Author's response

First we would like to thank the anonymous referees for the evaluation of our manuscript and for the valuable comments, suggestions and corrections. We would like to answer to the questions and concerns raised by them and provide additional information on the changes made in the revised version. All modifications of the original manuscript are highlighted in red.

Author's response to Anonymous Referee # 1

(2) Specific Comments

Referee #1 comments "The 'number of cells owned by each task' as a robust metric for the scalability limit does not appear similarly convincing to me". In the revised version, we kept the transition zone as the area between 600 and 150 owned cells per task, but removed the argumentation of it being a robust limit for the breakdown of the parallel efficiency. Instead, we focussed on the minimum in measured absolute runtimes, i.e., on the number of cells owned per task below which the absolute runtimes increase again. For this definition, there is indeed a fairly robust number of 150 cells per task for all three test cases in Section 2 for the Intel-based systems. For the Blue Gene system Juqueen, the picture is less clear, since two of the three test cases (120km, 100-25km) are too small for application on the machine. We weakened our conclusion and now state that for the three test cases considered here, 150 cells per task is a good estimate below which the absolute runtimes increase and thus a parallelisation on larger numbers of tasks becomes pointless. This also fits better to Table G2, in which we compare "cheapest" and "fastest" parallelisations for the different meshes. It is worth noting that the numbers mentioned here are supported by the additional plots requested by Anonymous Referee #2, which display the scaling of the 120km and the 100-25km test case in the same way as it was already done for the 60-12km test case (previous Figure 12, now Figure 6).

We added the missing cases to the Tables D1-F1 and incorporate all the corrections mentioned as "more specific comments" in the respective places.

Regarding the question on page 7007, lines 18-22: We agree that choosing two cases within the transition zone is not particularly useful. We therefore repeated two profiling exercises for the 60-12km mesh with 4096 tasks and 8192 tasks on Juqueen (130 and 65 owned cells per task). For the latter one, the impact of halo-exchange on the parallel performance is clearly visible as it requires longer (in terms of total runtime) to complete than the 2048-task and the 4096-task run. These three runs are now used for the profiling. The comparison of the three runs is summarised in Section 2.7 (previously Section 2.6) and Table G1. This table was introduced instead of the pie chart plots (old Figure 14) as requested by Anonymous Referee #2.

(3) Technical corrections

Discussion of units (tasks, nodes) and appropriate measures for the parallel performance: The unit “tasks” is used for the scaling experiments in section 2. Section 4, the extreme scaling experiment, now consistently uses “nodes” as unit. Table H2 now lists the number of nodes and the number of tasks for the cheapest runs as well as for the fastest runs to facilitate the interpretation and comparison of the results.

Author’s response to Anonymous Referee # 2

(1) General Comments

We appreciate the concerns raised by the referee regarding the significance of the conclusion made, based on a single 11-month comparison of MPAS with and without grid refinement. We placed the corresponding Section 3 in the context of a pilot study and rephrased the conclusions drawn from it in the abstract, the main text (Section 3) and the Conclusions (Section 5) accordingly. We shortened Section 3 by removing the paragraph on re-analysis data, which does not contribute directly to the overall conclusions. We would also like to bring to the referee's attention that we have already applied for computational resources to conduct an ensemble modelling experiment with various MPAS meshes for West Africa with the focus on seasonal predictions. With regards to the spin up time for soil, we have conducted several experiments with WRF and with WRF-Hydro (see interactive discussion), which suggest that a spin up time of 2 years is sufficient for continental West Africa, and that even a spin up time of 1 year might be acceptable. A detailed discussion of this topic is beyond the scope of this paper and therefore we mention it only briefly in the manuscript.

(2) Specific Comments

Section 2.1: We condensed the important information into one table (new Table 1). Section 2.1 now summarises the main features of the four HPC sites in brief. Further, Section 2.6 has been added and the relevant discussion of the differences between the performance of MPAS on the four HPC systems has been moved and condensed there. This also includes the old Figure 12, which compares the total runtime as function of number of tasks (now Figure 6 and for all three test cases, as suggested).

Section 2.3: Following the referee's suggestion, we fitted a modified Amdahl law to the three test cases, separately for the Intel architecture and the Blue Gene architecture. We added this information to the manuscript in form of a table (new Table 2), and plotted the so-obtained fitting curves in the new Figure 6 for all three test cases. This is now part of the new Section 2.6.

Section 2.2: A model top of 30km with 41 vertical levels on a hybrid height-based terrain-following coordinate is used in this study. Being developed out of a regional climate modelling context, the version of MPAS-A used here has not been adapted to extend higher into the atmosphere. In more recent versions, the model top has been extended to 1mb (around 42km). We agree that a higher model top (and the implementation of a coupler between the three MPAS components) would be required if MPAS was to be used as a full earth system model in the future. This information has been added in brief to the introduction (Section 1).

Regarding the measurements of the performance data: The performance data was derived from two to three test runs, depending on how close the first two measurements were: For sufficiently close results from the first two runs, the average was taken from these runs. Otherwise, a third run was conducted and the average was taken from all three runs. This information has been added as last sentence in Section 2.2.

Page 7000, line 27: Indeed, we mistakenly stated that the relationship is linear. We fitted a power law commvol = A * tasks**B to the data for each test case and obtained an exponent B=0.52, independent of the mesh. This exponent of 0.52 agrees with what we calculated from the uniform mesh plot. We added these fits to the three panels in the old Figure 8 (now Figure 3) and corrected the text accordingly.

Use of figures: We followed the referee's suggestion and merged Figures 1-3. Old Figures 4, 6 and 11 are combined to the new Figure 2. Old Figure 12 for the 60-12km mesh and equivalent figures for the remaining two test cases are now combined into new Figure 6 for an easier comparison. Old Figure 5 and 10 were merged into new Figure 4.

Section 2.4: We shortened the discussion and dropped old Figure 9, but kept old Figure 7 (now Figure 5). We feel that the aspect of contiguous and non-contiguous partitions and the variability in the communication properties for individual patches (not existent for regular grids) are worth mentioning, even though they have no significant influence in this study.

Section 2.6: We dropped panels (d)-(f) of old Figure 13 (new Figure 7) as suggested and modified the text accordingly. We also replaced the pie charts (old Figure 14) with new Table G1, which summarises the percentage of time spent for the different categories (physics, communication, dynamics, I/O). As suggested by Anonymous Referee #1, rather than presenting the details for two model runs within the transition zone, we kept the 2024-task run for the 60-12km mesh and added two runs with 4096 and 8192 tasks on the same mesh. For the latter one, the impact of halo-exchange on the parallel performance is clearly visible as it requires longer (total runtime) to complete than the 2048-task run and the 4096-task run. These three runs are now used for the profiling on Juqueen. The comparison of the three runs is summarised in Section 2.7 (previously Section 2.6) and the new table G1.

Section 3 and model spin up time: see above

Section 3: Orography in (old) Figure 15 (now Figure 8): This issue was corrected.

Section 3 and general comment on colour maps and line plots: Old Figures 15 to 19 are now Figures 8 to 12. We introduced proper colour maps for the filled contour plots and labelled axes and colour bars with the appropriate units as suggested by the referee. Also, the line plots now use different strokes for each line.

Section 4.3, NaNs: The default compiler flags used in MPAS for Blue Gene systems do not enable floating-point error trapping via the “-qfltttrap” option. Without this flag, floating-point exceptions like a division by zero or the use of a NaN as an operand do not generate signals that would cause the program to halt. We updated the discussion in Section 4.3 to emphasise that for this reason, the model does not simply halt upon encountering a NaN.

Section 4, discussion of units (tasks, nodes): The unit “tasks” is used in the scaling experiments in Section 2. Section 4, the extreme scaling experiment, now consistently uses nodes. Table H2 now lists the number of nodes and the number of tasks for the cheapest runs as well as for the fastest runs to facilitate the interpretation of the results.

Section 4, old Figure 21 (new Figure 13): We adapted the figure so that each column is labelled by category and time spent in seconds.

(3) Technical Corrections

Both typos are corrected in the revised version of the manuscript.

Other modifications and suggestions from the journal and other discussion members

Following the Executive Editor’s suggestion, we added a version number to the title and also introduced a code availability section at the end of the manuscript (new appendix A). The model code and test cases are available as http://dx.doi.org/10.1594/PANGAEA.849428.

The last sentence of the first paragraph in the abstract now reflects better our motivation to work with MPAS in a climate modelling context.

Section 1 (Introduction): Page 5 now contains additional minor details on the MPAS model.

Section 2.2 (MPAS-A code): Page 10 now provides a formula to calculate the number of ghost/ halo cells for a given number of owned cells, used later in Section 2.7 (previous Section 2.6).

The discussion of interconnect speeds on page 13 (Section 2.3, regular 120km grid) has been improved for better consistency. This also includes the discussion of the 420-task run on on Jtest-full and Jtest-half, which was mentioned by Anonymous Referee #1.

Section 2.6 was introduced to summarise the differences of the four HPC systems in one place, rather than being split across other sections. This also includes the new Figure 6 (previous Figure 12) and the fits to the measured runtime curves using a modified version of Amdahl's law (see also Table 2).

Further, the spelling of “Blue Gene” (rather than “Bluegene”) was corrected in the paper, and the Yellowstone supercomputer is now referred to as the NCAR-Wyoming Supercomputing Center's (NWSC) Yellowstone machine.

Manuscript prepared for Geosci. Model Dev. Discuss.
with version 2015/04/24 7.83 Copernicus papers of the LaTeX class copernicus.cls.
Date: 13 November 2015

Towards convection-resolving, global atmospheric simulations with the Model for Prediction Across Scales (MPAS) v3.1: an extreme scaling experiment

D. Heinzeller[1], M. G. Duda[2], and H. Kunstmann[1,3]

[1]Karlsruhe Institute of Technology, Institute of Meteorology and Climate Research, Kreuzeckbahnstr. 19, 82467 Garmisch-Partenkirchen, Germany
[2]National Center for Atmospheric Research, Mesoscale and Microscale Meteorology Laboratory, 3090 Center Green Drive, Boulder, CO 80301, USA
[3]University of Augsburg, Institute of Geography, Alter Postweg 118, 86159 Augsburg, Germany

Correspondence to: D. Heinzeller (heinzeller@kit.edu)

Abstract

The Model for Prediction Across Scales (MPAS) is a novel set of earth-system simulation components and consists of an atmospheric model, an ocean model and a land-ice model. Its distinct features are the use of unstructured Voronoi meshes and C-grid discretisation to address shortcomings of global models on regular grids and of limited area models nested in a forcing data set, with respect to parallel scalability, numerical accuracy and physical consistency. This concept allows to include the feedback of regional land use information on weather and climate at local *and* global scale in a consistent way, impossible to achieve with traditional limited area modelling approaches.

Here, we present an in-depth evaluation of MPAS with regards to technical aspects of performing model runs and scalability for three medium-size meshes on four different High Performance Computing sites with different architectures and compilers. We uncover model limitations and identify new aspects for the model optimisation that are introduced by the use of unstructured Voronoi meshes. We further demonstrate the model performance of MPAS in terms of its capability to reproduce the dynamics of the West African Monsoon and its associated precipitation in a pilot study. Constrained by available computational resources, we compare 11-month runs for two meshes with observations and a reference simulation from the Weather Research & Forecasting (WRF) model. We show that MPAS can reproduce the atmospheric dynamics on global and local scale in this experiment, but identify a precipitation excess for the West African region.

Finally, we conduct extreme scaling tests on a global 3 km mesh with more than 65 million horizontal grid cells on up to half a million cores. We discuss necessary modifications of the model code to improve its parallel performance in general and specific to the HPC environment. We confirm good scaling (70 % parallel efficiency or better) of the MPAS model and provide numbers on the computational requirements for experiments with the 3 km mesh. In doing so, we show that global, convection-resolving atmospheric simulations with MPAS are within reach of current and next generations of high-end computing facilities.

1 Introduction

The weather- and climate-modelling community is currently seeing a shift in paradigm from limited area models towards novel approaches involving global, complex and irregular meshes. Yet, regional models are commonly used in numerical weather prediction and to study past, current and future climate at high spatial and temporal resolution over areas of specific interest. A wealth of such models exists nowadays, which differ in their discretisation of the computational grid, their implementation of the numerical solvers, their parameterisation of physical processes, and most notably in their simulation results (e.g., Smiatek et al., 2009; Nikulin et al., 2012). Despite these differences, they share the common principle of nested modelling: Regional climate information is generated by supplying a set of initial conditions as well as time-varying lateral boundary conditions (LBCs; large-scale atmospheric fields such as wind, temperature, geopotential height and hydrometeors) and lower boundary conditions (sea-surface temperature, sea ice) to the regional model. The idea behind this approach is that the LBCs keep the regional climate model (RCM) solution consistent with the forcing atmospheric circulation, while small-scale patterns are generated with higher accuracy due to the increase in temporal and spatial resolution. Sub-grid scale processes in the RCM are included through parameterisations, which can be entirely different from those of the forcing global circulation model (GCM).

Supplying lateral boundary conditions to nested models can cause severe problems, up to the point where the RCM solution becomes inconsistent with the forcing data (Davies, 1983; Warner et al., 1997; Harris and Durran, 2010; Park et al., 2014). Starting off as an initial-value problem, the RCM solution gradually becomes a boundary value problem, which from a mathematical point of view represents a fundamentally ill-posed boundary value problem (Staniforth, 1997; Laprise, 2003). This is less of an issue in the context of numerical weather prediction (NWP), where typical model runtimes are 3 to 15 days, than in seasonal forecasting (weeks to months) and in regional climate modelling (years to centuries).

A common problem for both short- and long-term forecasting is that the solution of the RCM seems to vary with the size of the computational domain, as well as location and season (Caya and Biner, 2004; Leduc and Laprise, 2008; Caron, 2013). Several authors have shown that nudging techniques (grid or spectral nudging) towards the large-scale features of the forcing model can reduce these adverse effects, but that they can also hide model biases (von Storch et al., 2000; Miguez-Macho et al., 2004).

