalars). By the same token, derivatives of scalar quantities can be calculated at the edges of the volumes. In this way it is possible to calculate derivatives over only one grid length and the effective spatial model resolution can be increased by a factor of two in comparison to non-staggered grids.

By default, the advection terms in Eqs. (1)–(5) are discretized using ~~the an upwind-biased 5th-order scheme after Wicker and Skamarock (2002)~~. ~~Alternatively, the 2nd-order scheme after Piacsek and Williams (1970) is available.~~ ~~Discretization in time is achieved using a differencing scheme in combination with a 3rd-order Runge–Kutta time-stepping scheme (Williamson, 1980) as standard.~~ ~~Alternatively, time-stepping scheme after Williamson (1980).~~ ~~Wicker and Skamarock (2002) compared different time- and advection differencing schemes and found that this combination give the best results regarding accuracy and algorithmic simplicity. However, the 5th-order differencing scheme is known to be overly dissipative. It is thus also possible to use a 2nd-order scheme after Piacsek and Williams (1970).~~ ~~The latter scheme is non-dissipative, but it suffers from immense numerical dispersion. Time discretization can also be achieved using 2nd-order Runge–Kutta or 1st-order Euler schemes~~ ~~can be used.~~

## 2.4 Pressure solver

The Boussinesq approximation requires incompressibility of the flow, but the integration of the governing equations formulated in Sect. 2.1 does not provide this feature. Divergence

of the flow field is thus inherently produced. Hence, a predictor-corrector method is used where an equation is solved for the modified perturbation pressure after every time step (e.g. Patrinos and Kistler, 1977). In a first step, the pressure term  $-(1/\rho_0)\partial\pi^*/\partial x_i$  is excluded from Eq. (1) during time integration. This yields a preliminary velocity  $u_{i,\text{pre}}^{t+\Delta t}$  at time  $t + \Delta t$ . Emerging divergences can then be attributed to the pressure term. Subsequently, the prognostic velocity can be decomposed in a second step as

$$u_i^{t+\Delta t} = u_{i,\text{pre}}^{t+\Delta t} - \Delta t \cdot \frac{1}{\rho_0} \frac{\partial \pi^{*t}}{\partial x_i}. \quad (23)$$

The third step then is to stipulate incompressibility for  $u_i^{t+\Delta t}$

$$\frac{\partial}{\partial x_i} u_i^{t+\Delta t} = \frac{\partial}{\partial x_i} \left( u_{i,\text{pre}}^{t+\Delta t} - \Delta t \cdot \frac{1}{\rho_0} \frac{\partial \pi^{*t}}{\partial x_i} \right) \stackrel{!}{=} 0. \quad (24)$$

The result is a Poisson equation for  $\pi^*$ :

$$\frac{\partial^2 \pi^{*t}}{\partial x_i^2} = \frac{\rho_0}{\Delta t} \frac{\partial u_{i,\text{pre}}^{t+\Delta t}}{\partial x_i}. \quad (25)$$

The exact solution of Eq. (25) would give a  $\pi^*$  that yields a  $u_i^{t+\Delta t}$  free of divergence when used in Eq. (23). In practice, a numerically efficient reduction of divergence by several orders of magnitude is found to be sufficient. Note that the differentials in Eqs. (23)–(25) are used for convenience and that the model code uses finite differences instead. When employing a Runge–Kutta time stepping scheme, the formulation above is used to solve the Poisson equation for each substep.  $\pi^*$  is then calculated from its weighted average over these substeps.

In case of cyclic lateral boundary conditions, the solution of Eq. (25) is achieved by using a direct fast Fourier transform (FFT). The Poisson equation is Fourier transformed in both horizontal directions, the resulting tri-diagonal matrix is solved along the  $z$  direction, and then transformed back ([see e.g. Schumann and Sweet, 1988](#)). PALM provides

the inefficient but less restrictive Singleton-FFT (Singleton, 1969) and the well optimized Temperton-FFT (Temperton, 1992). External FFT libraries can be used as well, with the FFTW (Frigo and Johnson, 1998) being the most efficient one. Alternatively, the iterative multigrid scheme can be used (e.g. Hackbusch, 1985). This scheme uses an iterative successive over-relaxation (SOR) method for the inner iterations on each grid level. The convergence of this scheme is steered by the number of so-called V- or W-cycles to be carried out for each call of the scheme and by the number of SOR iterations to be carried out on each grid level. As the multigrid scheme does not require periodicity along the horizontal directions, it allows for using non-cyclic lateral boundary conditions.

## 2.5 Boundary conditions

PALM offers a variety of boundary conditions. Dirichlet or Neumann boundary conditions can be chosen for  $u$ ,  $v$ ,  $\theta$ ,  $q_v$ , and  $p^*$  at the bottom and top of the model. For the horizontal velocity components the choice of Neumann (Dirichlet) boundary conditions yields free-slip (no-slip) conditions. Neumann boundary conditions are also used for the SGS-TKE. Kinematic fluxes of heat and moisture can be prescribed at the surface instead (Neumann conditions) of temperature and humidity (Dirichlet conditions). At the top of the model, Dirichlet boundary conditions can be used with given values of the geostrophic wind. By default, the lowest grid level ( $k = 0$ ) for the scalar quantities and horizontal velocity components is not staggered vertically and defined at the surface ( $z = 0$ ). In case of free-slip boundary conditions at the bottom of the model, the lowest grid level is defined below the surface ( $z = -0.5 \cdot \Delta z$ ) instead. Vertical velocity is assumed to be zero at the surface and top boundaries, which implies using Neumann conditions for pressure.

Following Monin–Obukhov similarity theory (MOST) a constant flux layer can be assumed as boundary condition between the surface and the first grid level where scalars and horizontal velocities are defined ( $k = 1$ ,  $z_{MO} = 0.5 \cdot \Delta z$ ). It is then required to provide the roughness lengths for momentum  $z_0$  and heat  $z_{0,h}$ . Momentum and heat fluxes as well as the horizontal velocity components are calculated using the following framework. The formulation is theoretically only valid for horizontally-averaged quantities. In PALM we assume that

MOST can be also applied locally and we therefore calculate local fluxes, velocities, and scaling parameters.

Following MOST, the vertical profile of the horizontal wind velocity  $u_h = (u^2 + v^2)^{\frac{1}{2}}$  is given in the surface layer by

$$\frac{\partial u_h}{\partial z} = \frac{u_*}{\kappa z} \underline{\Psi} \Phi_m \left( \frac{z}{L} \right), \quad (26)$$

where  $\kappa = 0.4$  is the Von Kármán constant and  $\underline{\Psi} \Phi_m$  is the similarity function for momentum in the formulation of Businger–Dyer (see e.g. Panofsky and Dutton, 1984)

$$\underline{\Psi} \Phi_m = \begin{cases} 1 + 5 \frac{z}{L} & \text{for } \frac{z}{L} \geq 0 \\ (1 - 16 \frac{z}{L})^{-\frac{1}{4}} & \text{for } \frac{z}{L} < 0. \end{cases} \quad (27)$$

Here,  $L$  is the Obukhov length, calculated as

$$L = \frac{\theta_v(z) u_*^2}{\kappa g (\theta_* + 0.61 \theta(z) q_* + 0.61 q_v(z) \theta_*)} \frac{\theta_v(z) u_*^2}{\kappa g [\theta_* + 0.61 \theta(z) q_* + 0.61 q_v(z) \theta_*]}. \quad (28)$$

The scaling parameters  $\theta_*$  and  $q_*$  are defined by MOST as:

$$\theta_* = -\frac{\overline{w'' \theta''}_0}{u_*}, \quad q_* = -\frac{\overline{w'' q''}_0}{u_*}, \quad (29)$$

with the friction velocity  $u_*$  defined as

$$u_* = \left[ (\overline{w'' w''}_0)^2 + (\overline{v'' w''}_0)^2 \right]^{\frac{1}{4}}. \quad (30)$$

In PALM,  $u_*$  is calculated from  $u_h$  at  $z_{\text{MO}}$  by vertical integration of Eq. (26) over  $z$  from  $z_0$  to  $z_{\text{MO}}$ .



From Eqs. (26) and (30) it is possible to derive a formulation for the horizontal wind components, viz.

$$\frac{\partial u}{\partial z} = \frac{-\overline{u''w''}_0}{u_*\kappa z} \underline{\Psi}\Phi_m\left(\frac{z}{L}\right) \text{ and } \frac{\partial v}{\partial z} = \frac{-\overline{v''w''}_0}{u_*\kappa z} \underline{\Psi}\Phi_m\left(\frac{z}{L}\right). \quad (31)$$

Vertical integration of Eq. (31) over  $z$  from  $z_0$  to  $z_{\text{MO}}$  then yields the surface momentum fluxes  $\overline{u''w''}_0$  and  $\overline{v''w''}_0$ .

The formulations above all require knowledge of the scaling parameters  $\theta_*$  and  $q_*$ . These are deduced from vertical integration of

$$\frac{\partial \theta}{\partial z} = \frac{\theta_*}{\kappa z} \underline{\Psi}\Phi_h\left(\frac{z}{L}\right) \text{ and } \frac{\partial q_v}{\partial z} = \frac{q_*}{\kappa z} \underline{\Psi}\Phi_h\left(\frac{z}{L}\right) \quad (32)$$

over  $z$  from  $z_{0,h}$  to  $z_{\text{MO}}$ . The similarity function  $\underline{\Psi}\Phi_h$  is given by

$$\underline{\Psi}\Phi_h = \begin{cases} 1 + 5\frac{z}{L} & \text{for } \frac{z}{L} \geq 0 \\ (1 - 16\frac{z}{L})^{-1/2} & \text{for } \frac{z}{L} < 0. \end{cases} \quad (33)$$

Note that this implementation of MOST in PALM requires the use of data from the previous time step. The following steps are thus carried out in sequential order. First of all,  $\theta_*$  and  $q_*$  are calculated by integration of Eq. (32) using the value of  $z_{\text{MO}}/L$  from the previous time step. Second, the new value of  $z_{\text{MO}}/L$  is derived from Eq. (28) using the new values of  $\theta_*$  and  $q_*$ , but using  $u_*$  from the previous time step. Then, the new values of  $u_*$ , and subsequently  $\overline{u''w''}_0$  as well as  $\overline{v''w''}_0$  are calculated by integration of Eqs. (26), and (31), respectively. At last, Eq. (29) is employed to calculate the new surface fluxes  $\overline{w''\theta''}_0$  and  $\overline{w''q''}_0$ . In the special case, when surface fluxes are prescribed instead of surface temperature and humidity, the first and last steps are omitted and  $\theta_*$  and  $q_*$  are directly calculated using Eq. (29) instead.

Furthermore, the flat bottom of the model can be replaced by a Cartesian topography (see Sect. 2.5.4).

By default, lateral boundary conditions are set to be cyclic in both directions. Alternatively, it is possible to opt for non-cyclic conditions in one direction, i.e., a laminar or turbulent inflow boundary (see Sect. 2.5.1) and an open outflow boundary on the opposite site (see Sect. 2.5.3). The boundary conditions for the other direction have to remain cyclic.

In order to prevent gravity waves from being reflected at the top boundary, a sponge layer (Rayleigh damping) can be applied to all prognostic variables in the upper part of the model domain (Klemp and Lilly, 1978). Such a sponge layer should be applied only within the free atmosphere, where no turbulence is present.

The model is initialized by horizontally homogeneous vertical profiles of potential temperature, specific humidity (or a passive scalar), and the horizontal wind velocities. The latter can be also provided from a 1-D precursor run (see Sect. 3.5). [Uniformly distributed random perturbations with a user-defined amplitude can be imposed to the fields of the horizontal velocities components to initiate turbulence.](#)

### 2.5.1 Laminar and turbulent inflow boundary conditions

In case of laminar inflow, Dirichlet boundary conditions are used for all quantities, except for the SGS-TKE  $e$  and perturbation pressure  $\pi^*$  for which Neumann boundary conditions are used. Vertical profiles, as taken for the initialization of the simulation, are used for the Dirichlet boundary conditions. In order to allow for a fast turbulence development, random perturbations can be imposed on the velocity fields within a certain area behind the inflow boundary (inlet). These perturbations may persist for the entire simulation. For the purpose of preventing gravity waves from being reflected at the inlet, a relaxation area can be defined after Davies (1976). So far, it was found to be sufficient to implement this method for temperature only. This is hence realized by an additional term in the prognostic equation for

$\theta$  (see Eq. 3):

$$\frac{\partial \theta}{\partial t} = \dots - C_{\text{relax}} (\theta - \theta_{\text{inlet}}). \quad (34)$$

Here,  $\theta_{\text{inlet}}$  is the stationary inflow profile of  $\theta$ , and  $C_{\text{relax}}$  is a relaxation coefficient, depending on the distance  $d$  from the inlet, viz.

$$C_{\text{relax}}(d) = \begin{cases} F_{\text{inlet}} \cdot \sin^2\left(\frac{\pi}{2} \frac{D-d}{D}\right) & \text{for } d < D, \\ 0 & \text{for } d \geq D, \end{cases} \quad (35)$$

with  $D$  being the length of the relaxation region and  $F_{\text{inlet}}$  being a damping factor.

### 2.5.2 Turbulence recycling

If non-cyclic horizontal boundary conditions are used, PALM offers the possibility of generating time-dependent turbulent inflow data by using a turbulence recycling method. The method follows the one described by Lund et al. (1998), with the modifications introduced by Kataoka and Mizuno (2002). Figure 3 gives an overview of the recycling method used in PALM. The turbulent signal  $\varphi'(y, z, t)$  is taken from a recycling plane which is located at a fixed distance  $x_{\text{recycle}}$  from the inlet:

$$\varphi'(y, z, t) = \varphi(x_{\text{recycle}}, y, z, t) - \langle \varphi \rangle_y(z, t), \quad (36)$$

where  $\langle \varphi \rangle_y(z, t)$  is the line average of a prognostic variable  $\varphi \in \{u, v, w, \theta, e\}$  along  $y$  at  $x = x_{\text{recycle}}$ .  $\varphi'(y, z)$  is then added to the mean inflow profile  $\langle \varphi_{\text{inflow}} \rangle_y(z)$  at  $x_{\text{inlet}}$  after each time step:

$$\varphi_{\text{inlet}}(y, z, t) = \langle \varphi_{\text{inlet}} \rangle_y(z) + \phi(z) \varphi'(y, z, t), \quad (37)$$

with the inflow damping function  $\phi(z)$ , which has a value of 1 below the initial boundary layer height, and which is linearly damped to 0 above, in order to inhibit growth of the

boundary layer depth.  $\langle \varphi_{\text{inlet}} \rangle_y(z)$  is constant in time and either calculated from the results of the precursor run or prescribed by the user. The distance  $x_{\text{recycle}}$  has to be chosen much larger than the integral length scale of the respective turbulent flow. Otherwise, the same turbulent structures could be recycled repeatedly, so that the turbulence spectrum is illegally modified. It is thus recommended to use a precursor run for generating the initial turbulence field of the main run. The precursor run can have a comparatively small domain along the horizontal directions. In that case the domain of the main run is filled by cyclic repetition of the precursor run data. Note that the turbulence recycling has not been adapted for humidity and passive scalars so far.

Turbulence recycling is frequently used for simulations with urban topography. In such a case, topography elements should be placed sufficiently downstream of  $x_{\text{recycle}}$  to prevent effects on the turbulence at the inlet.

### 2.5.3 Open outflow boundary conditions

At the outflow boundary (outlet), the velocity components  $u_i$  meet radiation boundary conditions, viz.

$$\frac{\partial u_i}{\partial t} + U_{u_i} \frac{\partial u_i}{\partial n} = 0, \quad (38)$$

as proposed by Orlanski (1976). Here  $\partial/\partial n$  is the derivative normal to the outlet (i.e.,  $\partial/\partial x$  in Fig. 3) and  $U_{u_i}$  a transport velocity which includes wave propagation and advection. Rewriting Eq. (38) yields the transport velocity

$$U_{u_i} = - \left( \frac{\partial u_i}{\partial t} \right) \left( \frac{\partial u_i}{\partial n} \right)^{-1} \quad (39)$$

that is calculated at interior grid points next to the outlet at the preceding time step for each velocity component. If the transport velocity, calculated by means of Eq. (39), is outside the range  $0 \leq U_{u_i} \leq \Delta/\Delta t$ , it is set to the respective threshold value that is exceeded. Because this local determination of  $U_{u_i}$  can show high variations in case of complex turbulent flows,

it is averaged laterally to the direction of the outflow, so that it varies only in the vertical direction. Alternatively, the transport velocity can be set to the upper threshold value ( $U_{u_i} = \Delta/\Delta t$ ) for the entire outlet. Eqs. (38) and (39) are discretized using an upstream method following Miller and Thorpe (1981). As the radiation boundary condition does not ensure conservation of mass, a mass flux correction can be applied at the outlet.

#### 2.5.4 Topography

The Cartesian topography in PALM is generally based on the mask method (Briscolini and Santangelo, 1989) and allows for explicitly resolving solid obstacles such as buildings and orography. The implementation makes use of the following simplifications:

1. the obstacle shape is approximated by (an appropriate number of) full grid cells to fit the grid, i.e., a grid cell is either 100 % fluid or 100 % obstacle,
2. so far, only bottom surface-mounted obstacles are permitted (no holes or overhanging structures),
3. the obstacles are fixed (not moving).

These simplifications transform the 3-D obstacle dimension to a 2.5-D topography. This reduced dimension format is conform to the Digital Elevation Model (DEM) format. DEMs of city morphologies have become increasingly available worldwide due to advances in remote sensing technologies. Consequently, it is sufficient to provide 2-D topography height data to mask obstacles and their faces in PALM. The model domain is then separated into three subdomains (see Fig. 4):

- A. grid points in free fluid without adjacent walls, where the standard PALM code is executed,
- B. grid points next to walls that require extra code (e.g., wall functions)
- C. grid points within obstacles that are excluded from calculations.

Additional topography code is only executed in grid volumes of subdomain B. The faces of the obstacles are always located where the respective wall-normal velocity components  $u$ ,  $v$ , and  $w$  are defined (cf. Fig. 2) so that the impermeability boundary condition can be implemented by setting the respective wall-normal velocity component to zero.

An exception is made for the 5th-order advection scheme, where the numerical stencil at grid points adjacent to obstacles would require data within the obstacle. In order to avoid this behavior, the order of the advection scheme is successively degraded at respective grid volumes adjacent to obstacles, i.e., from the 5th-order to 3rd-order at the second grid point above/beside an obstacle and from the 3rd-order to a 2nd-order at grid points directly adjacent to an obstacle.

Wall surfaces in PALM can be aligned horizontally (bottom surface or rooftop, i.e., always facing upwards) or vertically (facing north, east, south or west direction). At horizontal surfaces, PALM allows to either specify the surface values ( $\theta$ ,  $q_v$ ,  $s$ ) or to prescribe their respective surface fluxes. The latter is the only option for vertically oriented surfaces. Simulations with topography require the application of MOST between each wall surface and the first computational grid point. For vertical walls, neutral stratification is assumed for MOST. [The topography implementation has been validated by Letzel et al. \(2008\) and Kanda et al. \(2013\)](#). Park and Baik (2013) have recently extended the vertical wall boundary conditions for non-neutral stratifications [and validated their results against wind tunnel data](#). Up to now, however, these modifications are not included in ~~the~~ PALM 4.0. [Figure 5 shows exemplarily the development of turbulence structures induced by a densely built-up artificial island off the coast of Macau, China \(see also animation in Knoop et al., 2014\)](#). [The approaching flow above the sea exhibits relatively weak turbulence due to the smooth water surface. Within the building areas, strong turbulence is generated by additional wind shear \(due to the walls of isolated buildings\) and due to a general increase in surface roughness.](#)

The technical realization of the topography will be outlined in Sect. 4.3.

## 2.6 Large-scale forcing

Processes occurring on larger scales (LS) than usually considered in LES and which are affecting the local LES scales have to be prescribed by additional source terms. These LS processes include pressure gradients via the geostrophic wind, subsidence and horizontal advection of scalars. In case of cyclic boundary conditions, this forcing is prescribed homogeneously in the horizontal directions and thus depends on height and time only. The relation between LS pressure ( $p_{LS}$ ) gradient and geostrophic wind is given by

$$\frac{\partial p_{LS}}{\partial x_i} = -\rho_0 \varepsilon_{i3j} f_3 u_{g,j} \quad (40)$$

and enters Eq. (1). LS vertical advection (subsidence or ascent) tendencies can be prescribed for the scalar prognostic variables  $\varphi \in \{\theta, q, s\}$  by means of

$$\left. \frac{\partial \varphi}{\partial t} \right|_{SUB} = -w_{LS} \frac{\partial \varphi}{\partial z}. \quad (41)$$

The so-called subsidence velocity  $w_{LS}$  and the geostrophic wind components  $u_g$  and  $v_g$  can either be prescribed gradient-wise or they can be provided in an external file. Moreover, an external pressure gradient can be applied for simulations with Coriolis force switched off, which is usually required for simulations to be compared with wind tunnel experiments.

To account for less-idealized flow situations, time-dependent surface fluxes (or surface temperature and humidity) can be prescribed. Moreover, LS horizontal advective (LSA) tendencies can be added to the scalar quantities by means of

$$\left. \frac{\partial \varphi}{\partial t} \right|_{LSA} = - \left( u_{LS} \frac{\partial \varphi_{LS}}{\partial x} + v_{LS} \frac{\partial \varphi_{LS}}{\partial y} \right). \quad (42)$$

These tendencies are typically derived from larger scale models or observations and should be spatially averaged over a large domain so that local-scale perturbations are avoided.

Newtonian relaxation (nudging) towards given large-scale profiles  $\varphi_{\text{LS}}$  can be used for  $\varphi \in \{u, v, \theta, q, s\}$  via

$$\left. \frac{\partial \varphi}{\partial t} \right|_{\text{NUD}} = - \frac{\langle \varphi \rangle - \varphi_{\text{LS}}}{\tau_{\text{LS}}} . \quad (43)$$

$\tau_{\text{LS}}$  is a relaxation time-scale which, on the one hand, should be chosen large enough in the order of several hours to allow an undisturbed development of the small-scale turbulence in the LES model. On the other hand it should be chosen small enough to account for synoptic disturbances (Neggers et al., 2012). In this way, the nudging can prevent considerable model drift in time.

## 2.7 Ocean option

PALM allows for studying the OML by using an ocean option where the sea surface is defined at the top of the model, so that negative values of  $z$  indicate the depth. Hereafter, we keep the terminology and use the word *surface* and index 0 for variables at the sea surface and top of the ocean model. For a list of ocean specific parameters, see Table 3. The ocean version differs from the atmospheric version by a few modifications, which are handled in the code by distinction of cases, so that both versions share the same basic code. In particular, seawater buoyancy and static stability depend not only on  $\theta$ , but also on the salinity  $S_a$ . In order to account for the effect of salinity on density, a prognostic equation



is added for Sa (in PSU): [practical salinity unit](#):

$$\frac{\partial \text{Sa}}{\partial t} = -\frac{\partial u_j \text{Sa}}{\partial x_j} - \frac{\partial}{\partial x_j} \left( \overline{u_j'' \text{Sa}''} \right) + \Phi \Psi_{\text{Sa}}, \quad (44)$$

where  $\Phi \Psi_{\text{Sa}}$  represents sources and sinks of salinity. Furthermore,  $\theta_v$  is replaced by potential density  $\rho_\theta$  in the buoyancy term of Eq. (1)

$$+ g \frac{\theta_v - \langle \theta_v \rangle}{\langle \theta_v \rangle} \delta_{i3} \rightarrow -g \frac{\rho_\theta - \langle \rho_\theta \rangle}{\langle \rho_\theta \rangle} \delta_{i3}, \quad (45)$$

in the stability related term of the SGS-TKE equation (Eq. 16)

$$+ \frac{g}{\theta_{v,0}} \overline{u_3'' \theta_v''} \rightarrow + \frac{g}{\rho_{\theta,0}} \overline{u_3'' \rho_\theta''} \quad (46)$$

as well as in the calculation of the mixing length (Eq. 15)

$$\left( \frac{g}{\theta_{v,0}} \frac{\partial \theta_v}{\partial z} \right)^{-\frac{1}{2}} \rightarrow \left( \frac{g}{\rho_{\theta,0}} \frac{\partial \rho_\theta}{\partial z} \right)^{-\frac{1}{2}}. \quad (47)$$

$\rho_\theta$  is calculated from the equation of state of seawater after each time step using the algorithm proposed by Jackett et al. (2006). The algorithm is based on polynomials depending on Sa,  $\theta$ , and  $p$  (see Jackett et al., 2006, Table A2). At the moment, only the initial values of  $p$  enter this equation.

