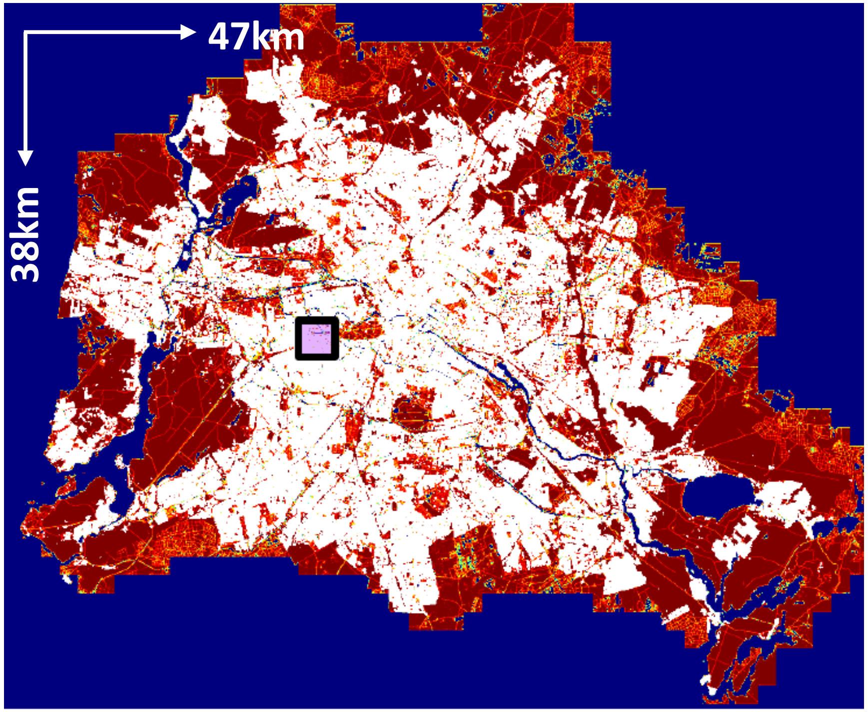
Grid nesting

palmgroup

Institute of Meteorology and Climatology, Leibniz Universität Hannover

Why is there a need for a grid nesting?

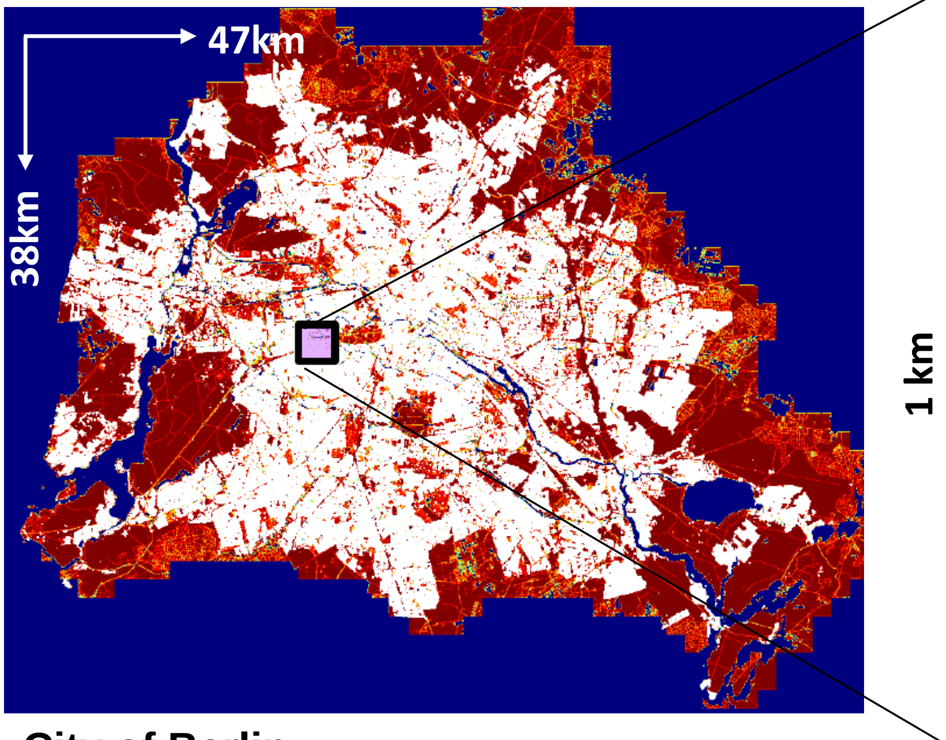


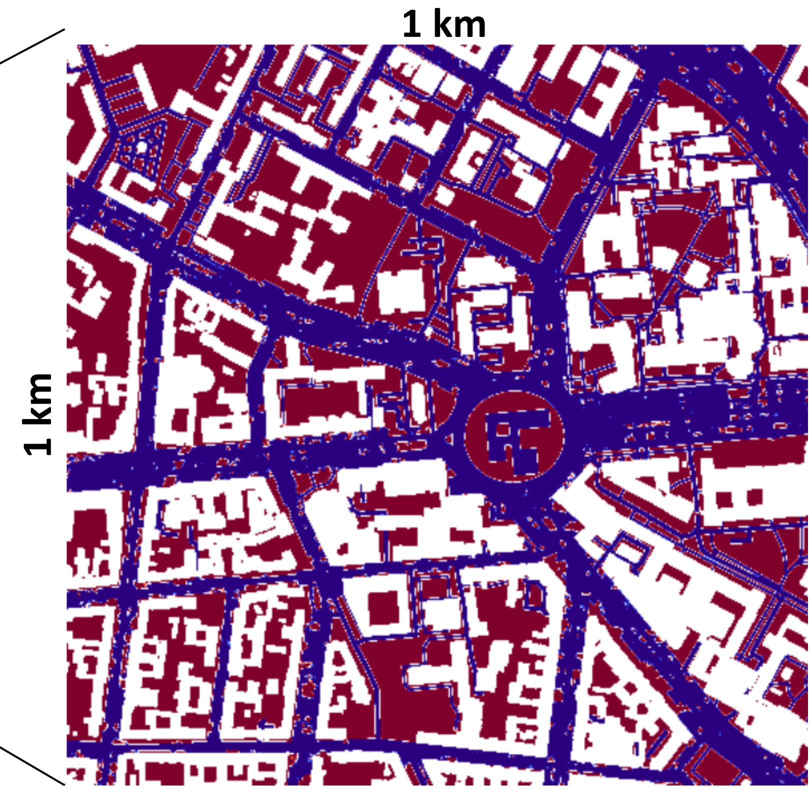
City of Berlin

- **10 m resolution: 10^9-10^{10} grid points**
- **1 m resolution: 10^{12}-10^{13} grid points**

Not feasible for detailed parameter studies, even with 10 m resolution everywhere!

Why is there a need for a grid nesting?





City of Berlin

- **50 m resolution: 10^7-10^8 grid points**

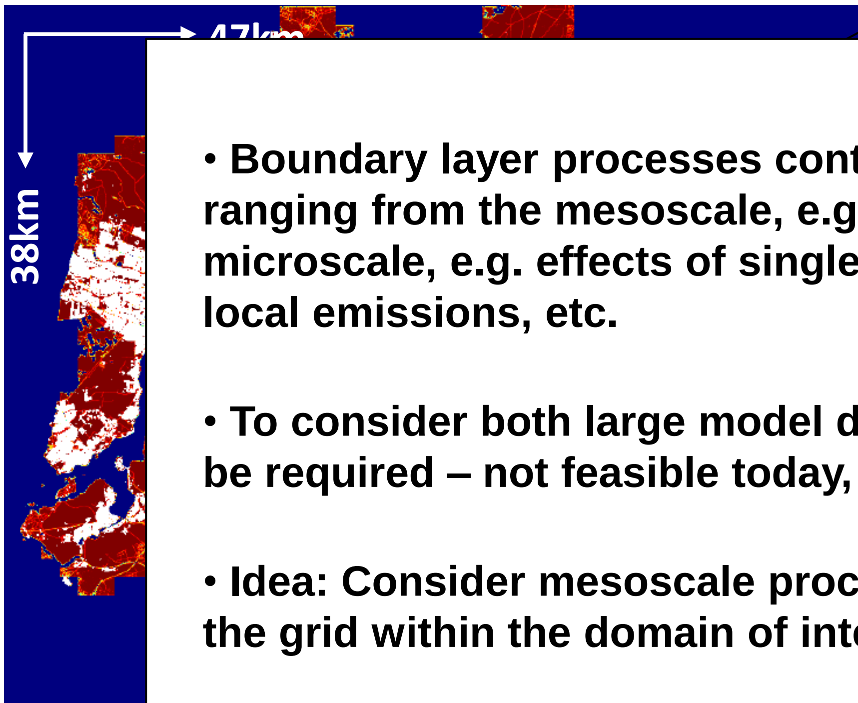
50 m resolution is feasible, but it might be too coarse for using it everywhere!

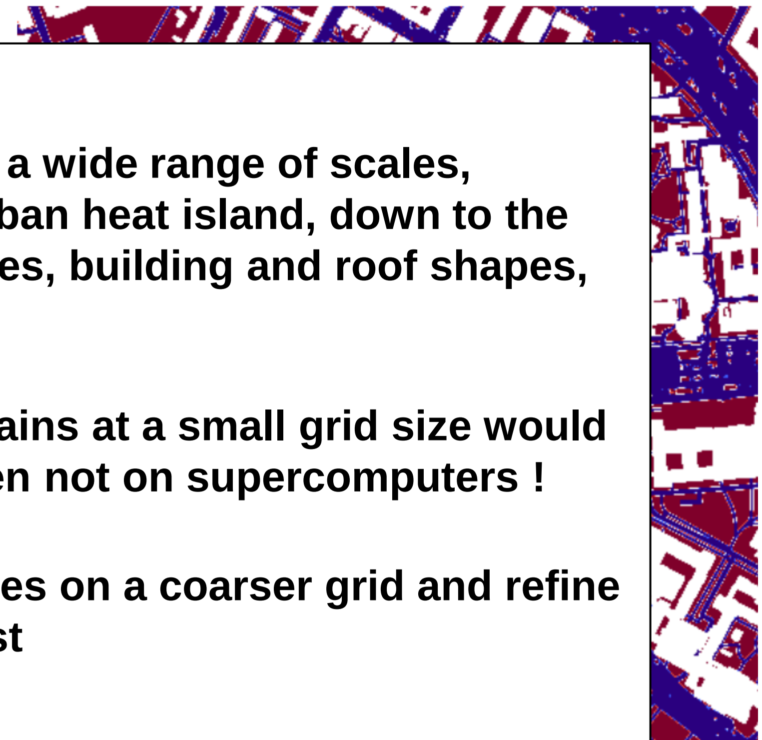
Increase grid resolution for domain of interest

- **1 m resolution: 10^8 grid points**

Attention: Too high grid spacing ratios can create unrealistic results

Why is there a need for a grid nesting?





- Boundary layer processes contain a wide range of scales, ranging from the mesoscale, e.g. urban heat island, down to the microscale, e.g. effects of single trees, building and roof shapes, local emissions, etc.

- To consider both large model domains at a small grid size would be required – not feasible today, even not on supercomputers !

- Idea: Consider mesoscale processes on a coarser grid and refine the grid within the domain of interest

City o

- **50 m resolution: 10^7-10^8 grid points**

50 m resolution is feasible, but it might be too coarse!