Further, the technique of grid nesting introduces discontinuities in spatial resolution between the regional model and the coarser-grid driving model, as well as between the nests within the regional model itself. For a typical refinement ratio of 3, two-thirds of the spatial wave-number spectrum present in the fine mesh are absent in the coarse mesh (Park et al., 2014). This implies that (a) these features must be spun up for inflows into the higher-resolution domain, and that (b) these wave numbers are reflected at the domain boundary for outflows from the high-resolution domain to the coarser domain. To address the latter issue, filters that are efficient over a large range of wave numbers are required. The temporal interpolation required by nesting can introduce further numerical artefacts, in particular when interpolating forcing LBCs, usually available at 3–6 h timesteps, to the model integration time step of the high-resolution domain (typically 6 s per 1 km grid size).

One obvious solution is to avoid using LBCs and nesting by running a global model at the resolution required for the area of interest. This, however, is prohibitively expensive or simply not feasible, even on the latest generations of supercomputers. An intermediate approach therefore is to run a global model at a moderate resolution and use a smooth mesh transition on a variable-resolution grid, where filters are efficient at the local scale of the corresponding grid cell (Ringler et al., 2011). Beside the here-discussed MPAS model, few other recent developments such as ICON (ICOsahedral Non-hydrostatic model, Zaengel et al., 2015), adopt this strategy. Applying such models for mid- and long-term regional climate simulations has only recently become possible and requires substantial computational power.

The Model for Prediction Across Scales (MPAS)[1] is a recent numerical modelling framework that includes an atmospheric model MPAS-A (Skamarock et al., 2012), an ocean model MPAS-O (Ringler et al., 2013), and a land-ice model MPAS-LI. The MPAS model is a collaborative project, led by the National Center for Atmospheric Research (NCAR) and the Los Alamos National Laboratory (LANL). The three components of MPAS in principle form a so-called Earth System Model (ESM), but a coupler between them is not yet available. A common feature of the three MPAS constituents are unstructured, centroidal Voronoi meshes (spherical centroidal Voronoi tessellations, SCVTs Du et al., 2003), which allow the generation of global, irregular, variable-resolution meshes with smooth transitions between areas of different refinement.

The atmospheric model MPAS-Atmosphere is a global, fully-compressible non-hydrostatic model using finite-volume numerics. Based on the Voronoi mesh, the model uses a C-grid staggering for the state variables (i. e., wind components are modelled at the faces of every cell, and the prognosed component of the wind is orthogonal to the cell face) as described in Thuburn et al. (2009) and Ringler et al. (2010) (see Fig. 1, upper panels, for illustration). The governing equations can then be cast in a way such that energy, momentum and water vapour content are conserved (more precisely: potential temperature, mass and scalar mass; see Skamarock et al., 2012, Sect. 2). The MPAS-A model builds on existing, well-established techniques of the Advanced Research Weather Research and Forecasting model (WRF-ARW, Skamarock et al., 2008), for example the split-explicit time integration scheme for the treatment of gravity waves and horizontally propagating acoustic waves. It also contains a subset of WRF's physics parameterisations that are suitable for climate modelling purposes. While WRF uses a terrain-following hydrostatic pressure coordinate for the vertical discretisation, MPAS employs a height-based terrain-following vertical coordinate. The latter discretisation reduces artificial circulations caused by inaccuracies in the horizontal pressure gradient term (Klemp, 2011). It should be noted here that because MPAS was developed out of a regional climate modelling context, the typical model tops in MPAS-A are around 30 km to 42 km, while current earth system models work with model

[1] http://mpas-dev.github.io

tops of up to 100 km. At the time of writing, work is underway to increase the model tops in MPAS-A for future applications as earth system modelling component.

Until recently, advances in computational power following Moore's Law were mainly driven by transistor speed and energy scaling, as well as by micro-architecture advances. Physical limitations and practical energy concerns will create new challenges for continued performance scaling in the coming decades. In consequence, large-scale parallelism and the use of accelerators will be required to achieve performance and energy efficiency (Borkar and Chien, 2011). Hence, a key aspect of modern numerical codes is their ability to scale on massively parallel systems. The quasi-uniform centroidal Voronoi meshes used by MPAS are similar to icosahedral (hexagonal) meshes and can provide nearly uniform resolution over the globe, as opposed to latitude–longitude grids that require polar filtering to overcome the issue of converging grid lines at the poles. Grids requiring polar filtering or spherical transform methods do not scale very well on massively parallel systems (Skamarock et al., 2012). With MPAS, an efficient parallelisation can be achieved by aligning all grid cells in a 1-D array, with the vertical coordinate stacked on top as the second dimension (MacDonald et al., 2011). Good scaling has been achieved in early weak and strong scaling tests. However, a thorough investigation of the scalability of MPAS on parallel and massively parallel systems has not yet been conducted.

In this study, we investigate the performance of MPAS for different problem sizes on four HPC facilities in Europe. For each problem, strong scaling tests are conducted on all four platforms, which cover a range of different architectures to reflect the large variety of computational systems available for research. Additionally, we conduct extreme scaling tests using a 3 km global mesh to study the scalability of MPAS up to nearly half a million tasks and to demonstrate that global, convection-resolving simulations are becoming possible. We explore the limits of the MPAS model when its parallel efficiency breaks down and identify opportunities for improvement. We further derive estimates on the feasibility to conduct longer runs at convection-resolving resolution on current HPC facilities.

We also assess the quality of the MPAS model output in terms of its accuracy for climate modelling. We have chosen to study the particular problem of reproducing the characteris-

tics of the West African Monsoon (WAM). The WAM is the most prominent feature of the West African climate and accounts for the majority of the annual precipitation. Differential heating of the ocean and the land surface cause a seasonal change of the large-scale wind systems during boreal summer, which results in the migration of the inter-tropical convergence zone (ITCZ) and the associated rain band northwards over the West African continent. The WAM is driven by a complex and not yet fully understood interplay of various dynamical features (e.g., Sultan et al., 2003; Grist and Nicholson, 2001; Nicholson and Webster, 2007). Global circulation models (GCMs) often fail to reproduce this annual movement of the ITCZ due to their limited temporal and spatial resolution (e.g., Hourdin et al., 2010; Sylla et al., 2010). Despite their deficiencies discussed above, regional climate models can improve the representation of precipitation in comparison to their forcing data set (Nikulin et al., 2012; Klein et al., 2015). Variable-resolution meshes permit resolving the region of interest (greater West Africa in this case) at high resolution, while keeping the model aligned with large-scale features outside of this area. It is hoped that this will lead to an improvement of the representation of the WAM. It also opens up the possibility to study processes such as the teleconnection between the Indian Monsoon and the West African Monsoon (Rodwell and Hoskins, 1996), or the impact of land use changes on weather and climate in a consistent approach.

The paper is organised as follows: in Sect. 2, we introduce the HPC facilities used for this study, provide details about the MPAS-A code, and present the scaling experiments with moderate problem sizes. We continue in Sect. 3 with an analysis of the physical accuracy of the MPAS model in comparison to observational data and data from own regional climate modelling experiments. Section 4 is devoted to the extreme scaling tests, and Sect. 5 summarises our findings and gives an outlook on future modelling activities.

2 Scaling experiments for moderate problem sizes

We perform strong scaling experiments for three different meshes and on four HPC facilities in Europe. The problem sizes range from a regular 120 km mesh with 40 962 cells as the

smallest problem to a large, variable-resolution 60–12 km mesh with 535 554 grid cells. An intermediate test case with a variable-resolution 100–25 km mesh with 163 842 grid cells is studied as well.

2.1 HPC facilities

Two of the four HPC systems, the Très Grand Centre de Calcul (TGCC) Curie and the Forschungszentrum Jülich (FZJ) Juqueen, belong to the largest machines in Europe and are part of the PRACE Tier-0 pool[2]. The technical specifications for each of the systems are summarised in Table 1, a brief summary is given in the following.

TGCC Curie.[3] The TGCC Curie went into service in 2012 and consists of 360 "fat nodes" and 16 "hybrid nodes", not used in this study, and 5040 "thin nodes" with 2 eight-core Intel Sandy Bridge CPUs. An InfiniBand QDR network is used for both the compute network and the I/O to the global LUSTRE file system. With 3 different node types, Curie addresses a wide range of scientific challenges.

FZJ Juqueen.[4] The FZJ Juqueen is an IBM Blue Gene/Q system and was installed in 2012/13. It hosts 28 racks with 1024 nodes per rack and 16 cores per node, which totals to 458 752 physical cores. Simultaneous multi-threading (SMT) is supported by the hardware, but not used in this study due to the lack of threading in MPAS-A (see below). A 5-D Torus interconnect is used as compute network, while I/O is redirected to dedicated I/O nodes using a 10 Gb Ethernet to connect to the GPFS file system. With a large number of relatively slow CPUs and a small memory per core, Juqueen most resembles the future massively parallel systems described above. As such, porting and scaling experiments of numerical codes onto this architecture are as challenging as instructive for future applications.

[2] http://www.prace-ri.eu/prace-resources

[3] http://www-hpc.cea.fr/en/complexe/tgcc-curie.htm

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

SCC ForHLR1.[5] The ForHLR1 is the most recent addition to the SCC's high performance computing systems, installed in September 2014. It hosts different types of nodes to cater for the various needs of the modelling community: the workhorse for parallel applications, and used in this study, are 512 "thin nodes" with 2 ten-core Intel Ivy Bridge CPUs, connected via an Infiniband FDR interconnect. A central LUSTRE filesystem is attached to the nodes, using the same Infiniband interconnect for I/O as for the compute network.

FZJ Juropatest.[6] The FZJ Juropatest cluster is a small but cutting-edge prototype system and consists of 70 T-Platform V210s blades with 2 fourteen-core Intel Haswell processors. The MPI communication is realised over Infiniband FDR, and the I/O to the central GPFS file system is routed via 10 Gb Ethernet. While optimising the MPAS model for the Haswell features and instruction sets is beyond the scope of this study, it will become an inevitable step in future model development and tuning.

On Juropatest, we conduct two sets of runs for each of the test cases: for the first set (*Jtest-half* in the following), we use only one of the two available fourteen-core CPUs in each node, which implies a similar number of cores per node for Curie and Juropatest or, in other words, a similar number of nodes for the same total number of tasks. In this configuration, each task is bound to one core on the node. For the second set (*Jtest-full* in the following), we use both CPUs, i. e., 28 cores on each node to exploit the capabilities of the Juropatest system and possible memory bandwidth limitations of MPAS-A.

2.2 MPAS-A code

For the strong-scaling studies in this paper, we use MPAS-A v3.1, released on 24 November 2014. This release of the model employs a horizontal domain decomposition for parallel execution, and parallelisation is implemented using MPI only; in this version of the code, no threading is used. The MPAS code is written almost entirely in Fortran 2003, with a few minor parts written in C. The MPAS build system uses only the `make` utility, with set-

[5] http://www.scc.kit.edu/dienste/forhlr.php

[6] http://www.fz-juelich.de/ias/jsc/EN/Expertise/Supercomputers/JUROPATEST/JUROPATEST_node.html

tings for different compilers and architectures described as different targets in the top-level `Makefile`; see Appendix B for the compiler flags used in this study.

The optimal parallelisation and distribution of the cells of the Voronoi mesh for a given number of tasks is treated as a graph partitioning problem. The dual mesh of a Voronoi tessellation is a Delaunay triangulation, which immediately provides the connectivity graph for the primal (i. e., Voronoi) cells in the mesh. In MPAS, the graph partitioning is computed as a separate pre-processing step, for which the METIS software is used[7]. An optimal partitioning distributes equal work (by proxy, the number of cells) to each task while minimising the edge cut (assumed to model the communication between tasks). METIS uses a multilevel k-way partitioning scheme, which produces partitions of comparable quality to traditional multilevel bisection algorithms and is about two orders of magnitude faster (Karypis and Kumar, 1998). The resulting graph partitioning can be critical for the model performance due to, for example, a large overhead of communication and computational imbalances between the individual partitions.

At start-up, the MPAS-A model reads a file that assigns Voronoi cells to each of the MPI tasks according to a partitioning produced by METIS. The set of cells assigned to an MPI task is referred to as a "block", and the cells in this assignment are referred to as the "owned" cells. The dynamical solver in MPAS-A requires stencils of cells in order to apply various operators, and as part of the model start-up, referred to internally as the "bootstrapping" process, a pre-determined number of layers of halo cells (sometimes referred to as "ghost" cells in other modelling systems) are added around each block. Although the number of halo cells can vary between different MPAS models, as illustrated in Fig. 1, lower left panel, MPAS-A adds two layers of halo cells around each block of cells. A lower bound for the number of halo cells N_h for a given number of owned cells N_o can be estimated as

$$N_\mathrm{h} = \pi \left(\sqrt{\frac{N_\mathrm{o}}{\pi}} + 2 \right)^2 - N_\mathrm{o} \,. \tag{1}$$

[7] http://glaros.dtc.umn.edu/gkhome/views/metis

At points in the MPAS-A dynamical solver where current values of fields in halo cells are required, values are communicated between tasks, from owned cells to ghost cells, with point-to-point MPI communications.

An important aspect and common bottleneck in numerical weather prediction and climate modelling is disk I/O, since large 4-D fields such as temperature, geopotential height, or wind components need to be written to disk frequently. In MPAS v3.1, I/O is facilitated by the parallel I/O library PIO v1.7.1, a wrapper with an easy-to-use API that encapsulates the complexity of parallel I/O for a number of supported formats: binary, serial NetCDF[8], Parallel-NetCDF[9], and recently (since v.1.9.14) parallel NetCDF-4 through PHDF5[10] (Dennis et al., 2013). PIO is compiled without further optimisation (standard settings) on all four machines. The HDF5, NetCDF and Parallel-NetCDF libraries are provided as modules on all four systems.