The ocean is driven by prescribed fluxes of momentum, heat and salinity at the top. The boundary conditions at the bottom of the model can be chosen as for atmospheric runs, including the possibility to use topography at the sea bottom.

Note that the current version of the ocean option does not account for the effect of surface waves (e.g., Langmuir circulation and wave-breaking). Parametrization schemes might, however, be provided within the user interface (see Sect. 4.5) and have been used, e.g., by Noh et al. (2004). The ocean option in its current state was recently used for simulations

of the ocean mixed layer by Esau (2014), who investigated indirect air–sea interactions by means of the atmosphere–ocean coupling scheme that will be described in Sect. 2.8. Note that most previous PALM studies of the OML used the atmospheric code, subsequent inversion of the  $z$ -axis and appropriate normalization of the results, instead of using the relatively new ocean option (e.g. Noh et al., 2004, 2009).

## 2.8 Coupled atmosphere–ocean simulations

A coupled mode for the atmospheric and oceanic versions of PALM has been developed in order to allow for studying the interaction between turbulent processes in the ABL and OML. The coupling is realized by the online exchange of information at the sea surface (boundary conditions) between two PALM runs (one atmosphere and one ocean). The atmospheric model uses a constant flux layer and transfers the kinematic surface fluxes of heat and moisture as well as the momentum fluxes to the oceanic model. Flux conservation between the ocean and the atmosphere requires an adjustment of the fluxes for the density of water  $\rho_{1,0}$ :

$$\begin{aligned}\overline{w''u''}_0|_{\text{ocean}} &= \frac{\rho_0}{\rho_{l,0}} \overline{w''u''}_0, \\ \overline{w''v''}_0|_{\text{ocean}} &= \frac{\rho_0}{\rho_{l,0}} \overline{w''v''}_0.\end{aligned}\quad (48)$$

Since evaporation leads to cooling of the surface water, the kinematic flux of heat in the ocean depends on both the atmospheric kinematic surface fluxes of heat and moisture and is calculated by

$$\overline{w''\theta''}_0|_{\text{ocean}} = \frac{\rho_0}{\rho_{l,0}} \frac{c_p}{c_{p,l}} \left( \overline{w''\theta''}_0 + \frac{L_N}{c_p} \overline{w''q''}_0 \right).\quad (49)$$

Here,  $c_{p,l}$  is the specific heat of water at constant pressure. Since salt does not evaporate, evaporation of water also leads to an increase in salinity in the ocean subsurface. This process is modeled after Steinhorn (1991) by a negative (downward) salinity flux at the sea surface:

$$\overline{w''S''}_0|_{\text{ocean}} = -\frac{\rho_0}{\rho_{l,0}} \frac{S}{1000 \text{ PSU} - S} \overline{w''q''}_0.\quad (50)$$

Sea surface values of potential temperature and the horizontal velocity components are transferred as surface boundary conditions to the atmosphere:

$$\theta_0 = \theta_0|_{\text{ocean}}, u_0 = u_0|_{\text{ocean}}, v_0 = v_0|_{\text{ocean}}.\quad (51)$$

The time steps for atmosphere and ocean are set individually and are not required to be equal. The coupling is then executed at a user-prescribed frequency. At the moment, the coupling requires equal extents of the horizontal model domains in both atmosphere and ocean. In order to account for the fact that eddies in the ocean are generally smaller but usually have lower velocities than in the atmosphere, it is beneficial to use different grid spacings in both models (i.e., finer grid ~~resolution~~ spacing in the ocean model). In

this case, the coupling is realized by a two-way bi-linear interpolation of the data fields at the sea surface. Furthermore, it is possible to perform uncoupled precursor runs for both atmosphere and ocean, followed by a coupled restart run. In this way it is possible to reduce the computational load due to different spin-up times in atmosphere and ocean.

As mentioned above, this coupling has been successfully applied for the first time in the recent study of Esau (2014). Furthermore, we would encourage the atmospheric and oceanic scientific community to consider the coupled atmosphere–ocean LES technique for further applications in the future.

### **3 Embedded models**

PALM offers several optional embedded models that can be switched on for special purposes. In this section we will describe the embedded cloud microphysics model (Sect. 3.1, Table 4), the LPM for use of Lagrangian particles as passive tracers (Sect. 3.2, Table 5), the LCM which uses the LPM for the simulation of explicit cloud droplets and aerosols (Sect. 3.3), and the canopy model (Sect. 3.4, Table 6). Moreover, we will outline the one-dimensional (1-D) version of PALM in Sect. 3.5, which is used for creating steady-state wind profiles to be used as initialization of the 3-D model.

#### **3.1 Cloud microphysics**

PALM offers an embedded bulk cloud microphysics representation that takes into account the liquid water specific humidity and warm (i.e., no ice) cloud-microphysical processes.

Therefore, PALM solves the prognostic equations for the total water content

$$q = q_v + q_l, \quad (52)$$

instead of  $q_v$ , and for a linear approximation of the liquid water potential temperature (e.g. Emanuel, 1994)

$$\theta_l = \theta - \frac{L_V}{c_p \Pi} q_l, \quad (53)$$

instead of  $\theta$  as described in Sect. 2.1. Since  $q$  and  $\theta_l$  are conserved quantities for wet adiabatic processes, condensation/evaporation is not considered for these variables.

Liquid phase microphysics are parametrized following the two-moment scheme of Seifert and Beheng (2001, 2006), which is based on the separation of the droplet spectrum into droplets with radii  $< 40 \mu\text{m}$  (cloud droplets) and droplets with radii  $\geq 40 \mu\text{m}$  (rain droplets). The model predicts the first two moments of these partial droplet spectra, namely cloud and rain droplet number concentration ( $N_c$  and  $N_r$ , respectively) as well as cloud and rain water specific humidity ( $q_c$  and  $q_r$ , respectively). Consequently,  $q_l$  is the sum of both  $q_c$  and  $q_r$ . The moments' corresponding microphysical tendencies are derived by assuming the partial droplet spectra to follow a gamma distribution that can be described by the predicted quantities and empirical relationships for the distribution's slope and shape parameters. For a detailed derivation of these terms, see Seifert and Beheng (2001, 2006).

We employ the computational efficient implementation of this scheme as used in the UCLA-LES (Savic-Jovicic and Stevens, 2008) and DALES (Heus et al., 2010) models. We

thus solve only two additional prognostic equations for  $N_r$  and  $q_r$ :

$$\frac{\partial N_r}{\partial t} = -u_j \frac{\partial N_r}{\partial x_j} - \frac{\partial}{\partial x_j} \left( \overline{u_j'' N_r''} \right) + \underline{\Phi} \Psi_{N_r}, \quad (54)$$

$$\frac{\partial q_r}{\partial t} = -u_j \frac{\partial q_r}{\partial x_j} - \frac{\partial}{\partial x_j} \left( \overline{u_j'' q_r''} \right) + \underline{\Phi} \Psi_{q_r}, \quad (55)$$

with the sink/source terms  $\underline{\Phi} \Psi_{N_r}$  and  $\underline{\Phi} \Psi_{q_r}$ , and the SGS fluxes

$$\overline{u_j'' N_r''} = -K_h \frac{\partial q_r}{\partial x_j} \quad (56)$$

$$\overline{u_j'' q_r''} = -K_h \frac{\partial N_r}{\partial x_j} \quad (57)$$

with  $N_c$  and  $q_c$  being a fixed parameter and a diagnostic quantity, respectively.

In the next subsections we will describe the diagnostic determination of  $q_c$ . From Sect. 3.1.2 on, the microphysical processes considered in the sink/source terms of  $\theta_1$ ,  $q$ ,  $N_r$  and  $q_r$ ,

$$\underline{\Phi} \Psi_{\theta_1} = -\frac{L_v}{c_p \bar{\Pi}} \varphi_q, \quad (58)$$

$$\underline{\Phi} \Psi_q = \frac{\partial q}{\partial t} \Big|_{\text{sed, c}} + \frac{\partial q}{\partial t} \Big|_{\text{sed, r}}, \quad (59)$$

$$\underline{\Phi} \Psi_{N_r} = \frac{\partial N_r}{\partial t} \Big|_{\text{auto}} + \frac{\partial N_r}{\partial t} \Big|_{\text{sfl/brk}} + \frac{\partial N_r}{\partial t} \Big|_{\text{evap}} + \frac{\partial N_r}{\partial t} \Big|_{\text{sed, r}}, \quad (60)$$

$$\underline{\Phi} \Psi_{q_r} = \frac{\partial q_r}{\partial t} \Big|_{\text{auto}} + \frac{\partial q_r}{\partial t} \Big|_{\text{accr}} + \frac{\partial q_r}{\partial t} \Big|_{\text{evap}} + \frac{\partial q_r}{\partial t} \Big|_{\text{sed, r}}, \quad (61)$$

are used in the formulations of Seifert and Beheng (2006) unless explicitly specified. Section 3.1.8 gives an overview of the necessary changes for the turbulence closure (cf. Sect. 2.2) using  $q$  and  $\theta_1$  instead of  $q_v$  and  $\theta$ , respectively.

### 3.1.1 Diffusional growth of cloud water

The diagnostic estimation of  $q_c$  is based on the assumption that water supersaturations are immediately removed by the diffusional growth of cloud droplets only. This can be justified since the bulk surface area of cloud droplets exceeds that of rain drops considerably (Stevens and Seifert, 2008). Following this saturation adjustment approach,  $q_c$  is obtained by

$$q_c = \max(0, q - q_r - q_s), \quad (62)$$

where  $q_s$  is the saturation specific humidity. Because  $q_s$  is a function of  $T$  (not predicted),  $q_s$  is computed from the liquid water temperature  $T_l = \Pi \theta_l$  in a first step:

$$q_s(T_l) = \frac{R_d}{R_v} \frac{p_{v,s}(T_l)}{p - (1 - R_d/R_v) p_{v,s}(T_l)}, \quad (63)$$

using an empirical relationship for the saturation water vapor pressure  $p_{v,s}$  (Bougeault, 1981):

$$p_{v,s}(T_l) = 610.78 \text{ Pa} \cdot \exp\left(17.269 \frac{T_l - 273.16 \text{ K}}{T_l - 35.86 \text{ K}}\right). \quad (64)$$

$q_s(T)$  is subsequently calculated from a 1st-order Taylor series expansion of  $q_s$  at  $T_l$  (Sommeria and Deardorff, 1977):

$$q_s(T) = q_s(T_l) \frac{1 + \beta q}{1 + \beta q_s(T_l)}, \quad (65)$$

with

$$\beta = \frac{L_v^2}{R_v c_p T_l^2}. \quad (66)$$

### 3.1.2 Autoconversion

In the following Sects. 3.1.2–3.1.4 we describe collision and coalescence processes by applying the stochastic collection equation (e.g. Pruppacher and Klett, 1997, Chap. 15.3) in the framework of the described two-moment scheme. As two species (cloud and rain droplets, hereafter also denoted as  $c$  and  $r$ , respectively) are considered only, there are three possible interactions affecting the rain quantities: autoconversion, accretion, and self-collection. Autoconversion summarizes all merging of cloud droplets resulting in rain drops ( $c + c \rightarrow r$ ). Accretion describes the growth of rain drops by the collection of cloud droplets ( $r + c \rightarrow r$ ). Selfcollection denotes the merging of rain drops ( $r + r \rightarrow r$ ).

The local temporal change of  $q_r$  due to autoconversion is

$$\left. \frac{\partial q_r}{\partial t} \right|_{\text{auto}} = \frac{K_{\text{auto}}}{20 m_{\text{sep}}} \frac{(\mu_c + 2)(\mu_c + 4)}{(\mu_c + 1)^2} q_c^2 m_c^2 \cdot \left[ 1 + \frac{\Psi_{\text{auto}}(\tau_c)}{(1 - \tau_c)^2} \frac{\Phi_{\text{auto}}(\tau_c)}{(1 - \tau_c)^2} \right] \rho_0. \quad (67)$$

Assuming that all new rain drops have a radius of  $40 \mu\text{m}$  corresponding to the separation mass  $m_{\text{sep}} = 2.6 \times 10^{-10} \text{ kg}$ , the local temporal change of  $N_r$  is

$$\left. \frac{\partial N_r}{\partial t} \right|_{\text{auto}} = \rho \left. \frac{\partial q_r}{\partial t} \right|_{\text{auto}} \frac{1}{m_{\text{sep}}}. \quad (68)$$

Here,  $K_{\text{auto}} = 9.44 \times 10^9 \text{ m}^3 \text{ kg}^{-2} \text{ s}^{-1}$  is the autoconversion kernel,  $\mu_c = 1$  is the shape parameter of the cloud droplet  $\Gamma$ -distribution and  $m_c = \rho q_c / N_c$  is the mean mass of cloud droplets.  $\tau_c = 1 - q_c / (q_c + q_r)$  is a dimensionless timescale steering the autoconversion similarity function

$$\Psi \Phi_{\text{auto}} = 600 \cdot \tau_c^{0.68} (1 - \tau_c^{0.68})^3. \quad (69)$$

The increase of the autoconversion rate due to turbulence can be considered optionally by an increased autoconversion kernel depending on the local kinetic energy dissipation rate after Seifert et al. (2010).



### 3.1.3 Accretion

The increase of  $q_r$  by accretion is given by:

$$\left. \frac{\partial q_r}{\partial t} \right|_{\text{accr}} = K_{\text{accr}} q_c q_r \underline{\Psi}\Phi_{\text{accr}}(\tau_c) (\rho_0 \rho)^{\frac{1}{2}}, \quad (70)$$

with the accretion kernel  $K_{\text{accr}} = 4.33 \text{ m}^3 \text{ kg}^{-1} \text{ s}^{-1}$  and the similarity function

$$\underline{\Psi}\Phi_{\text{accr}} = \left( \frac{\tau_c}{\tau_c + 5 \times 10^{-5}} \right)^4. \quad (71)$$

Turbulence effects on the accretion rate can be considered after using the kernel after Seifert et al. (2010).

### 3.1.4 Selfcollection and breakup

Selfcollection and breakup describe merging and splitting of rain drops, respectively, which affect the rain water drop number concentration only. Their combined impact is parametrized

as

$$\left. \frac{\partial N_r}{\partial t} \right|_{\text{slf/brk}} = -(\Psi\Phi_{\text{break}}(r) + 1) \left. \frac{\partial N_r}{\partial t} \right|_{\text{self}}, \quad (72)$$

with the breakup function

$$\Psi\Phi_{\text{break}} = \begin{cases} 0 & \text{for } \tilde{r}_r < 0.15 \times 10^{-3} \text{ m,} \\ K_{\text{break}}(\tilde{r}_r - r_{\text{eq}}) & \text{otherwise,} \end{cases} \quad (73)$$

depending on the volume averaged rain drop radius

$$\tilde{r}_r = \left( \frac{\rho q_r}{\frac{4}{3} \pi \rho_{l,0} N_r} \right)^{\frac{1}{3}}, \quad (74)$$

the equilibrium radius  $r_{\text{eq}} = 550 \times 10^{-6} \text{ m}$  and the breakup kernel  $K_{\text{break}} = 2000 \text{ m}^{-1}$ . The local temporal change of  $N_r$  due to selfcollection is

$$\left. \frac{\partial N_r}{\partial t} \right|_{\text{self}} = K_{\text{self}} N_r q_r (\rho_0 \rho)^{\frac{1}{2}}, \quad (75)$$

with the selfcollection kernel  $K_{\text{self}} = 7.12 \text{ m}^3 \text{ kg}^{-1} \text{ s}^{-1}$ .

### 3.1.5 Evaporation of rain water

The evaporation of rain drops in subsaturated air (relative water supersaturation  $S < 0$ ) is parametrized following Seifert (2008):

$$\left. \frac{\partial q_r}{\partial t} \right|_{\text{evap}} = 2\pi G S \frac{N_r \lambda_r^{\mu_r+1}}{\Gamma(\mu_r+1)} f_v \rho, \quad (76)$$

where

$$G = \left[ \frac{R_v T}{K_v p_{v,s}(T)} + \left( \frac{L_v}{R_v T} - 1 \right) \frac{L_v}{\lambda_h T} \right]^{-1}, \quad (77)$$

with  $K_v = 2.3 \times 10^{-5} \text{ m}^2 \text{ s}^{-1}$  being the molecular diffusivity water vapor in air and  $\lambda_h = 2.43 \times 10^{-2} \text{ W m}^{-1} \text{ K}^{-1}$  being the heat conductivity of air. Here,  $N_r \lambda_r^{\mu_r+1} / \Gamma(\mu_r+1)$  denotes the intercept parameter of the rain drop gamma distribution with  $\Gamma$  being the gamma-function. Following Stevens and Seifert (2008), the slope parameter reads as

$$\lambda_r = \frac{((\mu_r+3)(\mu_r+2)(\mu_r+1))^{\frac{1}{3}}}{2 \cdot \tilde{r}_r}, \quad (78)$$

with  $\mu_r$  being the shape parameter, given by

$$\mu_r = 10 \cdot (1 + \tanh(1200 \cdot (2 \cdot \tilde{r}_r - 0.0014))). \quad (79)$$

In order to account for the increased evaporation of falling rain drops, the so-called ventilation effect, a ventilation factor  $f_v$  is calculated optionally by a series expansion considering the rain drop size distribution (Seifert, 2008, Appendix).

The complete evaporation of rain drops (i.e., their evaporation to a size smaller than the separation radius of  $40 \mu\text{m}$ ) is parametrized as

$$\left. \frac{\partial N_r}{\partial t} \right|_{\text{evap}} = \gamma \frac{N_r}{\rho q_r} \left. \frac{\partial q_r}{\partial t} \right|_{\text{evap}}, \quad (80)$$

with  $\gamma = 0.7$  (see also Heus et al., 2010).

### 3.1.6 Sedimentation of cloud water

As shown by Ackerman et al. (2009), the sedimentation of cloud water has to be taken in account for the simulation of stratocumulus clouds. They suggest the cloud water sedimentation flux to be calculated as

$$F_{q_c} = k \left( \frac{4}{3} \pi \rho_l N_c \right)^{-2/3} (\rho q_c)^{5/3} \exp(5 \ln^2 \sigma_g), \quad (81)$$

based on a Stokes drag approximation of the terminal velocities of log-normal distributed cloud droplets. Here,  $k = 1.2 \times 10^8 \text{ m}^{-1} \text{ s}^{-1}$  is a parameter and  $\sigma_g = 1.3$  the geometric SD of the cloud droplet size distribution (Geoffroy et al., 2010). The tendency of  $q$  results from the sedimentation flux divergences and reads as

$$\left. \frac{\partial q}{\partial t} \right|_{\text{sed, c}} = - \frac{\partial F_{q_c}}{\partial z} \frac{1}{\rho}. \quad (82)$$

### 3.1.7 Sedimentation of rain water

The sedimentation of rain water is implemented following Stevens and Seifert (2008). The sedimentation velocities are based on an empirical relation for the terminal fall velocity after Rogers et al. (1993). They are given by

$$w_{N_r} = \left( 9.65 \text{ m s}^{-1} - 9.8 \text{ m s}^{-1} (1 + 600 \text{ m} / \lambda_r)^{-(\mu_r + 1)} \right), \quad (83)$$

and

$$w_{q_r} = \left( 9.65 \text{ m s}^{-1} - 9.8 \text{ m s}^{-1} (1 + 600 \text{ m} / \lambda_r)^{-(\mu_r + 4)} \right). \quad (84)$$

The resulting sedimentation fluxes  $F_{N_r}$  and  $F_{q_r}$  are calculated using a semi-Lagrangian scheme and a slope limiter (see Stevens and Seifert, 2008, their Appendix A). The resulting tendencies read as

$$\left. \frac{\partial q_r}{\partial t} \right|_{\text{sed, r}} = - \frac{\partial F_{q_r}}{\partial z}, \quad \left. \frac{\partial N_r}{\partial t} \right|_{\text{sed, r}} = - \frac{\partial F_{N_r}}{\partial z}, \quad \text{and} \quad \left. \frac{\partial q}{\partial t} \right|_{\text{sed, r}} = \frac{\partial q_r}{\partial t} \Big|_{\text{sed, r}}. \quad (85)$$

### 3.1.8 Turbulence closure

Using bulk cloud microphysics, PALM predicts liquid water temperature  $\theta_l$  and total water content  $q$  instead of  $\theta$  and  $q_v$ . Consequently, some terms in Eq. (19) are unknown. We thus follow Cuijpers and Duynkerke (1993) and calculate the SGS buoyancy flux from the known SGS fluxes  $\overline{w''\theta_l''}$  and  $\overline{w''q''}$ . In unsaturated air ( $q_c = 0$ ) Eq. (19) is then replaced by

$$\overline{w''\theta_v''} = K_1 \cdot \overline{w''\theta_l''} + K_2 \cdot \overline{w''q''}, \quad (86)$$

with

$$K_1 = 1 + \left( \frac{R_v}{R_d} - 1 \right) \cdot q, \quad (87)$$

$$K_2 = \left( \frac{R_v}{R_d} - 1 \right) \cdot \theta_l, \quad (88)$$

and in saturated air ( $q_c > 0$ ) by

$$K_1 = \frac{1 - q + \frac{R_v}{R_d}(q - q_l) \cdot \left( 1 + \frac{L_v}{R_v T} \right)}{1 + \frac{L_v^2}{R_v c_p T^2}(q - q_l)}, \quad (89)$$

$$K_2 = \left( \frac{L_v}{c_p T} K_1 - 1 \right) \cdot \theta. \quad (90)$$

### 3.1.9 Recent applications

The two-moment cloud microphysics scheme has been used within the framework of the HD(CP)<sup>21</sup> project to produce LES simulation data for the evaluation and benchmarking of ICON-LES (Dipankar et al., 2015). Figure 6 contains a snapshot from such a benchmark simulation, where three continuous days were simulated. The figure shows the simulated

<sup>1</sup>High Definition Clouds and Precipitation for Climate Prediction, <http://www.hdcp2.eu>

clouds including precipitation events on 26 April 2013 during a frontal passage. Moreover, cloud microphysics have been recently used for investigations of shadow effects of shallow convective clouds on the ABL and their feedback on the cloud field (not yet published).

## 3.2 Lagrangian particle model (LPM)

The embedded LPM allows for studying transport and dispersion processes within turbulent flows. In the following we will describe the general modeling of particles, including passive particles that do not show any feedback on the turbulent flow. In Sect. 3.3 we will describe the use of Lagrangian particles as explicit cloud droplets.