Increase grid resolution for domain of interest

- **1 m resolution: 10^8 grid points**

Presentation outline

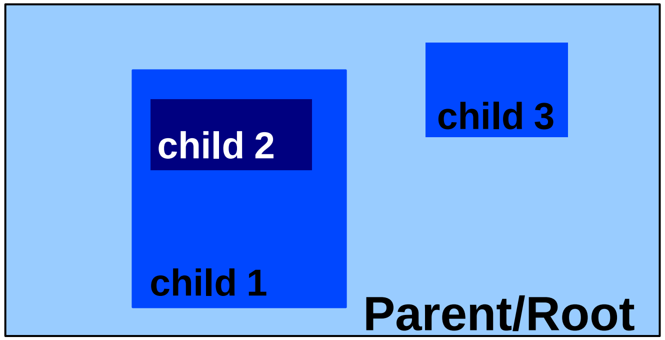
- General/Technical information about the self-nesting method → PALM's first nesting system
  - Steering
  - Examples
- Mesoscale (Offline) nesting in large-scale models (e.g., COSMO) → PALM's second nesting system
- Outlook and open points

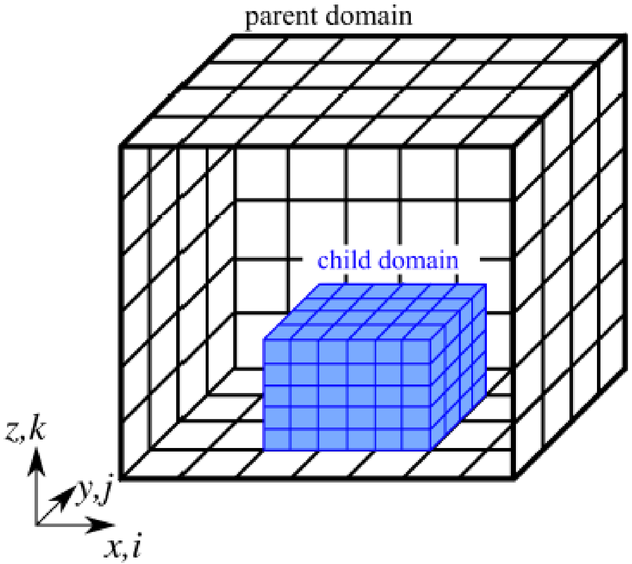
Basics of the nesting

- **Goal:** Reduce computational costs significantly
  - Enable simulations with a large domain and detailed analysis within domain of interest
  - Enable industrial application of LES with PALM (urban environments, site assessment in wind energy)
- **Idea:**
  - High grid resolution within domain of interest
  - Coarse grid resolution of other/outer parts of model domain
- **Precondition/Requirement:**
  - Nested domain results should correspond to results of the “classical” non-nested simulation, where a uniform grid spacing identical to the grid spacing in the nested domain is used in the whole domain

General principles 3D nesting – Domain Structure

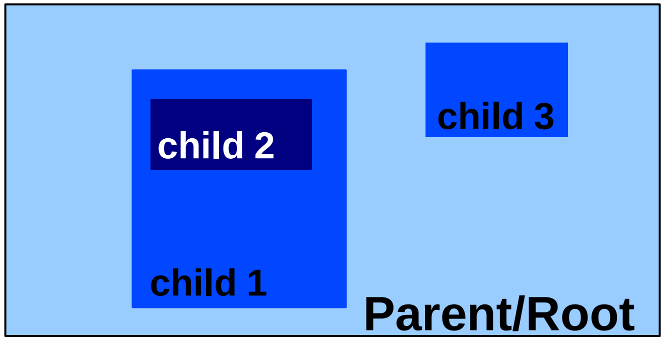
- Self-nesting is a PALM-PALM-coupling with two or more simulations running in parallel to each other with a continuous communication at runtime.
- One root domain (outermost and coarsest-resolution LES domain) and up to 63 child domains embedded into the root model are possible
- Childs can be recursively nested within each other → model domain can be parent and child at the same time (see child 1)
- Child domains can also be parallel to each other sharing same parent domain





General principles 3D nesting – Domain Structure

- All child domains must be completely inside their parent domain → no overlapping of parallel child domains → they have one parent domain

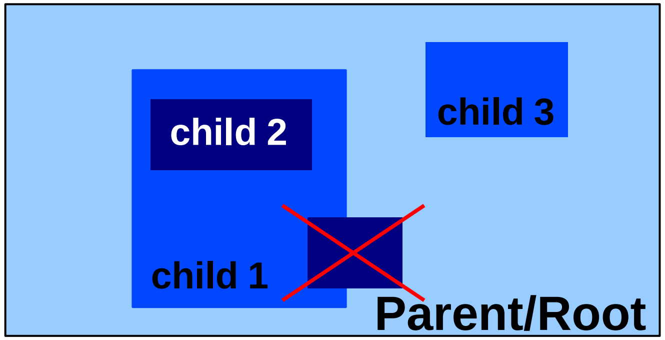


- Outer boundaries of child domain must match the underlying parent grid lines in all directions, lower boundary surface-bound

- Inside child domain all parent-grid lines must match the corresponding child-grid lines

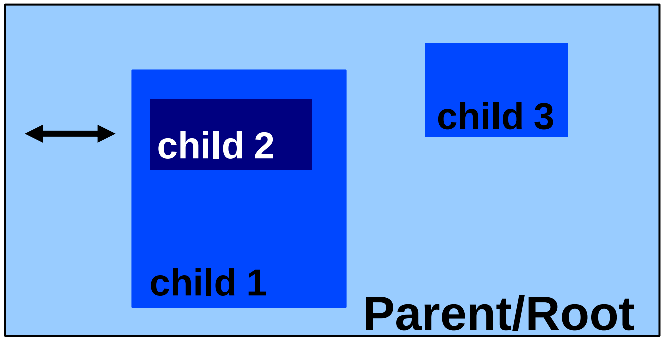
→ Grid-spacing ratios in each direction must be integer valued.

→ Vertical grid stretching is only allowed in the root domain above the top level of the highest nested domain

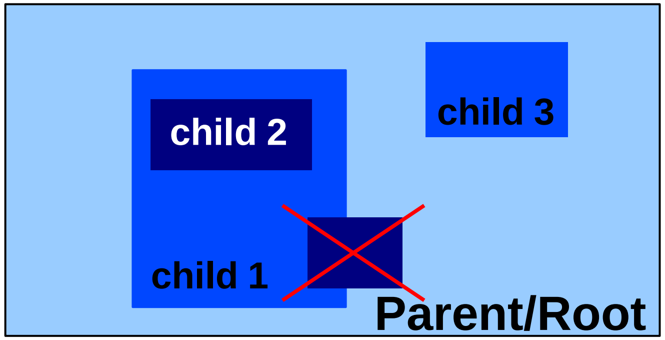


General principles 3D nesting – Domain Structure

- 2D Domain decomposition of child domains during parallelization must be realized in a way that the sub-domain size is never smaller than the parent grid spacing in the respective direction



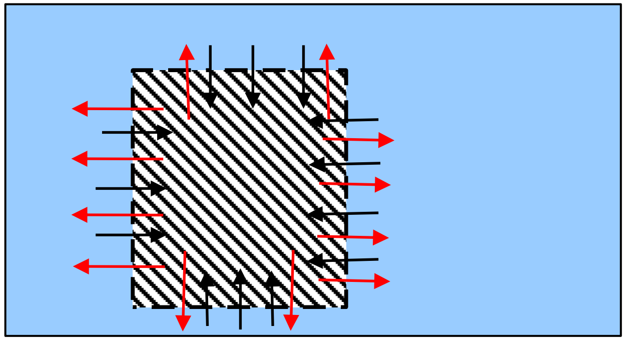
- 4 parent grid cells between the boundaries of child and parent domains are necessary



General principles 3D nesting – Data transfer

- Two-way (default mode) or one-way coupling is possible
- All prognostic variables are coupled except the SGS-TKE *e* (has no real benefit and a coupling of *e* is everything else than straightforward since it strongly depends on grid resolution)
- Most important requirements for the nesting algorithm: Minimizing flux conservation errors and enabling complex topography

Data transfer ?



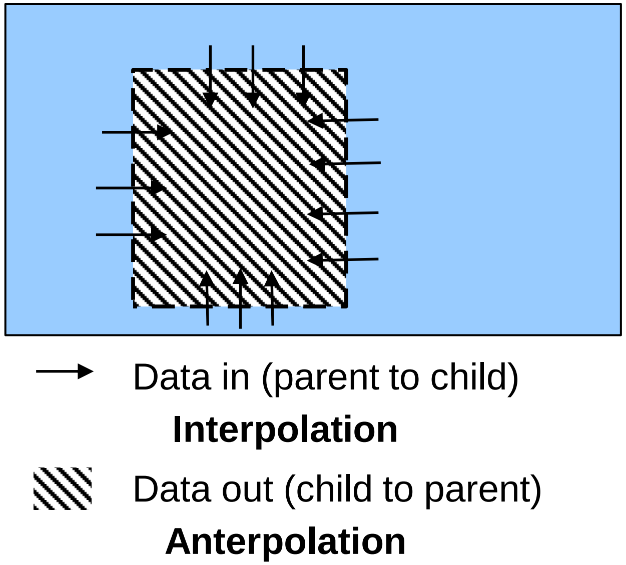
Data in ? (parent to child)

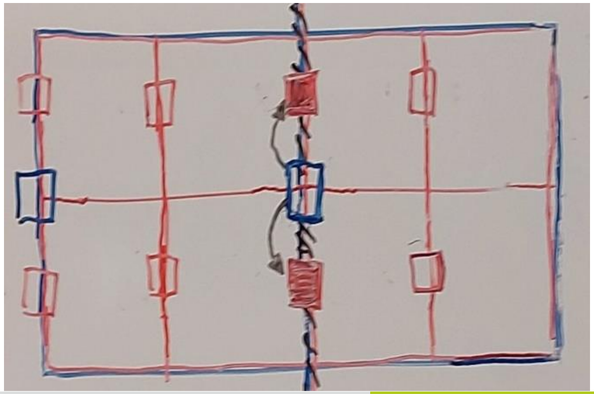
Data out ? (child to parent)

General principles 3D nesting – Data transfer

- **Two-way coupling/nesting**:
  - The focus is on both parent and child domain (e.g., dispersion scenarios)
  - Child domain obtains boundary-conditions from parent through zero order or linear interpolation
  - For boundary-normal velocity components, the original parent values are used (e.g., left child boundary u-component, values are set directly on the boundary → zero order interpolation):