Unless stated otherwise, all experiments are conducted with double precision floating point precision, 41 atmospheric levels, 4 soil levels, a full suite of physics and dynamics (see Appendix C for details), and standard disk I/O. Each experiment is run for 24 h model time, during which an initial conditions file is read (`init.nc`), diagnostic output files are written every 3 h (`diag.nc`), and a final restart file and a comprehensive output file are written at the end (`restart.nc`, `output.nc`). The model integration time step depends on the grid resolution and is mentioned in the individual sections below. Note that for variable resolution meshes, the global time step is determined by the smallest grid size. By default, all tasks participate in the parallel I/O. Each experiment is repeated once or twice, depending on how close the measured runtimes are, to account for fluctuations of single experiments.

2.3 Regular 120 km grid

The first and smallest test case consists of a global, regular 120 km mesh with 40 962 grid cells, which is roughly comparable in resolution to a 284×142 latitude–longitude grid. It is

[8] http://www.unidata.ucar.edu/software/netcdf

[9] http://trac.mcs.anl.gov/projects/parallel-netcdf

[10] http://www.hdfgroup.org/HDF5

thus in the range of current earth system models. A model integration time step of 150 s is adopted. For a resolution of 120 km, this is an extremely conservative setting (1.25 s per km grid size) for MPAS-A, which implies 576 integration time steps for a 24 h test run, compared to 120–144 integration time steps for typical values of 5–6 s per km grid size. This increases the time spent for the actual parallel integration relative to that for model initialisation and disk I/O. The decomposition of the 120 km grid for 64 tasks is illustrated as an example in Fig. 1, lower right panel, while Fig. 2, left panel, shows the scaling plots on the four HPC facilities described above. For an easier comparison of the scalability of the different test cases, the scaling is displayed as parallel efficiency (i. e., the ratio of real scaling and ideal scaling) vs. the number of tasks (bottom horizontal axis) and number of cells owned per task (top horizontal axis). Table D1 provides further details about the scaling, whereas Table H1 summarises the size of the files to be read and written during one model run.

Previous scaling tests on the NCAR-Wyoming Supercomputing Center's (NWSC) Yellowstone machine[11] suggest that for regular meshes, the parallel efficiency of MPAS-A is correlated with the number of cells owned per task. Considering the time required for the solver only, i. e., neglecting the setup costs and the disk I/O, a parallel efficiency of close to 70 % is obtained for more than 160 cells per task. Here, we include the setup costs (bootstrapping and reading of initial conditions file) and the output to disk in the scaling to emphasise the importance of all aspects of the system – from filesystem performance to compute performance to the speed of the interconnect – and to estimate the necessary resource requirements. It should be noted that this can have a negative and noticeable influence on the parallel efficiency, depending on the performance of the parallel I/O operations and the ratio of the time spent for the setup of the model and the actual time integration. Hence, the threshold of 160 cells owned per task for the breakdown of the parallel efficiency should be considered as a lower limit. In Sect. 2.7, we provide a detailed analysis of the costs of the individual steps for the Jtest-full system.

On the three Linux-cluster type systems, test runs are conducted on single nodes up to 32 (Curie), 20 (ForHLR1), 15 (Jtest-full) and 30 (Jtest-half) nodes. In the following, we refer

[11] https://www2.cisl.ucar.edu/resources/yellowstone

to "good scaling" as a parallel efficiency of $\approx 70\,\%$ or more. Good scaling is achieved up to 6 nodes on Curie (96 tasks, or 427 grid cells per task), 8 nodes on ForHLR1 (160 tasks, or 256 grid cells per task), 8 nodes on Jtest-full (224 tasks, or 183 grid cells per task), and 15 nodes on Jtest-half (210 tasks, or 195 grid cells per task). Comparing Curie and Jtest-half, it is evident that a single Haswell CPU with 14 cores outperforms two Sandy Bridge CPUs with 2×8 cores on the same board, and that (b) the parallel efficiency decreases faster with the number of nodes on Curie. This is probably related to the interconnect: while Juropatest (as well as Yellowstone) uses Infiniband FDR Full Fat Tree technology (Fourteen Data Rate, theoretical effective, aggregated throughput $56\,\mathrm{Gb\,s^{-1}} = 7\,\mathrm{GB\,s^{-1}}$) for the inter-process communication (MPI) and a separate 10 Gb Ethernet connection for I/O operations, Curie uses QDR Full Fat Tree technology (Quad Data Rate, theoretical effective, aggregated throughput $32\,\mathrm{Gb\,s^{-1}} = 4\,\mathrm{GB\,s^{-1}}$) for the inter-process communication *and* for I/O operations. The transition zone for the breakdown of the parallel efficiency between 600 and 150 owned cells per task is indicated as shaded blue area in Fig. 2. A comparison of the absolute runtimes on Jtest-half and Jtest-full shows that runs with 28 cores per node are 5–15 % slower than runs with 14 cores per node, which is presumably due to memory bandwidth bottlenecks. An exception here is the 420-task run (30 nodes on Jtest-half, 15 nodes on Jtest-full), for which the increase in inter-node communication is the limiting factor (see also the discussion in Sect. 2.4).

The minimum job (and block) size on Juqueen is 32 nodes or 512 tasks, which corresponds to only around 80 cells owned by each task. The parallel efficiency drops rapidly with increasing number of nodes, since this problem size is simply too small for application on Juqueen. Figure 3, left panel, displays the communication volume as function of the number of tasks, which follows a power law with index 0.52 up to about 1000 tasks (approx. 40 owned cells per task). A runaway growth can be seen for larger numbers of tasks. At the same time, the graph partitions become increasingly non-contiguous (not displayed). It should be noted that the communication volume and number of non-contiguous partitions are computed by METIS and as such are a function of the partitioning of the cells only, which essentially assumes a single layer of ghost cells. Since MPAS-A exchanges two lay-

ers of ghost cells at maximum, the actual number of ghost cells, edges and vertices can be slightly different. A detailed study of the impact of these graph properties will be given in the following section.

Table H2 lists the cheapest model runs in terms of CPUh spent per 24 h model integration and the fastest runs in terms of realtime per 24 h model integration for the four HPC sites. It is important to remember that while the Jtest-half runs use only 50 % of the available cores on each node, the computational costs for the full node (28 CPUh per node per hour real-time) are charged for the model run, since the node is not available for other users or jobs. Also, a one-to-one relation of CPUh between Linux cluster-type machines and an IBM Blue Gene is not meaningful. By comparing typical calls for proposals for the different HPC systems, a conversion factor of $1:16$ seems to be reasonable, i. e., to consider one entire node with 16 cores on Juqueen as equivalent to one core on the other systems. However, since applications for computing resources usually demand estimates for the required amount of CPUh, we list the actual CPUh here.

2.4 Variable 100–25 km grid

The second test case employs an irregular mesh with a variable resolution ranging from 100 km for most parts of the globe to 25 km for a circular area spanning about 60°, and centred on West Africa (lat $= 12.5°$ N, lon $= 0°$ E). An integration time step of 120 s is used. The mesh as well as the frequency distribution of cell sizes are displayed in Fig. 4, top panels, the scaling is illustrated in Fig. 2, middle panel, and summarised in Table E1.

Test runs are conducted on single nodes up to 192 nodes on Curie (3072 tasks, 53 owned cells per task), 60 nodes on ForHLR1 (1200 tasks, 137 owned cells per task), 50 nodes on Jtest-full (1400 tasks, 117 owned cells per task), and 55 nodes on Jtest-half (770 tasks, 213 owned cells per task). Good scaling is achieved up to 32 nodes on Curie (512 tasks, 320 owned cells per task), 20 nodes on ForHLR1 (400 tasks, 410 owned cells per task), 25 nodes on Jtest-full (700 tasks, 234 owned cells per task), and 35 nodes on Jtest-half (490 tasks, 334 owned cells per task). The parallel efficiency on average drops from about 90

to 40 % within the transition zone (shaded area in Fig. 2, 600 to 150 cells owned per task), similar to the first test case on the regular 120 km mesh.

Notably different to the previous test case are the measured runtimes for Jtest-full and Jtest-half: for small numbers of tasks, the Jtest-full runs show a worse performance due to the aforementioned memory bandwidth limitations. For large numbers of tasks (nodes), the increase in inter-node MPI communication, which impacts the Jtest-half runs more than the Jtest-full runs, becomes the limiting factor and decreases the performance of the Jtest-half runs below that of the Jtest-full runs. With respect to the remaining HPC systems, a clear separation of the parallel efficiency by interconnect technology as for the 120 km test case cannot be seen here, due to the following reasons:

Firstly, the disk I/O demand scales with the number of grid cells and is larger by a factor of 4 for this mesh (see Table H1). As we will see in the following sections, in particular for the extreme scaling experiments in Sect. 4, the disk I/O becomes increasingly important for larger problem sizes and can consume a significant part of the total runtime. The I/O is routed differently at the four HPC sites, the central storage systems have different bandwidths and block sizes, and the parallel I/O libraries might perform differently, depending on the compilers and compilation flags. Additionally, in this test case we adopt an integration time step of 120 s (4.8 s per km grid size), which implies a smaller fraction of the total time spent for the actual time integration relative to the disk I/O.

Secondly, the graph partitioning adds another layer of complexity and variability to the performance diagnostics. Figure 3, middle panel, shows that the above-mentioned power-law relationship between the communication volume and the number of tasks also holds for the variable 100–25 km mesh up to about 40 owned cells per task (4000 tasks). Contrary to the 120 km regular mesh, METIS tends to create more non-contiguous partitions for complex mesh structures (not displayed here). This variability introduced by the graph partitioning is unpredictable and may have significant impact on the model performance. If not instructed otherwise, the graph partitioning tool METIS aims at minimising the edge cut (communication volume), which potentially comes at the cost of having non-contiguous partitions. As discussed earlier in Sect. 2.2, halo cells are added around each patch of the

individual tasks, for which communication with the neighbouring tasks is required. The additional amount of communication caused by halo cells around non-contiguous partitions can be substantial, in particular if the number of cells owned per task is small (i. e., the ratio of halo cells to owned cells is large).

To investigate the effect of non-contiguous partitions on the parallel efficiency, we analyse one partition with 300 tasks for the 100–25 km mesh on ForHLR1 (546 owned cells per task), for which METIS by default produces a non-contiguous partition. We create an additional, contiguous partition using the command line arguments `-contig -minconn` for METIS, which results in an increase of the edge cut from 58 031 to 58 870 (1.4 %). Figure 5 displays the total number of cells (nCellsByTask), the number of owned cells (nCellsSolveByTask), and the number of halo cells per task (nHaloCellsByTask) as ratio between the non-contiguous and the contiguous partition. Since the communication volume increases for the contiguous partition, the total number of cells per task on average is larger, too. The average number of owned cells is identical, since the number of cells of the graph does not change. Notably, task 200 has a 1.3 times larger number of halo cells for the non-contiguous partition, since its partition consists of two separate patches, which implies a larger number of neighbouring tasks and of surrounding halo cells.

To eliminate the influence of the disk I/O on the runtimes for the two partitions in this test, we switch off the output to disk. We find that the measured runtimes for the model integration is practically identical for the two runs (251 s non-contiguous vs. 255 s contiguous). For 300 tasks, the average ratio of halo cells to owned cells is 1 : 2.8, which might be too small to see the effect of the additional halo cells in the non-contiguous partition. We therefore repeat the test for non-contiguous and contiguous partitions for 2520 tasks (65 owned cells per task), with a corresponding ratio of 1.2 : 1 halo cells to owned cells. Even in this case, the measured runtimes for the model integration are nearly identical (45.2 s contiguous vs. 45.6 s non-contiguous). We conclude therefore that the impact of non-contiguous partitions on the runtime is negligible for any reasonable number of tasks for a given mesh.

Although the number of grid cells is 4 times larger for this test case than for the regular 120 km mesh, the problem size is still too small for application on Juqueen. The two smallest

possible parallel runs with 512 and 1024 tasks correspond to 320 and 160 cells owned per task, for which the decrease in parallel efficiency is 20 %. Runs with larger number of tasks all have parallel efficiencies of less than 60 %. Table H2 lists the cheapest and fastest model runs for the four HPC sites.

2.5 Variable 60–12 km grid

The third moderately-sized scaling test consists of a variable resolution mesh with maximum grid spacing 60 km and minimum grid spacing of 12 km. The refinement area is an approximate ellipse, illustrated in Fig. 4, lower panels, and encompasses the whole of North and Central Africa, extends as far as India in the East and covers a large part of the Atlantic Ocean in the West. This particular mesh is useful for studying the teleconnection between the Indian and Atlantic Ocean and the monsoon systems in East and West Africa. The total number of grid cells is 535 554, which corresponds to 1034×517 grid points on a regular latitude–longitude grid and thus is in the range of current re-analyses. A time step of 72 s (6 s per km grid size) is adopted.

Figure 2, right panel, and Table F1 summarise the scaling of this mesh on the four systems. As for the variable 100–25 km mesh, a separation of the parallel performance by interconnect cannot be detected, due to the increasing variability introduced by the disk I/O (see Table H1). While the number of tasks is sufficiently large to obtain a constant power law relationship between the communication volume and the number of tasks up to 16 384 tasks on Juqueen (Fig. 3, right panel), the complexity of the mesh leads to more non-contiguous partitions (not displayed).

Tests runs are conducted on single nodes up to 384 nodes on Curie (6144 tasks, 87 owned cells per task), 120 nodes on ForHLR1 (2400 tasks, 223 owned cells per task), 55 nodes on Jtest-full (1540 tasks, 348 owned cells per task), and 55 nodes on Jtest-half (770 tasks, 696 owned cells per task). Good scaling is achieved up to 48 nodes on Curie (768 tasks, 697 owned cells per task), 45 nodes on ForHLR1 (900 tasks, 595 owned cells per task), and up to the maximum number of 55 nodes on Jtest-full and Jtest-half. In the

transition zone between 600 and 150 owned cells per task, shaded in blue, the parallel efficiency on Curie and ForHLR1 drops off quickly from about 80 % to as low as 30 %.