### 3.2.1 Formulation of the LPM

Lagrangian particles can be released in prescribed source volumes at different points in time. The particles then obey

$$\frac{dx_{p,i}}{dt} = u_{p,i}(t) \quad (91)$$

where  $x_{p,i}$  describes the particle location in  $x_i$  direction ( $i \in \{1, 2, 3\}$ ) and  $u_{p,i}$  is the respective velocity component of the particle. Particle trajectories are calculated by means of the turbulent flow fields provided by PALM for each time step. The location of a certain particle at time  $t + \Delta t_L$  is calculated by

$$x_{p,i}(x_{ps,i}, t + \Delta t_L) = x_{p,i}(x_{ps,i}, t) + \int_t^{t+\Delta t_L} u_{p,i}(\hat{t}) d\hat{t}, \quad (92)$$

where  $x_{ps,i}$  is the spatial coordinate of the particle source point and  $\Delta t_L$  is the applied time step in the Lagrangian particle model. Note that the latter is not necessarily equal to the time step of the LES model. The integral in Eq. (92) is evaluated using either a Runge–Kutta (2nd- or 3rd-order) or the (1st-order) Euler time-stepping scheme.

The velocity of a weightless particle that is transported passively by the flow is determined by

$$u_{p,i} = u_i(x_{p,i}), \quad (93)$$

and for non-passive particles (e.g., cloud droplets) by

$$\frac{du_{p,i}}{dt} = \frac{1}{\tau_p} \left( u_i(x_{p,i}) - u_{p,i} \right) - \delta_{i3} \left( 1 - \frac{\rho_0}{\rho_{p,0}} \right) g, \quad (94)$$

considering Stoke's drag, gravity and buoyancy (on the right-hand side, from left to right). Note that Eq. (94) is solved analytically assuming all variables but  $u_{p,i}$  as constants for one time step. Here,  $u_i(x_{p,i})$  is the velocity of air at the particles location gathered from the eight adjacent grid points of the LES by tri-linear interpolation (see Sect. 4.2). Since Stoke's drag is only valid for radii  $\leq 30 \mu\text{m}$  (e.g. Rogers and Yau, 1989), a non-linear correction is applied to the Stokes's drag relaxation time scale:

$$\tau_p^{-1} = \frac{9\nu\rho_0}{2r^2\rho_{p,0}} \cdot (1 + 0.15 \cdot Re_p^{0.687}). \quad (95)$$

Here,  $r$  is the radius of the particle,  $\nu = 1.461 \times 10^{-5} \text{m}^2 \text{s}$  the molecular viscosity of air, and  $\rho_{p,0}$  the density of the particle. The particle Reynolds number is given by

$$Re_p = \frac{2r |u_i(x_{p,i}) - u_{p,i}|}{\nu}. \quad (96)$$

Following Lamb (1978) and the concept of LES modeling, the Lagrangian velocity of a weightless particle can be split into a resolved-scale contribution  $u_p^{\text{res}}$  and an SGS contri-

bution  $u_p^{\text{sgs}}$ :

$$u_{p,i} = u_{p,i}^{\text{res}} + u_{p,i}^{\text{sgs}}. \quad (97)$$

$u_{p,i}^{\text{res}}$  is determined by interpolation of the respective LES velocity components  $u_i$  to the position of the particle. The SGS part of the particle velocity at time  $t$  is given by

$$u_{p,i}^{\text{sgs}}(t) = u_{p,i}^{\text{sgs}}(t - \Delta t_L) + du_{p,i}^{\text{sgs}}, \quad (98)$$

where  $du_{p,i}^{\text{sgs}}$  describes the temporal change of the SGS particle velocity during a time step of the LPM based on Thomson (1987). Note that the SGS part of  $u_{p,i}$  in Eq. (92) is always computed using the (1st-order) Euler time-stepping scheme. Weil et al. (2004) developed a formulation of the Langevin-equation under assumption of isotropic Gaussian turbulence in order to treat the SGS particle dispersion in terms of a stochastic differential equation. This equation reads as

$$du_{p,i}^{\text{sgs}} = -\frac{3c_{\text{sgs}}C_L\epsilon}{4} \frac{u_{p,i}^{\text{sgs}}}{e} \Delta t_L + \frac{1}{2} \left( \frac{1}{e} \frac{de}{\Delta t_L} u_{p,i}^{\text{sgs}} + \frac{2}{3} \frac{\partial e}{\partial x_i} \right) \Delta t_L + (c_{\text{sgs}}C_L\epsilon)^{\frac{1}{2}} d\zeta_i \quad (99)$$

and is used in PALM for the determination of the change in SGS particle velocities. Here,  $C_L = 3$  is a universal constant ( $C_L = 4 \pm 2$ , see Thomson, 1987).  $\zeta$  is a vector composed of Gaussian-shaped random numbers, with each component neither spatially nor temporally correlated. The factor

$$c_{\text{sgs}} = \frac{\langle e \rangle}{\langle e_{\text{res}} \rangle + \langle e \rangle}, \quad 0 \leq c_{\text{sgs}} \leq 1, \quad (100)$$

where  $e_{\text{res}}$  is the resolved-scale TKE as resolved by the numerical grid, assures that the temporal change of the modeled SGS particle velocities is, on average (horizontal mean), smaller than the change of the resolved-scale particle velocities (Weil et al., 2004). Values of  $e$  and  $\epsilon$  are provided by the SGS model (see Eqs. 16 and 18, respectively). The first term on the right-hand side of Eq. (99) represents the influence of the SGS particle velocity from



the previous time step (i.e., inertial “memory”). This effect is considered by the Lagrangian time scale after Weil et al. (2004):

$$\tau_L = \frac{4}{3} \frac{e}{c_{\text{sgs}} C_L \epsilon}, \quad (101)$$

which describes the time span during which  $u_p^{\text{sgs}}(t - \Delta t_L)$  is correlated to  $u_p^{\text{sgs}}(t)$ . The applied time step of the particle model hence must not be larger than  $\tau_L$ . In PALM, the particle time step is set to be smaller than  $\tau_L/40$ . The second term on the right-hand side of Eq. (99) ensures that the assumption of well-mixed conditions by Thomson (1987) is fulfilled on the subgrid scales. This term can be considered as drift correction, which shall prevent an over-proportional accumulation of particles in regions of weak turbulence (Rodean, 1996). The third term on the right-hand side of Eq. (99) is of stochastic nature and describes the SGS diffusion of particles by a Gaussian random process. For a detailed derivation and discussion of Eq. (99) see Thomson (1987), Rodean (1996) and Weil et al. (2004).

The required values of the resolved-scale particle velocity components,  $e$ , and  $\epsilon$  are obtained from the respective LES fields using the eight adjacent grid points of the LES and tri-linear interpolation on the current particle location (see Sect. 4.2). An exception is made in case of no-slip boundary conditions set for the resolved-scale horizontal wind components below the first vertical grid level above the surface. Here, the resolved-scale particle velocities are determined from MOST (see Sect. 2.5) in order to capture the logarithmic wind profile within the height interval of  $z_0$  to  $z_{\text{MO}}$ . The available values of  $u_*$ ,  $\overline{w''u''}_0$ , and  $\overline{w''v''}_0$  are first bi-linearly interpolated to the horizontal location of the particle. In a second step the velocities are determined using Eqs. (30)–(31). Resolved-scale horizontal velocities of particles residing at height levels below  $z_0$  are set to zero. The LPM allows to switch off the transport by the SGS velocities.

### 3.2.2 Boundary conditions and release of particles

Different boundary conditions can be used for particles. They can be either reflected or absorbed at the surface and top of the model. The lateral boundary conditions for particles can either be set to absorption or cyclic conditions.

The user can explicitly prescribe the release location and events as well as the maximum lifetime of each particle. Moreover, the embedded LPM provides an option for defining different groups of particles. For each group the horizontal and vertical extension of the particle source volumes as well as the spatial distance between the released particles can be prescribed individually for each source area. In this way it is possible to study the dispersion of particles from different source areas simultaneously.

### 3.2.3 Recent applications

The embedded LPM has been recently applied for the evaluation of footprint models over homogeneous and heterogeneous terrain (Steinfeld et al., 2008; Markkanen et al., 2009, 2010; Sühling et al., 2014). For example, Steinfeld et al. (2008) calculated vertical profiles of crosswind-integrated particle concentrations for continuous point sources and found good agreement with the convective tank experiments of Willis and Deardorff (1976), as well as with LES results presented by Weil et al. (2004). Moreover, Steinfeld et al. (2008) calculated footprints for turbulence measurements and showed the benefit of the embedded LPM for footprint prediction compared to Lagrangian dispersion models with fully parametrized turbulence. Noh et al. (2006) used the LPM to study the sedimentation of inertial particles in the OML. Moreover, the LPM has been used for visualizing urban canopy flows as well as dust-devil-like vortices (Raasch and Franke, 2011).

## 3.3 Lagrangian cloud model (LCM)

The LCM is based on the formulation of the LPM (Sect. 3.2). For the LCM, however, the Lagrangian particles are representing droplets and aerosols. The droplet advection and sedimentation is given by Eqs. (94) and (95) with  $\rho_{p,0} = \rho_{l,0}$ . ~~The LCM has been~~

recently applied by Riechelmann et al. (2012) and Lee et al. (2014) for studying turbulence and droplet dynamics in convective clouds. At present it is computationally not feasible to simulate a realistic amount of particles. A single Lagrangian particle thus represents an ensemble of identical particles (i.e., same radius, velocity, mass of solute aerosol) and is referred to as “super-droplet”. The number of particles in this ensemble is referred to as the “weighting factor”. For example,  $q_l$  of a certain LES grid volume results from all Lagrangian particles located therein considering their individual weighting factor  $A_n$ :

$$q_l = \frac{4/3\pi\rho_{l,0}}{\rho_0\Delta V} \sum_{n=1}^{N_p} A_n r_n^3, \quad (102)$$

with  $N_p$  being the number of particles inside the grid volume of size  $\Delta V$ , and  $r_n$  being the radius of the particle. The concept of weighting factors and super-droplets in combination with LES has been also used similarly by Andrejczuk et al. (2008) and Shima et al. (2009) for warm clouds, as well as by Sölch and Kärcher (2010) for ice clouds.

### 3.3.1 Diffusional growth

The growth of a particle by diffusion of water vapor, i.e., condensation and evaporation, is described by

$$r \frac{dr}{dt} = \frac{f_v}{F_D + F_k} (S - S_{eq}), \quad (103)$$

with the coefficients

$$F_D = \frac{R_v T}{K_v p_{v,s}(T)} \rho_{l,0} \text{ and } F_k = \left( \frac{L_v}{R_v T} - 1 \right) \frac{L_v}{\lambda_h T} \rho_{l,0}, \quad (104)$$

depending primarily on the diffusion of water vapor in air and heat conductivity of air, respectively.  $f_v$  is the ventilation factor, which accounts for the increased diffusion of water

vapor, particularly the accelerated evaporation of large drops precipitating from a cloud (e.g. Pruppacher and Klett, 1997, Chap. 13.2.3):

$$f_v = \begin{cases} 1 + 0.09 \cdot Re_p & \text{for } Re_p < 2.5, \\ 0.78 + 0.28 \cdot Re_p^{0.5} & \text{otherwise.} \end{cases} \quad (105)$$

Here,  $Re_p$  is particle Reynolds number. The relative water supersaturation  $S$  is computed from the LES values of  $\theta$  and  $q_v$ , tri-linearly interpolated to the particle's position. The equilibrium saturation term  $S_{eq}$  considers the impact of surface tension as well as the physical and chemical properties of the solute aerosol on the equilibrium saturation of the droplet. In order to take into account these effects, the optional activation model for fully soluble aerosols must be switched on:

$$S_{eq} = \begin{cases} 0 & \text{without activation,} \\ A_{eq} r^{-1} - B_{eq} r^{-3} & \text{with activation,} \end{cases} \quad (106)$$

with coefficients for surface tension

$$A_{eq} = \frac{2\vartheta}{\rho_{l,0} R_v T}, \quad (107)$$

and physical and chemical properties

$$B_{eq} = \frac{F_{vH} m_s M_l}{\frac{4}{3} \pi \rho_{l,0} M_s}. \quad (108)$$

Here,  $\vartheta$  is the temperature-dependent surface tension, and  $M_l = 18.01528 \text{ g mol}^{-1}$  the molecular mass of water. Depending on the simulation setup (e.g., continental or maritime conditions), the physical and chemical properties of the aerosol, its mass  $m_s$ , molecular mass  $M_s$ , and the van't Hoff factor  $F_{vH}$ , indicating the degree of the solute aerosol's dissociation, are prescribed. As discussed by Hoffmann et al. (2015), the aerosol mass (or

equivalently aerosol radius) can be specified by an additional particle feature allowing the initialization of aerosol mass distributions, i.e., varying aerosol masses among the simulated particle ensemble.

In summary, diffusional growth is the major coupling between the LES and LCM model. The change of water vapor during one time step is considered in the prognostic equations for potential temperature (see Eq. 3) and specific humidity (see Eq. 4) by

$$\Phi\Psi_{q_v} = \frac{1}{\Delta t} \frac{\frac{4}{3}\pi\rho_{l,0}}{\rho_0\Delta V} \sum_{n=1}^{N_p} A_n (r_n^{*3} - r_n^3). \quad (109)$$

Here,  $r_n$  and  $r_n^*$  are the radius of the  $n$ th droplet before and after diffusional growth, respectively. Since the diffusional growth (see Eq. 103) is a stiff differential equation, we use a 4th-order Rosenbrock-method (Press et al., 1996; Grabowski et al., 2011), adapting its internal time step for both a computationally efficient and numerically accurate solution.

### 3.3.2 Collision and coalescence

Collision and coalescence are computed using a statistical approach that allows the collision of all droplets that are currently located in the same LES grid volume. For this purpose, two quantities are predicted: the weighting factor, i.e., the number of droplets represented by a super-droplet, and the bulk mass of all droplets represented by a super droplet,  $m_n = A_n(4/3)\pi\rho_l r_n^3$ . For the collision of a super-droplet with a super-droplet smaller in radius, we assume that the larger droplets merges with a certain amount of smaller droplets. Thereby, the weighting factor of the larger super-droplet is kept constant, while bulk mass and consequently radius increase (see Fig. ??7a). On the other hand, the weighting factor and bulk mass of the smaller super-droplet decrease according to the amount of droplets lost to the larger super-droplet, keeping the smaller super-droplet's radius constant. As described in Riechelmann et al. (2015), we allow the droplets represented by a single super-droplet to collide among each other. These internal collisions only decrease the weighting factor of the super-droplet but not the bulk mass. Consequently, internal collisions increase

the super-droplet's radius (see Fig. ??7b). The collision kernel  $K$ , which describes the collision probability of two droplets, can either be a purely gravitational one (Hall, 1980) or including turbulence effects (Ayala et al., 2008).

We arrange the droplets by radius such that  $r_n \leq r_{n+1}$ . The weighting factor after one collision/coalescence time step then reads as

$$A_n^* = A_n - K(r_n, r_n) \frac{1}{2} \frac{A_n(A_n - 1)}{\Delta V} \Delta t - \sum_{m=n+1}^{N_p} K(r_m, r_n) \frac{A_n A_m}{\Delta V} \Delta t. \quad (110)$$

The asterisk denotes a quantity after one collision/coalescence time step. On the right-hand side, we consider the initial weighting factor (first term), the loss of droplets due to internal collisions (second term), and the loss of droplets due to collision with all larger droplets (third term). Note that collision with smaller droplets does not change the weighting factor of the larger droplet.

Since the mass of all droplets represented by a single super-droplet is not a useful quantity, we predict the volume averaged radius of all droplets represented by a super-droplet directly:

$$r_n^* = \left( \frac{m_n^*}{\frac{4}{3}\pi\rho_{l,0}A_n^*} \right)^{\frac{1}{3}} \quad (111)$$

$$= \left[ \left( r_n^3 + \sum_{m=1}^{n-1} K(r_n, r_m) \frac{A_m}{\Delta V} r_m^3 \Delta t - \sum_{m=n+1}^{N_p} K(r_m, r_n) \frac{A_m}{\Delta V} r_n^3 \Delta t \right) \cdot \left( 1 - K(r_n, r_n) \frac{1}{2} \frac{A_n - 1}{\Delta V} \Delta t - \sum_{m=n+1}^{N_p} K(r_m, r_n) \frac{A_m}{\Delta V} \Delta t \right)^{-1} \right]^{\frac{1}{3}}. \quad (112)$$

On the right-hand side, the nominator (first pair of round brackets) contains the initial mass (first term), the gain of mass due to collisions with all smaller droplets (second term), and the loss of mass due to collisions with all larger droplets (third term). The denominator (second pair of round brackets) is identical to Eq. (110) divided by  $A_n$ .

### 3.3.3 Recent applications

The LCM was validated against traditional bulk models for the BOMEX<sup>2</sup> experiment (see Siebesma et al., 2003) by Riechelmann et al. (2012). Riechelmann et al. (2012) and Lee et al. (2014) used the LCM for studying turbulence and droplet dynamics in convective clouds; Hoffmann et al. (2015) investigated cloud droplet activation. Figure 8 shows the spatial distribution of simulated droplets and their respective radius within a simulated cumulus cloud. It appears that the largest drops (in terms of radius) are located at the top and the edges of the cloud, whereas smaller droplets tend to be located at the cloud base.

<sup>2</sup>The Barbados Oceanographic and Meteorological Experiment

### 3.4 Canopy model

The embedded plant canopy model allows for studying the turbulent flow inside and above vegetation canopy. It is well-known that vegetation canopy effects on the surface–atmosphere exchange of momentum, energy and mass can be rather complex and can significantly modify the structure of the ABL, particularly in its lower part (e.g. Raupach et al., 1996; Dupont and Brunet, 2009). It is thus not possible to describe such processes by means of the roughness length and surface fluxes of sensible and latent heat. The canopy model in PALM accounts for the vertically extended drag, release of heat, plant evaporation and leaf-air-interactions that are functions of height within the canopy layer.

Dynamical effects of the plant canopy are based on the assumption that the canopy acts as a sink for momentum due to form (pressure) and viscous drag forces. This sink for momentum is modeled following Shaw and Schumann (1992) and Watanabe (2004) by adding the term  $C_{u_i}$  to Eq. (1):

$$\frac{\partial u_i}{\partial t} = \dots - \underbrace{c_d \text{LAD} \sqrt{u_i^2} u_i}_{C_{u_i}}. \quad (113)$$

Here,  $C_{u_i}$  represents the net resolved-scale dynamical effect of the canopy, averaged over the respective grid volume.  $c_d$  is the canopy drag coefficient with typical values around 0.2 (e.g. Cescatti and Marcolla, 2004), and LAD is the leaf area density (available leaf area per unit volume). As an example, LAD is rather constant with height within crop fields, whereas it is often very heterogeneous in forests, where most of the leaf area is concentrated in the trees' crown space (e.g. Yi, 2008).



The effect of the canopy on the SGS turbulence is considered by adding a similar sink term to the prognostic equation for SGS-TKE (see Eq. 16):

$$\frac{\partial e}{\partial t} = \dots - \underbrace{2c_d LAD \sqrt{u_i^2}}_{C_e} e. \quad (114)$$

This approach was suggested by Shaw and Schumann (1992) and is based on the assumption that SGS-TKE is dissipated by the canopy due to the rapid dissipation of wake turbulence in the lee of plant elements. This rapid break-down of turbulence is also known as the spectral shortcut (e.g. Shaw and Patton, 2003). This type of canopy model has been successfully applied by various authors to study turbulent flows inside and above homogeneous as well as heterogeneous canopies such as forest edges (Cassiani et al., 2008; Finnigan et al., 2009; Dupont and Brunet, 2009, among others).

In case of incoming solar radiation the plant canopy acts as a source for heat. It is assumed that this warming of the foliage by solar radiation results in a warming of the surrounding air. This process is considered by adding a source term  $C_\theta$  to the prognostic equation for  $\theta$  (see Eq. 3):

$$\frac{\partial \theta_1}{\partial t} = \dots + \underbrace{\frac{\partial Q_\theta}{\partial z}}_{C_\theta}. \quad (115)$$

In order to account for the fact that solar radiation can penetrate different layers of the canopy, based on the leaf area, an exponential decay function for the upward vertical kinematic heat flux  $Q_\theta$  after Brown and Covey (1966) is used.  $Q_\theta$  is derived at each height

inside the canopy by means of the downward cumulative leaf area index (LAI):

$$Q_{\theta}(z) = Q_{\theta}(z_c) \exp(-\eta \text{LAI}), \quad (116)$$

with

$$\text{LAI} = \int_z^{z_c} \text{LAD} dz \quad (117)$$

where  $Q_{\theta}(z_c)$  is the prescribed heat flux at the top of the canopy layer  $z_c$  and  $\eta$  is the extinction coefficient set to 0.6. Additionally, contributions by sinks/sources for  $q$  and  $s$  are considered in the canopy model by adding additional terms  $C_{\varphi}$  to the scalar transport equations (see Eqs. 4–5):

$$\frac{\partial \varphi}{\partial t} = \dots - \underbrace{c_{\varphi} \text{LAD} \sqrt{u_i^2}}_{C_{\varphi}} (\varphi - \varphi_{c,0}), \quad (118)$$

where  $\varphi \in \{q, s\}$  and  $c_{\varphi}$  is a user-defined scalar exchange coefficient.  $\varphi_{c,0}$  and  $\varphi$  are the scalar concentrations at a leaf surface and in the surrounding air volume, respectively. This approach is based on the assumption that the scalar sink/source strength depends on the concentration gradient between the leaf surface and the surrounding air (e.g. Watanabe, 2004).

### 3.4.1 Recent applications

PALM simulations with the embedded canopy model were recently performed by Kanani et al. (2014c) to study the flow adjustment downstream of a transition from an un-forested (clearing) to a forested surface patch. In this study the LES results were validated against multidimensional field and wind-tunnel data. In the high-resolution follow-up study of Kanani-Sühring and Raasch (2015), a detailed analysis of the turbulent scalar transport

within the canopy layer was successfully performed for the first time by means of LES. [Figure 9 shows exemplarily the flow at a forest edge, where an internal boundary layer developed above the forest due to the extended drag of the canopy. See also associated animations in Kanani et al. \(2014a\) and Kanani et al. \(2014b\).](#)

### 3.5 1-D model for precursor runs

The initial profiles of the horizontal wind components in PALM can be prescribed by the user by piecewise linear gradients or by directly using observational data. Alternatively, a 1-D model can be employed to calculate stationary boundary-layer wind profiles. This is particularly useful in neutral stratification, where inertial oscillations can persist for several days in case that non-balanced profiles are used for initialization. By employing the embedded computationally inexpensive 1-D model with a Reynolds-average based turbulence parametrization, these oscillations can be significantly damped. A stationary state of the wind profiles can thus be provided much faster in the 3-D model. The arrays of the 3-D variables are then initialized with the (stationary) solution of the 1-D model. These variables are  $u_i$  where  $i \in \{1, 2\}$ ,  $e$ ,  $K_h$ ,  $K_m$  and, with MOST applied between the surface and the first vertical grid level, also  $Rif$ ,  $u_*$  as well as  $\overline{u_i''u_3''}$  (where  $i \in \{1, 2\}$ ).