Data transfer

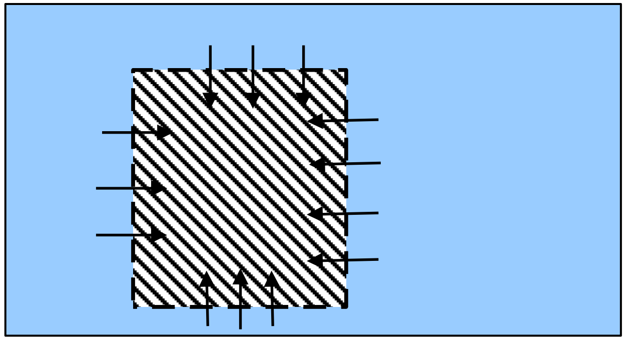




General principles 3D nesting – Data transfer

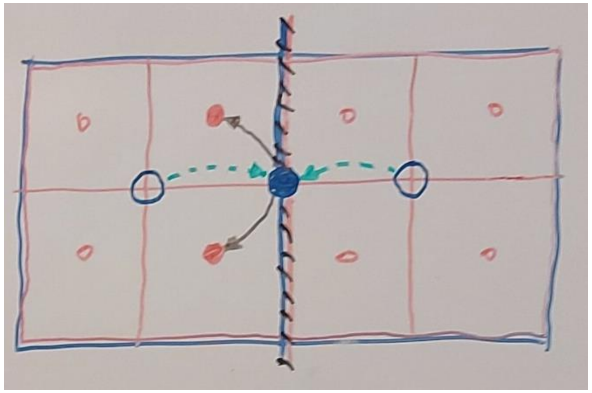
- **Two-way coupling/nesting**:
  - The focus is on both parent and child domain (e.g., dispersion scenarios)
  - Child domain obtains boundary-conditions from parent through zero order or linear interpolation
  - For scalars, averaged parent values from the nest boundary are used (e.g., left child boundary, values are set for the first ghost point based on two surrounding parent grid values → linear interpolation):

Data transfer



→ Data in (parent to child)
Interpolation

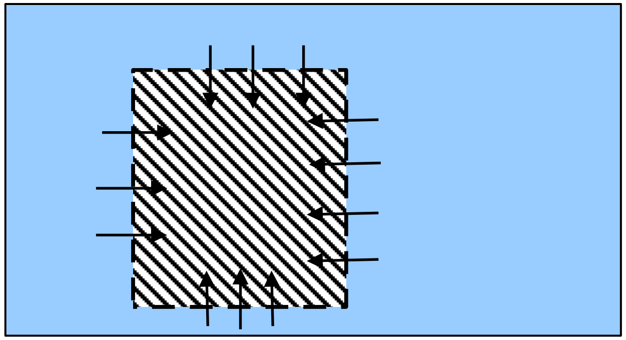
Data out (child to parent)
Anterpolation



General principles 3D nesting – Data transfer

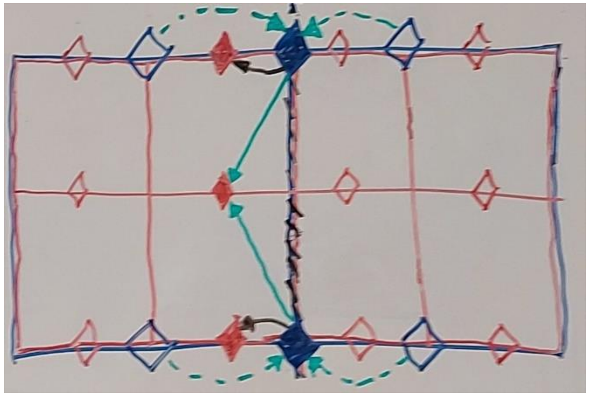
- **Two-way coupling/nesting**:
  - The focus is on both parent and child domain (e.g., dispersion scenarios)
  - Child domain obtains boundary-conditions from parent through zero order or linear interpolation
  - For staggered velocity components with respect to the boundary-normal velocity the following formula is used (e.g., left child boundary v-component, values are set for the first ghost point based on four surrounding parent grid values → linearly interpolated twice or once):

Data transfer



→ Data in (parent to child)
Interpolation

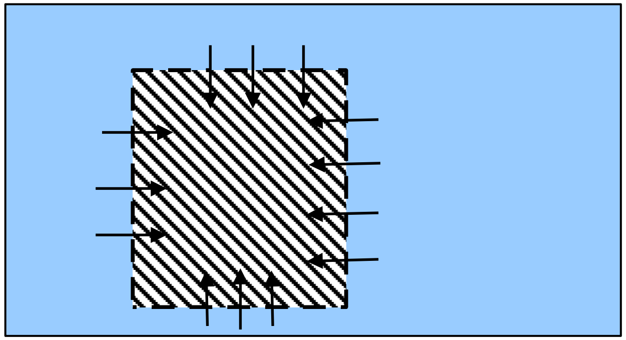
Data out (child to parent)
Anterpolation

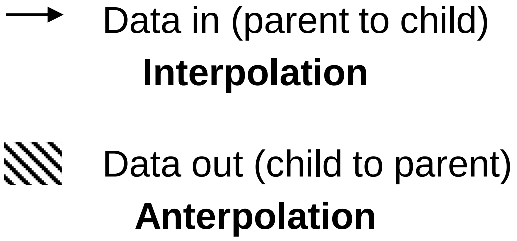


General principles 3D nesting – Data transfer

- **Two-way coupling/nesting**:
  - For *e*, a simple Neumann condition (zero-gradient) is used on child domain boundaries
  - The reason behind the (randomly appearing) interpolation scheme is explained in detail in "A Nested Multi-Scale System Implemented in the Large-Eddy Simulation Model PALM" by Hellsten et al. and goes far beyond an introduction

Data transfer





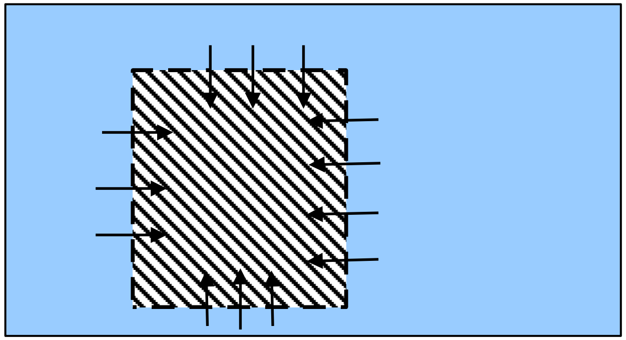
General principles 3D nesting – Data transfer

- **Two-way coupling/nesting**:
  - Child influences parent through anterpolation
  - Mapping the fine-grid solution back to the parent domain
  - Averaging over one parent-domain grid volume around the parent grid node of the variable in question (i.e., top-hat filtering)

$$\widehat{\phi}_{I,J,K} = \frac{1}{N_{I,J,K}} \sum_{i_1(I)}^{i_2(I)} \sum_{j_1(J)}^{j_2(J)} \sum_{k_1(K)}^{k_2(K)} \phi_{i,j,k}$$

  - Buffer zones of two prognostic gird points, where the anterpolation is omitted, are applied next to the child-domain boundaries (except bottom boundary) to avoid an unstable feedback loop between anterpolation and interpolation

Data transfer



→ Data in (parent to child) **Interpolation**

▨ Data out (child to parent) **Anterpolation**

→ The results must always be examined regarding unphysical feedbacks between parent and child

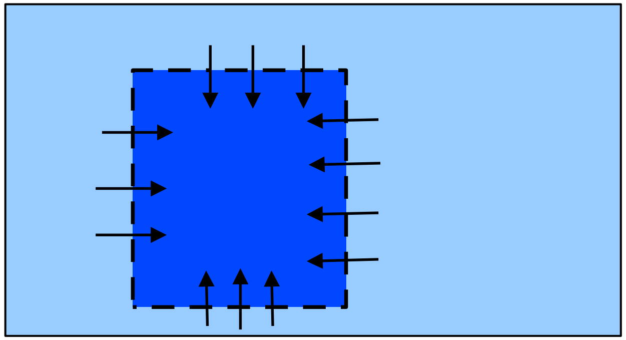
General principles 3D nesting – Data transfer

- **One-way coupling/nesting**:
  - The focus is only on the child domain (e.g., complex terrain)
  - Anterpolation step is omitted (saves computational costs)
  - Parent simulation is independent from child simulation – no feedback
  - Decoupling of turbulence may lead to strong discontinuities

  → The results of parent and child may become very different from each other but unphysical feedbacks between parent and child are uncritical

- Coupling operations are made at each Runge-Kutta time sub-step just before the pressure solver independent from the coupling method

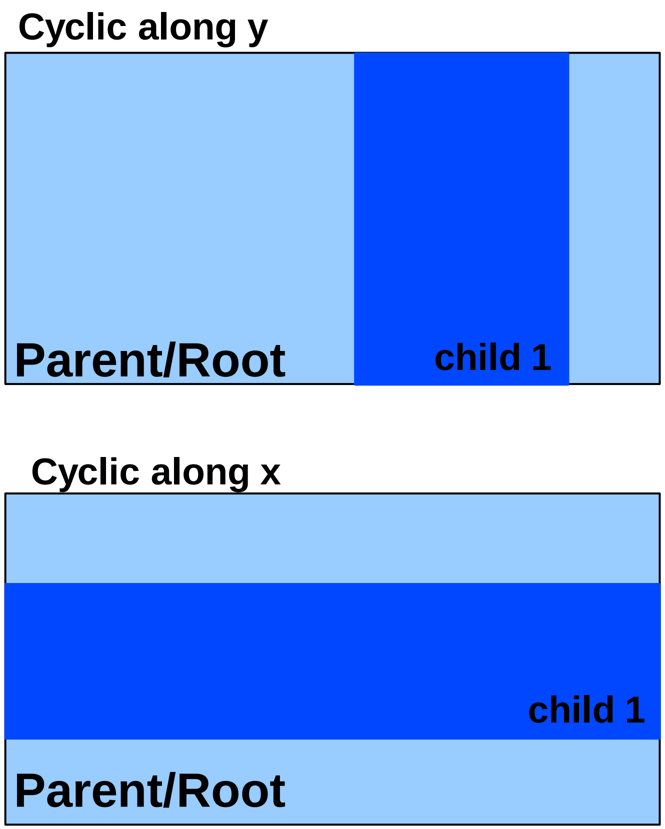
Data transfer



→ Data in (parent to child)
Interpolation

Special setups – Domain Structure

- 2D or even 1D nesting can also be applied
- 2D: Root domain size exactly matches the extension of the child in one horizontal direction
- 1D: All domains have the same horizontal extent → pure vertical nesting

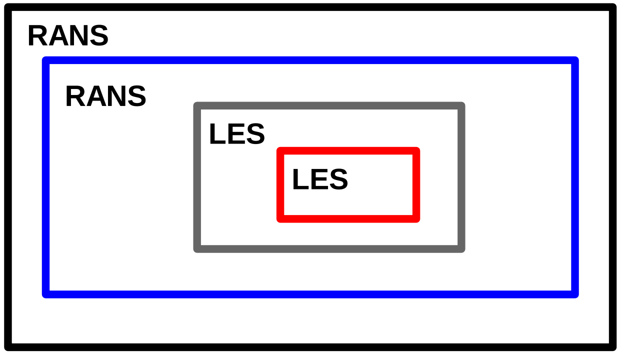


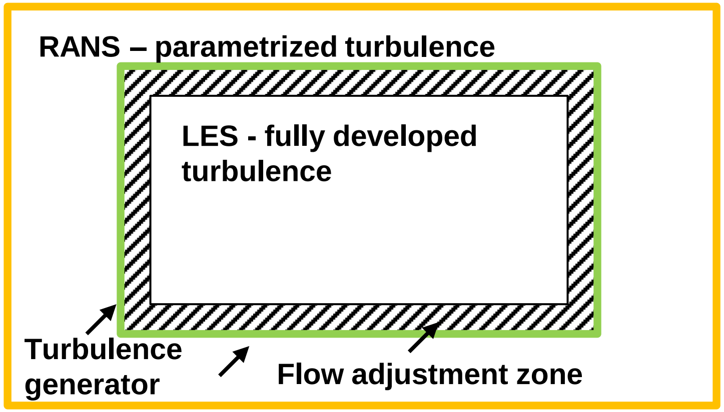
General principles – Initialization, boundary conditions, pressure solver