Juropatest shows the best, but most irregular scaling of the three Linux-cluster systems. With a maximum of 55 available nodes on the system, the parallel performance is better than 70 % for both Jtest-half and Jtest-full. The parallel efficiencies for ForHLR1 and Curie follow a more regular trend, although the number of non-contiguous partitions is highly variable for ForHLR1, but not for Curie. This adds further support to our conclusion that the impact of non-contiguous partitions on the runtime is negligible for any reasonable number of tasks for a given mesh. With 535 554 grid cells, this test case also shows good scaling on Juqueen up to 128 nodes (2048 tasks, 262 owned cells per task). For larger number of tasks, the parallel efficiency drops rapidly and constantly and the number of non-contiguous partitions becomes highly variable.

Table H2 lists the cheapest model runs in terms of CPUh spent per 24 h model integration and the fastest runs in terms of realtime per 24 h model integration for the four HPC sites.

2.6 Comparison of HPC systems

We further address the question how the total runtimes compare on the various HPC systems. Figure 6 displays the total runtimes for the 24 h model runs on the four systems. For all test cases, the three Linux-type systems line up quite well, which means that the absolute runtimes are very close for similar parallel decompositions and that differences in the processor architectures do not translate into model speedup. Due to the large differences in tasks between the Linux-cluster systems and the Blue Gene system, the problem size must be large enough for a fair comparison. Among the three test cases, only the 60–12 km mesh test case satisfies this requirement.

A common feature across the three test cases is a minimum of the total runtimes on the three Linux-type systems around 150 owned cells per task. The reason for the increase in total runtimes for smaller numbers of owned cells per task will be discussed in the following section. Here, we attempt to fit a modified version of Amdahl's law (Amdahl, 1967), which also accounts for the observed increase in runtimes when the parallel performance breaks

down:

$$T(n) = T(1)\left(A + \frac{1 - A - (nB)^X}{n} + (nB)^X\right). \quad (2)$$

Here, n denotes the number of parallel tasks and $T(n)$ the total runtime in seconds. The first term on the r. h. s. of Equation (2) represents the serial part of the code, the second term the part that scales perfectly, and the third term the part that takes longer when more tasks are used. This allows to quickly estimate the required resources for the test cases presented here for Linux-type systems (and Blue Gene/Q systems) with similar specifications.

The coefficients A and B are allowed to vary across the three test cases and two different classes of architectures. The power law index X, on the other hand, is derived as the best fit for the three test cases on the Linux-type systems and the 60–12 km on the Blue Gene system. This results in a value of $X = 2.85$ with a good fit to all four curves, which indicates that the increase in total runtimes below a certain number of owned cells per task occurs for the same reason (see also the discussion in the following Sect. 2.7).

Table 2 summarises the coefficients obtained from fitting Equation (2) to each of the six curves. Note that the fits to the 120 km mesh and the 100–25 km mesh on the Blue Gene system are listed here as well, but should be used with precaution. The minimum of $T(n)$ is obtained by setting $\partial T(n)/\partial n = 0|_{n=n_{\mathrm{min}}}$ and solving for the number of owned cells per task $N_{\mathrm{o,min}} = N_{\mathrm{cells}}/n_{\mathrm{min}}$. This leads to minimum values of $N_{\mathrm{o,min}} = \{143, 146, 182\}$ owned cells per task and $T_{\mathrm{min}} = \{116, 177, 402\}$ s for the three Linux-type systems on the 120 km mesh, the 100–25 km mesh, and the 60–12 km mesh, respectively. For the Blue Gene system, only the fit for the 60–12 km mesh test case can be considered as realistic and leads to $N_{\mathrm{o,min}} = 97$ owned cells per task and a minimum runtime of $T_{\mathrm{min}} = 2155$ s.

From the discussion in this section, we infer that a lower limit of about 150 owned cells per task is a good and quick estimate for the Linux-type systems below which the parallelisation becomes highly inefficient. The shift of the minimum value to larger numbers of tasks (lower numbers of owned cells per task) on the Blue Gene system can be attributed to the performance of the faster 5-D Torus interconnect relative to that of the slower CPU cores. The resulting fitting curves are also displayed in Fig. 6.

2.7 Breakdown of parallel performance

In the following, we investigate the reasons for the breakdown of the parallel performance. In the previous scaling tests, we include the model setup – in principle, bootstrapping and reading of the initial conditions file – as well as the parallel output to disk in the measurements. Here, we split up the total computational costs into three different steps: time integration, model setup, and disk I/O. Over longer runs, the model initialisation costs are amortised and the parallel performance is determined by the numerical solver (time integration) and the parallel I/O (disk output, reading of boundary updates). Accordingly, Fig. 7 displays the total computational costs, the costs for the three steps, and for a combination of time integration and disk I/O for the Jtest-full system and all test cases. Each time, the parallel efficiency of the combined time integration and disk I/O, indicated with orange stars, starts to decrease for less than 600 owned cells per task. The time integration alone shows nearly ideal scaling down to 150 owned cells per task, while the model setup and disk I/O are only weakly dependent on the number of tasks.

According to Figs. 6 and 7, the absolute runtimes for all three test cases start to increase for less than 150 owned cells per task on the three Linux-cluster systems. For the Blue Gene system, this increase in absolute runtimes takes place at smaller numbers of owned cells per task, but the picture is less clear due to the small problem sizes. However, given the faster interconnect on the Blue Gene system and the fact that the model initialisation and disk I/O depend only weakly on the number of tasks, this suggests a breakdown of the parallel efficiency of the solver (i. e., the time integration) itself.

To exploit the limiting factors of the solver for large number of tasks, we use the parallel debugging and profiling tools Scalasca[12] and Score-P[13]. These tools are available as modules on Juqueen, hence we focus on three particular runs on the 60–12 km mesh on this system. We analyse in detail the 2048-task run (261 owned cells per task), the 4096-task

[12] http://www.scalasca.org

[13] http://www.vi-hps.org/projects/score-p

run (130 owned cells per task), and the 8192-task run (65 owned cells per task), for which the latter one takes longer than the former two runs (Fig. 6).

Halo cells are added around each patch of the individual tasks, for which communication with the neighbouring tasks is required. The larger the number of tasks, the smaller the number of owned cells per tasks, and the larger the ratio between halo cells and owned cells. A lower bound for the number of halo cells N_h for a given number of owned cells N_o is given by Equation (1) and corresponds to a ratio of $N_\mathrm{o} : N_\mathrm{h} = 261 : 127 = 2.1$ for the 2048-task run, $131 : 94 = 1.4$ for the 4096-task run, and $65 : 70 = 0.94$ for the 8192-task run.

Table G1 summarises the percentages of time spent during the time integration step for both runs. Leaving aside the model initialisation (initial bootstrapping, reading of initial conditions), the total time spent for communication increases from 35 % for 2048 tasks to 70 % for 8192 tasks. A detailed analysis of the Scalasca report reveals that most of this time is "wasted" in `MPI_WAIT` during the exchange of 2-D and 3-D halo fields. About 16 % of the time is spent for parallel I/O, namely for building and writing output streams and for reading boundary updates (sea-surface temperature, sea-ice fraction). It should be noted here that the selection of fields written to disk is customisable at runtime using simple ASCII files. Table H1 reports on the number of 3D and 4D fields written to disk in our tests.

The Scalasca reports also reveal that computational imbalances in the parallelisation can have serious impacts. For the example of the 2048-task run on the 60–12 km mesh, the average time required to update the boundary information (sea-surface temperature, sea-ice fraction) is about 11.8 s per task. However, one single task takes 19.3 s for the same action and blocks all remaining tasks in their execution. Since the model synchronisation takes place when exchanging halo cell data, the different computing times of the model physics appear in the time spent for communication.

These results highlight the importance of an efficient parallelisation and fast interconnects between the compute nodes and to the central storage, in particular for future applications on massively parallel systems and for the extreme scaling tests in Sect. 4.

3 Reproducing the dynamics of the West African Monsoon

The second important aspect of a numerical weather prediction and climate model is its accuracy in reproducing observed meteorological conditions on global, regional and local level. As described in the introduction, the West African Monsoon has turned out to be a notoriously difficult problem in climate modelling, since it is a complex interplay of various dynamical, microphysical and surface-related processes across scales. The common understanding is that the intensity of the monsoon and the associated precipitation on local scales are regulated by the driving, large-scale atmospheric circulation. This picture is challenged and complicated by a recent study of Klein et al. (2015), who found that processes on local scale such as mesoscale convection and precipitation events, can have a noticeable influence through feedback effects on the entire monsoon system. Examples therefore are enhanced moisture transport and circulation, and strengthening of westward traveling disturbances (African Easterly Waves).

In this study, we attempt a first and brief evaluation of the ability of MPAS-A to reproduce the dynamics of the West African Monsoon. Due to computational limitations for this short comparison, we are restricted to 11-month long model runs. We focus on the onset of the monsoon season in June/July 1982 for two of the meshes presented above, namely the regular 120 km mesh and the variable 60–12 km mesh. Both models are initialised in September 1981 using CFSR data (NCEP Climate Forecast System Reanalysis, Saha et al., 2010) at $0.5^\circ \times 0.5^\circ$ resolution as initial conditions. Daily updates of the sea-surface temperature are taken from the NOAA Optimum Interpolation Sea Surface Temperature Analysis (NOAA OI SST, Reynolds et al., 2002) at $0.25^\circ \times 0.25^\circ$ resolution. The period from September 1981 to beginning of 1982 is considered as spin up time for the model, in particular for the soil conditions. An analysis of the soil properties over the 11-month runs is undertaken to investigate whether such a short spin up time is sufficient. The model output is compared to different sets of observational data to account for the large uncertainty in the gridded observational products in the data-sparse region of West Africa (see, for example, Lorenz and Kunstmann, 2012; Sylla et al., 2013).

For near-surface air temperature and precipitation, we refer to (1) the Climate Research Unit (CRU) high-resolution gridded time-series dataset v. 3.22 (Harris et al., 2014), and (2) the University of Delaware (UDEL) Air Temperature & Precipitation long term monthly means V3.01 (Willmott and Matsuura, 2014) at $0.5^\circ \times 0.5^\circ$. Additional observational data for near-surface air temperature is obtained from the Global Historical Climate Network (GHCN) gridded 2 m temperature dataset V2 (Fan and van den Dool, 2008) at $0.5^\circ \times 0.5^\circ$. For precipitation, we also use the Global Precipitation Climatology Centre (GPCC) Full Data Reanalysis Version 6 Monthly Means (Schneider et al., 2011), also at $0.5^\circ \times 0.5^\circ$.

We also compare the MPAS-A model output to reference data obtained from a set of novel regional climate simulations over West Africa within the WASCAL program[14]. This data is produced using the regional climate model WRF at 12 km resolution with initial and lateral boundary conditions provided by the ERA-Interim re-analysis (Dee et al., 2011, 80 km resolution). The WRF model uses a setup that is optimised for the region of West Africa, following a detailed analysis of the monsoon dynamics for different WRF model configurations by Klein et al. (2015). An extensive documentation and analysis of this reference data set will be given in a forthcoming paper. The WRF model run covers a region from 25° W to 25° E and 5° S to 25° N and is initialised in January 1979. Spectral nudging is applied to the WRF limited area model to keep it on track with the large-scale features of the driving ERA-Interim re-analysis. The sea-surface temperature is updated every 6 h from the ERA-Interim SST data.

The WRF model domain lies entirely within the refinement zone of the variable 60–12 km mesh. Hence, the resolution of the MPAS-A runs is 120 km for the regular mesh (MPAS-120r hereafter) and 12 km for the variable mesh (MPAS-12v hereafter). Figure 8 shows the model topography of the WRF reference model at 12 km resolution (WRF-12r hereafter) and the MPAS-A models. Also indicated is a classification of the land area into five agro-climatical zones, which will be used in the further analysis. Naturally, the topography is nearly identical for WRF-12r and MPAS-12v. Minor differences can be seen along the coast lines and inland water bodies, which are caused by the different grids and by the need to re-grid the MPAS

[14] http://www.wascal.org

model output onto a regular lat-lon grid. This re-gridding is performed with an NCL[15] script `mpas_to_latlon.py` (L. Fowler, personal communication, 2014), which adds another step to the post-processing tasks and is computationally quite demanding: For example, a minimum of 1.5 GB of memory and 10 s runtime is required to re-grid one 3-D variable of MPAS-12v for one time step. For MPAS-120v, the terrain is smoothed out and the coast line (often referred to as landmask in the models) is ill-represented. This has negative effects when verifying the model output over regions such as Guinea, as we will see below.

In Fig. 9, we analyse the spatial distribution of the near-surface temperature for July 1982 (top panel) and its annual cycle between September 1981 and July 1982 (bottom panel). The observations provide data over land only and show noticeable spatial differences in the position and intensity of the Saharan Heat Low (SHL), in particular between CRU/UDEL and GHCN, and minor differences over Ghana and along the coastline. Regarding the model runs, WRF-12r matches the position and temperature of the SHL best and reproduces the observed temperatures. Both MPAS models show a slightly colder surface temperature distribution for July 1982, and the cold bias in MPAS-120r is larger on average. The sea-surface temperature distribution is similar for the two MPAS runs, since they are using the same SST data for their daily updates. However, MPAS-120r shows strong artefacts along the coastline of the Gulf of Guinea, which is due to its inaccurate landmask. The WRF-12r SST, which is updated every 6 h from ERA-Interim data, is colder over the Golf of Guinea, but otherwise shows the same patterns. With respect to the temporal evolution, the annual cycle is reproduced well for both MPAS models, however a significant cold bias is detected over most of the land area from the time of model initialisation in September 1981 up to May 1982. From June to July 1982, this bias seems to nearly vanish with the exception of the coarser MPAS-120v over Guinea due to the limited resolution of the coast line. The observational data sets displayed in the lower panel agree in general, but show small differences for the Saharan region (only CRU and GHCN are displayed for clarity; UDEL is very close to CRU).