The 1-D model assumes the profiles of  $\theta$  and  $q_v$ , as prescribed by the user, to be constant in time. The model solves the prognostic equations for  $u_i$  and  $e$ :

$$\frac{\partial u_i}{\partial t} = -\varepsilon_{i3j} f_3 u_j + \varepsilon_{i3j} f_3 u_{g,j} - \frac{\partial \overline{u_i'' u_3''}}{\partial x_3} \quad (119)$$

and

$$\frac{\partial e}{\partial t} = -\frac{\partial \overline{u'' w''}}{\partial z} - \frac{\partial \overline{v'' w''}}{\partial z} - \frac{g}{\theta} \frac{\partial \overline{w'' \theta''}}{\partial z} - \frac{\partial \overline{w'' e''}}{\partial z} - \epsilon. \quad (120)$$

The dissipation rate is parametrized by

$$\epsilon = 0.064 \frac{e^{\frac{3}{2}}}{l} \quad (121)$$

after Detering and Etling (1985). The mixing length is calculated after Blackadar (1997) as

$$l = \frac{\kappa z}{1 + \frac{\kappa z}{l_{\text{Bl}}}} \text{ with } l_{\text{Bl}} = 2.7 \times 10^{-4} \sqrt{u_g^2 + v_g^2}. \quad (122)$$

The turbulent fluxes are calculated using a 1st-order closure:

$$\overline{u_i'' u_3''} = -K_m \frac{\partial u_i}{\partial x_3}, \quad \overline{w'' \theta''} = -K_h \frac{\partial \theta}{\partial z}, \quad \overline{w'' e''} = -K_m \frac{\partial e}{\partial z}, \quad (123)$$

where  $K_m$  and  $K_h$  are calculated as

$$K_m = c_m \sqrt{e} \begin{cases} l & \text{for } Ri \geq 0 \\ l_{\text{Bl}} & \text{for } Ri < 0 \end{cases}, \quad (124)$$

$$K_h = \frac{\Psi_h}{\Psi_m} \frac{\Phi_h}{\Phi_m} K_m \quad (125)$$

with the similarity functions  $\Psi_h$  and  $\Psi_m$ ,  $\Phi_h$  and  $\Phi_m$  (see Eqs. 33 and 27, respectively), using the gradient Richardson number:

$$Ri = \frac{\frac{g}{\theta_v} \frac{\partial \theta}{\partial z}}{\left( \left( \frac{\partial u}{\partial z} \right)^2 + \left( \frac{\partial v}{\partial z} \right)^2 \right)} \frac{\frac{g}{\theta_v} \frac{\partial \theta}{\partial z}}{\left[ \left( \frac{\partial u}{\partial z} \right)^2 + \left( \frac{\partial v}{\partial z} \right)^2 \right]} \cdot \begin{cases} 1 & \text{for } Ri \geq 0, \\ (1 - 16 \cdot Ri)^{\frac{1}{4}} & \text{for } Ri < 0. \end{cases} \quad (126)$$

Note that the distinction of cases in Eq. (126) is done with the value of  $Ri$  from the previous

Moreover, a Rayleigh damping can be switched on to speed up the damping of inertial oscillations. The 1-D model is discretized in space using finite differences. Discretization in time is achieved using the 3rd-order Runge–Kutta time-stepping scheme (Williamson, 1980). Dirichlet boundary conditions are used at the top and bottom boundaries of the model, except for  $e$ , for which Neumann conditions are set at the surface (see also Sect. 2.5).

## 4 Technical realization

The model has been developed to run on Unix platforms. The PALM code is written according to the Fortran standard and split into several source code files. In the following Sect. 4.1 we will give a condensed overview of the general code structure and program flow. The embedded LPM requires a special data structure, which has been recently changed, in order to handle ~~immense numbers~~ billions of particles. We will thus devote Sect. 4.2 to this new particle structure.

The PALM code is optimized for use on massively parallel architectures using the Message Passing Interface (MPI, e.g. Gropp et al., 1999) and Open Multiprocessing (OpenMP)<sup>3</sup> (see Sect. 4.4).

The model steering is achieved by Fortran `NAMELIST` parameter lists that have to be provided by the user. The model operation will be described in detail in Sect. 4.6. The code also offers an interface that can be used to add user code extensions, and which will be described in detail in Sect. 4.5. Data handling in PALM (see Sect. 4.7) is mainly based on the Network Common Data Form (`netCDF`)<sup>4</sup>. Restart data are written in Fortran binary format. Finally, Sect. 6 deals with the code management.

---

<sup>3</sup><http://www.openmp.org>

<sup>4</sup><http://www.unidata.ucar.edu/software/netcdf>

## 4.1 General code structure

The PALM source code layout follows similar coding standards that have been developed for other community models like NEMO ocean dynamics model<sup>5</sup>. Special emphasis is given on providing extensive comment sections within the code in order to illustrate the functionality of specific model parts.

The source code is subdivided into a series of Fortran files. Most of them contain single subroutines only. These are called from the main PALM routine (`pal_m.f90`) and wherever needed. Each file features a header, containing a description and its history of modifications. The data handling between the subroutines is usually realized via Fortran modules defined in a separate file (`modules.f90`) instead of using parameter lists. The code contains several machine dependent segments, e.g., calls of routines from external libraries such as MPI, netCDF and FFTW<sup>6</sup>, and which may not be available on some machines. These segments are activated/deactivated using C-preprocessor directives, which allow to compile alternative parts of the code.

Three-dimensional arrays of prognostic variables ( $u = u$ ,  $v = v$ ,  $w = w$ ,  $\theta = \text{pt}$ ,  $q_v = q$ ,  $s = s$ ,  $e = e$  and  $Sa = sa$ ) are stored at the last two time levels of the Runge–Kutta substeps. These arrays are declared as (e.g., the  $u$ -wind component)  $u(k, j, i)$  on the respective subdomain of each processor, including ghost point layers ( $nb_{gp} = 3$  by default) for data exchange between the processors (see also Sect. 4.4):

```
u(nzb:nzt+1, nysg:nyng, nxlg:nxrg)
```

with `nzb` and `nzt` being the domain bounds of the bottom and top of the model. The lateral subdomain bounds (including ghost layers) are given by

<sup>5</sup>[http://www.nemo-ocean.eu/content/download/250/1629/file/coding\\_rules\\_OPA9.pdf](http://www.nemo-ocean.eu/content/download/250/1629/file/coding_rules_OPA9.pdf)

<sup>6</sup><http://www.fftw.org>

```
nysg = nys - nbgp,  
nyng = nyn + nbgp,  
nxlg = nxl - nbgp,  
nxrg = nxr + nbgp,
```

with `nys`, `nyn`, `nxl`, and `nxr` being the true subdomain bounds in south, north, west and east direction, respectively. For optimization, most of the 3-D variables are declared as pointers, e.g., for  $u$  and  $v$ :

```
REAL, DIMENSION(:, :, :), POINTER :: u, v
```

which speeds up the swapping of time levels after each time step, as it is not required to move the data in the memory.

A condensed overview of the program flow of PALM is shown in Fig. 10. At the beginning of the model run (hereafter referred to as “job”), the model setup is read from a Fortran `NAMELIST` file that is provided by the user, and optionally additional files for large-scale forcing and topography. PALM allows for conducting so-called restart jobs and job chains, where long-lasting model runs can be split into smaller ones. This does not only meet the requirements of most supercomputing systems, it also provides the user the opportunity to modify the setup between runs, or e.g., performing a set of parameter studies based on the same precursor run. For job chains, the current state of the model is saved as binary data at the end of the run and read as input for the subsequent restart run. After model initialization, possibly using a 1-D model precursor run (see Sect. 3.5), the time integration loop is executed until a termination is initiated. The latter might be caused by either the fact, that the desired simulation time has been reached, or by the need to initiate a restart of the job chain. The latter can be the case when the current job is running out of CPU time, or when the user has manually forced a restart. PALM can be used on cache-optimized as well as on vector processors. Moreover, General Purpose Computing on Graphics Processing Units (GPGPU) can be used. Each machine architecture requires specially optimized code to be executed within computationally expensive loops of the prognostic equations. This

is realized by a Fortran `INTERFACE` so that different code branches are executed in the `prognostic_equations.f90` subroutine.

In most cases, the large computational grid with very large number of grid points does not allow for processing the raw model data in a post-processing step, because then the input/output (I/O) time and the required hard disc space would easily exceed the available resources. Therefore, PALM calculates many standard quantities (e.g., variances, turbulent fluxes, and even higher order moments) online during the run. Also, temporal averages of vertical profiles, cross-sections, and 3-D data can be created this way. The user interface allows the user to easily extent this output (see Sect. 4.5).

After each time step it is checked whether data output (see also Sect. 4.7) is required, depending on the user settings.

## 4.2 Particle code structure

This section will give a brief summary of the particle code structure and the changes carried out for PALM 4.0. These changes are aiming at reaching a significantly improved performance of the LPM in comparison to the previous versions described by Steinfeld et al. (2008) and Riechelmann et al. (2012).

Each particle is defined by its features, which are stored as components of a Fortran 95 derived data type (e.g. Metcalf et al., 2004, Chap. 2.9):

```
TYPE particle_type
    REAL :: x, y, z, radius, age, ...
END TYPE particle_type
```

Here, `x`, `y`, `z`, `radius` and `age` are some components of the derived data type of the intrinsic data type `REAL`. Several other components of all intrinsic data types (or even other derived data types) can be defined (e.g., location, velocity). In general, the particles are stored in an allocatable array of the derived data type



TYPE(`particle_type`), DIMENSION(:), ALLOCATABLE :: `particles`  
 An element of `particles` defines a complete particle with its entire features, which can be accessed by the selector %, e.g., the radius and age of the particles by

```
particles(n)%radius
```

and

```
particles(n)%age,
```

respectively, where `n` is the index of a certain particle. In the old PALM version, all particles of the respective subdomain were stored in such a 1-D array.

Since many quantities derived from the LPM depend solely on the particles located in a certain grid volume, e.g., the collision and coalescence process of the LCM (see Sect. 3.3.2), the order in which these particles are stored in memory determines heavily the CPU time for the LPM. In general,  $N^2$  operations, where  $N$  is the number of all simulated particles, are needed to identify the particles located in the vicinity of another particle (see Riechelmann et al., 2012). In the previous versions of the LPM, this amount of operations was reduced to  $N$  by sorting the particles according to the grid volumes in which they are located. However, due to the large number of  $\mathcal{O}(10^6)$  particles stored, sorting was inefficient and also demanded a temporary array of the same size during sorting.

Therefore, from PALM 4.0 on, all particles are stored in a new array-structure based on another derived data type named `particle_grid_type`, which contains, as a component, a 1-D array of the derived data type `particle_type`:

```
TYPE particle_grid_type
```

```
    TYPE(particle_type), DIMENSION(:),           &
        ALLOCATABLE :: particles
```

```
END TYPE particle_grid_type
```

Note that the individual particle features are still accessible as components of `particles`. An allocatable three-dimensional array of `particle_grid_type` is defined

```
TYPE(particle_grid_type), DIMENSION(:, :, :), &  
    ALLOCATABLE :: particle_grid
```

and allocated using the same dimensions as used for a scalar of the LES model. In this way, all particles currently located in a certain LES grid volume are permanently stored in the particle array, assigned to this grid volume:

```
particle_grid(k, j, i)%particles(1:n_par)
```

Here, `n_par` is the number of particles located in the grid volume defined by the indices `k`, `j`, and `i`. The small size of this particle array at each grid volume (typically containing  $\mathcal{O}(10^2)$  particles) allows the de-allocation and allocation of the particle array during the simulation adapting its size to the number of required particles. This was (practically) not possible in the previous version of the LPM due to the large size of the full particle array ( $\mathcal{O}(10^6)$  particles), which required a temporary array of the same size during re-allocation. A temporary array is still required in the present version, but its size could be reduced by four orders of magnitude. However, as a particle moves from one grid volume to another, its data has to be copied from the 1-D array of the previous grid volume to the 1-D array of the new volume, and finally deleted from previous one, which consumes CPU time itself. Overall, the new particle structure reduces the CPU time of the LPM by 9%, since sorting of particles is not required anymore. Moreover, large temporary arrays are no longer required, which increases the available memory by almost a factor of two (which doubles the hypothetical amount of allocatable particles for future studies).

From PALM 4.0 on, the LPM features an optimized version of the tri-linear interpolation of LES data fields on the location of the particle. In general, the particles located in a certain grid volume are stored in an arbitrary order. Because of the staggered grid, indices of the eight surrounding grid points required for interpolation may differ from particle to particle (e.g., a particle in the lower left corner of a scalar grid box requires other data for interpolation than a particle in the upper right corner). This would require to re-calculate the respective index values for every new particle. By dividing every grid volume in eight subgrid boxes, two in every spatial direction, the same set of LES data can be used for all

particles located in the same subgrid box (see example in Fig. 11). Therefore, the particles belonging to the same subgrid box are stored contiguously in memory reducing the CPU time substantially for the different subroutines depending on the interpolation of LES fields substantially (e.g., advection by 64 %, condensational growth by 50 %, whole LPM by 22 %), whereas the time needed for additional sorting increases the CPU time by only 3 %.

In summary, these optimizations reduce the CPU time of the LPM by 40 % and almost halve its memory demand. For simulations with hundreds of millions of particles, the LPM consumes more than 95 % of the overall CPU time of PALM and the memory demand of the particles is the limiting factor for these simulations (see high-end applications, e.g. Riechelmann et al., 2012; Lee et al., 2014; Sühling et al., 2015). The present version of the LPM now allows for larger amounts of particles.

### 4.3 Topography implementation

The topography implementation described in Sect. 2.5.4 allows the use of 2-D topography height data in PALM. Currently, the topography data has to be provided within a rastered ASCII file. After reading and mapping of these data to the horizontal grid in PALM, they can be directly incorporated into the standard loop structure of the Fortran code as lower vertical index for all integration loops. Therefore, PALM employs two 2-D height index arrays (e.g., `nzb_w_inner(j, i)` and `nzb_w_outer(j, i)`) for the velocity component  $w$  to separate the domain into four regions based on the vertical index  $k$  (see Fig. 4):

- A.  $0 \leq k < \text{nzb\_w\_inner}$ , grid points within obstacles or in the ground that are excluded from calculations,
- B.  $\text{nzb\_w\_inner} \leq k < \text{nzb\_w\_outer}$ , grid points next to vertical walls, where wall-bounded code is executed,
- C.  $k = \text{nzb\_w\_inner} = \text{nzb\_w\_outer}$ , grid points next to horizontal walls, where wall-bounded code is executed,
- D. all other  $k$ , grid points in free fluid.

The additional topography code is executed in regions B and C only. As the velocity components are defined on a different (staggered) grid than the scalar quantities (see Fig. 2), three extra pairs of 2-D height index arrays are defined; two for the horizontal velocities and one for scalar quantities (e.g., `nzb_s_inner` and `nzb_s_outer` for scalar quantities).

#### 4.4 Parallelization and optimization

The parallelization of the code is achieved by a 2-D domain decomposition method along the  $x$  and  $y$  direction with equally sized subdomains. The method has not been changed in general since the formulation given by Raasch and Schröter (2001). In the following we will show that this method still allows for sufficient scalability on up to 50 000 processor cores (also referred to as processor elements, PEs). Ghost layers are added at the side boundaries of the subdomains in order to account for the local data dependencies, which are caused by the need to compute finite differences at these positions. The number of ghost layers that are used in PALM depend on the order of the advection scheme, with three layers for the 5th-order Wicker–Skamarock scheme and one layer for the 2nd-order Piacsek–Williams scheme. Ghost layer data are exchanged after every time step. An anti-cyclic index order (i.e.,  $(k, j, i)$ ) is chosen for the 3-D arrays in order to speed up the data exchange. The anti-cyclic order guarantees that the ghost layer data are stored as long consecutive blocks in the memory, which allows to access them in the fastest way.

The solution of the Poisson equation is complicated by the 2-D decomposition, because non-local data dependencies appear in all three directions, if the equation is solved with the FFT-method (see Sect. 2.4). Due to the domain decomposition, the processor elements cannot perform standard FFTs along  $x$  or  $y$  direction as their memory contains only a part of the full data. The general method to overcome the problem is to re-order the 3-D-pressure/divergence data among the PEs using a transposition technique described in Raasch and Schröter (2001). The transposition is done using the MPI-routine `MPI_ALLTOALL` and requires an additional re-sorting of the data in the local memory of the PEs before and after `MPI_ALLTOALL` is called. A similar method with `MPI_ALLTOALL`, replaced by `MPI_SENDRECV`, has been recently presented by Sullivan and Patton (2011).

Only local data dependencies appear if the Poisson equation is solved with the multigrid scheme. However, this method requires frequent exchange of ghost layers during every iteration step of the SOR-solver, as well as for the restriction and prolongation step. The amount of ghost layer data rapidly decreases for the coarser grid levels, so that the MPI-transfer time may become latency bound. The domain decomposition effects the coarsening of grid levels at the point, when the subdomain array of the current grid level contains only a single grid point along one of the spatial directions. In case that the number of grid points of the total (coarsened) domain allows further coarsening, array data from all subdomains are gathered and further processed on the main PE (hereafter PE 0), and results are redistributed to the other PEs in the respective prolongation step. However, this method is very inefficient and not used by default. Instead, coarsening is just stopped at that level, where subdomains contain only two grid points along at least one of the three spatial directions. The precision of the multigrid method depends on the iteration count. Using two W-cycles and two SOR iterations for each grid level typically reduces the velocity divergence by about 4 orders of magnitude, which turned out to be sufficient in most of our applications. With these settings, and for a numerical grid of about  $2000^3$  gridpoints, the multigrid method requires about the same time as the FFT Poisson-solver, and for larger grids it is even faster than the FFT solver.

The scaling behavior of PALM 4.0 is presented in Fig. 12a for a test case with  $2160^3$  grid points and the FFT Poisson solver. Tests have been performed on the [Cray-XG30](#) [Cray-XC40](#) of the North-German Computing Alliance (HLRN). The machine has 1128 compute nodes, each equipped with two 12-core Intel-Haswell CPUs, plus 744 compute nodes equipped with two 12-core Intel-Ivy Bridge CPUs, and an Aries-interconnect. Additionally, runs with  $4320^3$  grid points have been carried out with up to 43200 cores, starting with a minimum of 11520 cores (see Fig. 12b). Runs with less cores could not be carried out as the data would not have fit into the memory.

Ideally, for a so-called strong scaling test, where the same setup is run on different numbers of cores, the wallclock time of a run should decrease by a factor of two if the core number is doubled, which is shown in Fig. 12a and b (black lines). Figure 12b shows that

the code scales very well up to 20000 cores and still acceptable for larger numbers (gray line). The decreasing scalability for larger core numbers is mainly caused by a performance drop of the `MPI_ALLTOALL` routine (brown line). In contrast, the pure computational part, i.e., the calculation of the prognostic equations (red line), scales up perfectly to the maximum number of cores.

While the general parallelization methods used in version 4.0 do not differ from the first version, a large number of code optimizations have been carried out since then. Only the most important ones shall be briefly discussed at this point, namely the scalar optimization for different processor types; and overlapping of computation and inter-processor communication.

The original PALM code calculated the different contributions to the tendency terms (i.e., advection, buoyancy, diffusion, etc.) and the final prognostic equation for each prognostic quantity in separate 3-D-loops over the three spatial directions, ~~like~~

e.g. In case of large 3-D-arrays that do not fit into the cache of cache based processors like Intel-Xeon or AMD-Athlon, the array data has to be reloaded from the main memory for each 3-D-loop, which is extremely time consuming. For this reason, the outer loops over  $i$  and  $j$  have been extracted from each 3-D-loop, now forming a 2-D-loop over all tendencies and prognostic equations:

e.g.

```
DO  i = nx1, nxr
  DO  j = nys, nyn

      DO  k = nzb+1, nzt
!--      advection term
          tend(k, j, i) = ...
      ENDDO

      DO  k = nzb+1, nzt
!--      diffusion term
```

```
tend(k, j, i) = tend(k, j, i) + ...  
ENDDO
```

```
... ! further tendencies
```

```
ENDDO  
ENDDO
```

In this way, array data used in the first loop can be re-used from the cache by the following loops, since the size of 1-D-data columns along  $k$  is usually small enough to fit completely into the cache. Figure 12a shows that this loop-structure gives a performance gain for the computation of the prognostic equations of 40% compared with the 3-D loop structure. The overall performance of the code improves by about 15%. Nonetheless, both methods are implemented in the code in separate branches, since the 3-D loops give a much better performance on vector based hardware like NEC-SX or accelerator boards (e.g., Nvidia K20), since they allow the compilers to generate much longer vectors than for the single loops along the  $z$ -direction.

From Fig. 12b it is evident that for large-size setups with huge number of grid points and more than a few thousand of PEs, the solution of the Poisson equation dominates the time consumption of the simulation. This is because the FFT and the data transpositions with `MPI_ALLTOALL` scale less well than the other parts of the code. The FFT time increases ~~nonlinear~~ nonlinearly with  $N \log(N)$ , where  $N$  is the total number of grid points along  $x$  or  $y$ . The `MPI_ALLTOALL` time also increases ~~nonlinear~~ nonlinearly with the number of cores. While the scaling problem is not easy to address, the Poisson solver can be ~~speed~~ sped up by overlapping the computation (FFT and tri-diagonal equation solver) and the communication among the PEs (`MPI_ALLTOALL`). PALM solves the Poisson equation in different steps in a sequential order. So far, first, the complete 3-D data subdomain array is transposed, followed by the FFT along  $x$ , followed by the next transposition, followed by the FFT along  $y$ , etc. The FFT calculation cannot start unless the complete 3-D array is transposed. Now, in PALM 4.0, the 3-D arrays are processed in 2-D slices (e.g. in  $z$ - $x$  slices

for the transposition from  $z$  to  $x$ ). The slices are processed in a loop along the remaining direction (which is  $y$  in this case) with alternating transfer (`MPI_ALLTOALL` of a 2-D slice) and computation (FFT of this slice). This allows some overlapping of the computation and communication parts because of the following reason: on the Cray XC30 system mentioned above, for example, every compute node is populated with two processor dies, containing an Intel CPU with 12 cores each. This allows 24 MPI processes on the node. However, every node is equipped with two MPI channels only. If a data transfer is issued on all MPI processes simultaneously, the transfer cannot be done totally in parallel because individual MPI processes have to wait for the free channel. This behavior allows computation and transfer in parallel. For example, if PE 0 is the first to get a free MPI channel, it can start computation as soon as the transfer has been finished. All other PEs consecutively start computation after transfer. When the last transfer is finished, PE 0 has already finished computation and can immediately start the next transfer. The fact that not all PEs have simultaneous access to an MPI channel allows for parallel transfer and computation without any extra programming effort, such as asynchronous `MPI_ALLTOALL` or doing the transfer using hyperthreads.

Breaking up the workload as described above also improves performance due to better cache utilization, because the transposed data are still in the cache when needed by the FFT. The difference in performance between the sequential and the overlapping method are displayed in Fig. 12a. The FFT-Poisson-solver is ~~speed~~ speed up about 15% for up to 2500 cores in case that overlapping is used. For higher core numbers, the current realization of overlapping becomes inefficient because the data chunks handled by `MPI_ALLTOALL` get too small. The latency thus dominates the transfer time. In order to overcome this problem, several 2-D slices can be transposed at the same time. We will implement this technique in one of the next PALM revisions in the near future.

Beside the parallelization by domain decomposition, PALM is also fully parallelized on loop-level using the shared-memory OpenMP programming model and can be run in so-called hybrid mode, e.g., with two MPI processes and 12 OpenMP threads per MPI process started on each node.