- Child domain is by default initialized with 3D volume data from parent, any other initialization, e.g. **`'set_constant_profiles'`** will be overwritten.
- Boundary conditions at lateral and top boundaries of nested domains are internally set to **`'nested'`**
  - Zero-gradient conditions for pressure (Neumann condition)
  - Dirichlet conditions for prognostic quantities derived from interpolation
  - For the root domain of a nested run the default is as usual (e.g., **`'cyclic'`** for lateral boundaries)
  - Exception: pure vertical nesting (lateral boundaries of parent and child are the same), where still cyclic lateral boundary conditions are applied as default

Nesting for RANS/LES mode

- PALM can run either in LES or in RANS mode – different turbulence closures (two for each)
- Nesting can be applied for both modes:
  - RANS – RANS nesting (1-way or 2-way coupling)
  - LES – LES nesting (1-way or 2-way coupling)
  - RANS – LES nesting (1-way coupling only) - mechanism requires to initiate turbulence at lateral boundaries – synthetic turbulence generator





Technical Implementation – PMC

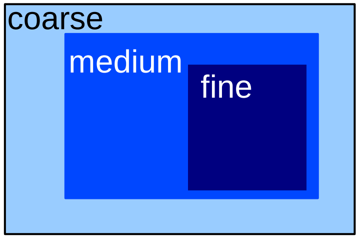
- Main challenge is the two-level parallelism:
  → Domains run in parallel and they are internally parallelized
- PALM Model Coupler (**PMC**), written by an external programmer (Klaus Ketelsen), handles data transfer
- It uses one-sided MPI communication, also called remote memory access (RMA), together with MPI windows (shared memory regions) for data transfer
- PMC can rather be seen as a black box and should never be touched
- It contains *pmc_child_mod, pmc_general_mod, pmc_handle_communicator_mod, pmc_mpi_wrapper_mod,* and *pmc_parent_mod*

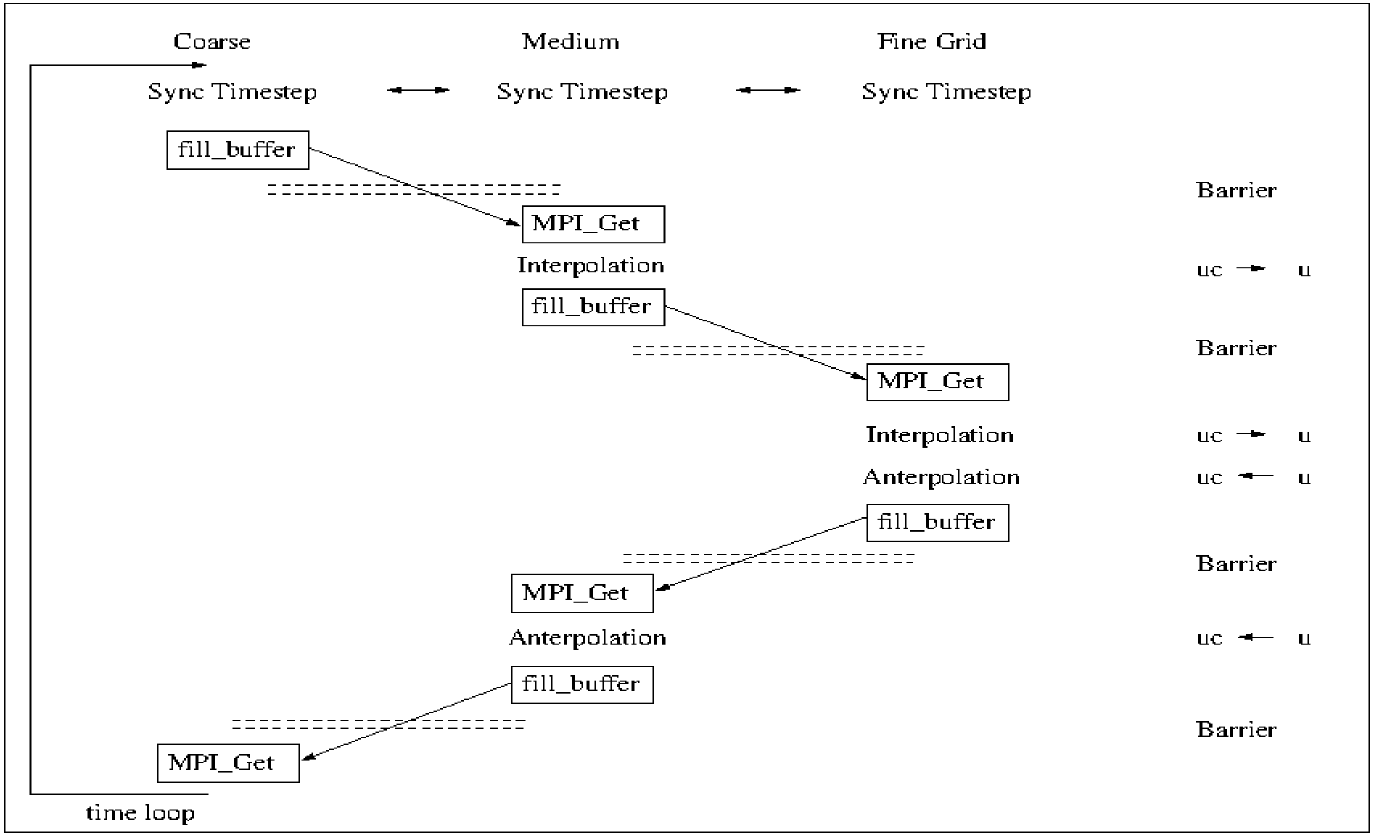
Technical Implementation – PMC interface

- Interface which contains all required subroutines, etc. for nesting – provides “easy” way to add new prognostic quantities → PMC interface
- PMC is called from the module *pmc_interface_mod*
- *pmc_interface_mod* contains all interpolation and anterpolation algorithms as well as other necessary operations (e.g., initialization operations)
- Interface has been mainly developed by Antti Hellsten, a collegue from Helsinki, Finland
- A publication called “A Nested Multi-Scale System Implemented in the Large-Eddy Simulation Model PALM” is available since 2021.
- A special interface to the Lagrangian Particle Model is also available to handle particle transfer between parent and child → *pmc_particle_interface*

Data Exchange – Two-way nesting

- **Cascade mode** - Overlap mode - Mixed mode (default) (only of relevance for recursively nested child domains)



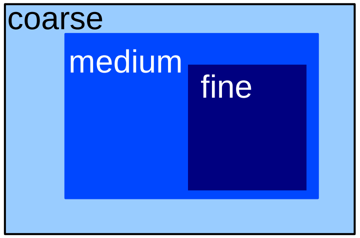


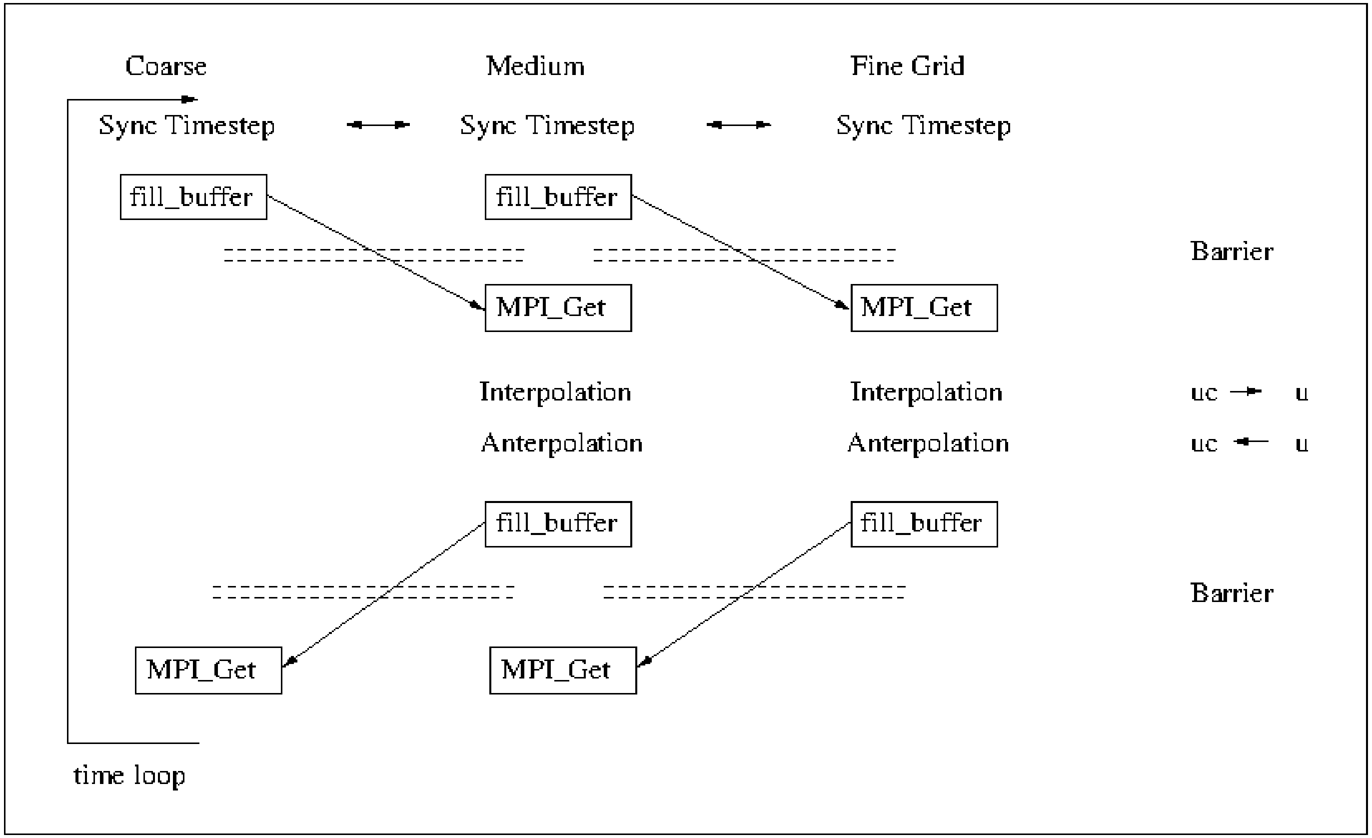
From coarse to fine:
Child waits until it has received data from the coarse model, does the interpolation, and then sends the data to the finer model

From fine to coarse:
Parent waits until it has received data from the finer model, does the anterpolation, and then sends the data to the coarser model

Data Exchange – Two-way nesting

- Cascade mode **- Overlap mode** - Mixed mode (default) (only of relevance for recursively nested child domains)