[15] http://http://www.ncl.ucar.edu

Figure 10 likewise displays the spatial distribution of the monthly precipitation for July 1982 (top panel) and its annual cycle (bottom panel) for the three observational data sets and the three models. At the onset of the monsoon season, the rain band is centred over 10° N for all three observational data sets. Areas of high precipitation are found over Guinea, Sierra Leone and the Senegal in the West, and over Nigeria, Cameroon and the Central African Republic in the East. While the spatial distribution hardly shows any difference between the observational data sets, the temporal evolution deviates for the Saharan and Guinea regions. For the Saharan region, small absolute values and a very small number of actual observations lead to large relative uncertainties. For Guinea, the spatial interpolation along the coast line leads to differences in the timing of the maximum precipitation (April 1982 for CRU, May 1982 for GPCC; as for temperature, UDEL follows CRU closely).

The three models successfully reproduce the location of the rain band, a fact that should not be taken for granted. In the case of WRF, Klein et al. (2015) demonstrate that the representation of the monsoon dynamics is largely determined by the microphysics and the planetary boundary layer schemes, while the cumulus scheme seems to play more of a role on daily time scales and for the actual amount of precipitation triggered by mesoscale convection. The WRF model configuration chosen here uses the WSM5 microphysics scheme, the ACM2 planetary boundary layer parameterisation and the Grell–Freitas cumulus scheme (see Wang et al., 2014, for a summary of the WRF physics options and further references). It is highly optimised for the region and thus not only matches the location of the rain band, but also reproduces the observed precipitation patterns over the Soudano, the Sahel and the Sahelo regions. The two MPAS model runs, on the other hand, use an "out-of-the-box" setup, which consists of the WSM6 microphysics scheme (similar to WSM5 with graupel as additional hydrometeor), the YSU planetary boundary layer parameterisation and the Kain–Fritsch cumulus scheme. This particular combination produces excessive precipitation during the peak of the monsoon in WRF due to a non-linear response of convective precipitation events to the dynamics, and it seems to exhibit the same behaviour in the two MPAS models.

With respect to the annual cycle of precipitation, the WRF model shows excessive precipitation for all months and an early onset of the monsoon season for the Soudano, Sahel and Sahelo regions. The excess in rainfall over all land area is mainly caused by an overestimation over Guinea, which receives most rainfall over the year (more than 3000 mm year^{-1}, compared to less than 500 mm year^{-1} in the Sahelo region). The "out-of-the-box" MPAS models match the observations better, in particular the timing of the onset of the rainy season and the precipitation over Guinea. The coarser MPAS-120r is closer to the observed precipitation over the inland areas than the higher-resolution MPAS-12v during the onset of the monsoon in June/July 1982. This, however, is not a signal of a higher accuracy of the coarser model: at 120 km resolution, the Kain–Fritsch cumulus scheme is less active than at 12 km resolution, and consequently its wet bias is reduced.

To support this further, Fig. 11a displays the mean sea level pressure map for July 1982. WRF-12r and MPAS-12r both show a distinct area of low pressure, the Saharan Heat Low (SHL), alongside with the South–North pressure gradient that is causing the movement of the ITCZ and the associated monsoon rain band. This SHL is much less pronounced in MPAS-120r. In both MPAS models, the region of low pressure is displaced by about 10° to the East compared to WRF. Lavaysse et al. (2009) derive a climatological mean position of 3° W, 23° N from re-analysis data between 1979 to 2001, which coincides well with WRF.

However, the Saharan Heat Low is not only a region of low surface pressure but also of high surface temperatures. Comparing Fig. 9 and Fig. 11a, it is apparent that the areas of low pressure and high temperature are co-located for WRF-12r. The high-temperature regions of MPAS-120r and MPAS-12v match the WRF SHL position albeit a cold bias of about 2 °C, while the low-pressure region is shifted to the East. This shift is caused by the overall cold bias in the two MPAS models: the SHL, like any other non-frontal thermal low, is formed by rapid solar heating of the land surface and the near-surface layers, which leads to a rising of hot and dense air and to the formation of a stationary low pressure area. Thus, the cold bias in the MPAS models implies a less intense deepening of the pressure system at the expected location of the SHL. As discussed above, the general cold bias over all land areas persists from the onset of the MPAS model runs in September 1981 and only

starts to vanish from June to July 1982 on. This is reflected in the temporal evolution of the mean sea level pressure (Fig. 12, left panel), integrated over all land areas, which is highest for MPAS-120r, lowest for WRF-12r, in-between for MPAS-12v, and merging towards each other from June 1982 on.

A final investigation of the soil properties in Figs. 11b and c and 12 (middle and right panel) reveal the root cause of the cold temperature bias and the associated displacement of the low pressure system. Starting from the MPAS model initialisation in September 1981, the soil temperature is about 2 °C lower then for the WRF model run, which is initialised in January 1979 and thus has more than 2 years to spin up the land surface model (NOAH LSM for both WRF and MPAS). Over the course of the simulated 11 months, the MPAS soil temperatures start to converge towards the WRF soil temperature: in July 1982, the soil temperatures distributions in Fig. 11b are already quite similar. However, the relative soil moisture maintains a constant bias for most of the time and still shows spatial differences in July 1982. This is due to the fact that between the MPAS model initialisation in September 1981 and the onset of the monsoon season in 1982, virtually no precipitation occurs over large fractions of the land area and that the inherently low soil moisture simply has not possibility to adapt.

We therefore conclude from this section that the global-to-variable resolution mesh concept and its implementation in MPAS-A have the potential to reproduce the dynamics of the West African Monsoon and its associated precipitation. In this single, 11-month experiment, the "out-of-the-box" setup of MPAS can compete with the optimised WRF setup with respect to the timing of the onset of the rainy season. The high-resolution MPAS-12v model shows a better performance than the coarse-resolution MPAS-120r model in general, but in particular for the coastal regions. Deficiencies in the location of the Saharan Heat Low and a cold bias in the surface temperature are apparent in both MPAS model runs, which are caused to a large extend by an insufficient spin up time of the models. Detailed investigations of the WRF reference run, not discussed here, suggest that a spin up time of at least one year is required in order to adjust the soil moisture. The MPAS runs also tend to over-estimate the

monsoon precipitation, which we attribute to the combination of physical parameterisations of the "out-of-the-box" setup.

4 Extreme scaling experiment at very high resolution

In this section, we evaluate the scalability and limitations of the MPAS-A model for a very large mesh on a massively parallel system. This experiment is motivated by the following two aspects:

1. Future HPC environments will most likely be massively parallel systems, with the number of cores per node and the number of nodes increasing much faster than the speed of the individual cores. Models such as MPAS-A have to be able to scale on these systems in order to be used successfully in the future.

2. The typical model resolution of global and regional NWP and climate models has increased continuously over the past few decades. Currently, the European Centre for Medium-Range Weather Forecasts (ECMWF) is leading the field with an operational global NWP model at 14 km resolution[16], and limited area models have been taken down to sub-km resolution. With this experiment, we want to demonstrate that convection-resolving (below about 4 km grid size, see Weisman et al., 1997; Prein et al., 2013, for example), global atmospheric simulations are possible on current HPC environments.

To create a large-enough problem for these tests at a convection-resolving resolution, we use a regular, global 3 km mesh with more than 65 million horizontal grid cells and 41 vertical levels. Up to now, only a few real-data simulations on this mesh have been conducted on NWSC's Yellowstone supercomputer using a maximum of 16 384 MPI tasks (approx. 4000 owned cells per task); a set of scaling benchmarks based on an idealised case have

[16] http://www.ecmwf.int/en/forecasts/datasets/dataset-i-i-atmospheric-fields-high-resolution-forecast

also been run on up to 131 072 MPI tasks on Edison, a Cray XC30 at the National Energy Research Scientific Computing Center[17].

Among the four HPC sites presented earlier, only the FZJ Juqueen offers a large enough number of cores to conduct this extreme scaling experiment. With a maximum of 458 752 cores, these tests require scaling out to the full system, which is not possible during normal operations. However, following the 3rd Juqueen Tuning and Porting Workshop in February 2015, a few selected applications were invited to conduct extreme scaling tests on the full machine during a period of 24 h. The results presented in the following were obtained during this event, which is summarised in a technical report by Brömmel et al. (2015).

4.1 Model configuration and experiment preparation

Several modifications of the MPAS-A code are required to conduct this experiment. Firstly, it is no longer possible to use the standard NetCDF large-file format (CDF-2), since the maximum number of elements for some of the 4-D variables exceeds the internal limit of 2 billion values for any particular record. One possible solution is to use the CDF-5 extension of CDF-2, which is supported by the Parallel-NetCDF library. However, only very few applications understand this format, and initial testing on Juqueen revealed problems with reading correct data in massively parallel read operations. Another solution, which is adopted here, is to use the newer NetCDF-4 format, which supports parallel I/O through PHDF5. This requires upgrading the parallel I/O library PIO from v1.7.1 to 1.9.15, and modifying the I/O framework of the MPAS-A model code. Motivated by this study, MPAS-A v4.0, released at the time of writing, supports NetCDF-4 I/O without any need to modify the software framework.

Further, the generation of the model input data becomes a large computational problem which cannot be fit on a single machine due to time constraints and memory limitations. In MPAS release v3.1, the pre-processing of the data is partly a serial process running on one CPU core only. Hence, a parallelisation of the pre-processing is required in addition to the

[17] https://www.nersc.gov/users/computational-systems/edison

above changes to the I/O routines. These changes are applied to the basic MPAS-A v3.1 code, and this modified version is used in the following scaling experiments.

Another difference from the default model configuration presented in Sect. 2.2 is that each test is run for 1 h model time only. This implies that the update of the surface data (sea-surface temperature, sea-ice fraction), which occurs every 24 h in the above moderate scaling tests, is dropped. Also, with a global resolution of 3 km, it is generally agreed that no convection scheme is required, since the microphysics scheme is able to generate the convective precipitation systems on the grid scale. These modifications are reflected in the model namelist in Appendix D.

The pre-processing consists of two steps. First, a static data set is produced (`static.nc`), which maps invariants such as terrain height, landmask and land use classification onto the 3 km mesh. These invariant fields are required as input for the subsequent generation of the initial conditions (`init.nc`), in part because the terrain field is necessary for the generation of the height-based vertical grid. The parallel pre-processing steps require special mesh partitions, different from the partitions generated by METIS and used for the model runs. Here, both steps are conducted with 576 cores, spread across 80 nodes on the ForHLR1 to provide sufficient memory for each task. Each step takes about 1 h 15 m realtime and requires around 4.6 TB of the available 5.1 TB of memory. The resulting initial conditions file has a size of 1.2 TB and is transferred to Juqueen over the 10 Gb s^{-1} internet connection between the two HPC sites. Since both pre-processing steps are only required once, no further investigation or optimisation of the runtimes is attempted.

4.2 First attempts and optimisations

Initial test runs at 3 km resolution revealed previously unknown problems on the system. As described in Sect. 2.2, a bootstrapping step is required during the model initialisation to set up the grid and instruct individual tasks with whom to share information about neighbouring grid cells. In the MPAS code, this is implemented using hash tables. In order to complete the bootstrapping in a reasonable time, the hash table size (parameter `TABLESIZE`) is increased from the default value 27 183 to 6 000 000. After this adjustment, the bootstrapping

step takes between 18 and 29 min on Juqueen, depending on the number of MPI tasks (see Table 3). A second bottleneck is the reading of the initial conditions file. Performance improvements for this step are achieved by setting two runtime environment variables that were presented during the tuning and porting workshop (`BGLOCKLESSMPIO_F_TYPE`, `ROMIO_HINTS`), and by optimising the number of I/O tasks. While in the previous scaling tests all tasks participate in the I/O, we find improvements when using only 128 I/O tasks per 1024 nodes (1 rack, 16 384 tasks), with an average read performance of 1.2 GB s^{-1} and an average write performance of 0.6 GB s^{-1}, respectively. Due to the large number of parallel tasks in this extreme scaling experiment, we use the unit "nodes" instead of "tasks" throughout this section, where 1024 nodes correspond to 1 rack or 16 384 tasks.

With these optimisations, the parallel reading of the initial conditions file improves slightly with the number of tasks, since they are located in different racks. Conversely, the bootstrapping step takes longer for larger numbers of tasks. Hence, the overall model initialisation takes approximately 45 min for the 3 km mesh and varies only slightly with the number of nodes. One notable exception here is the run on 8192 nodes, for which the initial I/O is only 50 % of that of the other runs. The exact reasons for this behaviour needs to be investigated, but we think that this combination of file size and I/O tasks is a sweet spot on Juqueen.

4.3 Execution of extreme scaling tests

The substantial memory requirements for the 3 km mesh do not allow to run the model on 1024 or 2048 nodes. The baseline for our scaling experiment is therefore the run on 4096 nodes (4 racks, 65 536 MPI tasks, 512 I/O tasks, 65 TB memory). Contrary to the model initialisation, the time integration step scales very well up to the entire machine, with a parallel efficiency of 87 % for 24 576 nodes (393 216 tasks, 167 cells per task) compared to the baseline (see Table 3). The test run on the full system with 28 672 nodes (458 752 tasks, 143 cells per task) shows a lower performance than the run on 24 racks and a parallel efficiency of nearly 70 %, which confirms the above-mentioned limit of approximately 150 owned cells per task, below which the parallelisation becomes inefficient.

All scaling tests are conducted with an 18 s model integration time step for a 1 h model time. However, we find that in order to keep the model stable when starting off the 3 km mesh from initial conditions derived from a 48 km re-analysis dataset (CFSR), a more conservative time step is required. MPAS currently lacks a dynamical initialisation system (e.g., digital filters, adaptive time-stepping), which could avoid this issue. Model instabilities leading to NaNs can affect the performance in different ways: (1) the performance might increase in case if-NaN-tests may cause the code to return early from computationally intensive physics routines, or (2) the performance might decrease due to the continuous generation of floating-point exceptions. We note that on Blue Gene systems, MPAS does not by default include the "-qfloattrap" compiler flag to trap floating-point exceptions, and, consequently, the model will continue to run even when the simulation generates NaNs. We therefore repeat the runs for 4096, 8192, and 16 384 nodes with a 12 s time step to obtain a stable model run. The measured realtimes for the three 12 s runs are very close to 1.5 times the realtimes for the corresponding 18 s runs, which gives us confidence that we can scale the results for 24 576 and 28 672 nodes with an 18 s time step to a 12 s time step, despite the fact that the runs with an 18 s time step produced NaNs.