A typical PALM setup uses 2-D domain decomposition with one MPI process on every processor core. The hybrid mode normally does not give advantages, because the OpenMP parallelization creates another synchronization level so that the total computation time will not decrease (typically it even increases by a few percent). Anyhow, for the following special cases hybrid parallelization may have some advantages:

- with many processor cores per CPU, a 1-D domain decomposition plus OpenMP parallelization may show better performance because the number of transpositions is reduced from 6 to 2,
- since the hybrid mode enlarges the subdomain sizes (because of less MPI processes), load imbalance problems caused by e.g. inhomogeneously distributed buildings or clouds may be reduced, because larger subdomains provide a better chance to get about the same number of buildings/clouds per subdomain,
- for the multigrid Poisson solver the hybrid mode allows to generate more grid levels on the subdomains because of their larger size, which may help to improve the solver convergence.

Since the speedup behavior depends on many factors like the problem size, the virtual 2-D processor grid, the network, etc., the actual speedup is difficult to predict and should be tested individually for every setup.

The data exchange in case of coupled ocean–atmosphere runs is realized with MPI. There are two methods available. The first one, based on MPI-1, splits the default communicator (`MPI_COMM_WORLD`) into two parts, with the respective number of PEs assigned to the atmosphere and ocean part as given by external parameters. The second method starts the respective number of MPI tasks for the atmosphere and the ocean independently (e.g., by two calls of `mpiexec`) and is using MPI-2 routines `MPI_COMM_CONNECT` and `MPI_COMM_ACCEPT` to couple them.

## 4.5 User interface

PALM offers a flexible interface that allows for adding user-specific calculations and code extensions. Also, the data output of user-defined quantities, such as 2-D/3-D data as well as time series, vertical profiles and spectra can be accomplished in a convenient manner. The implementation of such user-defined code is realized in the form of subroutine calls, which are made at several places in the model code. These subroutines have predefined names. Some of the entry points for the subroutine calls are shown in Fig. 10. Their basic versions are a part of the default model code and labeled as `user_***.f90`. These basic versions perform no actions and thus act as pure templates. For example, the subroutine `user_init.f90` reads

```
SUBROUTINE user_init

  USE control_parameters
  USE user

  IMPLICIT NONE

  !
  !-- Here the user defined initialization
  !-- follow:

END SUBROUTINE user_init
```

and can be extended according to the needs of the user.

By default, quantities in the time series and horizontally averaged vertical profile data output always refer to the total model domain (see also Sect. 4.7). The user interface, however, allows for defining up to 9 user-defined (horizontal) subdomains for which the output of time series and profiles is automatically added to the output data. Besides the output of profiles and time series for user-defined horizontal domains, PALM offers a very

flexible masked data output, controlled by a set of `NAMELIST` parameters. This feature allows to output quantities at different mask locations, e.g., 3-D volume data or 2-D cross sections of arbitrary extension within the model domain, 0-D or 1-D data at any positions and of any amount.

## 4.6 Model operation

The compilation and execution of PALM is controlled via a Unix shell scripts named `mbuild` and `mrun`, using `bash/ksh` syntax. `mbuild` compiles the default code using the Unix `make` mechanism. Compiler options, including C-preprocessor directives and required library paths (e.g., for `netCDF` or `FFT`), are given in a configuration file (default name `.mrun.config`). The configuration file allows for setting different compilers and options in separate blocks. The compiled source code (object files) is stored in a so-called depository folder (one folder for each option block). `mrun` takes care of the compilation (main program and user interface files only) and job submission/execution of PALM, including the handling of I/O files. The `mrun` command has a number of options to control the program execution. The execution is also controlled by the configuration file, which provides machine- and user-specific settings such as compiler options and library paths (see above), and I/O file locations. Basically, `mrun` performs the following tasks in sequential order:

1. create a unique temporary working directory for the job,
2. copy input files and user-defined code required for the job to the temporary directory,
3. copy pre-compiled PALM routines to the temporary directory,
4. compile the main program using the precompiled object files and the user code,
5. execute the program,
6. copy the model output files from the temporary directory to a directory specified by the user,

## 7. delete the temporary working directory.

Since each job runs in a unique temporary directory (see task 1), several jobs can run at the same time without interfering each other. The I/O files are handled (tasks 3 and 6) via so-called file connection statements, which allow to manage these files in a flexible way and to keep them in a well organized folder structure. A typical file connection statement for an input file reads

```
PARIN in d3# ~/palm/current_version/JOBS/INPUT _p3d
```

where the first column gives the local filename in the temporary working directory that must correspond to the filename in the `OPEN` statement in the `PALM` source code. The second column provides a file attribute (where `in` means that it is an input file), and the third column is the activating string that defines whether this file connection statement is carried out in the respective job. The fourth column gives the folder name where the permanent (input) file is provided by the user. Finally, the sixth column gives the suffix of the permanent file.

The full name of the permanent file results from the folder name, the suffix, and the value of the `mrunc` option `-d`, which defines the so-called basename of all files handled by `mrunc`, e.g., the `mrunc` call

```
mrunc -d example_cbl -r "d3#"...
```

defines the filename to be

```
~/palm/current_version/JOBS/INPUT/example_cbl_p3d
```

and which will be copied to `PARIN` in the temporary working directory (task 2) due to the setting of the activation string with the option `-r`. Besides, it is possible to organize jobs using the string `$fname` in the folder name column of the connection statement:

```
PARIN in d3# ~/palm/current_version/JOBS/$fname/INPUT _p3d
```

Here, the value of `$fname` is given by the `-d` option during the `mrunc` call (here `example_cbl`) and all job files can be stored accordingly.

The `mr` script never replaces or overwrites existing files. New so-called cycle numbers are created instead. For example, the file `example_cb1_d3d` (3-D data) has been created within a first model run. Then a second call of `mr` and subsequent model execution will create a new file, named `example_cb1_d3d.1`, etc.

While some I/O operates on single files only (e.g., output of `cpu` measurements), other data (e.g. restart data) I/O is done by each core separately. In such cases, filenames provided by the file connection statements are interpreted as directory names. Each core then opens a file, named `#####` in the respective directory, where the hashes stand for a six digit integer, declaring the rank of the MPI process in the MPI communicator in PALM.

Each simulation setup can be complemented by a separate set of user interface routines that replace the template files in the default code at compile time (see task 2). In this way, PALM executables will be dynamically created for each setup, based on the same default code, but with unique user code extensions. This also has the advantage, that it is generally possible to update the PALM version without the need of adapting own user-defined code. User interface routines for different setups can be stored in different folders, which are accessed by `mr` using the basename mechanism as for I/O file described above.

At the beginning of task 4, various checks are performed on the parameter files and the provided user interface. Therewith, illegal model settings are trapped and reported to the user with a unique error message identifier. Moreover, several runtime and netCDF errors are captured by PALM in this way. A comprehensive online database provides additional information on each error message identifier (see Sect. 6).

Furthermore, `mr` can be used to generate batch jobs on local and remote hosts, and it also controls the automatic generation of restart jobs/job chains. For convenience, an optional graphical user interface has been developed as a wrapper for `mr`, called `mrGUI`, providing an intuitive access to the `mr` script (see Fig. 13).

## 4.7 Data handling

Due to the enormous amount of data that comes along with computationally expensive LES, the data handling plays a key role for the performance of LES models and for data analysis

during post-processing. PALM is optimized to pursue the strategy of performing data operations to great extent online during the simulation instead of postpone these operations to the post-processing. In this way, the data output (e.g., of huge 4-D data, or temporal averages) can be significantly reduced. In order to allow the user to perform own calculations during runtime, the user interface offers a wide range of possibilities, e.g., for defining user-defined output quantities (see Sect. 4.5).

PALM allows data output for different quantities as time series, (horizontally-averaged) vertical profiles, 2-D cross sections, 3-D volume data, and masked data (see Sect. 4.5). All data output files are in netCDF format, which can be processed by different public domain and commercial software. NetCDF data can also be easily read from Fortran programs, provided that a netCDF library is available. The netCDF libraries currently support three different binary formats for netCDF files: classic, 64-bit offset, and netCDF-4. The latter was introduced in netCDF version 4.0 and is based on the HDF5<sup>7</sup> data format. PALM is able to handle all three netCDF formats and also supports parallel I/O for netCDF-4.

For visualization of the netCDF data generated by PALM, several NCAR Command Language (NCL)<sup>8</sup> scripts are available that allow a quick overview of the simulation data. For advanced visualizations, we have developed a tool that converts PALM data into the `vdF` data format of the Open Source software VAPOR<sup>9</sup> (Clyne et al., 2007). Animations using PALM data and VAPOR have been recently published by Maronga et al. (2013a), Knoop et al. (2014), and Kanani et al. (2014a, b).

## 4.8 Code management and regulations

~~The PALM code is freely available<sup>10</sup> and distributed under the GNU General Public License v3<sup>10</sup>. For code management, versioning and revision control the PALM group runs an~~

---

<sup>7</sup><http://www.hdfgroup.org/HDF5>

<sup>8</sup><http://www.ncl.ucar.edu>

<sup>9</sup><http://www.vapor.ucar.edu>

<sup>10</sup>~~The code can be downloaded at~~

<sup>10</sup>

Apache Subversion<sup>10</sup> (svn) server at IMUK. The PALM code can be downloaded via the svn server, which is also integrated in a web-based project management and bug-tracking system using the software Trac<sup>10</sup>. In this way, PALM users can use the web interface to browse through the code, view recent code modifications, and to submit bug reports via a ticketing system directly to the code developers. Furthermore, a model documentation, a detailed user manual as well as an online tutorial are available on the Trac server and are constantly kept up to date by the PALM group. Code updates and development is generally reserved to the PALM group in order to keep the code structure clean, consistent, and uniform. However, we encourage researchers to contact us for collaborative code development that might be suitable to enter the default PALM code. We also appreciate suggestions for future PALM developments.

## 5 PALM applications: status quo and future perspectives

### 5.1 Past and current research fields

PALM has been applied for numerous boundary layer research studies over the years. For example, coherent structures in the convective ABL have been simulated by Raasch and Franke (2011) (dust devil-like vortices), [see also visualization, Maronga et al., 2013a](#)) and under neutral conditions at a forest edge by Kanani et al. (2014c) and Kanani-Sühring and Raasch (2015) (using the canopy model). [Classic vertical profiles of temperature, humidity, fluxes, structure parameters, and variances, as well as horizontal cross-sections and turbulence spectra for the convective ABL were shown e.g. by Maronga and Raasch \(2013\) and Maronga et al. \(2013b\)](#). Moreover, Hellsten and Zilitinkevich (2013) used PALM to investigate the role of convective structures and background turbulence in the ABL. The model has been also applied for the stable boundary layer in the scope of an LES intercomparison (Beare et al., 2006).

---

<sup>10</sup>

<sup>10</sup>

The investigation of effects of land surface heterogeneity on the convective boundary layer has been one of the core areas of research with PALM. The early studies used idealized surface heterogeneity, i.e., stripes or checkerboard patterns (Raasch and Harbusch, 2001; Kim et al., 2004; Letzel and Raasch, 2003; Inagaki et al., 2006), whereas recent studies incorporated more complex surface configurations using the irregularly distributed land use classes as observed during the LITFASS-2003 field experiment (see Beyrich and Mengelkamp, 2006; Maronga and Raasch, 2013; Sühling and Raasch, 2013; Maronga et al., 2014; Sühling et al., 2015). Moreover, PALM has been applied to study the flow over arctic ice leads and during cold-air outbreaks (e.g. Lüpkes et al., 2008; Gryschka et al., 2008, 2014). PALM has also been used several times to evaluate in situ measurement systems and strategies, e.g., for acoustic tomography, eddy covariance measurements, airborne flux observations, and scintillometers (e.g. Weinbrecht et al., 2004; Kanda et al., 2004; Sühling and Raasch, 2013; Maronga et al., 2013b). Steinfeld et al. (2008), Markkanen et al. (2010), and Sühling et al. (2015) used the embedded LPM to determine accurate footprint estimations for tower and eddy covariance measurements.

The possibility of using Cartesian topography as surface boundary condition (see Sect. 2.5.4) has facilitated the simulation of the urban boundary layer and studying the flow around buildings ([first validation in Letzel et al., 2008](#)) ([first validation against wind tunnel data from Martinuzzi and Tropea \(1993\) in Letzel et al. \(2008\)](#)). The research fields ranged from development of better urban parametrization schemes to the investigation of the ventilation at pedestrian level in densely built-up cities. The flow around street canyons and idealized buildings has been subject of several studies (e.g. Inagaki et al., 2011; Abd Razak et al., 2013; Park et al., 2012; Park and Baik, 2013; Yaghoobian et al., 2014). With increasing computational resources, it also has become possible to use PALM to simulate entire city quarters (Letzel et al., 2012; Kanda et al., 2013). Lately, PALM has been also used to simulate the entire city of Macau, ~~China~~, with a model domain of about  $6 \text{ km} \times 5 \text{ km}$  and a [grid-resolution-fine spacing](#) of only 1 m (see Keck et al., 2012; Knoop et al., 2014). In addition to applications for the urban boundary layer, the topography option was used to study atmospheric Kármán vortex streets (Heinze et al., 2012).



Investigations on cloud-topped boundary layers have been another core area of the research with PALM. Cloudy boundary layers have been simulated using bulk cloud microphysics [for cold-air outbreaks](#) by Gryschka et al. (2008) as well as [for the analysis of 2nd-order budgets in cloud-topped boundary layers for BOMEX and DYCOMS-II](#)<sup>10</sup> (see [Stevens et al., 2005](#)) [experiments by](#) Heinze et al. (2015). Recently, the embedded LCM has been employed for studying the effect of turbulence on the droplet dynamics and growth (Lee et al., 2014; Riechelmann et al., 2015), and for investigating the entrainment (of aerosols) at the edges of cumulus clouds (Hoffmann et al., 2015, 2014).

[Furthermore, PALM has been applied to simulations of wind turbine and wind farm wakes \(e.g. Witha et al., 2014; Dörenkämper et al., 2015\)](#).

Finally, PALM has been used to study several aspects of the OML (e.g. Noh et al., 2003, 2009, 2011; Wakata, 2011), and recently to investigate the feedback between atmospheric and oceanic turbulence (Esau, 2014).

## 5.2 Current and future developments

The most serious model modification in the near future will be related to the change from the currently used incompressible conservation equations (see Sect. 2.1), to an anelastic approximation. In the anelastic equations the density of the fluid can vary with height, so that it will be possible to study both shallow and deep convection and therefore will also allow a better representation of larger-scale processes. This change will be accompanied by adding the ice phase for clouds. In the long term, we plan to add the option of a fully compressible code. On the one hand, this will render the pressure solver unnecessary and thus help to overcome the transposition bottleneck for core numbers  $> 100\,000$  (see, e.g., Fig. 12a). On the other hand, the high propagation velocity of sound waves requires the implementation of a time-splitting algorithm with a considerably smaller time step than in the incompressible system (see e.g. Wicker and Skamarock, 2002).

---

<sup>10</sup>[The Second Dynamics and Chemistry of Marine Stratocumulus field study](#)

The research group of Matthias Mauder at Karlsruhe Institute of Technology is currently working on a vertical grid nesting of PALM similar to Sullivan et al. (1986). The self-nesting will allow using very high grid resolutions within the atmospheric surface layer, and relatively low resolutions above, which would reduce the computational load for investigations of the surface layer by up to 90%. After sufficient validity checks, the nesting technique will be incorporated in the default code. [Also, a full 3-D two-way self-nesting of PALM is currently developed by Antti Hellsten at the Finnish Meteorological Institute.](#)

[Furthermore, wind turbine parametrizations developed by the research group of Detlev Heinemann at University of Oldenburg will be included in the default code in the near future.](#)

The rapid radiative transfer model ([RRTM, Clough et al., 2005](#)) ([for global models](#)) ([RRTMG, Clough et al., 2005](#)) has been recently coupled ~~successfully~~ to PALM to allow a better representation of radiative effects in clouds and during nighttime. ~~We intend to implement the RRTM model into the default code in an upcoming PALM release.~~ In order to allow feedback between radiative effects and the surface/soil, a land surface model (LSM) ~~implementation is currently under way and a beta version has been already~~ [has already been](#) implemented. The LSM is a modified version of the Tiled European Centre for Medium-Range Weather Forecasts Scheme for Surface Exchanges over Land (TESSEL, van den Hurk et al., 2000; Balsamo et al., 2009) and the derivative implemented in DALES. It consists of a solver for the energy balance and a four layer soil scheme, taking into account soil properties and vegetation. [Both the RRTMG implementation and the LSM will be part of the next PALM release.](#)

[In order to allow for a sufficient representation of SGS turbulence when using relatively coarse meshes, we intend to implement the dynamic Smagorinsky turbulence closure model after Esau \(2004\).](#)

Finally, we plan to add an option to use viscous topography for urban LES and complex terrain [after Mason and Sykes \(1978\)](#), where topography in PALM will be represented by grid volumes with infinite viscosity instead of using solid elements. Unlike the present im-

plementation, where grid volumes can either be set to topography or fluid, sloping surfaces will be better represented by adjusting the viscosity of the respective grid volumes.

### 5.3 Future perspectives

At the moment, LES remains a pure research tool, which can be used to tackle fundamental and initial research questions, and that often requires the world's largest supercomputers. In the mid term (next 5–10 years), however, further increasing capacities of supercomputers and alternative hardware, such as multiple GPUs and the Intel MIC coprocessor computer architecture, might alter the situation.

At present we are porting the PALM code to use it on multiple Nvidia GPUs. Instead of using GPU programming models such as OpenCL<sup>11</sup> or CUDA<sup>12</sup>, which requires re-writing of the existing code (see e.g. Schalkwijk et al., 2012, which have ported the DALES code), we have chosen the new OpenACC<sup>13</sup> programming model. Like OpenMP, OpenACC is based on directives which are placed, e.g., in front of loops and interpreted as comment lines by standard compilers. This allows us to use the same code on any kind of hardware, avoiding redundant model development in completely different branches. In order to minimize the time consuming data transfer between the host (CPU) memory and the device (GPU) memory, almost the complete PALM code is run on the GPU and data are only transferred for I/O purposes. PALM 4.0 is able to run on a single GPU, but only some basic PALM features have been ported so far (FFT Poisson solver, dry prognostic equations). This version has been selected to be part of the SPECACCEL benchmark<sup>14</sup>. Multiple-GPU usage is currently implemented using so-called CUDA-aware MPI implementations, which allow to send data from the GPU-memory directly to the network adapter without staging through the host memory.

---

<sup>11</sup><https://www.khronos.org/opencv>

<sup>12</sup>[http://www.nvidia.com/object/cuda\\_home\\_new.html](http://www.nvidia.com/object/cuda_home_new.html)

<sup>13</sup><http://www.openacc-standard.org/>

<sup>14</sup><http://www.spec.org/accel>

Within the foreseeable future, the LES technique will become a rewarding alternative for operational forecasts, particularly of local near-surface high-risk conditions such as strong wind gust, dense fog, or pollutant dispersion in urban environments. End-users will be airport operators, city planners, and consultants that currently rely on information from mesoscale models. LES might also be employed for improving the nowcast of convective shower cells. Moreover, the site assessment, which usually involves long-term measurements, is currently a major expense factor during planning of wind parks. The usage of LES might shorten this procedure significantly and thus reduce costs. In order to enable such applications, however, it will be necessary to achieve a highly-optimized parallelization of the model. This is particularly true as the number of processors of supercomputer clusters will further increase.

While past LES research has mainly focused on the convective and neutral boundary layer, we observe increasing interest in the stable regime (e.g. Stoll and Porté-Agel, 2008; Zhou and Chow, 2014). This trend will continue in the future and allow more rigorous investigations of the stable and very stable boundary layer, where the largest eddies are only of size of a few meters. This trend is surely linked to increasing computational power and hence the possibility of using fine enough grid ~~resolutions~~ spacings in LES of 2 m and below. Also, the transition periods in the afternoon and early morning will be object of future research (e.g. Edwards et al., 2014). The optimization and scalability of PALM and future developments like the vertical self-nesting of PALM or the multiple GPU adaptation that will be available in the near future will support the usage of PALM for such applications.

## 6 Code availability

The PALM code is freely-available<sup>15</sup> and distributed under the GNU General Public License v3<sup>16</sup>. For code management, versioning and revision control the PALM group runs an

<sup>15</sup>The code can be downloaded at <http://palm.muk.uni-hannover.de>

<sup>16</sup><http://www.gnu.org/copyleft/gpl.html>

Apache Subversion<sup>17</sup> (svn) server at IMUK. The PALM code can be downloaded via the svn server, which is also integrated in a web-based project management and bug-tracking system using the software Trac<sup>18</sup>. In this way, PALM users can use the web interface to browse through the code, view recent code modifications, and to submit bug reports via a ticketing system directly to the code developers. Furthermore, a model documentation, a detailed user manual as well as an online tutorial are available on the Trac server and are constantly kept up to date by the PALM group. Code updates and development is generally reserved to the PALM group in order to keep the code structure clean, consistent, and uniform. However, we encourage researchers to contact us for collaborative code development that might be suitable to enter the default PALM code. We also appreciate suggestions for future PALM developments.

## 7 Summary

In this technical overview paper, we described the current version 4.0 of the well established LES model PALM that has been developed at Leibniz Universität Hannover, Germany. PALM has been successfully applied over the last 15 years for a variety of boundary layer research questions related to both turbulence in the atmospheric and oceanic boundary layer. The aim of this paper was to create a detailed, yet condensed, reference work of the technical realization and special features of PALM.

It was shown that the model is highly optimized for use on massively parallel computer architectures, showing a high performance on up to 50000 processor cores. Owing to the high scalability, the model is suitable for carrying out computationally expensive simulations for large domain and very high grid resolutions. Moreover, PALM features embedded models, namely a LPM/LCM for simulating passive particles as well as explicit cloud droplets using the concept of super-droplets. Alternatively, a two-moment microphysics scheme is implemented for studying boundary layer clouds. A simple canopy model allows for study-

---

<sup>17</sup><http://subversion.apache.org>

<sup>18</sup><http://trac.edgewall.org>

ing the effect of vegetation on the boundary layer. Furthermore, a Cartesian topography is implemented that is most useful for simulations of the urban boundary layer. A surface coupling can be used that allows to resolve feedback processes between the atmospheric and oceanic versions of PALM.

Furthermore, we gave an overview of the technical realization. This included the general Fortran code structure, the structure of the Lagrangian particles, which require special treatment, as well as parallelization and optimization on supercomputer clusters and novel hardware and techniques such as GPGPU.

We also described planned model developments, such as the change to an anelastic approximation that will allow to simulate deep convection, to include ~~a vertical~~ self-nesting of the model, and a full coupling of PALM with land surface and radiation models.

Finally, we would like to encourage interested researchers in both the atmospheric and oceanic boundary layer community to try out PALM. The model can be freely downloaded from <http://palm.muk.uni-hannover.de> and used under the GNU GPL. The PALM web page does not only provide the model code and a full documentation, it also offers an extensive tutorial section allowing a quick introduction to the model usage.