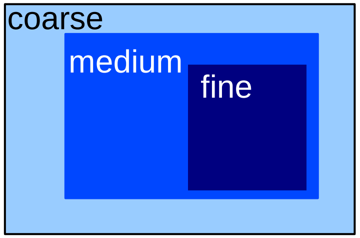


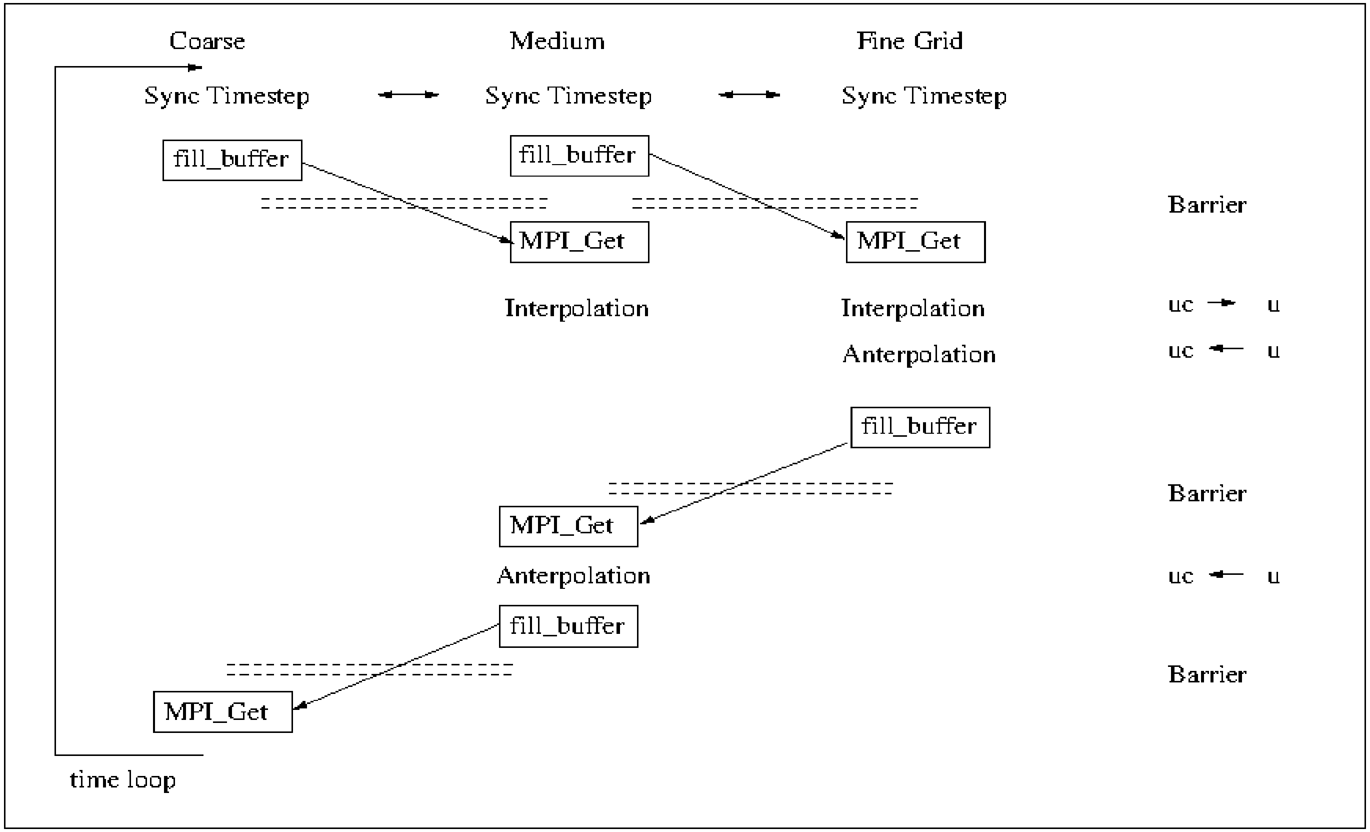
From coarse to fine:
All parents immediately send data after timestep synchronization. The childs fetch the data via MPI_Get and do the interpolation

From fine to coarse:
Anterpolation can also be done simultaneously for all models. Afterwards the data is transferred to the coarse model in parallel

Data Exchange – Two-way nesting

- Cascade mode **-** Overlap mode - **Mixed mode (default)** (only of relevance for recursively nested child domains)





From coarse to fine:
Overlap mode

From fine to coarse:
Cascade mode

Parameter file

- Each domain has its own parameter file:
  → *_p3d* (*PARIN*), *_p3d_N02* (*PARIN_N02)*,…
- Additional *NAMELIST* group name: *nesting_parameters*
  → provides information about all domains
  → only in *PARIN* (root model)
- Other input files (e.g topography, static and dynamic driver) are given for each domain
  → using domain tags e.g., *_static_N02*, *static_N03*, …
- Data output for each domain
  → using domain tags e.g. *_rc* (*RUN_CONTROL)*, *_rc_N02 (RUN_CONTROL_N02)*, ...

Parameter file – example parent (example_p3d)

```
                 name      id     parent_id  npe_total lower_left_x
                                                          lower_left_y

&nesting_parameters
         domain_layouts = 'palm_coarse', 1, -1, 64,   0.0,   0.0,
                          'palm_fine',   2,  1, 64, 320.0, 160.0,

         nesting_mode              = 'two-way',
         nesting_datatransfer_mode = 'mixed',
/

&initialization_parameters
    nx = 127, ny = 63, nz = 32,
...
```

Parameter file – example child (example_p3d_N02)

No NAMELIST group &nesting_parameters

```
&nesting_parameters
         domain_layouts = 'palm_coarse', 1, -1, 64, 0.0,   0.0,
                          'palm_fine',   2,  1, 64, 320.0, 160.0,

         nesting_mode                 = 'two-way',
         nesting_datatransfer_mode = 'cascade',
/

&initialization_parameters
    nx = 127, ny = 63, nz = 32,
...
```

Miscellaneous

- Assure that the total number of given cores match the sum of cores given for each domain
- Take care of a reasonable load balance between child and parent, i.e., subdomain sizes in terms of grid points should be comparable. Otherwise the parent always waits for the child or vice versa.

```
palmrun -r example -a “d3#” -X 128 ...
```

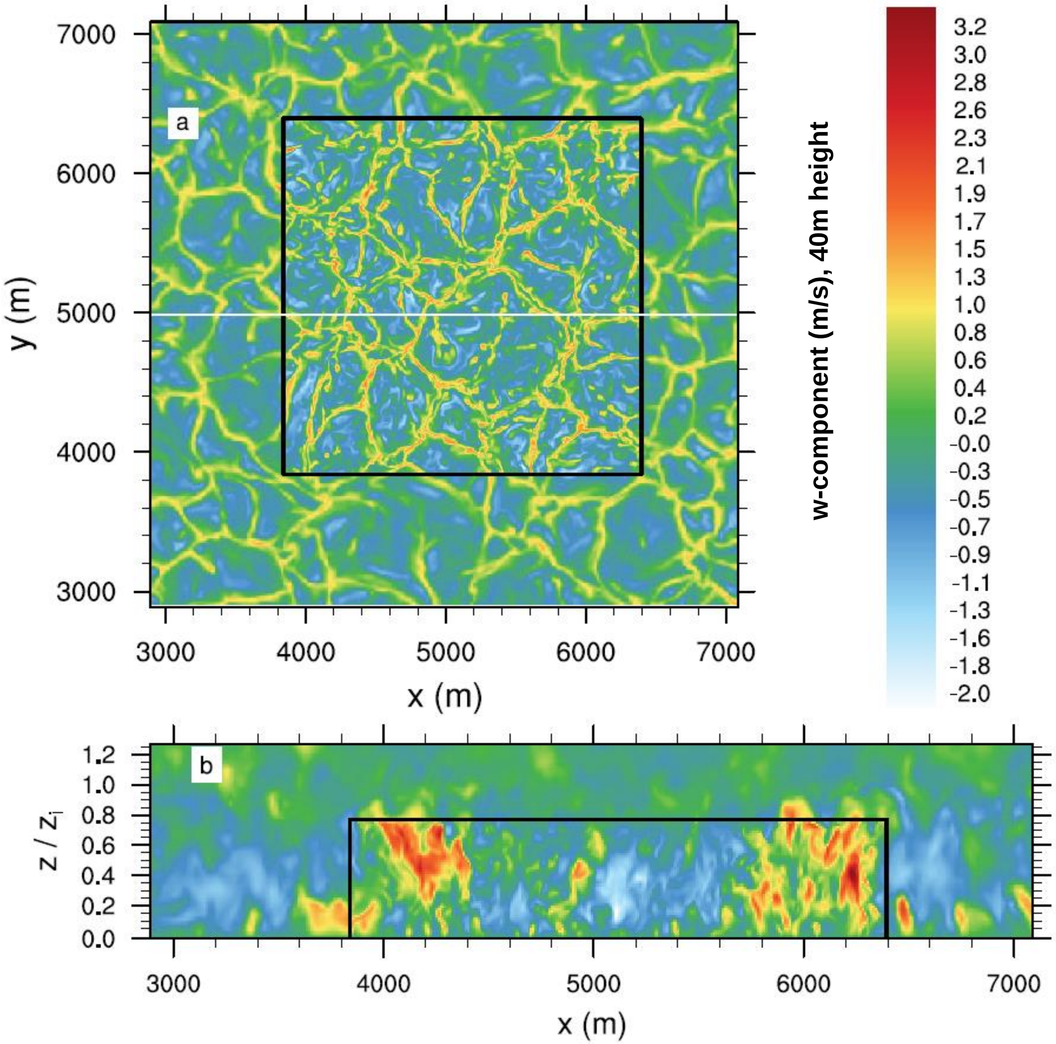
```
&nesting_parameters
        domain_layouts = 'palm_coarse', 1, -1, 64,  0.0,   0.0,
                         'palm_fine',   2,  1, 64, 320.0, 160.0,

        nesting_mode             = 'two-way',
        nesting_datatransfer_mode = ‘mixed',
/
```

Examples

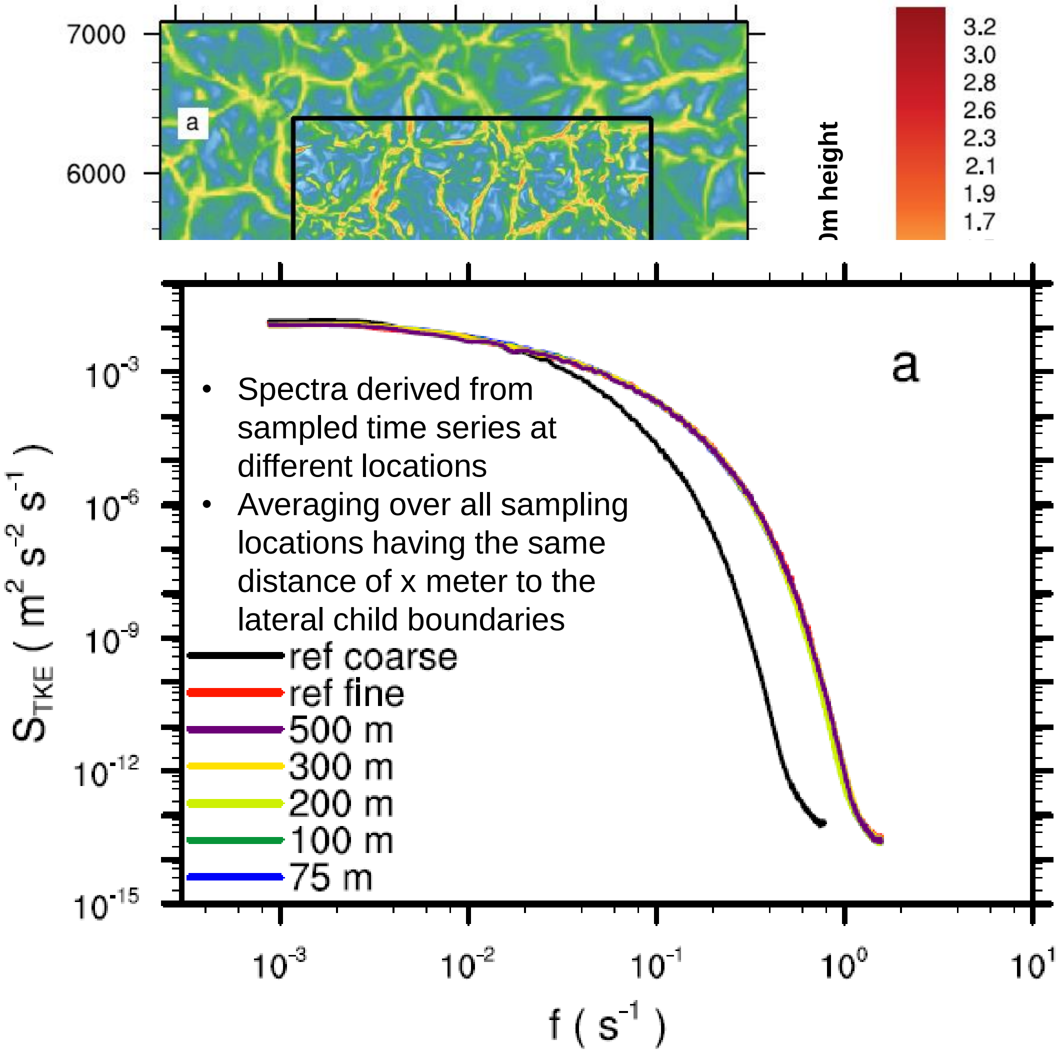
- Pure convective boundary layer with zero mean wind (homogeneously heated, flat terrain)
- Neutral boundary layer with background wind (purely shear-driven, flat-terrain)
- Neutrally-stratified urban boundary layer over a regular staggered arrangement of building cubes