Table 3 and Fig. 13 summarise the required times of the individual steps of the 3 km runs with an 18 s integration time step. Due to walltime constraints, we only conduct runs without writing output to disk. The last column in Table 3 estimates how many hours the 3 km model can be advanced within 24 h walltime, and is calculated as follows: a 12 s model integration time step is assumed, and the realtime required is scaled from the 18 s runs by a factor of 1.5 for 24 576 and 28 672 nodes, for which no 12 s runs are conducted. For a typical production run, diagnostic output files of 13 GB size are written every 3 h model time, while comprehensive output files of approximately 250 GB size are written every 24 h model time. A restart file of 2.1 TB size is written at the end of the model run. Based on a parallel write performance of 0.6 GB s^{-1}, we make a conservative estimate that roughly two hours of the 24 h walltime will be used up by writing these files to disk. Tables H1 and H2 list the file sizes and the cheapest and fastest model runs for the 3 km mesh on Juqueen.

We conclude from this extreme scaling test that the dynamical solver of MPAS scales on massively parallel systems out to hundreds of thousands of cores. Our results confirm that the model behaves similar for the 3 km mesh than for the significantly smaller problem sizes and that the parallel efficiency is limited by the same factors, namely the increasing number of halo cells and amount of communication for large number of tasks. This occurs around 150 owned cells per task, which corresponds to roughly 27 300 nodes for the 3 km mesh. However, we find that the model initialisation and the disk I/O become increasingly important and at the same time difficult to improve for extremely large test cases. Compared to the model integration, the time required for the model initialisation and for reading and writing data is largely independent of the number of tasks. For a maximum walltime of 24 h on Juqueen, these steps consume up to 3 h or 10–15 % of the total job time – in other words, hundreds of thousands of CPUh. The cheapest run on Juqueen utilises 4096 nodes (4 racks, 65 536 tasks), consumes 1.3 Mio CPUh for a 24 h model integration and gives a speedup of 1.2 $\times$ realtime. For 24 576 nodes (24 racks, 393 216 tasks), a speedup of 6.3 $\times$ realtime is achieved at the slightly higher expense of 1.5 Mio CPUh.

5 Conclusions

In this study, we analyse the atmospheric model MPAS-A in detail for its numerical performance and for its physical accuracy. We conduct scaling tests for three medium-size problems using regular and variable meshes of different complexity on four different HPC facilities. We confirm an overall good scaling ($\gtrapprox$ 70 % parallel efficiency) of MPAS across all systems and find that a value 150 cells owned by each task provides a good and quick estimate for the breakdown of the parallel performance when setup and I/O costs are included in the scaling. This limit depends weakly on the interconnect of the system (absolute but also relative to the core speed), with faster interconnects corresponding to lower values, but also on the I/O performance. Accordingly, it is slightly lower for the Blue Gene system (between 100 owned cells per task for the 60–12 km mesh and 150 owned cells per task for the 3 km mesh). Taking into account that the setup costs are amortised over

longer runs, MPAS-A maintains a parallel efficiency of 80 % or better for more than 150 owned cells (100 owned cells on Juqueen) per task. Based on these findings, we provide numbers on the typical file sizes and optimal model configurations for conducting research and operational runs on the different HPC systems.

An in-depth analysis of the properties of different graph partitions for one of the meshes shows that the impact of non-contiguous graph partitions in form of changes in the number of halo cells is negligible for any reasonable number of tasks for a given problem size. We further employ the parallel profiling tools Scalasca and Score-P to identify the bottlenecks in the MPAS-A code when the parallel performance breaks down. Our findings confirm that most of the time in such cases is spent waiting during the communication with neighbouring tasks, but also demonstrate the negative impact that computational imbalances can have on the model performance.

We also study the accuracy of MPAS-A for one common and challenging problem in climate research, namely the capability to reproduce the dynamics of the West African Monsoon and its associated precipitation. We conduct 11-month simulations for two meshes, a regular 120 km mesh and a variable 60–12 km mesh, and compare the model output to a number of observation data sets, selected re-analyses and a reference model run. The reference model run is chosen from a novel set of regional climate simulations over West Africa within the framework of the WASCAL programme and employs the regional climate model WRF, from which MPAS inherits several aspects of the dynamical solver and all of its physical parameterisation schemes.

We find that MPAS-A is able to model the monsoon dynamics and the northwards movement of the monsoon rain band in this pilot study. Despite using an "out-of-the-box" configuration of the model, both runs reproduce the timing of the onset of the monsoon season better than the optimised WRF reference run. We find that the precipitation in the early monsoon season is overestimated, which we attribute to the choice of physics parameterisations. The MPAS model runs also show a cold bias in the near-surface temperature and consequently fail to place the Saharan Heat Low at the correct location, which we believe stems from a too short spin up time of the model. We would like to stress that our pilot

study aims primarily at investigating the feasibility to conduct climate modelling research with MPAS for the West African region. In order to be able to draw general conclusions on the ability of MPAS to reproduce observed climatological properties, an ensemble of model runs over longer time periods with sufficiently long spin up times is required.

In the last part of this study, we conduct extreme scaling tests on a global 3 km mesh with more than 65 million grid cells on up to 458 752 cores on Juqueen, the IBM Blue Gene/Q at the Forschungszentrum Jülich. We describe the issues that arise when attempting such an experiment for the first time – up to now, MPAS-A has been run for real-data cases, which include a full physics suite, on a maximum number of 16 384 tasks on Yellowstone – and provide solutions that allow to conduct the scaling test in the first place and improve the model performance. We find that the model scales very well up to the entire machine with a parallel performance of nearly 70 % for 458 752 tasks. We confirm that the limitations and rules to estimate the scaling, derived for moderate problem sizes, are also valid for the extremely large test case. This gives confidence for planning model experiments and estimating required runtimes, storage and computational resources.

Furthermore, we identify additional aspects in the model that become increasingly relevant for larger problem sizes: the model initialisation and the disk I/O. We describe strategies to improve the performance of the model, which are partly machine-dependent. We further give estimates on required runtimes and resources for conducting scientific experiments with the 3 km mesh on Juqueen.

Our next steps will be to conduct a number of longer simulation experiments on regular and variable-resolution meshes with a moderate number of grid cells. Specifically, we plan to pursue the study on the dynamics of the West African Monsoon using a variable-resolution grid such as the 60–12 km mesh and a regular grid with a similarly fine resolution. This will allow us to compare the accuracy of the model after a full spin up of the soil conditions and to assess the impact of the variable mesh on the model results. It will also allow us to study physical processes such as the teleconnection between the oceans and the African monsoon systems, and investigate the impact of climate and land use changes in a consistent approach.

In conclusion, the MPAS-A model is a novel atmospheric model that scales well on a range of architectures for small up to extremely large numbers of tasks. Based on an unstructured Voronoi mesh, it allows to conduct global simulations with local refinement regions and smooth transition in-between them. This makes it possible to study local-scale processes in regions of interest with a full coupling to the large-scale motions and a physical consistency within the model. This is shown here in a pilot study of the West African Monsoon. We also demonstrate that it is possible to conduct global, convection-resolving atmospheric simulations with MPAS on current and future massively parallel systems. However, it is evident that the application of models such as MPAS for extremely large problem sizes and numbers of tasks require substantial efforts to optimise the model to the problem *and* to the machine it is run on. In order to do so, interdisciplinary approaches and more intensive training of scientist on hardware, software and programming techniques are paramount.

Appendix A: Code availability

Supplementary material and accompanying information are available at http://dx.doi.org/10.1594/PANGAEA.849428. These contain a modified version of the MPAS release v3.3, which is identical to v3.1 used in this publication, but includes minor fixes for model stability and extensions for detailed measurements of the performance of the individual components of the model. This version can be used to reconstruct the three medium-sized test cases in Sect. 2 and the 11-month study on the West African Monsoon in Sect. 3. Compressed archives of the required input data for the three test cases are included in the supplementary material. The initial conditions provided for the 60–12 km variable grid test case (Sect. 2.5) are identical to those used in Sect. 3.

Appendix B: MPAS compiler flags

We use the following compiler optimisation flags for the Intel compilers icc and ifort on Curie, ForHLR1 and Juropatest:

```
CFLAGS = "-O3"
FFLAGS = "-real-size 64 -O3 -convert big_endian -FR"
```

For the IBM XL compilers mpixlc_r and mpixlf95_r on Juqueen, the following flags are used:

```
CFLAGS = "-O3 -qstrict -qarch=qp -qtune=qp"
FFLAGS = "-O3 -qstrict -qarch=qp -qtune=qp -qrealsize=8"
```

Appendix C: Configuration for moderate problem sizes

The following model configuration in terms of the usual namelist (`namelist.atmosphere`) is used for the experiments in Sect. 2.3–2.5 (regular 120 km mesh, variable 100–25 km mesh, variable 60–12 km mesh). Differences in the setup (e.g., model integration time step) are indicated in namelist-style comments. Details about the structure of the namelist file and the available options can be found in the MPAS-Atmosphere Model User's Guide (Duda et al., 2014).

```
&nhyd_model
    config_dt           = 150.0 # 120km
    config_dt           = 120.0 # 100-25km
    config_dt           =  72.0 # 60-12km
    config_start_time   = '1981-09-02_00:00:00'
    config_stop_time    = '1981-09-03_00:00:00'
    config_run_duration = '24:00:00'
    config_len_disp     = 120000.0 # 120km
```

```
    config_len_disp     =  25000.0 # 100-25km
    config_len_disp     =  12000.0 # 60-12km
/

&damping
    config_zd = 22000.0
    config_xnutr = 0.2
/

&io
    config_pio_num_iotasks = 0
    config_pio_stride = 1
/

&decomposition
   config_block_decomp_file_prefix = 'part.'
/

&restart
    config_do_restart =.true.
/

&physics
    config_frac_seaice         =.true.
    config_sst_update          =.true.
    config_sstdiurn_update     =.true.
    config_deepsoiltemp_update =.true.
    config_bucket_update       = '24:00:00'
    config_bucket_rainc        = 100.0
```

```
    config_bucket_rainnc       = 100.0
    config_bucket_radt         = 1.0e9
    config_microp_scheme       = 'wsm6'
    config_convection_scheme   = 'kain_fritsch'
    config_lsm_scheme          = 'noah'
    config_pbl_scheme          = 'ysu'
    config_gwdo_scheme         = 'off'
    config_radt_cld_scheme     = 'cld_incidence'
    config_radt_lw_scheme      = 'rrtmg_lw'
    config_radt_sw_scheme      = 'rrtmg_sw'
    config_sfclayer_scheme     = 'monin_obukhov'
/
```

Appendix D: Configuration for extreme scaling tests

For the 3 km extreme scaling tests on Juqueen, the following model configuration is used. For details, the reader is referred to the MPAS-Atmosphere Model User's Guide (Duda et al., 2014).

```
&nhyd_model
    config_dt           = 12.0
    config_start_time   = '1981-09-01_00:00:00'
    config_stop_time    = '1981-09-01_01:00:00'
    config_run_duration = '01:00:00'
    config_len_disp     = 3000.0
/

&damping
    config_zd = 22000.0
```

```
    config_xnutr = 0.2
/

&io
    config_pio_num_iotasks = 512 # for 4 racks, 128 per rack
    config_pio_stride = 128
/

&decomposition
   config_block_decomp_file_prefix = 'part.'
/

&restart
    config_do_restart =.false.
/

&physics
    config_frac_seaice          =.false.
    config_sst_update           =.false.
    config_sstdiurn_update      =.false.
    config_deepsoiltemp_update  =.false.
    config_bucket_update        = '24:00:00'
    config_bucket_rainc         = 100.0
    config_bucket_rainnc        = 100.0
    config_bucket_radt          = 1.0e9
    config_microp_scheme        = 'wsm6'
    config_convection_scheme    = 'off'
    config_lsm_scheme           = 'noah'
    config_pbl_scheme           = 'ysu'
```

```
    config_gwdo_scheme          = 'off'
    config_radt_cld_scheme      = 'cld_incidence'
    config_radt_lw_scheme       = 'rrtmg_lw'
    config_radt_sw_scheme       = 'rrtmg_sw'
    config_sfclayer_scheme      = 'monin_obukhov'
/
```

Author contributions. All experiments in this study were carried out by D. Heinzeller with active support of M. G. Duda. The moderately-sized scaling experiments presented in Sect. 2 were designed and analysed by D. Heinzeller, while the extreme scaling experiment was designed and realised in a collaborate effort of D. Heinzeller and M. G. Duda (Sect. 4). The analysis of the model performance with respect to the representation of the West African Summer Monsoon and its associated precipitation was conducted by D. Heinzeller and H. Kunstmann (Sect. 3).

Acknowledgements. We acknowledge that the results of this research have been achieved using the PRACE Research Infrastructure resources TGCC Curie and FZJ Juqueen based in France and Germany. Access to SCC ForHLR1 was granted through a test account, and to FZJ Juropatest through an ongoing research project on FZJ Juropa. The authors acknowledge the European Centre for Medium-Range Weather Forecasts (ECMWF) for the dissemination of ERA-Interim, the National Centers for Environment Prediction (NCEP) for the dissemination of CFSR, and the Global Modeling and Assimilation Office (GMAO) and the GES DISC for the dissemination of MERRA. Further, we acknowledge the NOAA/OAR/ESRL PSD, Boulder, Colorado, USA, for providing the UDEL air temperature and precipitation, GHCN air temperature, GPCC precipitation, and NOAA OI SST sea-surface temperature data sets. We also thank PRACE for funding the 3rd Juqueen Porting and Tuning Workshop 2015 as part of the PRACE Advanced Training Centres courses. The authors are particularly grateful for the extensive and valuable support of the Jülich Supercomputing Centre (JSC) support team during the extreme scaling experiment: Dirk Brömmel, Wolfgang Frings, Markus Geimer, Klaus Görgen, Sabine Grießbach, Lars Hoffmann, Catrin Meyer, Michael Rambadt, Michael Stephan, and Brian Wylie. We also thank the two anonymous referees for useful comments

and corrections that helped to improve this manuscript.