*Acknowledgements.* First of all we would like to thank all previous and current code developers for their contributions, such as Jens Fricke, Michael Schröter, Gerald Steinfeld, and Jörg Uhlenbrock (in alphabetical order). Furthermore we thank all work groups around the world that use PALM and thus support our model development, namely the groups of Jong-Jin Baik (Seoul National University, Korea), Igor Esau (Nansen Environmental and Remote Sensing Center, Norway), [Detlev Heinemann \(University of Oldenburg, Germany\)](#), Antti Hellsten (Finnish Meteorological Institute, Finland), Manabu Kanda and Atsushi Inagaki (Tokyo Institute of Technology, Japan), Edward Ng (Chinese University of Hong Kong, China), Yign Noh (Yonsei University, Korea), Jun Tanimoto and Aya Hagishima (Kyushu University, Japan) and Jochen Reuder (University of Bergen, Norway). Moreover, the PALM group would like to thank Gerd Brandt and Gabriel Gaus at HLRN for their technical support. Most of the model test runs during the development phases of PALM have been performed at the supercomputers of HLRN, which is gratefully acknowledged. The PALM code development has been partly funded by the German Research Foundation (DFG) under grants RA 617/3, RA 617/6, RA 617/16, and RA 617/27-1. [NCL \(The NCAR Command Language \(Version 5.2.1\) \[Software\]](#),

[2010, Boulder, CO:UCAR/NCAR/CISL/VETS, doi:10.5065/D6WD3XH5](#) and VAPOR<sup>19</sup> were used for data visualization of Figures 5, 6, 8, and 9. We acknowledge support by DFG and Open Access Publishing Fund of Leibniz Universität Hannover. We appreciate the anonymous reviewers for their valuable comments that helped to improve the manuscript.

## References

- Abd Razak, A., Hagishima, A., Ikegaya, N., and Tanimoto, J.: Analysis of airflow over building arrays for assessment of urban wind environment, *Build. Environ.*, 59, 56–65, 2013.
- Ackerman, A. S., vanZanten, M. C., Stevens, B., Savic-Jovicic, V., Bretherton, C. S., Chlond, A., Golaz, J.-C., Jiang, H., Khairoutdinov, M., Krueger, S. K., Lewellen, D. C., Lock, A., Moeng, C.-H., Nakamura, K., Petters, M. D., Snider, J. R., Weinbrecht, S., and Zuluaga, M.: Large-eddy simulations of a drizzling, stratocumulus-topped marine boundary layer, *Mon. Weather Rev.*, 137, 1083–1110, 2009.
- Andrejczuk, M., Reisner, J. M., Henson, B., Dubey, M. K., and Jeffery, C. A.: The potential impacts of pollution on a nondrizzling stratus deck: Does aerosol number matter more than type?, *J. Geophys. Res.*, 113, D19204, doi:10.1029/2007JD009445, 2008.
- Arakawa, A. and Lamb, V. R.: Computational design of the basic dynamical processes of the UCLA general circulation model, in: 1977: *General Circulation Models of the Atmosphere*, Methods in Computational Physics, edited by: Chang, J., 17, Berlin, 173–265, 1977.
- Ayala, O., Rosa, B., and Wang, L.-P.: Effects of turbulence on the geometric collision rate of sedimenting droplets. Part 2. Theory and parameterization, *New J. Phys.*, 10, 075016, doi:10.1088/1367-2630/10/7/075016, 2008.
- Balsamo, G., Vitebo, P., Beljaars, A., van den Hurk, B., Hirschi, M., Betts, A. K., and Scipal, K.: A revised hydrology for the ECMWF model: Verification from field site to terrestrial water storage and impact in the Integrated Forecast System, *J. Hydrometeorol.*, 10, 623–643, 2009.
- Beare, R. J., Cortes, M. A. J., Cuxart, J., Esau, I., Golaz, C., Holtslag, A. A. M., Khairoutdinov, M., Kosovic, B., Lewellen, D., Lund, T., Lundquist, J., McCabe, A., Macvean, M. K., Moene, A., Noh, Y., Poulos, G., Raasch, S., and Sullivan, P.: An intercomparison of large-eddy simulations of the stable boundary layer, *Bound.-Lay. Meteorol.*, 118, 247–272, 2006.

---

<sup>19</sup>[www.vapor.ucar.edu](http://www.vapor.ucar.edu)

- Beyrich, F. and Mengelkamp, H.-T.: Evaporation over a heterogeneous land surface: EVA\_GRIPS and the LITFASS-2003 experiment: an overview, *Bound.-Lay. Meteorol.*, 121, 5–32, 2006.
- Blackadar, A. K.: *Turbulence and Diffusion in the Atmosphere*, Springer, Berlin, Heidelberg, New York, 185 pp., 1997.
- Bougeault, P.: Modeling the trade-wind cumulus boundary layer. Part I: Testing the ensemble cloud relations against numerical data, *J. Atmos. Sci.*, 38, 2414–2428, 1981.
- Briscolini, M. and Santangelo, P.: Development of the mask method for incompressible unsteady flows, *J. Comput. Phys.*, 84, 57–75, 1989.
- Brown, K. W. and Covey, W.: The energy-budget evaluation of the micro-meteorological transfer process within a cornfield, *Agr. Meteorol.*, 3, 73–96, 1966.
- Cassiani, M., Katul, G. G., and Albertson, J. D.: The effects of canopy leaf area index on airflow across forest edges: large-eddy simulation and analytical results, *Bound.-Lay. Meteorol.*, 126, 433–460, 2008.
- Cescatti, A. and Marcolla, B.: Drag coefficient and turbulence intensity in conifer canopies, *Agr. Forest Meteorol.*, 121, 197–206, 2004.
- Clough, S. A., Shephard, M. W., Mlawer, E. J., Delamere, J. S., Iacono, M. J., Cady-Pereira, K., Boukabara, S., and Brown, P. D.: Atmospheric radiative transfer modeling: a summary of the AER codes, Short Communication, *J. Quant. Spectrosc. Ra.*, 91, 233–244, 2005.
- Clyne, J., Mininni, P., Norton, A., and Rast, M.: Interactive desktop analysis of high resolution simulations: application to turbulent plume dynamics and current sheet formation, *New. J. Phys.*, 301, 1–28, 2007.
- Cuipers, J. W. M. and Duynkerke, P. G.: Large eddy simulation of trade wind cumulus clouds, *J. Atmos. Sci.*, 50, 3894–3908, 1993.
- Davies, H. C.: A lateral boundary formulation for multi-level prediction models, *Q. J. Roy. Meteor. Soc.*, 102, 405–418, 1976.
- Deardorff, J. W.: The use of subgrid transport equations in a three-dimensional model of atmospheric turbulence, *J. Fluid. Eng.-T. ASME*, 95, 429–438, 1973.
- Deardorff, J. W.: Three-dimensional numerical study of the height and mean structure of a heated planetary boundary layer, *Bound.-Lay. Meteorol.*, 7, 81–106, 1974.
- Deardorff, J. W.: Stratocumulus-capped mixed layers derived from a three-dimensional model, *Bound.-Lay. Meteorol.*, 18, 495–527, 1980.
- Detering, H. W. and Etling, D.: Application of the  $E$ - $\epsilon$  turbulence model to the atmospheric boundary layer, *Bound.-Lay. Meteorol.*, 33, 113–133, 1985.



- [Dipankar, A., Stevens, B., Heinze, R., Moseley, C., Zängl, G., Giorgetta, M. and Brdar, D.: Large eddy simulation using the general circulation model ICON, \*J. Adv. Mod. Earth Syst.\*, \*\*07\*\*, doi:10.1002/2015MS000431, 2015.](#)
- [Dörenkämper, M., Witha, B., Steinfeld, G., Heinemann, D. and Kühn, M.: The impact of stable atmospheric boundary layers on wind-turbine wakes within offshore wind farms, \*J. Wind Eng. Ind. Aerodyn.\*, doi:10.1016/j.jweia.2014.12.011, 2015.](#)
- Dupont, S. and Brunet, Y.: Coherent structures in canopy edge flow: a large-eddy simulation study, *J. Fluid Mech.*, 630, 93–128, 2009.
- Edwards, J. M., Basu, S., Bosveld, F. C., and Holtslag, A. A. M.: The impact of radiation on the GABLS3 large-eddy simulation through the night and during the morning transition, *Bound.-Lay. Meteorol.*, 152, 189–211, 2014.
- Emanuel, K. A.: *Atmospheric Convection*, Oxford University Press, 1994.
- Esau, I.: Indirect air-sea interactions simulated with a coupled turbulence-resolving model, *Ocean Dynam.*, 64, 689–705, doi:10.1007/s10236-014-0712-y, 2014.
- [Esau, I.: Simulation of Ekman Boundary Layers by Large Eddy Model with Dynamic Mixed Subfilter Closure, \*Env. Fluid. Mech.\*, \*\*4\*\*, 273–303, 2004.](#)
- Finnigan, J. J., Shaw, R. H., and Patton, E. G.: Turbulence structure above a vegetation canopy, *J. Fluid Mech.*, 637, 387–424, 2009.
- Frigo, M. and Johnson, S. G.: FFTW: an adaptive software architecture for the FFT, in: Proc. of the 1998 IEEE International Conference on Acoustics, Speech and Signal Processing, 381–1384, 1998.
- Geoffroy, O., Brenguier, J.-L., and Burnet, F.: Parametric representation of the cloud droplet spectra for LES warm bulk microphysical schemes, *Atmos. Chem. Phys.*, 10, 4835–4848, doi:10.5194/acp-10-4835-2010, 2010.
- Grabowski, W. M., Andrejczuk, M., and Wang, L.-P.: Droplet growth in a bin warm-rain scheme with Twomey CCN activation, *Atmos. Res.*, 99, 290–301, 2011.
- Gropp, W., Lusk, E., and Skjellum, A.: *Using MPI: Portable Parallel Programming with the Message Passing Interface*, 2nd Edn., MIT Press, Cambridge, MA, 1999.
- Gryschka, M. and Raasch, S.: Roll convection during a cold air outbreak: a large-eddy simulation with stationary model domain, *Geophys. Res. Lett.*, 32, L14805, doi:10.1029/2005GL022872, 2005.

- Gryschka, M., Drüe, C., Etling, D., and Raasch, S.: On the influence of sea-ice inhomogeneities onto roll convection in cold-air outbreaks, *Geophys. Res. Lett.*, 35, L23804, doi:10.1029/2008GL035845, 2008.
- Gryschka, M., Fricke, J., and Raasch, S.: On the impact of forced roll convection on vertical turbulent transport in cold-air outbreaks, *J. Geophys. Res.*, 119, 12513–12532, doi:10.1002/2014JD022160, 2014.
- Hackbusch, W.: *Multigrid Methods and Applications*, Springer, Berlin, Heidelberg, New York, 378 pp., 1985.
- Hall, W. D.: A detailed microphysical model within a two-dimensional dynamic framework: model description and preliminary results, *J. Atmos. Sci.*, 37, 2486–2507, 1980.
- Harlow, F. H. and Welch, J. E.: Numerical calculation of time-dependent viscous incompressible flow of fluid with free surface, *Phys. Fluids*, 8, 2182–2189, 1965.
- Heinze, R., Raasch, S., and Etling, D.: The structure of Karman vortex streets in the atmospheric boundary layer derived from large eddy simulation, *Meteorol. Z.*, 21, 221–237, 2012.
- Heinze, R., Mironov, D., and Raasch, S.: Second-moment budgets in cloud-topped boundary layers: a large-eddy simulation study, *J. Adv. Model. Earth Syst.*, ~~under review~~, [7](#), doi:10.1002/2014MS000376, 2015.
- Hellsten, A. and Zilitinkevich, S.: Role of convective structures and background turbulence in the dry convective boundary layer, *Bound.-Lay. Meteorol.*, 149, 323–353, 2013.
- Heus, T., van Heerwaarden, C. C., Jonker, H. J. J., Pier Siebesma, A., 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 overview of its applications, *Geosci. Model Dev.*, 3, 415–444, doi:10.5194/gmd-3-415-2010, 2010.
- Hoffmann, F., Raasch, S., and Noh, Y.: Entrainment of aerosols and their activation in a shallow cumulus cloud studied with a coupled LCM-LES approach, *Atmos. Res.*, ~~submitted, 2014~~, [156](#), [43–57](#), doi:10.1016/j.atmosres.2014.12.008, [2015](#).
- Hoffmann, F., Siebert, H., Schumacher, J., Riechermann, T., Katzwinkel, J., Kumar, B., Götzfried, P., and Raasch, S.: Entrainment and mixing at the interface of shallow cumulus clouds: results from a combination of observations and simulations, *Meteorol. Z.*, 23, 349–368, doi:10.1127/0941-2948/2014/0597, ~~2014~~, [2014](#).
- Inagaki, A., Letzel, M. O., Raasch, S., and Kanda, M.: Impact of surface heterogeneity on energy imbalance: A study using LES, *J. Meteorol. Soc. Jpn.*, 84, 187–198, 2006.

- Inagaki, A., Castillo, M., Yamashita, Y., Kanda, M., and Takimoto, H.: Large-eddy simulation of coherent flow structures within a cubical canopy, *Bound.-Lay. Meteorol.*, 142, 207–222, 2011.
- Jackett, D. R., McDougall, T. J., Feistel, R., Wright, D. G., and Griffies, S. M.: Algorithms for density, potential temperature, conservative temperature, and the freezing temperature of seawater, *J. Atmos. Ocean. Tech.*, 23, 1709–1728, 2006.
- Kanani**
- [Kanani-Sühring](#), F. and Raasch, S.: Spatial variability of scalar concentrations and fluxes downstream of a clearing-to-forest transition: a large-eddy simulation study, *Bound.-Lay. Meteorol.*, [155](#), 1–27, doi:10.1007/s10546-014-9986-3, ~~2014~~, [2015](#).
- Kanani, F., Maronga, B., Knoop, H., and Raasch, S.: Large-eddy simulation of a forest-edge flow – adjustment of a turbulent flow to the changing surface conditions at a clearing-to-forest transition, *Computer animation*, doi:10.5446/14311, 2014a.
- Kanani, F., Maronga, B., Knoop, H., and Raasch, S.: Large-eddy simulation of the scalar transport in a forest-edge flow – spatial variability of the scalar distribution and the scalar transport downstream of a clearing-to-forest transition, *Computer animation*, doi:10.5446/14368, 2014b.
- Kanani, F., Träumner, K., Ruck, B., and Raasch, S.: What determines the differences found in forest edge flow between physical models and atmospheric measurements? – an LES study, *Meteorol. Z.*, 23, 33–49, 2014c.
- Kanda, M., Inagaki, A., Letzel, M. O., Raasch, S., and Watanabe, T.: LES study of the energy imbalance problem with eddy covariance fluxes, *Bound.-Lay. Meteorol.*, 110, 381–404, 2004.
- Kanda, M., Inagaki, A., Miyamoto, T., Gryschka, M., and Raasch, S.: A new aerodynamic parameterization for real urban surfaces, *Bound.-Lay. Meteorol.*, 148, 357–377, 2013.
- Kataoka, H. and Mizuno, M.: Numerical flow computation around aerolastic 3d square cylinder using inflow turbulence, *Wind Struct.*, 5, 379–392, 2002.
- Keck, M., Raasch, S., Letzel, M. O., Ng, E., and Ren, C.: High resolution large-eddy simulations of the urban canopy flow in Macau, *First International Education Forum on Energy and Environment, Hawaii's Big Island USA*, 2012.
- Kim, H.-J., Noh, Y., and Raasch, S.: Interaction between wind and temperature fields under the heterogeneous heat flux in the planetary boundary layer, *Bound.-Lay. Meteorol.*, 111, 225–246, 2004.
- Klemp, J. B. and Lilly, D. K.: Numerical simulation of hydrostatic mountain waves, *J. Atmos. Sci.*, 35, 78–107, 1978.

- Knoop, H., Keck, M., and Raasch, S.: Urban large-eddy simulation – influence of a densely build-up artificial island on the turbulent flow in the city of Macau, *Computer animation*, doi:10.5446/14368, 2014.
- Lamb, R. G.: A numerical simulation of dispersion from an elevated point source in the convective planetary boundary layer, *Atmos. Environ.*, 12, 1297–1304, 1978.
- Lee, J. H., Noh, Y., Raasch, S., Riechelmann, T., and Wang, L.-P.: Investigation of droplet dynamics in a convective cloud using a Lagrangian cloud model, *Meteorol. Atmos. Phys.*, 124, 1–21, doi:10.1007/s00703-014-0311-y, 2014.
- Letzel, M. O. and Raasch, S.: Large eddy simulation of thermally induced oscillations in the convective boundary layer, *J. Atmos. Sci.*, 60, 2328–2341, 2003.
- Letzel, M. O., Krane, M., and Raasch, S.: High resolution urban large-eddy simulation studies from street canyon to neighbourhood scale, *Atmos. Environ.*, 42, 8770–8784, 2008.
- Letzel, M. O., Helmke, C., Ng, E., An, X., Lai, A., and Raasch, S.: LES case study on pedestrian level ventilation in two neighbourhoods in Hong Kong, *Meteorol. Z.*, 21, 575–589, 2012.
- Lilly, D. K.: The presentation of small-scale turbulence in numerical simulation experiments, in: *Proc. IBM scientific Computing Symp. on Environmental Sciences*, Thomas J. Watson Research Center, Yorktown Heights, NY, 195–210, 1967.
- Lund, T. S., Wu, X., and Squires, K. D.: Generation of turbulent inflow data for spatially-developing boundary layer simulations, *J. Comput. Phys.*, 140, 233–258, 1998.
- Lüpkes, C., Gryanik, V., Witha, B., Gryschka, M., Raasch, S., and Gollnik, T.: Modeling convection over arctic leads with LES and a non-eddy-resolving microscale model, *J. Geophys. Res.*, 113, c09028, doi:10.1029/2007JC004099, 2008.
- Markkanen, T., Steinfeld, G., Kljun, N., Raasch, S., and Foken, T.: Comparison of conventional Lagrangian stochastic footprint models against LES driven footprint estimates, *Atmos. Chem. Phys.*, 9, 5575–5586, doi:10.5194/acp-9-5575-2009, 2009.
- Markkanen, T., Steinfeld, G., Kljun, N., Raasch, S., and Foken, T.: A numerical case study on footprint model performance under inhomogeneous flow conditions, *Meteorol. Z.*, 19, 539–547, 2010.
- Maronga, B.: Monin-Obukhov similarity functions for the structure parameters of temperature and humidity in the unstable surface layer: results from high-resolution large-eddy simulations, *J. Atmos. Sci.*, 71, 716–733, 2014.
- Maronga, B. and Raasch, S.: Large-eddy simulations of surface heterogeneity effects on the convective boundary layer during the LITFASS-2003 experiment, *Bound.-Lay. Meteorol.*, 146, 17–44, 2013.

- Maronga, B., Hoffmann, F., Riechelmann, T., and Raasch, S.: Large-eddy simulation of dust devils: Animation of dust devils in the convective boundary layer using a virtual dust, *Computer animation*, doi:10.5446/9352, 2013a.
- Maronga, B., Moene, A. F., van Dinter, D., Raasch, S., Bosveld, F., and Gioli, B.: Derivation of structure parameters of temperature and humidity in the convective boundary layer from large-eddy simulations and implications for the interpretation of scintillometer observations, *Bound.-Lay. Meteorol.*, 148, 1–30, 2013b.
- Maronga, B., Hartogensis, O. K., Raasch, S., and Beyrich, F.: The effect of surface heterogeneity on the structure parameters of temperature and specific humidity: a large-eddy simulation case study for the LITFASS-2003 experiment, *Bound.-Lay. Meteorol.*, 153, 441–470, 2014.
- [Martinuzzi, R. and Tropea, C.: The flow around a surface-mounted, prismatic obstacle placed in a fully developed channel flow, \*J. Fluids Eng.\*, 115, 85–92, 1993.](#)
- [Mason, P. J.: Large-eddy simulation: A critical review of the technique, \*Q. J. Roy. Meteor. Soc.\*, 120, 1–26, 1994.](#)
- Mason, P. J.: Large-eddy simulation of the convective atmospheric boundary layer, *J. Atmos. Sci.*, 46, 1492–1516, 1989.
- [Mason, P. J., and Sykes, R. I.: A simple cartesian model of boundary layer flow over topography, \*J. Comput. Phys.\*, 28, 198–210, 1978.](#)
- Metcalfe, M., Reid, J. K., and Cohen, M.: *Fortran 95/2003 Explained*, vol. 416, Oxford University Press, Oxford, 2004.
- Miller, M. J. and Thorpe, A. J.: Radiation conditions for the lateral boundaries of limited-area numerical models, *Q. J. Roy. Meteor. Soc.*, 107, 615–628, 1981.
- Moeng, C.-H.: A large-eddy-simulation model for the study of planetary boundary-layer turbulence, *J. Atmos. Sci.*, 41, 2052–2062, 1984.
- Moeng, C.-H. and Wyngaard, J. C.: Spectral analysis of large-eddy simulations of the convective boundary layer, *J. Atmos. Sci.*, 45, 3573–3587, 1988.
- Neggels, R. A. J., Siebesma, A. P., and Heus, T.: Continuous single-column model evaluation at a permanent meteorological supersite, *B. Am. Meteorol. Soc.*, 29, 91–115, 2012.
- Noh, Y., Cheon, W. G., and Raasch, S.: The role of preconditioning in the evolution of open-ocean deep convection, *J. Phys. Oceanogr.*, 33, 1145–1166, 2003.
- Noh, Y., Min, H. S., and Raasch, S.: Large eddy simulation of the ocean mixed layer: the effects of wave breaking and Langmuir circulation, *J. Phys. Oceanogr.*, 34, 720–735, 2004.

- Noh, Y., Kang, I. S., Herold, M., and Raasch, S.: Large-eddy simulation of particle settling in the ocean mixed layer, *Phys. Fluids*, 18, 085109, doi:10.1063/1.2337098, 2006.
- Noh, Y., Goh, G., Raasch, S., and Gryschka, M.: Formation of a diurnal thermocline in the ocean mixed layer simulated by LES, *J. Phys. Oceanogr.*, 39, 1244–1257, 2009.
- Noh, Y., Goh, G., and Raasch, S.: Examination of the mixed layer deepening process during convection using LES, *J. Phys. Oceanogr.*, 40, 2189–2195, 2010.
- Noh, Y., Goh, G., and Raasch, S.: Influence of Langmuir circulation on the deepening of the wind-mixed layer, *J. Phys. Oceanogr.*, 41, 472–484, 2011.
- Orlanski, I.: A simple boundary condition for unbounded hyperbolic flows, *J. Comput. Phys.*, 21, 251–269, 1976.
- Panofsky, H. A. and Dutton, J. A.: *Atmospheric Turbulence, Models and Methods for Engineering Applications*, John Wiley & Sons, New York, 1984.
- Park, S. B. and Baik, J.: A large-eddy simulation study of thermal effects on turbulence coherent structures in and above a building array, *J. Appl. Meteorol.*, 52, 1348–1365, 2013.
- Park, S. B., Baik, J., Raasch, S., and Letzel, M. O.: A large-eddy simulation study of thermal effects on turbulent flow and dispersion in and above a street canyon, *J. Appl. Meteorol. Clim.*, 51, 829–841, 2012.
- Patrinos, A. N. A. and Kistler, A. L.: A numerical study of the Chicago lake breeze, *Bound.-Lay. Meteorol.*, 12, 93–123, 1977.
- Piacsek, S. A. and Williams, G. P.: Conservation properties of convection difference schemes, *J. Comput. Phys.*, 198, 580–616, 1970.
- Press, W. H., Teukolsky, S. A., Vetterling, W. T., and Flannery, B. P.: *Numerical Recipes in Fortran 90: the Art of Parallel Scientific Computing*, 2nd Edn., Cambridge University Press, Cambridge, 1996.
- Pruppacher, H. R. and Klett, J. D.: *Microphysics of Clouds and Precipitation*, 2nd Edn., Kluwer Academic Publishers, Dordrecht, 1997.
- Raasch, S. and Etling, D.: Numerical simulation of rotating turbulent thermal convection, *Beitr. Phys. Atmos.*, 64, 185–199, 1991.
- Raasch, S. and Franke, T.: Structure and formation of dust-devil-like vortices in the atmospheric boundary layer – a high resolution numerical study, *J. Geophys. Res.*, 116, D16120, doi:10.1029/2011JD016010, 2011.