Pure convective boundary layer



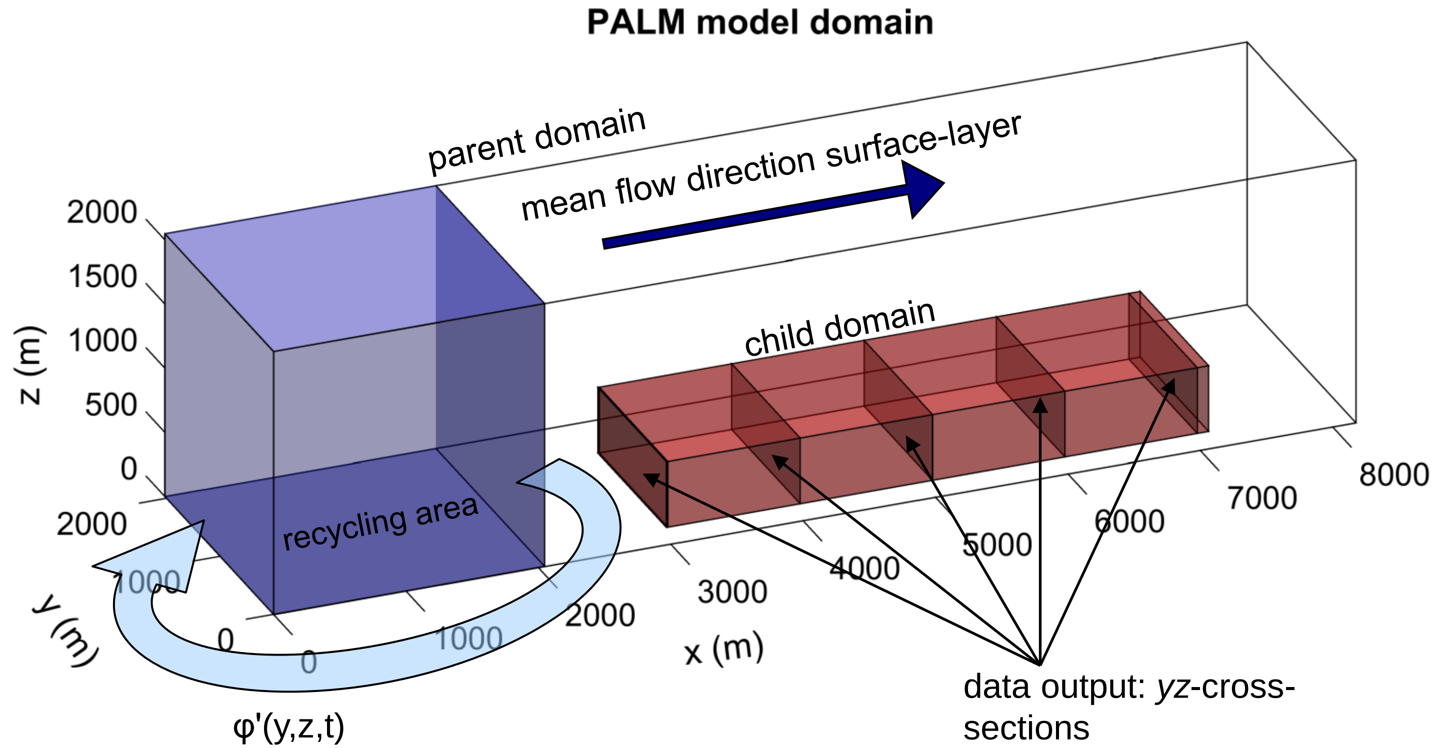
- Grid spacing ratio 20m:10m
- No discontinuities near boundaries in terms of shape and amplitude
- Finer structures within child domain with slightly stronger up/downdrafts
- Comparable size of hexagonal cells

Pure convective boundary layer



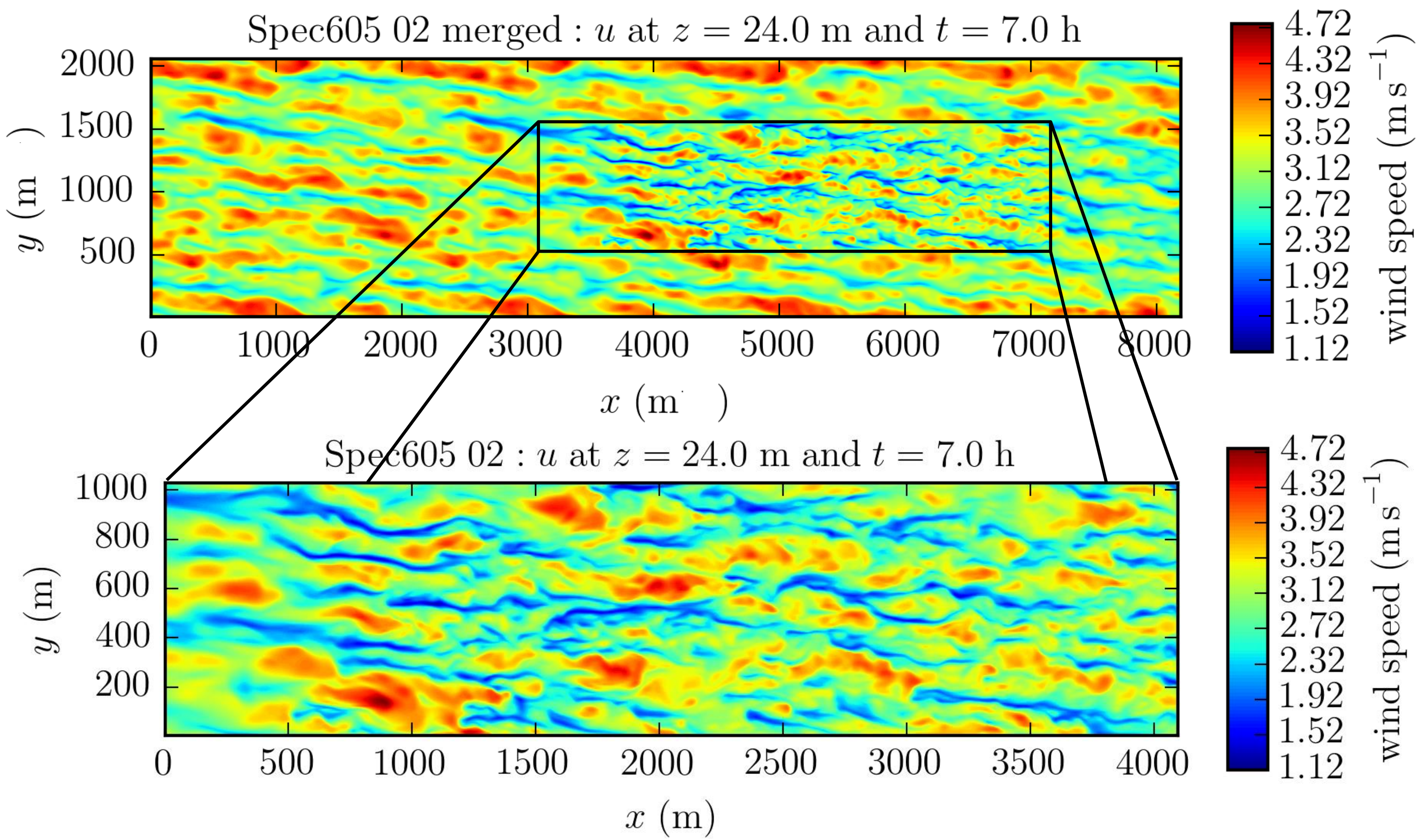
- Grid spacing ratio 20m:10m
- No discontinuities near boundaries in terms of shape and amplitude
- Finer structures within child domain with slightly stronger up/downdrafts
- Comparable size of hexagonal cells
- Better representation of spectral properties for fine-grid simulation
- Fine-grid simulation comparable to child solution independent of the distance to the boundaries
- In pure convective case almost no adjustment zone required since turbulence is mostly produced locally