The article processing charges for this open-access publication were covered by a Research Centre of the Helmholtz Association.

References

Amdahl, G. M.: Validity of the Single Processor Approach to Achieving Large-Scale Computing Capabilities, AFIPS Conference Proceedings, 30, 483–485, 1967.

Borkar, S. and Chien, A. A.: The future of microprocessors, Commun. ACM, 54, 67–77, 2011.

Brömmel, D., Frings, W., and Wylie, B.: Technical Report Juqueen Extreme Scaling Workshop 2015, Tech. rep., Jülich, Germany, available at: http://hdl.handle.net/2128/8435 (last access: 25 August 2015), 2015.

Caron, J.-F.: Mismatching perturbations at the lateral boundaries in limited-area ensemble forecasting: a case study, Mon. Weather Rev., 141, 356–374, doi:10.1175/MWR-D-12-00051.1, 2013.

Caya, D. and Biner, S.: Internal variability of RCM simulations over an annual cycle, Clim. Dynam., 22, 33–46, doi:10.1007/s00382-003-0360-2, 2004.

Davies, H. C.: Limitations of some common lateral boundary schemes used in regional NWP models, Mon. Weather Rev., 111, 1002–1012, doi:10.1175/1520-0493(1983)111<1002:LOSCLB>2.0.CO;2, 1983.

Dee, D. P., Uppala, S. M., Simmons, A. J., Berrisford, P., Poli, P., Kobayashi, S., Andrae, U., Balmaseda, M. A., Balsamo, G., Bauer, P., Bechtold, P., Beljaars, A. C. M., van de Berg, L., Bidlot, J., Bormann, N., Delsol, C., Dragani, R., Fuentes, M., Geer, A. J., Haimberger, L., Healy, S. B., Hersbach, H., Hólm, E. V., Isaksen, L., Kållberg, P., Köhler, M., Matricardi, M., Mcnally, A. P., Monge-Sanz, B. M., Morcrette, J. J., Park, B. K., Peubey, C. de Rosnay, P., Tavolato, C., Thépaut, J. N., and Vitart, F.: The ERA-Interim reanalysis: configuration and performance of the data assimilation system, Q. J. Roy. Meteor. Soc., 137, 553–597, doi:10.1002/qj.828, 2011.

Dennis, J. M., Edwards, J., Jacob, R., Loy, R., Mirin, A., and Vertenstein, M.: Parallel I/O library (PIO), available at: http://www.cesm.ucar.edu/models/pio (last access: 25 August 2015), 2013.

Du, Q., Gunzburger, M. D., and Ju, L.: Constrained centroidal voronoi tessellations for surfaces, SIAM J. Sci. Comput., 24, 1488–1506, doi:10.1137/S1064827501391576, 2003.

Duda, M., Fowler, L., Skamarock, W., Roesch, C., Jacobsen, D., and Ringler, T.: MPAS-Atmosphere Model User's Guide Version 3.0, available at: http://www2.mmm.ucar.edu/projects/mpas/mpas_atmosphere_users_guide_3.0.pdf (last access: 25 August 2015), 2014.

Fan, Y. and van den Dool, H.: A global monthly land surface air temperature analysis for 1948–present, J. Geophys. Res.-Atmos., 113, D01103, doi:10.1029/2007JD008470, 2008.

Grist, J. P. and Nicholson, S. E.: A study of the dynamic factors influencing the rainfall variability in the West African Sahel, J. Climate, 14, 1337–1359, doi:10.1175/1520-0442(2001)014<1337:ASOTDF>2.0.CO;2, 2001.

Harris, I., Jones, P. D., Osborn, T. J., and Lister, D. H.: Updated high-resolution grids of monthly climatic observations – the CRU TS3.10 Dataset, Int. J. Climatol., 34, 623–642, doi:10.1002/joc.3711, 2014.

Harris, L. M. and Durran, D. R.: An idealized comparison of one-way and two-way grid nesting, Mon. Weather Rev., 138, 2174–2187, doi:10.1175/2010MWR3080.1, 2010.

Hourdin, F., Musat, I., Guichard, F., Ruti, P. M., Favot, F., Filiberti, M. A., Pham, M. A. I., Grandpeix, J. Y., Polcher, J. A. N., Marquet, P., Boone, A., Lafore, J. P., Redelsperger, J. L., Dell'Aquila, A., Doval, T. L., Traore, A. K., and Gallée, H.: Amma-Model intercomparison project, B. Am. Meteorol. Soc., 91, 95–104, doi:10.1175/2009BAMS2791.1, 2010.

Karypis, G. and Kumar, V.: Multilevel k way partitioning scheme for irregular graphs, J. Parallel Distr. Com., 48, 96–129, 1998.

Klein, C., Heinzeller, D., Bliefernicht, J., and Kunstmann, H.: Variability of West African monsoon patterns generated by a WRF multi-physics ensemble, Clim. Dynam., doi:10.1007/s00382-015-2505-5, online first, 2015.

Klemp, J. B.: A terrain-following coordinate with smoothed coordinate surfaces, Mon. Weather Rev., 139, 2163–2169, doi:10.1175/MWR-D-10-05046.1, 2011.

Laprise, R.: Resolved scales and nonlinear interactions in limited-area models, J. Atmos. Sci., 60, 768–779, doi:10.1175/1520-0469(2003)060<0768:RSANII>2.0.CO;2, 2003.

Lavaysse, C., Flamant, C., Janicot, S., Parker, D. J., Lafore, J. P., Sultan, B., and Pelon, J.: Seasonal evolution of the West African heat low: a climatological perspective, Clim. Dynam., 33, 313–330, doi:10.1007/s00382-009-0553-4, 2009.

Leduc, M. and Laprise, R.: Regional climate model sensitivity to domain size, Clim. Dynam., 32, 833–854, doi:10.1007/s00382-008-0400-z, 2008.

Lorenz, C. and Kunstmann, H.: The hydrological cycle in three state-of-the-art reanalyses: intercomparison and performance analysis, J. Hydrometeorol., 13, 1397–1420, doi:10.1175/JHM-D-11-088.1, 2012.

MacDonald, A. E., Middlecoff, J., Henderson, T., and Lee, J.-L.: A general method for modeling on irregular grids, Int. J. High Perform. C., 25, 392–403, doi:10.1177/1094342010385019, 2011.

Miguez-Macho, G., Stenchikov, G. L., and Robock, A.: Spectral nudging to eliminate the effects of domain position and geometry in regional climate model simulations, J. Geophys. Res.-Atmos., 109, D13104, doi:10.1029/2003JD004495, 2004.

Nicholson, S. E. and Webster, P. J.: A physical basis for the interannual variability of rainfall in the Sahel, Q. J. Roy. Meteor. Soc., 133, 2065–2084, doi:10.1002/qj.104, 2007.

Nikulin, G., Jones, C., Giorgi, F., Asrar, G., Büchner, M., Cerezo-Mota, R., Christensen, O. B. s., Déqué, M., Fernandez, J., Hänsler, A., van Meijgaard, E., Samuelsson, P., Sylla, M. B., and Sushama, L.: Precipitation climatology in an ensemble of CORDEX-Africa regional climate simulations, J. Climate, 25, 6057–6078, doi:10.1175/JCLI-D-11-00375.1, 2012.

Park, S.-H., Klemp, J. B., and Skamarock, W. C.: A comparison of mesh refinement in the global MPAS-A and WRF models using an idealized normal-mode baroclinic wave simulation, Mon. Weather Rev., 142, 3614–3634, 2014.

Prein, A. F., Gobiet, A., Suklitsch, M., Truhetz, H., Awan, N. K., Keuler, K., and Georgievski, G.: Added value of convection permitting seasonal simulations, Clim. Dynam., 41, 2655–2677, doi:10.1007/s00382-013-1744-6, 2013.

Reynolds, R. W., Rayner, N. A., Smith, T. M., Stokes, D. C., and Wang, W.: An improved in situ and satellite SST analysis for climate, J. Climate, 15, 1609–1625, doi:10.1175/1520-0442(2002)015<1609:AIISAS>2.0.CO;2, 2002.

Rienecker, M. M., Suarez, M. J., Gelaro, R., Todling, R., Bacmeister, J., Liu, E., Bosilovich, M. G., Schubert, S. D., Takacs, L., Kim, G. K., Bloom, S., Chen, J., Collins, D., Conaty, A., Da Silva, A., Gu, W., Joiner, J., Koster, R. D., Lucchesi, R., Molod, A., Owens, T., Pawson, S., Pegion, P., Redder, C. R., Reichle, R., Robertson, F. R., Ruddick, A. G., Sienkiewicz, M., and Woollen, J.: MERRA: NASA's modern-era retrospective analysis for research and applications, J. Climate, 24, 3624–3648, doi:10.1175/JCLI-D-11-00015.1, 2011.

Ringler, T. D., Thuburn, J., and Klemp, J.: A unified approach to energy conservation and potential vorticity dynamics for arbitrarily-structured C-grids, J. Comput. Phys., 229, 3065–3090, 2010.

Ringler, T. D., Jacobsen, D., Gunzburger, M., Ju, L., Duda, M., and Skamarock, W.: Exploring a multiresolution modeling approach within the shallow-water equations, Mon. Weather Rev., 139, 3348–3368, doi:10.1175/MWR-D-10-05049.1, 2011.

Ringler, T. D., Petersen, M., Higdon, R. L., Jacobsen, D., Jones, P. W., and Maltrud, M.: A multi-resolution approach to global ocean modeling, Ocean Model., 69, 211–232, doi:10.1016/j.ocemod.2013.04.010, 2013.

Rodwell, M. J. and Hoskins, B. J.: Monsoons and the dynamics of deserts, Q. J. Roy. Meteor. Soc., 122, 1385–1404, doi:10.1002/qj.49712253408, 1996.

Saha, S., Moorthi, S., Pan, H. L., Wu, X., Wang, J., Nadiga, S., Tripp, P., Kistler, R., Woollen, J., Behringer, D., Liu, H., Stokes, D., Grumbine, R., Gayno, G., Wang, J., Hou, Y. T., Chuang, H. Y., Juang, H. M. H., Sela, J., Iredell, M., Treadon, R., Kleist, D., Van Delst, P., Keyser, D., Derber, J., Ek, M., Meng, J., Wei, H., Yang, R., Lord, St., Van Den Dool, H., Kumar, A., Wang, W., Long, C., Chelliah, M., Xue, Y., Huang, B., Schemm, J. K., Ebisuzaki, W., Lin, R., Xie, P., Chen, M., Zhou, S., Higgins, W., Zou, C. Z., Liu, Q., Chen, Y., Han, Y., Cucurull, L., Reynolds, R. W., Rutledge, G., Goldberg, M.: The NCEP climate forecast system reanalysis, B. Am. Meteorol. Soc., 91, 1015–1057, doi:10.1175/2010BAMS3001.1, 2010.

Schneider, U., Becker, A., Finger, P., Meyer-Christoffer, A., Rudolf, B., and Ziese, M.: GPCC full data reanalysis version 6.0 at 0.5°: monthly land-surface precipitation from rain-gauges built on GTS-based and historic data, MIMEO, doi:10.5676/DWD_GPCC/FD_M_V6_050, 2011.

Skamarock, W., Klemp, J., Dudhi, J., Gill, D., Barker, D., Duda, M., Huang, X.-Y., Wang, W., and Powers, J.: A Description of the Advanced Research WRF Version 3, NCAR Technical Note NCAR/TN-468+STR, doi:10.5065/D6DZ069T, 2008.

Skamarock, W. C., Klemp, J. B., Duda, M. G., Fowler, L. D., Park, S.-H., and Ringler, T. D.: A multiscale nonhydrostatic atmospheric model using centroidal voronoi tesselations and C-grid staggering, Mon. Weather Rev., 140, 3090–3105, doi:10.1175/MWR-D-11-00215.1, 2012.

Smiatek, G., Kunstmann, H., Knoche, R., and Marx, A.: Precipitation and temperature statistics in high-resolution regional climate models: evaluation for the European Alps, J. Geophys. Res., 114, D19107, doi:10.1029/2008JD011353, 2009.

Staniforth, A.: Regional modeling: a theoretical discussion, Meteorol. Atmos. Phys., 63, 15–29, doi:10.1007/BF01025361, 1997.

Sultan, B., Janicot, S., and Diedhiou, A.: The West African monsoon dynamics. Part I: Documentation of intraseasonal variability, J. Climate, 16, 3389–3406, doi:10.1175/1520-0442(2003)016<3389:TWAMDP>2.0.CO;2, 2003.

Sylla, M. B., Gaye, A. T., Jenkins, G. S., Pal, J. S., and Giorgi, F.: Consistency of projected drought over the Sahel with changes in the monsoon circulation and extremes in a regional climate model projections, J. Geophys. Res.-Atmos., 115, D16108, doi:10.1029/2009JD012983, 2010.

Sylla, M. B., Giorgi, F., Coppola, E., and Mariotti, L.: Uncertainties in daily rainfall over Africa: assessment of gridded observation products and evaluation of a regional climate model simulation, Int. J. Climatol., 33, 1805–1817, doi:10.1002/joc.3551, 2013.

Thuburn, J., Ringler, T. D., Skamarock, W. C., and Klemp, J. B.: Numerical representation of geostrophic modes on arbitrarily structured C-grids, J. Comput. Phys., 228, 8321–8335, doi:10.1016/j.jcp.2009.08.006, 2009.

von Storch, H., Langenberg, H., and Feser, F.: A spectral nudging technique for dynamical downscaling purposes, Mon. Weather Rev., 128, 3664–3673, doi:10.1175/1520-0493(2000)128<3664:ASNTFD>2.0.CO;2, 2000.