- Raasch, S. and Harbusch, G.: An analysis of secondary circulations and their effects caused by small-scale surface inhomogeneities using large-eddy simulation, *Bound.-Lay. Meteorol.*, 101, 31–59, 2001.
- Raasch, S. and Schröter, M.: PALM – a large-eddy simulation model performing on massively parallel computers, *Meteorol. Z.*, 10, 363–372, 2001.
- Raupach, M. R., Finnigan, J. J., and Brunet, Y.: Coherent eddies and turbulence in vegetation canopies: the mixing-layer analogy, *Bound.-Lay. Meteorol.*, 78, 351–382, 1996.
- Riechelmann, T., Noh, Y., and Raasch, S.: A new method for large-eddy simulations of clouds with Lagrangian droplets including the effects of turbulent collision, *New J. Phys.*, 14, 065008, doi:10.1088/1367-2630/14/6/065008, 2012.
- Riechelmann, T., Wacker, U., Beheng, K. D., Etling, D., and Raasch, S.: Influence of turbulence on the drip growth in warm clouds, part II: Sensitivity studies with a spectral bin microphysics and a lagrangian cloud model, *Meteorol. Z.*, submitted, 2015.
- Rodean, H. C.: Stochastic Lagrangian models of turbulent diffusion, *Meteor. Mon.*, 26, 1–84, doi:10.1175/0065-9401-26.48.1, 1996.
- Rogers, R. R. and Yau, M. K.: A short course in cloud physics, Pergamon Press, New York, 1989.
- Rogers, R. R., Baumgardner, D., Ethier, S. A., Carter, D. A., and Ecklund, W. L.: Comparison of raindrop size distributions measured by radar wind profiler and by airplane, *J. Appl. Meteorol.*, 32, 694–699, 1993.
- Saiki, E. M., Moeng, C.-H., and Sullivan, P. P.: Large-eddy simulation of the stably stratified planetary boundary layer, *Bound.-Lay. Meteorol.*, 95, 1–30, 2000.
- Savic-Jovcic, V. and Stevens, B.: The structure and mesoscale organization of precipitating stratocumulus, *J. Atmos. Sci.*, 65, 1587–1605, doi:10.1175/2007JAS2456.1, 2008.
- Schalkwijk, J., Griffith, E. J., Post, F. H., and Jonker, H. J. J.: High-performance simulations of turbulent clouds on a desktop PC, *B. Am. Meteorol. Soc.*, 93, 307–314, 2012.
- [Schumann, U. and Sweet, R. A.: Fast Fourier Transforms for Direct Solution of Poisson's Equation with Staggered Boundary Conditions , \*J. Comput. Phys.\*, 75, 123–137, 1988.](#)
- Schumann, U.: Subgrid scale model for finite difference simulations of turbulent flows in plane channels and annuli, *J. Comput. Phys.*, 18, 376–404, 1975.
- Seifert, A.: On the parameterization of evaporation of raindrops as simulated by a one-dimensional rainshaft model., *J. Atmos. Sci.*, 65, 3608–3619, doi:10.1175/2008JAS2586.1, 2008.
- Seifert, A. and Beheng, K. D.: A double-moment parameterization for simulating autoconversion, accretion and selfcollection, *Atmos. Res.*, 59, 265–281, 2001.

- Seifert, A. and Beheng, K. D.: A two-moment cloud microphysics parameterization for mixed-phase clouds. Part 1: Model description, *Meteorol. Atmos. Phys.*, 92, 45–66, 2006.
- Seifert, A., Nuijens, L., and Stevens, B.: Turbulence effects on warm-rain autoconversion in precipitating shallow convection, *Q. J. Roy. Meteor. Soc.*, 136, 1753–1762, 2010.
- Shaw, R. H. and Patton, E. G.: Canopy element influences on resolved- and subgrid-scale energy within a large-eddy simulation, *Agr. Forest Meteorol.*, 115, 5–17, 2003.
- Shaw, R. H. and Schumann, U.: Large-eddy simulation of turbulent flow above and within a forest, *Bound.-Lay. Meteorol.*, 61, 47–64, 1992.
- Shima, S.-I., Kusano, K., Kawano, A., Sugiyama, T., and Kawahara, S.: The super-droplet method for the numerical simulation of clouds and precipitation: a particle-based and probabilistic microphysics model coupled with a non-hydrostatic model, *Q. J. Roy. Meteor. Soc.*, 135, 1307–1320, 2009.
- [Siebesma, A. P., Bretherton, C. S., Brown, A., Chlond, A., Cuxart, J., Duynkerke, P. G., Jiang, H., Khairoutdinov, M., Lewellen, D., Moeng, C.-H., Sanchez, E., Stevens, B., and Stevens, D. E.: A large eddy simulation intercomparison study of shallow cumulus convection, \*J. Atmos. Sci.\*, 60, 1201–1219, 2003.](#)
- Singleton, R. C.: An algorithm for computing the mixed radix fast Fourier transform, *IEEE T. Acoust. Speech*, 17, 93–103, 1969.
- Smagorinsky, J.: General circulation experiments with the primitive equations, *Mon. Weather Rev.*, 91, 99–164, 1963.
- Sommeria, G. and Deardorff, J. W.: Subgrid-scale condensation in models of nonprecipitating clouds, *J. Atmos. Sci.*, 34, 344–355, 1977.
- Sölch, I. and Kärcher, B.: A large-eddy model for cirrus clouds with explicit aerosol and ice microphysics and Lagrangian ice particle tracking, *Q. J. Roy. Meteor. Soc.*, 136, 2074–2093, 2010.
- Sorbjan, Z.: A numerical study of daily transitions in the convective boundary layer, *Bound.-Lay. Meteorol.*, 123, 365–383, 2007.
- Steinfeld, G., Raasch, S., and Markkanen, T.: Footprints in homogeneously and heterogeneously driven boundary layers derived from a Lagrangian stochastic particle model embedded into large-eddy simulation, *Bound.-Lay. Meteorol.*, 129, 225–248, 2008.
- Steinhorn, I.: Salt flux and evaporation, *J. Phys. Oceanogr.*, 21, 1681–1683, 1991.
- Stevens, B. and Seifert, A.: Understanding macrophysical outcomes of microphysical choices in simulations of shallow cumulus convection, *J. Meteor. Soc. Jpn.*, 86, 143–162, 2008.



- [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. Weather Rev.\*, 133, 1443–1462, doi:10.1175/MWR2930.1, 2005, 1681–1683, 1991.](#)
- Stoll, R. and Porté-Agel, F.: Surface heterogeneity effects on regional-scale fluxes in stable boundary layers: surface temperature transitions, *J. Atmos. Sci.*, 66, 412–431, 2008.
- Stull, R. B.: *An Introduction to Boundary Layer Meteorology*, Kluwer Academic Publishers, Dordrecht, 666 pp., 1988.
- Sühring, M. and Raasch, S.: Heterogeneity-induced heat flux patterns in the convective boundary layer: can they be detected from observations and is there a blending height? – a large-eddy simulation study for the LITFASS-2003 experiment, *Bound.-Lay. Meteorol.*, 148, 309–331, 2013.
- Sühring, M., Maronga, B., Herbort, F., and Raasch, S.: On the effect of surface heat-flux heterogeneities on the mixed-layer top entrainment, *Bound.-Lay. Meteorol.*, 151, 531–556, 2014.
- Sühring, M., Kanani, F., Charuchittipan, D., Foken, T., and Raasch, S.: Footprint estimation for elevated turbulence measurements – a comparison between large-eddy simulation and a Lagrangian stochastic backward model, *Bound.-Lay. Meteorol.*, under review, 2015.
- Sullivan, P. E., McWilliams, J. C., and Moeng, C.-H.: A subgrid-scale model for large-eddy simulation of planetary boundary-layer flows, *Bound.-Lay. Meteorol.*, 71, 247–286, 1986.
- 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, 2395–2415, 2011.
- Sullivan, P. P., Moeng, C.-H., Stevens, B., Lenschow, D. H., and Mayor, S. D.: Structure of the entrainment zone capping the convective atmospheric boundary layer, *J. Atmos. Sci.*, 55, 3042–3064, 1998.
- Temperton, C.: A Generalized Prime Factor FFT Algorithm for Any  $N = (2^{**}P)(3^{**}Q)(5^{**}R)$ , *SIAM J. Sci. Stat. Comp.*, 13, 676–686, 1992.
- Thomson, D. J.: Criteria for the selection of stochastic models of particle trajectories in turbulent flows, *J. Fluid Mech.*, 180, 529–556, 1987.
- van den Hurk, B. J. J. M., Beljaars, A. C. M., and Betts, A. K.: Offline validation of the ERA-40 surface scheme, *Tech. Memo. 295, ECMWF*, 43 pp., 2000.
- Wakata, Y.: Dependence of seafloor boundary layer thickness on the overlying flow direction: a large eddy simulation study, *J. Oceanogr.*, 67, 667–673, 2011.

- Watanabe, T.: Large-eddy simulation of coherent turbulence structures associated with scalar ramps over plant canopies, *Bound.-Lay. Meteorol.*, 112, 207–341, 2004.
- Weil, J. C., Sullivan, P. P., and Moeng, C.-H.: The use of large-eddy simulations in Lagrangian particle dispersion models, *J. Atmos. Sci.*, 61, 2877–2887, 2004.
- Weinbrecht, S., Raasch, S., Ziemann, A., Arnold, K., and Raabe, A.: Comparison of large-eddy simulation data with spatially averaged measurements obtained by acoustic tomography – pre-suppositions and first results, *Bound.-Lay. Meteorol.*, 111, 441–465, 2004.
- Wicker, L. J. and Skamarock, W. C.: Time-splitting methods for elastic models using forward time schemes, *Mon. Weather Rev.*, 130, 2088–2097, 2002.
- Williamson, J. H.: Low-storage Runge–Kutta schemes, *J. Comput. Phys.*, 35, 48–56, 1980.
- Willis, G. E. and Deardorff, J. W.: A laboratory model of diffusion into the convective boundary layer, *Q. J. Roy. Meteorol. Soc.*, 102, 427–445, 1976.
- [Witha, B., Steinfeld, G., Dörenkämper, M. and Heinemann, D.: Large-eddy simulation of multiple wakes in offshore wind farms, \*J. Phys.: Conf. Ser.\*, 555, 012108, doi:10.1088/1742-6596/555/1/012108, 2014](#)
- Wyngaard, J. C., Peltier, L. J., and Khanna, S.: LES in the surface layer: surface fluxes, scaling, and SGS modeling, *J. Atmos. Sci.*, 55, 1733–1754, 1998.
- Yaghoobian, N., Kleissl, J., and Paw U, K. T.: An improved three-dimensional simulation of the diurnally varying street-canyon flow, *Bound.-Lay. Meteorol.*, 153, 251–276, doi:10.1007/s10546-014-9940-4, 2014.
- Yi, C.: Momentum transfer within canopies, *J. Appl. Meteorol.*, 47, 262–275, 2008.
- Zhou, B. and Chow, T. K.: Nested large-eddy simulations of the intermittently turbulent stable atmospheric boundary layer over real terrain, *J. Atmos. Sci.*, 71, 1021–1039, 2014.

**Table 1.** List of general model parameters.

Symbol	Value	Description
$c_m$	0.1	SGS model constant
$c_p$	$1005 \text{ J kg}^{-1} \text{ K}^{-1}$	Heat capacity of dry air at constant pressure
$g$	$9.81 \text{ m s}^{-2}$	Gravitational acceleration
$L_V$	$2.5 \times 10^6 \text{ J kg}^{-1}$	Latent heat of vaporization
$p_0$	1000 hPa	Reference air pressure
$R_d$	$287 \text{ J kg}^{-1} \text{ K}^{-1}$	Specific gas constant for dry air
$R_v$	$461.51 \text{ J kg}^{-1} \text{ K}^{-1}$	Specific gas constant for water vapor
$\kappa$	0.4	Kármán constant
$\rho$	$\text{kg m}^{-3}$	Density of dry air
$\rho_0$	$1.0 \text{ kg m}^{-3}$	Density of dry air at the surface
$\rho_{l,0}$	$1003 \text{ kg m}^{-3}$	Density of liquid water
$\Omega$	$0.729 \times 10^{-4} \text{ rad s}^{-1}$	Angular velocity of the Earth

**Table 2.** List of general symbols.

Symbol	Dimension	Description
$C_{\text{relax}}$	$\text{m}^{-1}$	Relaxation coefficient for laminar inflow
$D$	$\text{m}$	Length of relaxation area for laminar inflow
$d$	$\text{m}$	Distance to the inlet
$e$	$\text{m}^2 \text{s}^{-2}$	SGS-TKE
$F_{\text{inflow}}$	$\text{m}^{-1}$	Damping factor for laminar inflow
$f$	$\text{s}^{-1}$	Coriolis parameter
$K_h$	$\text{m}^2 \text{s}^{-1}$	SGS eddy diffusivity of heat
$K_m$	$\text{m}^2 \text{s}^{-1}$	SGS eddy diffusivity of momentum
$L$	$\text{m}$	Obukhov length
$l$	$\text{m}$	SGS mixing length
$l_{\text{BU}}$	$\text{m}$	Mixing length in the free atmosphere after Blackadar (1997)
$p$	$\text{hPa}$	Hydrostatic pressure
$p^*$	$\text{hPa}$	Perturbation pressure
$Q_{\theta}$	$\text{K m s}^{-1}$	Upward vertical kinematic heat flux
$q$	$\text{kg kg}^{-1}$	Total water content
$q_l$	$\text{kg kg}^{-1}$	Liquid water specific humidity
$q_v$	$\text{kg kg}^{-1}$	Specific humidity
$q_*$	$\text{kg kg}^{-1}$	MOST humidity scale
$Ri$		Bulk Richardson number
$s$	$\text{kg m}^{-3}$	Passive scalar
$U_{u_i}$	$\text{m s}^{-1}$	Transport velocity of the indexed velocity component at the outlet
$u_{g,i}$	$\text{m s}^{-1}$	Geostrophic wind components ( $u_{g,1} = u_g, u_{g,2} = v_g$ )
$u_i$	$\text{m s}^{-1}$	Velocity components ( $u_1 = u, u_2 = v, u_3 = w$ )
$u_{i,LS}$	$\text{m s}^{-1}$	Large-scale advection velocity components
$u_*$	$\text{m s}^{-1}$	Friction velocity
$x_i$	$\text{m}$	Coordinate on the Cartesian grid ( $x_1 = x, x_2 = y, x_3 = z$ )
$x_{\text{inlet}}$	$\text{m}$	Position of the inlet
$x_{\text{recycle}}$	$\text{m}$	Distance of the recycling plane from the inlet
$z_0$	$\text{m}$	Roughness length for momentum
$z_{0,h}$	$\text{m}$	Roughness length for heat
$z_{\text{MO}}$	$\text{m}$	Height of the constant flux layer (MOST)
$\alpha$		Angle between the $x$ -direction and the wind direction
$\Delta$	$\text{m}$	Nominal grid spacing
$\Delta$		Difference operator
$\Delta x, \Delta y, \Delta z$	$\text{m}$	Grid spacings in $x, y, z$ direction
$\Delta t$	$\text{s}$	Time step of the LES model
$\delta$		Kronecker-delta
$\epsilon$		Levi-Civita symbol
$\epsilon$	$\text{m}^2 \text{s}^{-3}$	SGS-TKE dissipation rate
$\theta$	$\text{K}$	Potential temperature
$\theta_{\text{zflow}}$	$\text{K}$	Laminar inflow profile of $\theta$
$\theta_l$	$\text{K}$	Liquid water potential temperature
$\theta_v$	$\text{K}$	Virtual potential temperature
$\theta_*$	$\text{K}$	MOST temperature scale
$\Pi$		Exner function
$\tau_{\text{LS}}$	$\text{s}$	Relaxation time scale for nudging
$\Phi_h$		Similarity function for heat
$\Phi_m$		Similarity function for momentum
$\varphi$		A prognostic variable ( $u, v, w, \theta/\theta_s, q_v/q_s, s, e$ )
$\varphi_{\text{LS}}$		Large-scale value of $\varphi$
$\Psi_{q_v}$	$\text{kg kg}^{-1} \text{s}^{-1}$	Source/sink term of $q_v$
$\Psi_s$	$\text{kg m}^{-3} \text{s}^{-1}$	Source/sink term of $s$

**Table 3.** List of ocean model parameters.

Symbol	Dimension/Value	Description
$c_{p,l}$	4218 J kg <sup>-1</sup> K <sup>-1</sup>	Heat capacity of water at constant pressure
Sa	PSU	Salinity
$\rho_\theta$	kg m <sup>-3</sup>	Potential density
$\Phi_{Sa}$ $\Psi_{Sa}$	PSU s <sup>-1</sup>	Source/sink term of Sa

**Table 4.** List of cloud physics parameters and symbols.

Symbol	Dimension/Value	Description
$A$		Particle weighting factor
$F_{q_c}$	$\text{kg m}^{-3} \text{m s}^{-1}$	Cloud water sedimentation flux
$F_{\text{VH}}$		Van't Hoff factor
$f_v$		Ventilation factor
$K$	$\text{m}^3 \text{s}$	Collision kernel
$K_{\text{accr}}$	$4.33 \text{ m}^3 \text{kg}^{-1} \text{s}^{-1}$	Accretion kernel
$K_{\text{auto}}$	$9.44 \times 10^9 \text{ m}^3 \text{kg}^{-2} \text{s}^{-1}$	Autoconversion kernel
$K_{\text{break}}$	$2000 \text{ m}^{-1}$	Breakup kernel
$K_{\text{self}}$	$7.12 \text{ m}^3 \text{kg}^{-1} \text{s}^{-1}$	Selfcollection kernel
$K_v$	$2.3 \times 10^{-5} \text{ m}^2 \text{s}^{-1}$	Molecular diffusivity of vapor in air
$M_l$	$18.01528 \text{ g mol}^{-1}$	Molar mass of water
$M_s$	$\text{g mol}^{-1}$	Molar mass of aerosol
$m$	$\text{kg}$	Mass of Lagrangian particle
$m_c$	$\text{kg}$	Volume-averaged droplet mass
$m_s$	$\text{kg}$	Mass of aerosol
$m_{\text{sep}}$	$2.6 \times 10^{-10} \text{ kg}$	Separation droplet mass
$N_c$	$\text{m}^{-3}$	Cloud droplet number concentration
$N_r$	$\text{m}^{-3}$	Rain drop number concentration
$p_{v,s}$	$\text{Pa}$	Saturation water vapor pressure
$q_c$	$\text{kg kg}^{-1}$	Cloud water specific humidity
$q_r$	$\text{kg kg}^{-1}$	Rain water specific humidity
$q_s$	$\text{kg kg}^{-1}$	Water saturation specific humidity
$Re_p$		Particle Reynolds number
$r$	$\text{m}$	Particle radius
$r_{\text{eq}}$	$550 \times 10^{-6} \text{ m}$	Breakup equilibrium radius
$\tilde{r}_r$	$\text{m}$	Volume-averaged rain drop radius
$S$		Water supersaturation
$S_{\text{eq}}$		Equilibrium saturation term
$T$	$\text{K}$	Actual temperature
$T_l$	$\text{K}$	Liquid water temperature
$w_{N_r}$	$\text{m s}^{-1}$	Rain water sedimentation velocity
$w_{q_c}$	$\text{m s}^{-1}$	Rain conc. sedimentation velocity
$\beta$		Coefficient for the approximation of $q_s$
$\Gamma$		Gamma function
$\gamma$	0.7	Constant for evaporation
$\Delta V$	$\text{m}^3$	Grid volume
$\vartheta$	$\text{kg s}^{-2}$	Surface tension
$\lambda_h$	$2.43 \times 10^{-2} \text{ W m}^{-1} \text{K}^{-1}$	Heat conductivity of air
$\lambda_r$	$\text{m}$	Rain drop slope parameter
$\mu_c$	1	Cloud droplet shape parameter
$\mu_r$	$\text{m}$	Rain drop shape parameter
$\nu$	$1.461 \times 10^{-5} \text{ m}^2 \text{s}^{-1}$	Molecular viscosity of air
$\tau_c$		Dimensionless cloud time-scale
$\Phi_{\text{accr}}$		Accretion similarity function
$\Phi_{\text{auto}}$		Autoconversion similarity function
$\Phi_{\text{break}}$		Breakup similarity function
$\Psi_{N_r}$	$\text{kg kg}^{-1} \text{s}^{-1}$	Source/sink term of $N_r$
$\Psi_q$	$\text{kg kg}^{-1} \text{s}^{-1}$	Source/sink term of $q$
$\Psi_{q_r}$	$\text{kg kg}^{-1} \text{s}^{-1}$	Source/sink term of $q_r$

**Table 5.** List of LPM symbols and parameters.

Symbol	Dimension/Value	Description
$C_L$	3	Constant in calculation of SGS particle velocity
$c_{\text{sgs}}$		Factor for relation between SGS and total TKE
$e_{\text{res}}$	$\text{m}^2 \text{s}^{-2}$	Resolved-scale TKE
$f_v$		Ventilation factor
$u_{p,i}$	$\text{m s}^{-1}$	Particle velocity components
$u_{p,i}^{\text{res}}$	$\text{m s}^{-1}$	Resolved particle velocity components
$u_{p,i}^{\text{sgs}}$	$\text{m s}^{-1}$	SGS particle velocity components
$u_*$	$\text{m s}^{-1}$	Friction velocity
$x_{p,i}$	m	Particle location
$x_{\text{ps},i}$	m	Particle source location
$\Delta t_L$	s	Time step of the LPM
$\zeta$		Vector composed of Gaussian-shaped random numbers
$\rho_{p,0}$	$\text{kg m}^{-3}$	Density of the particle
$\tau_L$	s	Lagrangian time scale
$\tau_p$	s	Stokes's drag relaxation time scale

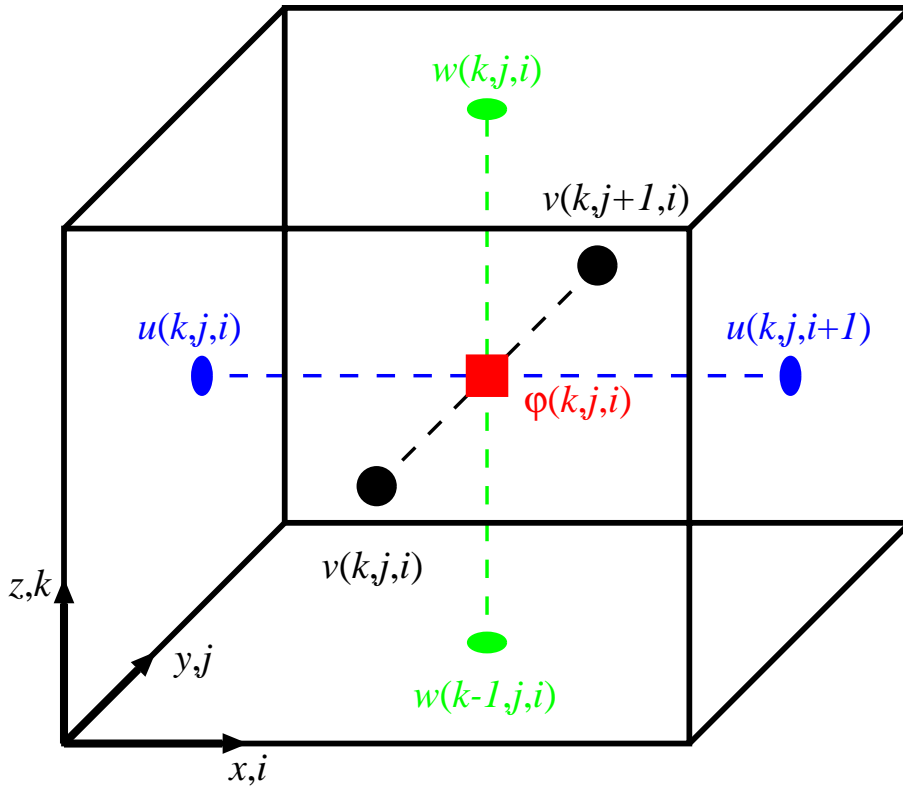
**Table 6.** List of canopy model parameters and symbols.