Neutral boundary layer

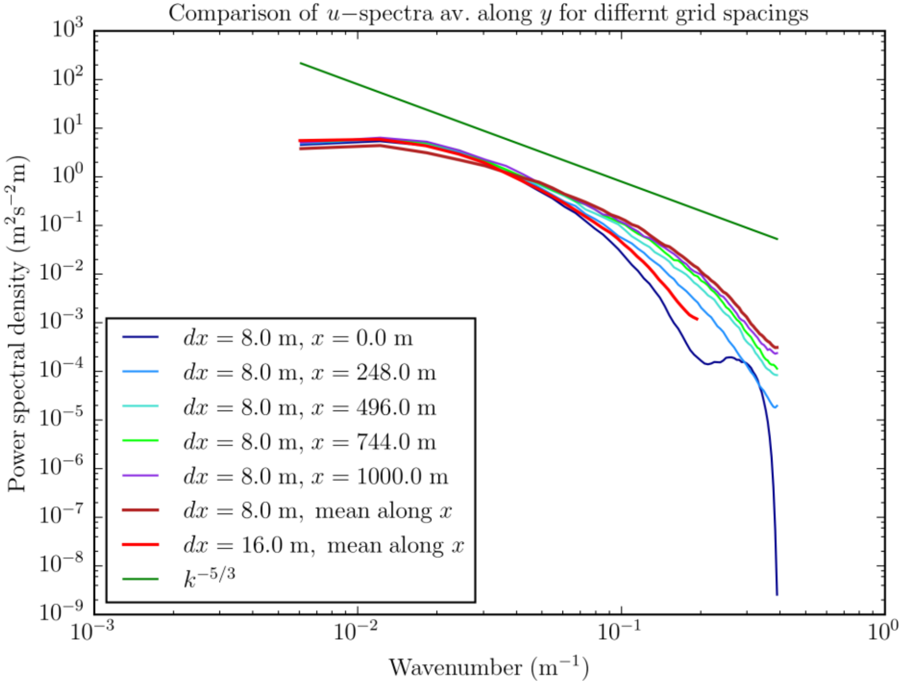


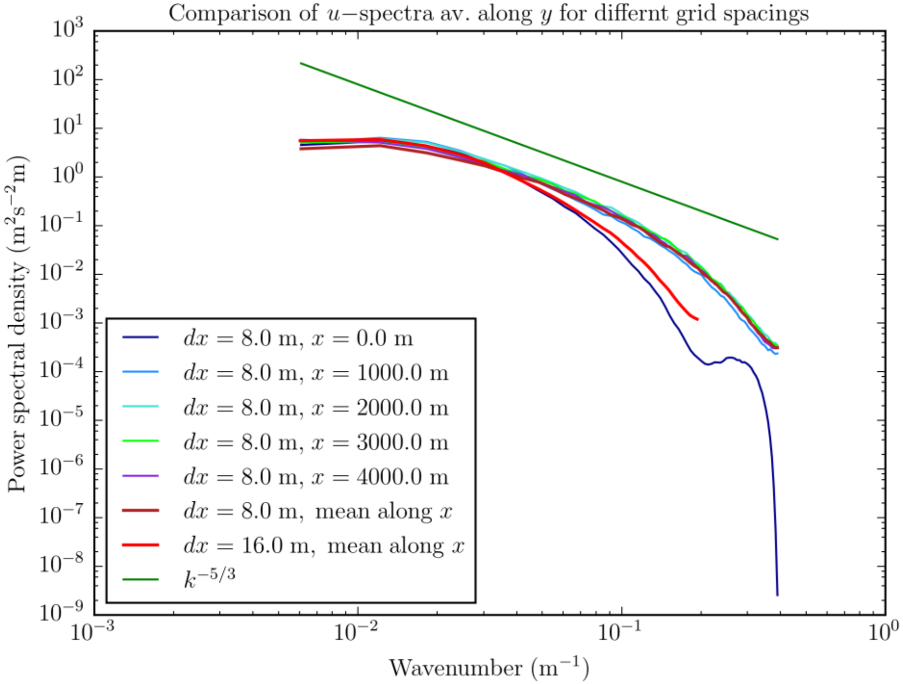
Grid for both domains: 512 x 128 x 64 gridpoints
Grid spacing: **16 m** (parent) and **8 m** (child)
Position nest: lower_left_x = 3072 m, lower_left_y = 512 m
PEs: each on 256
Coupling: Two-way

x-*y* cross-section of the streamwise velocity *u*

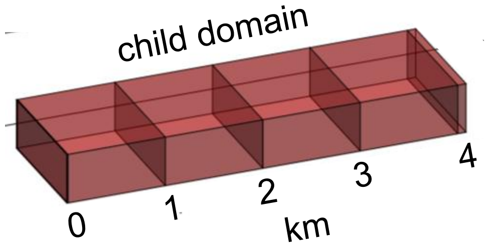


Spectra of *u*-component at different locations in streamwise direction





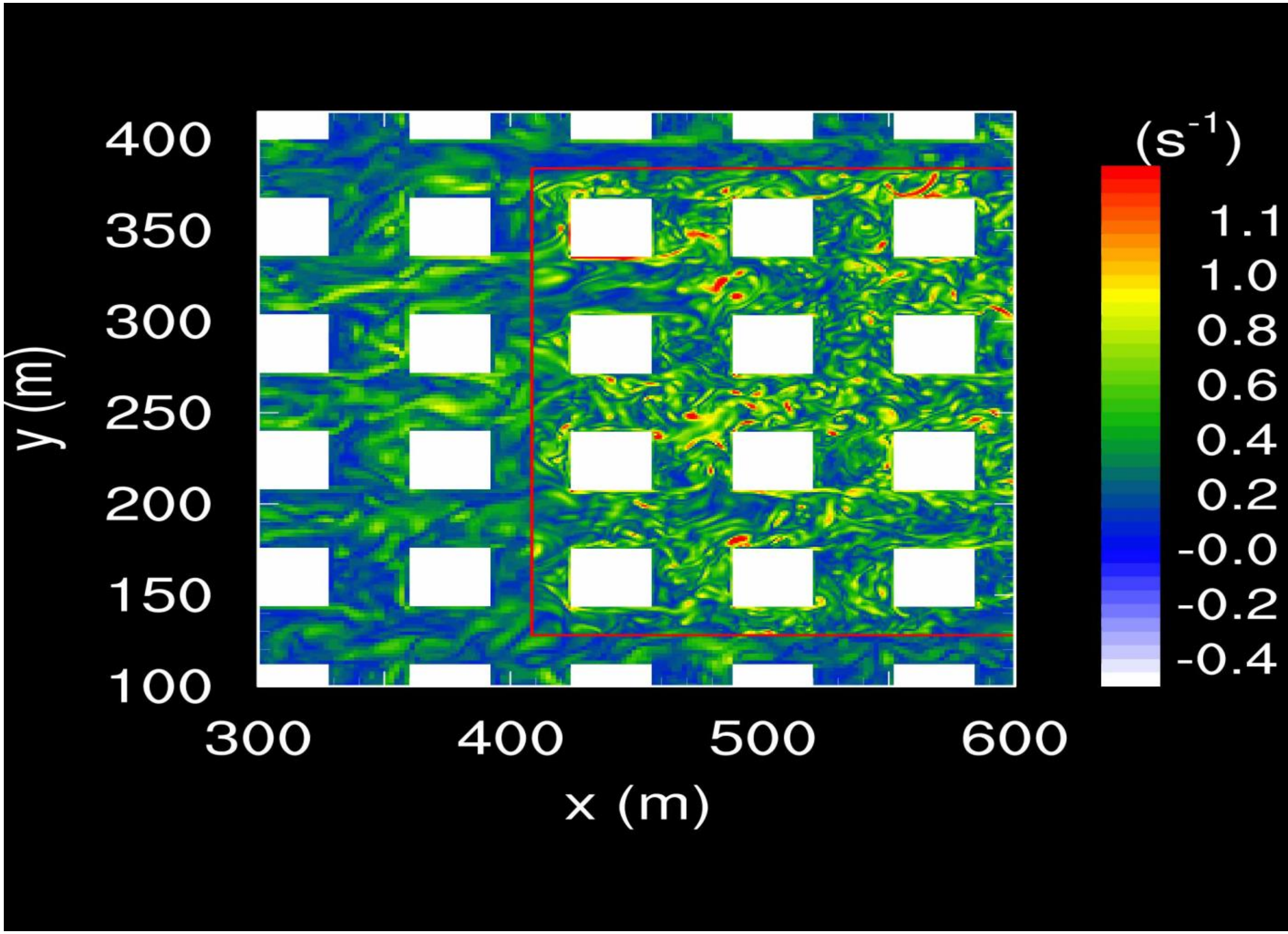
Spectra starts to overlap after more than 1 km behind the "inflow"
→ **Flow adjustment after 1-2 km**



Attention: Flow adjustment zone significantly increase with increasing grid spacing ratio

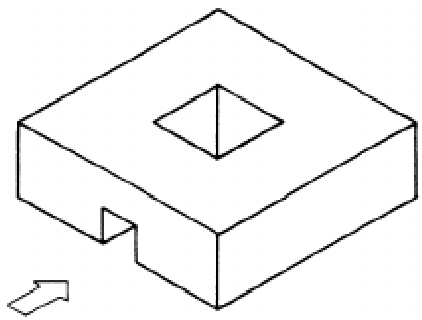
Urban boundary layer – pollutant dispersion on the city block scale

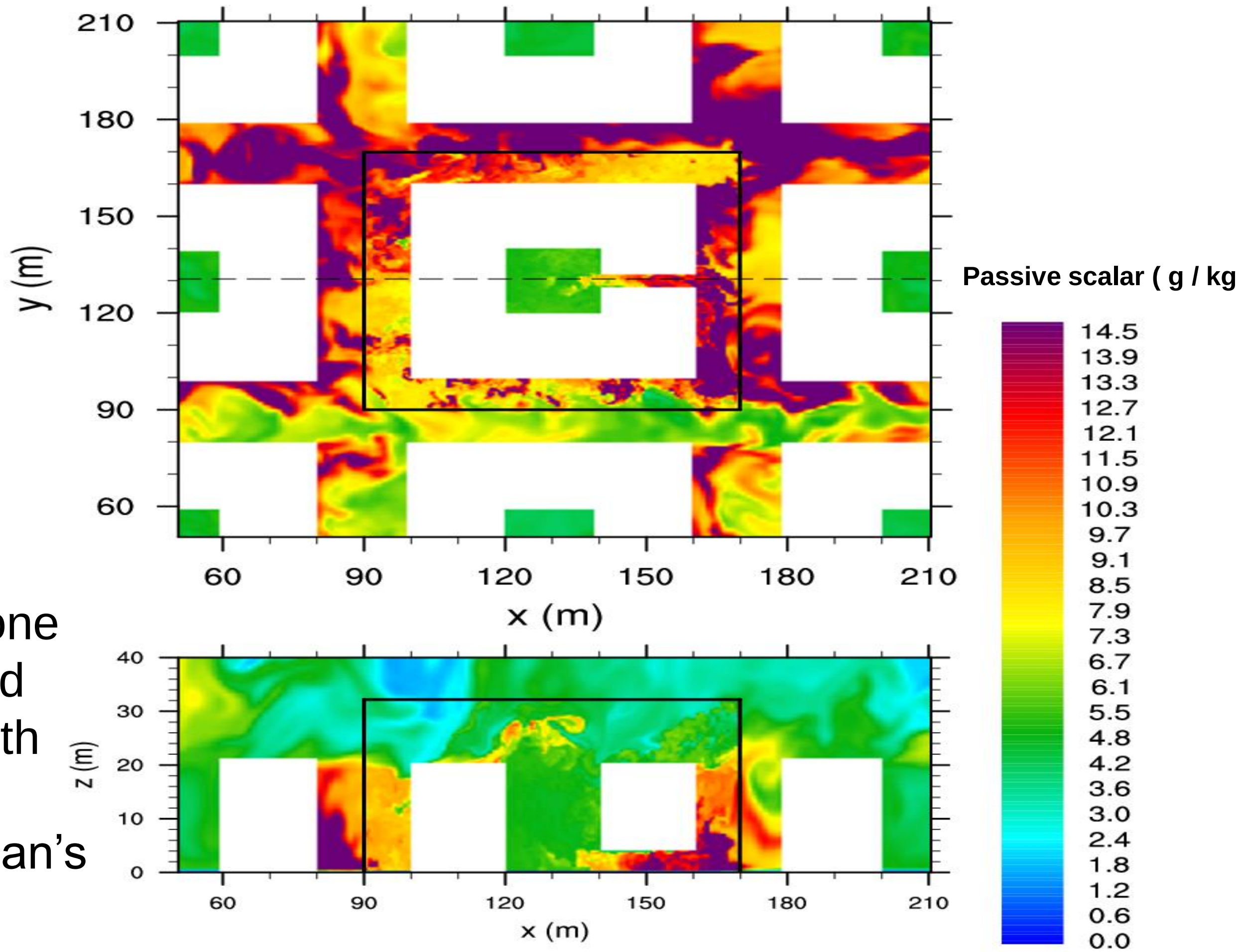
- Cutout of a nested pollutant dispersion simulation within an idealized city block
- Absolute value of rotation is shown
- Background wind from left to right
- Child domain shows much more details of the flow
- Vortices are often generated at the building's edges
- Back flow behind buildings



Urban boundary layer – pollutant dispersion on the building scale

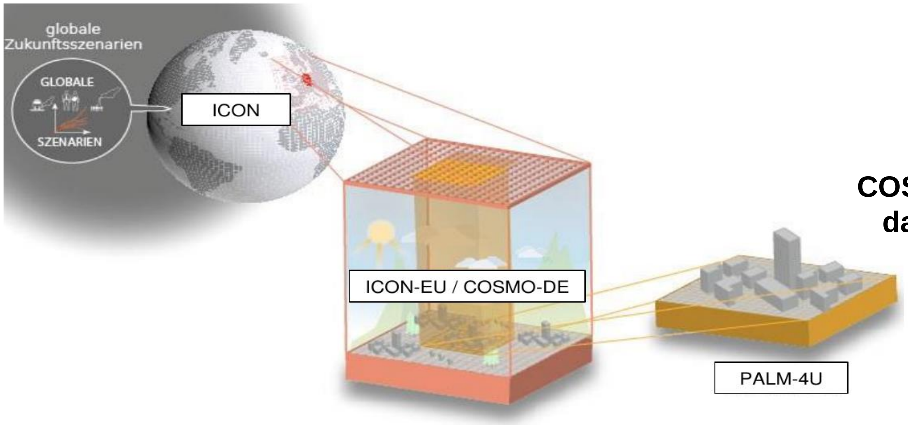
- How do courtyards and openings affect pollutant dispersion?

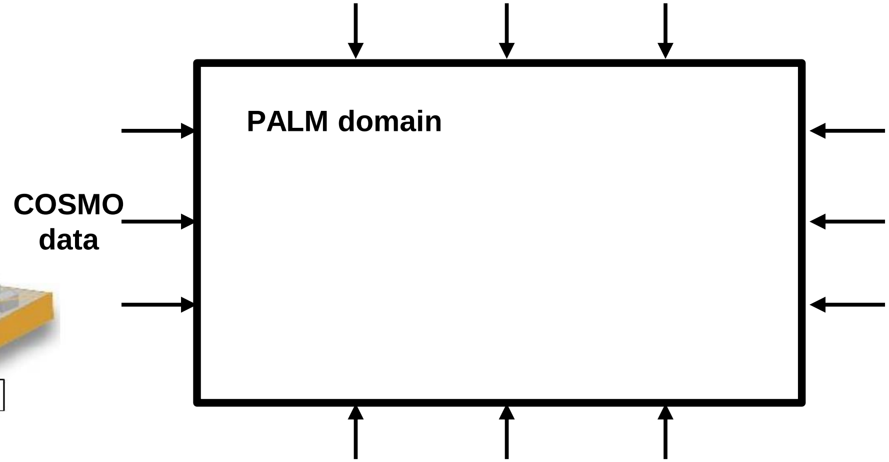




- The environment of just one single building was nested
- Flow features together with concentration enable an evaluation of the pedestrian's well-being

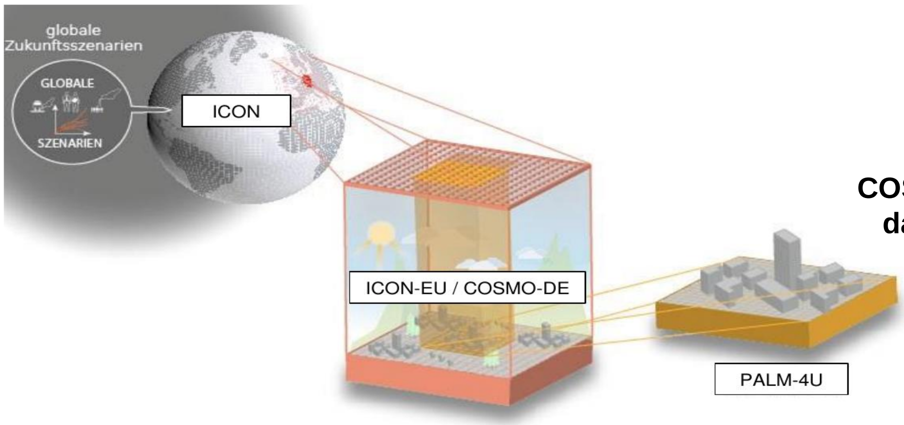
Mesoscale nesting within larger-scale models

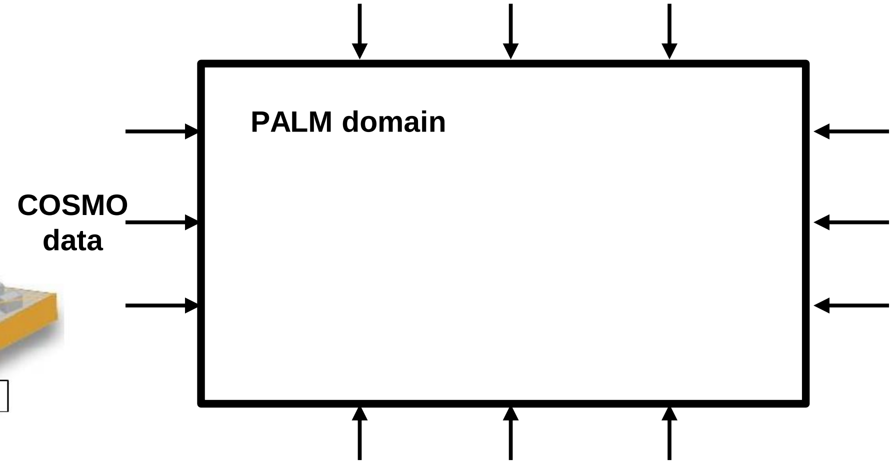




- Idea: consider changes in synoptic conditions
- Nest the model domain within larger-scale models, where the larger-scale model runs in advance
- Provide pre-processed (offline) time-dependent data from, e.g. COSMO or WRF model, at lateral and top boundaries of PALM domain via dynamic input file
- Boundary data is interpolated linearly in time

Mesoscale nesting within COSMO model

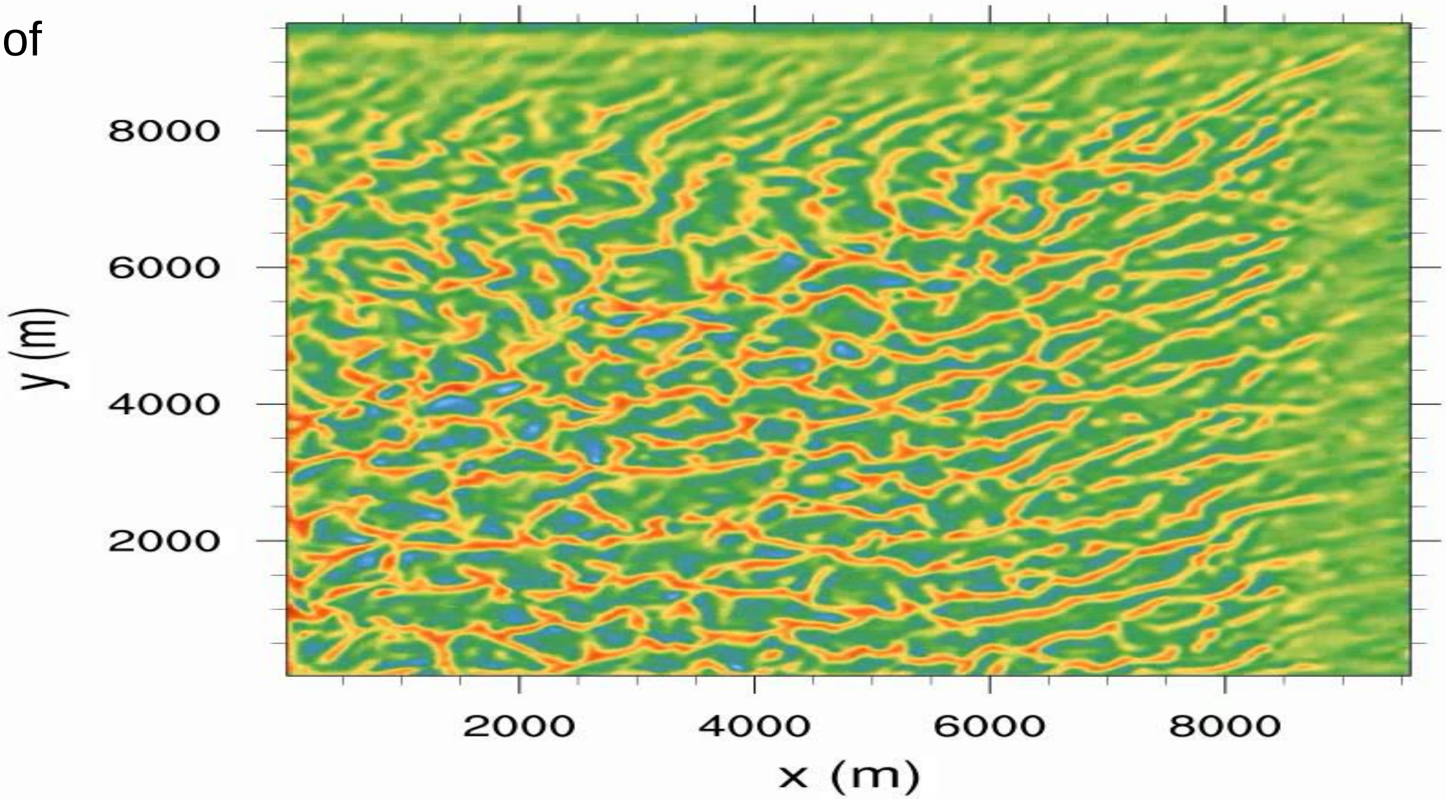




- COSMO-interface INIFOR - pre-processing tool (developed by Eckhard Kadasch, DWD)
  - **INI**tialization and **FOR**cing
  - Interpolation of COSMO data onto Cartesian grid
  - Provides initialization data of wind, temperature, humidity and soil temperature / moisture
  - Provides time-dependent information on boundaries (lateral and top) for all relevant quantities
  - Data is stored in “dynamic driver”, e.g., **`example_dynamic`**
  - Synthetic turbulence generator at lateral boundaries required to initiate turbulence

Mesoscale nesting in COSMO model

- COSMO forcing of 8 hrs
- Flow adjustment zone is clearly visible
- Synoptic wind comes from northeast and turns to eastern direction
- Combination of band-like and cellular patterns



Remarks

- Open points self nesting:
  - CBL test cases show secondary circulation in time-averaged fields with an upward motion in the child domain and downward motions at the child boundaries (is an inherent feature of the nesting)
  - Influence of coupling mode (one-way, two-way) must be analyzed in detail.
  - Less experiences in RANS-RANS and RANS-LES nesting
  - Particle nesting needs further testing especially regarding the treatment of stochastic subgrid-scale particle speeds
  - Elevated child domains (under development)

- Open points offline/mesoscale nesting:
  - Pre-processing tools currently exist for COSMO or WRF. Interfaces for further models (e.g. ICON) currently under development.

- Documentation at:
  https://palm.muk.uni-hannover.de/trac/wiki/doc/tec/nesting
  https://doi.org/10.5194/gmd-14-3185-2021 (Self-nesting - technical paper)
  https://doi.org/10.5194/gmd-14-5435-2021 (Mesoscale nesting - technical paper)

PALM online:
https://palm.muk.uni-hannover.de

Our YouTube channel:
youtube.com/user/palmhannover