Symbol	Dimension/Value	Description
$C_e$	$m^2 m^{-3}$	Canopy tendency for SGS-TKE
$C_{u_i}$	$m s^{-2}$	Canopy tendency for velocity components
$C_\theta$	$K s^{-1}$	Canopy tendency for potential temperature
$C_\varphi$	$kg m^{-3} s^{-1}, kg kg^{-3} s^{-1}$	Canopy tendency for scalar quantities ( $s, q$ )
$c_d$		Canopy drag coefficient
$c_\varphi$		Canopy scalar exchange coefficient
LAD	$m^2 m^{-3}$	Leaf area density
LAI	$m^2 m^{-2}$	Leaf area index
$z_c$	m	Canopy height
$\eta$	0.6	Canopy extinction coefficient
$\varphi_{c,0}$	$kg m^{-3}$	Scalar concentration at leaf surface

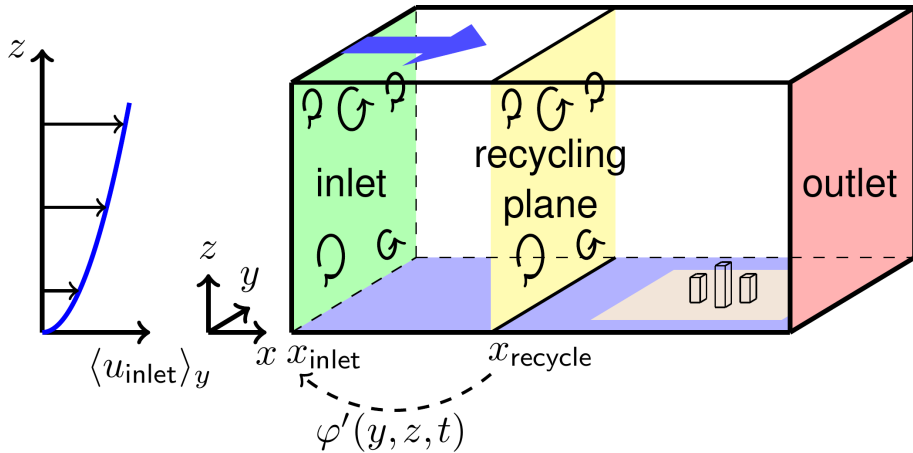




**Figure 1.** The PALM logo introduced in version 4.0.



**Figure 2.** The Arakawa staggered C-grid. The indices  $i, j, k$  refer to grid points in  $x, y$  and  $z$  direction, respectively. Scalar quantities  $\varphi$  are defined at the center of the grid volume, whereas velocities are defined at the edges of the grid volumes.



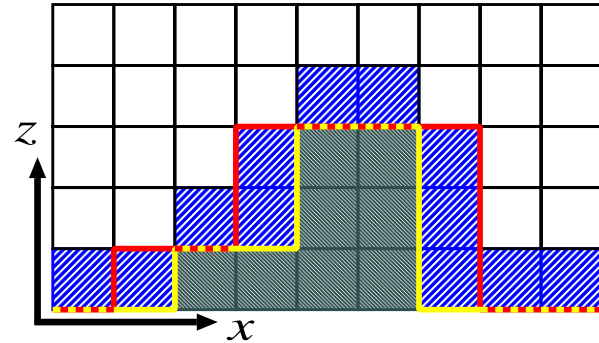
**Figure 3.** Schematic figure of the turbulence recycling method used for generation of turbulent inflow. The configuration represents exemplary conditions with a built-up analysis area (brown surface) and an open water recycling area (blue surface). The blue arrow indicates the flow direction.

index arrays

nzb\_w\_outer

nzb\_w\_inner

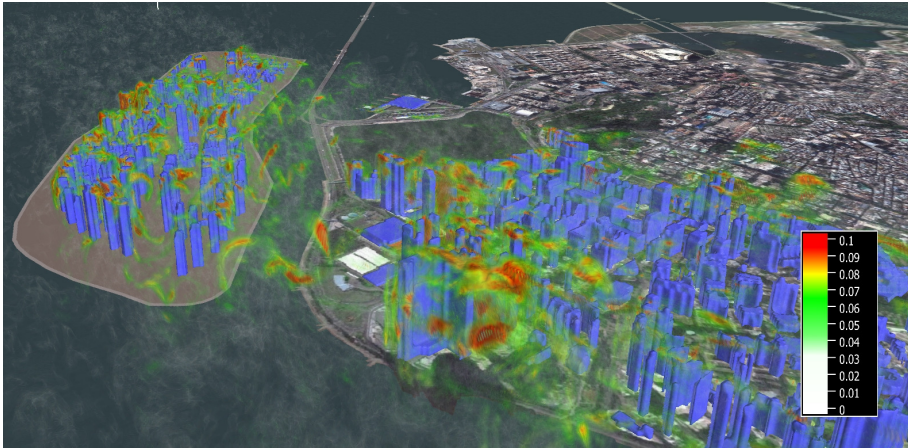
□ atmosphere  
code



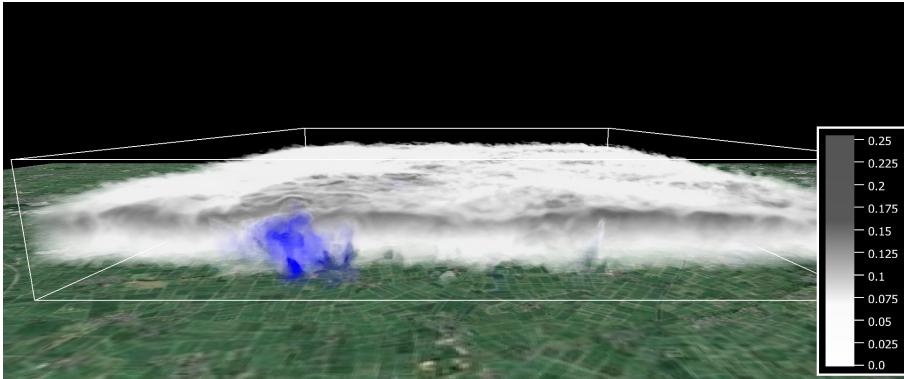
▨ wall-bounded  
code

■ no code  
(obstacle)

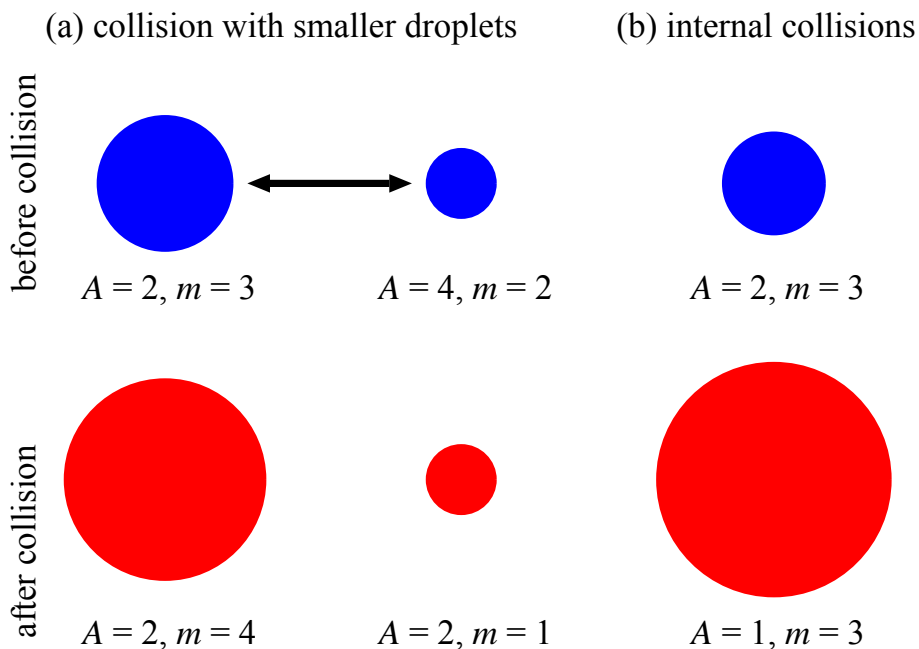
**Figure 4.** Sketch of the 2.5-D implementation of topography using the mask method (here for  $w$ ). The yellow and red lines represent the limits of the arrays  $nzb\_w\_inner$  and  $nzb\_w\_outer$  as described in Sect. 4.3, respectively.



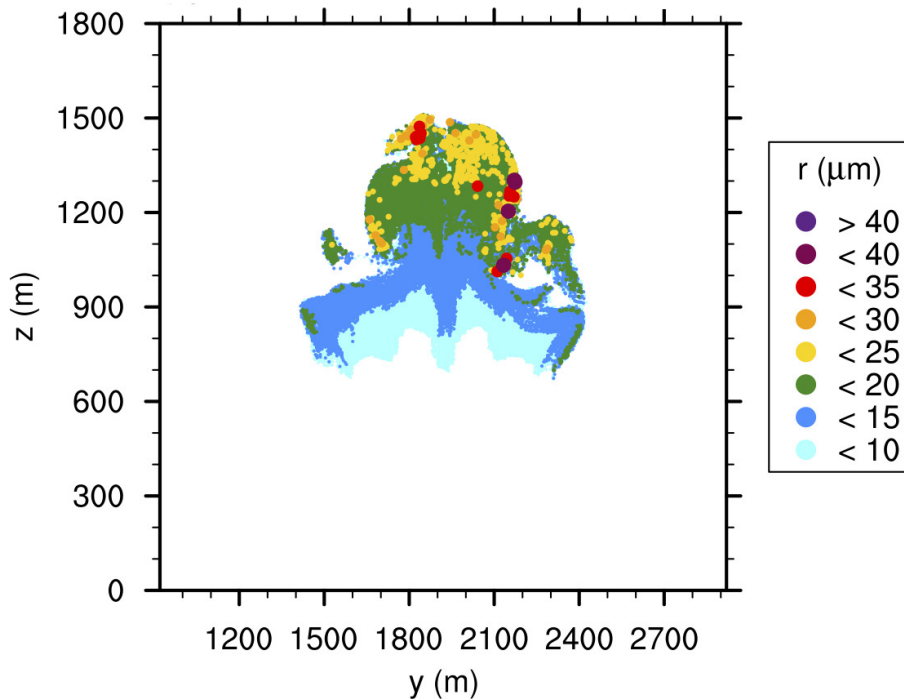
**Figure 5.** Snapshot of the absolute value of the 3-D rotation vector of the velocity field (red to white colors) for a simulation of the city of Macau, including a newly built-up artificial island (left). Buildings are displayed in blue. A neutrally-stratified flow was simulated with the mean flow direction from the upper-left to the bottom-right, i.e. coming from the open sea and flowing from the artificial island to the city of Macau. The figure shows only a subregion of the simulation domain that spanned a horizontal model domain of about  $6.1 \times 2.0 \times 1 \text{ km}^3$ , and with an equidistant grid spacing of 8 m. The copyright for the underlying satellite image is held by Cnes/Spot Image, Digitalglobe. For more details, see associated animation (Knoop et al., 2014).



**Figure 6.** Snapshot of the cloud field from a PALM run for three continuous days of the HD(CP)<sup>2</sup> Observational Prototype Experiment. Shown is the 3-D field of  $q_c$  (white to gray colors) as well as rain water ( $q_r > 0$ , blue) on 26 April 2013. The simulation had a grid-spacing of 50 m on a  $50 \times 50 \text{ km}^2$  domain. Large-scale advective tendencies for  $\theta_l$  and  $q$  were taken from COSMOE-DE (regional model of German Meteorological Service, DWD) analyses. The copyright for the underlying satellite image is held by Cnes/Spot Image, Digitalglobe.

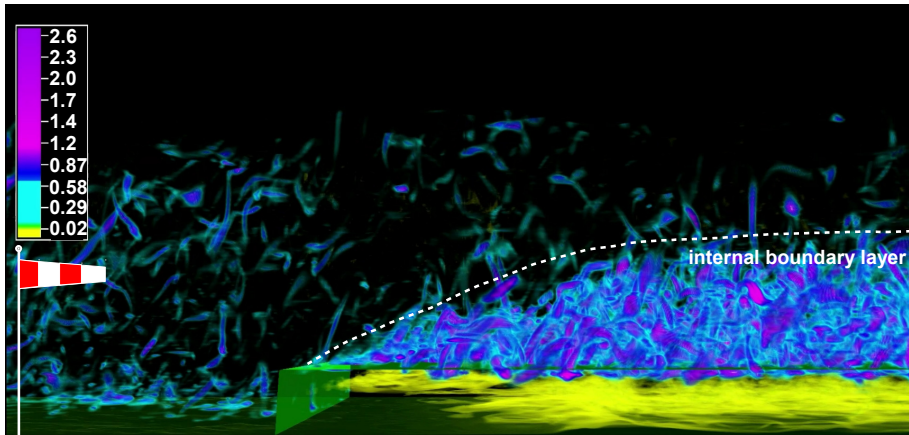


**Figure 7.** Illustration of (a) the collision of a super-droplet with a super-droplet smaller in radius, and (b) internal collisions of a single super-droplet. Blue (red) circles indicate super-droplets before (after) collision. Weighting factor ( $A$ ), and bulk mass ( $m$ ) are denoted in arbitrary units. The radius of the colored circle indicates the volume averaged radius of droplets represented by the super-droplet.



**Figure 8.** Distribution of droplets (colored dots) inside a shallow cumulus cloud simulated with PALM. The figure shows a vertical cross-section through the 3-D cloud. The color and size indicate the droplet's radius. The cloud has been triggered by a bubble of warm air (similar to Hoffmann et al., 2015). A grid spacing of 20 m was used and about 225 million particles were simulated in total.





**Figure 9.** Snapshot of the absolute value of the 3-D rotation vector of the velocity field above a forest canopy downstream of a grassland-to-forest transition (forest volume marked by green isosurface). Pink and yellow colors illustrate strong and weak turbulence, respectively. A neutrally-stratified open-channel flow was simulated with the mean flow direction from left to right, i.e. perpendicular to the windward forest edge, using an equidistant grid spacing of 3 m. The figure shows only a subregion of the simulation domain ( $2 \times 1 \times 0.4 \text{ km}^3$ ).

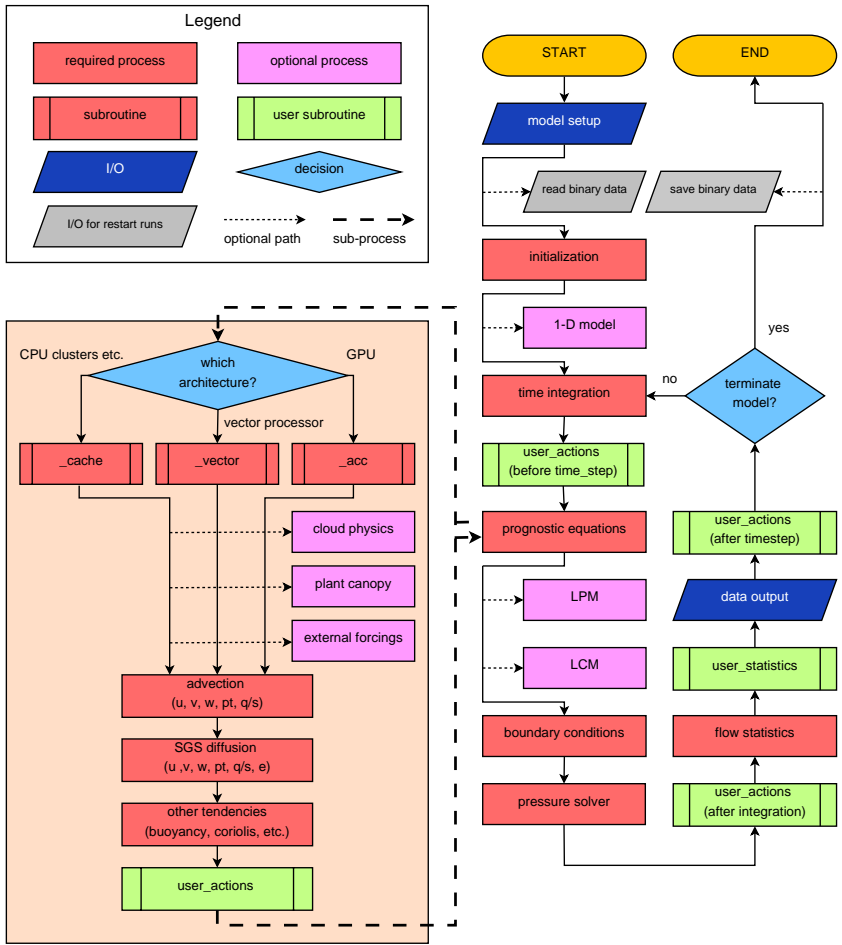
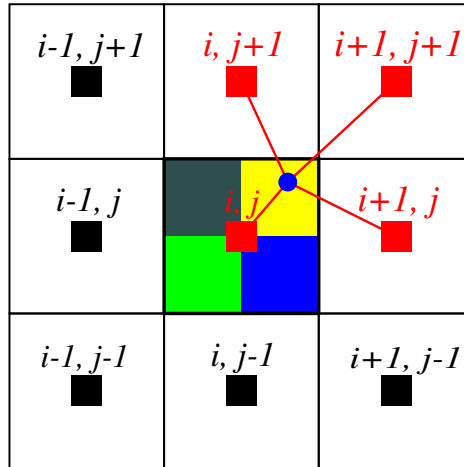
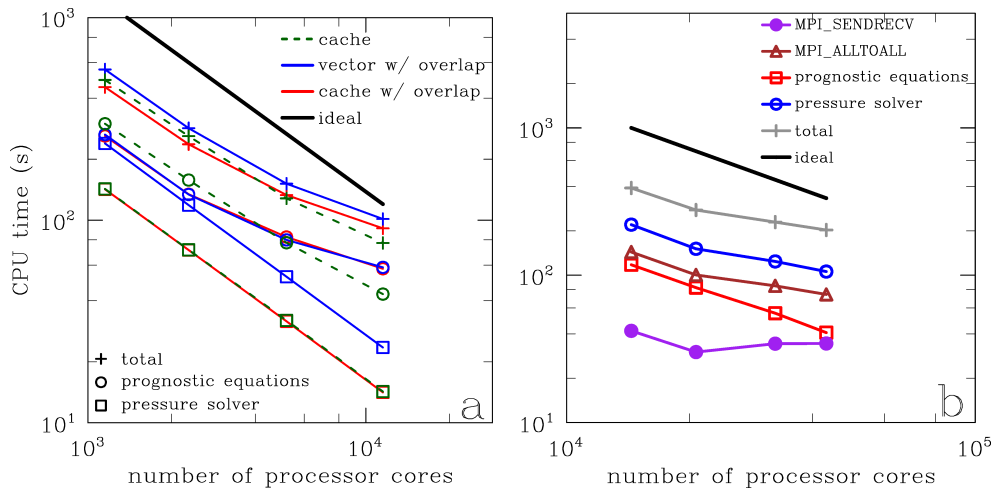


Figure 10. Simplified flowchart of PALM.



**Figure 11.** Two-dimensional example of the optimized interpolation algorithm. Interpolating a scalar quantity (e.g., temperature) bi-linearly on a particle (blue dot) located in a certain LES grid box (thick black line) includes four values of LES-data (red squares). Note that these values are the same for all particles located in the yellow subgrid box. Thus, by sorting all particles inside a grid box by their respective subgrid box, the indices required for interpolation need to be determined just once for all particles located in that subgrid box, and not repeatedly for all particle inside the entire grid box. This algorithm applies analogously for the velocity components located at the edges of the grid box.



**Figure 12.** Scalability of PALM 4.0 on the Cray XC30 supercomputer of HLRN. Simulations were performed with a computational grid of **(a)**  $2160^3$  and **(b)**  $4320^3$  grid points (Intel-Ivy Bridge CPUs). **(a)** shows data for up to 11520 PEs with cache (red lines) and vector (blue lines) optimization and overlapping during the computation (FFT and tri-diagonal equation solver, see Sect. 4.4) enabled (dashed green lines). Measurement data are shown for the total CPU time (crosses), the prognostic equations (circles), and for the pressure solver (boxes). **(b)** shows data for up to 43200 PEs and with both cache optimization and overlapping enabled. Measurement data is shown for the total CPU time (gray line), pressure solver (blue line), prognostic equations (red line), as well as the MPI calls MPI\_ALLTOALL (brown line) and MPI\_SENDRCV (purple line).

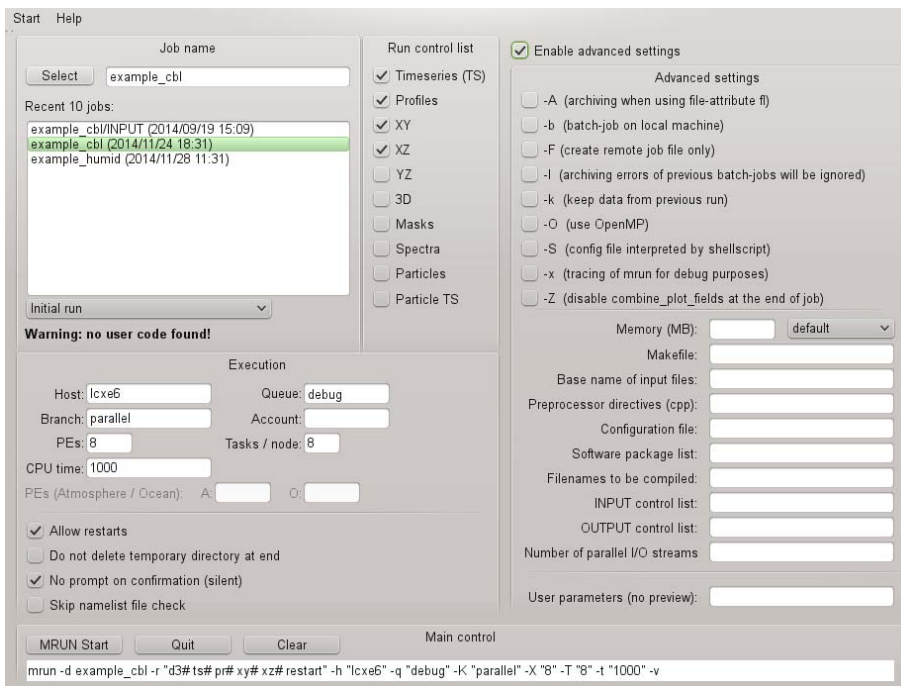


Figure 13. Screenshot of the mrunGUI program.