Wang, W., Bruyère, C., Duda, M., Dudhia, J., Gill, D., Kavulich, M., Keene, K., Lin, H.-C., Michalakes, J., Rizvi, S., Zhang, X., Berner, J., Smith, K., Beezley, J. D., Coen, J. L., Mandel, J., Chuang, H.-Y., McKee, N., Slovacek, T., and Wolff, J.: WRF Users Guide, available at: http://www2.mmm.ucar.edu/wrf/users/docs/user_guide_V3/ARWUsersGuideV3.pdf (last access: 25 August 2015), 2014.

Warner, T. T., Peterson, R. A., and Treadon, R. E.: A tutorial on lateral boundary conditions as a basic and potentially serious limitation to regional numerical weather prediction, B. Am. Meteorol. Soc., 78, 2599–2617, doi:10.1175/1520-0477(1997)078<2599:ATOLBC>2.0.CO;2, 1997.

Weisman, M. L., Skamarock, W. C., and Klemp, J. B.: The resolution dependence of explicitly modeled convective systems, Mon. Weather Rev., 125, 527–548, doi:10.1175/1520-0493(1997)125<0527:TRDOEM>2.0.CO;2, 1997.

Willmott, C. and Matsuura, K.: University of Delaware Air Temperature and Precipitation, Long Term Monthly Means V3.01, available at: http://www.esrl.noaa.gov/psd/data/gridded/data.UDel_AirT_Precip.html (last access: 25 August 2015), 2014.

Zaengel, G., Reinert, D., Rípodas, P., and Baldauf, M.: The ICON (ICOsahedral Non-hydrostatic) modelling framework of DWD and MPI-M: description of the non-hydrostatic dynamical core, Q. J. Roy. Meteor. Soc., 141, 563–579, 2015.

Table 1. Specifications of the four HPC facilities used for the medium-size scaling tests in Sect. 2.

	TGCC Curie*	FZJ Juqueen	SCC ForHLR1*	FZJ Juropatest
System	B510 Bullx	IBM Blue Gene/Q	Megware MiriQuid	T-Platform V210s
Processor	Intel Sandy Bridge E5-2680, 2.7 GHz	IBM PowerPC A2 1.6 GHz	Intel Ivy Bridge E5-2670v2, 2.4 GHz	Intel Haswell E5-2695v3, 2.3 GHz
Cores/node	2×8	1×16	2×20	2×14
Mem./node	64 GB	16 GB	64 GB	128 GB
Nodes total	5040	28 672	512	60
MPI network	Infiniband QDR 5 GB s^{-1}	5-D Torus 40 GB s^{-1}	Infiniband FDR 7 GB s^{-1}	Infiniband FDR 7 GB s^{-1}
Filesystem	LUSTRE 100 GB s^{-1}	GPFS 200 GB s^{-1}	LUSTRE 16 GB s^{-1}	GPFS 30 GB s-1
I/O network	same as MPI	10 Gb Ethernet**	same as MPI	10 Gb Ethernet
MPI library	Bull MPI	IBM MPI	Intel MPI	Intel MPI
Compiler	Intel icc/ifort	IBM bgxlc/bgxlf95	Intel icc/ifort	Intel icc/ifort
Peak perf.	2 Petaflops	5.9 Petaflops	216 Teraflops	72 Teraflops

* "thin nodes" only, ignoring "fat nodes" (larger memory) and "hybrid nodes" (GPU accelerators)
** using dedicated I/O nodes, typically 128 I/O nodes per 1024 nodes (1 rack)

Table 2. Fit coefficients of modified Amdahl's law (2) for the three test cases and the two classes of architectures, with a common value for the exponent $X = 2.85$. For the Blue Gene system, only the fit to for the 60–12 km mesh should be used to estimate the required computing times.

Architecture	T(1) [s]	A	B
120 km			
Linux-Cluster	18 652	1.5×10^{-3}	3.3×10^{-4}
Blue Gene	413 440	1.8×10^{-4}	3.3×10^{-5}
100–25 km			
Linux-Cluster	86 523	8.5×10^{-4}	5.2×10^{-5}
Blue Gene	1 274 880	2.4×10^{-4}	5.5×10^{-6}
60–12 km			
Linux-Cluster	422 424	4.9×10^{-4}	1.4×10^{-5}
Blue Gene	6 579 200	8.4×10^{-5}	6.1×10^{-6}

Table 3. MPAS-A 3 km global simulation experiment (strong scaling tests).

Nodes	Tasks (MPI only)	Bootstrapping [s]	Initial read [s]
4096	65 536	1260	1260
8192	131 072	1370	590
16 384	262 144	1560	1020
24 576	393 216	1680	1080
28 672	458 752	1740	1140
Nodes	Integration time f. 1 h model time [s]	Parallel efficiency integration only	Integration in 24 h walltime*
4096	1760	100.0 %	29 h
8192	960	91.2 %	53 h
16 384	490	90.1 %	104 h
24 576	335	87.7 %	152 h
28 672	360	69.5 %	141 h

* Estimated time extrapolated from 1 h integration, including initial bootstrapping/reading, output to disk, 12 s time step

Table D1. Scaling tests for the 120 km regular mesh on the four HPC sites.

HPC site	Nodes	Tasks	Realtime [s] per 24 h	CPUh per 24 h	Parallel efficiency	Cells/ task
Curie	1	16	1306	5.8	100.0 %	2560
	2	32	633	5.6	103.16 %	1280
	4	64	377	6.7	86.6 %	640
	6	96	305	8.1	71.37 %	427
	8	128	269	9.6	60.69 %	320
	10	160	236	10.5	55.34 %	256
	12	192	212	11.3	51.34 %	213
	16	256	187	13.3	43.65 %	160
	24	384	182	19.4	29.9 %	107
	32	512	181	25.7	22.55 %	80
ForHLR1	1	20	863	4.8	100.0 %	2048
	2	40	486	5.4	88.79 %	1024
	3	60	317	5.3	90.75 %	683
	4	80	255	5.7	84.61 %	512
	6	120	190	6.3	75.7 %	341
	8	160	149	6.6	72.4 %	256
	10	200	135	7.5	63.93 %	205
	12	240	128	8.5	56.18 %	171
	16	320	127	11.3	42.47 %	128
	18	360	131	13.1	36.6 %	114
	20	400	130	14.4	33.19 %	102
Jtest-full	1	28	651	5.1	100.0 %	1463
	2	56	333	5.2	97.75 %	731
	4	112	177	5.5	91.95 %	366
	6	168	133	6.2	81.58 %	244
	8	224	109	6.8	74.66 %	183
	10	280	102	7.9	63.82 %	146
	12	336	92	8.6	58.97 %	122
	15	420	91	10.6	47.69 %	98
Jtest-half	1	14	1172	9.1	100.0 %	2926
	2	28	628	9.8	93.31 %	1463
	4	56	317	9.9	92.43 %	731
	6	84	220	10.3	88.79 %	488
	8	112	167	10.4	87.72 %	366
	10	140	142	11.0	82.54 %	293
	15	210	95	11.1	82.25 %	195
	20	280	87	13.5	67.36 %	146
	25	350	88	17.1	53.27 %	117
	30	420	99	23.1	39.46 %	98
Juqueen	32	512	661	94.0	100.0 %	80
	64	1024	477	135.7	69.29 %	40
	96	1536	586	250.0	37.6 %	27
	128	2048	608	345.9	27.18 %	20
	192	3072	751	640.9	14.67 %	13

Table E1. Scaling tests for the 100–25 km variable mesh on the four HPC sites.

HPC site	Nodes	Tasks	Realtime [s] per 24 h	CPUh per 24 h	Parallel efficiency	Cells/ task
Curie	1	16	6086	27.05	100.0 %	10 240
	4	64	1528	27.16	99.57 %	2560
	8	128	782	27.8	97.28 %	1280
	16	256	424	30.15	89.71 %	640
	24	384	314	33.49	80.76 %	427
	32	512	265	37.69	71.77 %	320
	48	768	223	47.57	56.86 %	213
	64	1024	263	74.81	36.16 %	160
	96	1536	271	115.63	23.39 %	107
	128	2048	293	166.68	16.23 %	80
	192	3072	541	461.65	5.86 %	53
ForHLR1	1	20	4413	24.52	100.0 %	8192
	4	80	1129	25.09	97.72 %	2048
	8	160	555	24.67	99.39 %	1024
	10	200	474	26.33	93.1 %	819
	15	300	346	28.83	85.03 %	546
	20	400	316	35.11	69.83 %	410
	25	500	257	35.69	68.68 %	328
	30	600	239	39.83	61.55 %	273
	40	800	253	56.22	43.61 %	205
	45	900	196	49.0	50.03 %	182
	50	1000	208	57.78	42.43 %	164
	55	1100	217	66.31	36.98 %	149
	60	1200	210	70.0	35.02 %	137
Jtest-full	1	28	3361	26.14	100.0 %	5852
	2	56	1644	25.57	102.22 %	2926
	4	112	807	25.11	104.12 %	1463
	8	224	418	26.01	100.51 %	731
	10	280	349	27.14	96.3 %	585
	15	420	260	30.33	86.18 %	390
	20	560	215	33.44	78.16 %	293
	25	700	186	36.17	72.28 %	234
	30	840	168	39.2	66.69 %	195
	40	1120	180	56.0	46.68 %	146
	50	1400	186	72.33	36.14 %	117
Jtest-half	1	14	5405	42.04	100.0 %	11 703
	2	28	2766	43.03	97.7 %	5852
	4	56	1487	46.26	90.87 %	2926
	8	112	734	45.67	92.05 %	1463
	10	140	620	48.22	87.18 %	1170
	15	210	407	47.48	88.53 %	780
	20	280	323	50.24	83.67 %	585
	30	420	233	54.37	77.32 %	390
	35	490	201	54.72	76.83 %	334
	40	560	286	88.98	47.25 %	293
	50	700	229	89.06	47.21 %	234
	55	770	226	96.68	43.48 %	213
Juqueen	32	512	2250	320.0	100.0 %	320
	64	1024	1365	388.27	82.42 %	160
	96	1536	1323	564.48	56.69 %	107
	128	2048	1168	664.46	48.16 %	80
	160	2560	1004	713.96	44.82 %	64
	192	3072	940	802.13	39.89 %	53
	256	4096	700	796.44	40.18 %	40
	384	6144	554	945.49	33.84 %	27
	512	8192	562	1278.86	25.02 %	20
	1024	16 384	1749	7959.89	4.02 %	10

Table F1. Scaling tests for the 60–12 km variable mesh on the four HPC sites.

HPC site	Nodes	Tasks	Realtime [s] per 24 h	CPUh per 24 h	Parallel efficiency	Cells/ task
Curie	1	16	27 428	121.9	100.0 %	33 472
	4	64	7715	137.2	88.88 %	8368
	8	128	3955	140.6	86.69 %	4184
	16	256	2025	144.0	84.65 %	2092
	24	384	1349	143.9	84.72 %	1395
	48	768	777	165.8	73.54 %	697
	64	1024	690	196.3	62.11 %	523
	96	1536	567	241.9	50.39 %	349
	128	2048	484	275.3	44.27 %	262
	192	3072	505	430.9	28.29 %	174
	256	4096	628	714.5	17.06 %	131
	384	6144	604	1030.8	11.83 %	87
ForHLR1	2	40	10 967	121.9	100.0 %	13 389
	4	80	5645	125.4	97.14 %	6694
	8	160	2876	127.8	95.33 %	3347
	15	300	1584	132.0	92.31 %	1785
	20	400	1234	137.1	88.87 %	1339
	30	600	922	153.7	79.3 %	893
	40	800	720	160.0	76.16 %	669
	45	900	710	176.1	68.65 %	595
	50	1000	704	195.6	62.31 %	536
	60	1200	649	216.3	56.33 %	446
	80	1600	635	282.2	43.18 %	335
	100	2000	543	301.7	40.39 %	268
	120	2400	459	306.0	39.82 %	223
Jtest-full	1	28	17 151	133.4	100.0 %	19 127
	2	56	8552	133.0	100.27 %	9563
	8	224	2106	131.0	101.8 %	2391
	10	280	1704	132.5	100.65 %	1913
	15	420	1113	129.9	102.73 %	1275
	20	560	862	134.1	99.48 %	956
	25	700	716	139.2	95.82 %	765
	30	840	613	143.0	93.26 %	638
	40	1120	507	157.7	84.57 %	478
	50	1400	437	169.9	78.49 %	383
	55	1540	428	183.1	72.86 %	348
Jtest-half	1	14	29 000	225.6	100.0 %	38 254
	2	28	14 487	225.4	100.09 %	19 127
	4	56	7143	222.2	101.5 %	9563
	8	112	4126	256.7	87.86 %	4782
	10	140	2833	220.3	102.36 %	3825
	15	210	2133	248.9	90.64 %	2550
	20	280	1583	246.2	91.6 %	1913
	30	420	1097	256.0	88.12 %	1275
	40	560	784	243.9	92.47 %	956
	50	700	624	242.7	92.95 %	765
	55	770	621	265.7	84.91 %	696
Juqueen	32	512	11 800	1678.2	100.0 %	1046
	64	1024	6950	1976.9	84.89 %	523
	96	1536	4490	1915.7	87.6 %	349
	128	2048	4100	2332.4	71.95 %	262
	192	3072	2991	2552.3	65.75 %	174
	256	4096	2979	3389.4	49.51 %	131
	384	6144	2514	4290.6	39.11 %	87
	512	8192	2468	5616.1	29.88 %	65
	640	10 240	4462	12 691.9	13.22 %	52
	1024	16 384	9863	44 887.6	3.74 %	33

Table G1. Scalasca/Score-P profile of percentages of time spent during the model integration, i. e., ignoring the model initialisation, for the 60–12 km mesh on Juqueen. The most significant contributors to each of the categories are listed for completeness.

Step in time integration	2048-task run	4096-task run	8192-task run
Physics	13 %	9 %	4 %
Microphysics			
Radiation			
PBL/CU scheme			
Land surface model			
Communication	35 %	41 %	70 %
All-to-all min/max values			
Exchange halo fields			
Dynamics	36 %	34 %	11 %
Acoustic time step			
Advance scalars			
Compute/add tendencies			
Solve diagnostics			
I/O	16 %	16 %	15 %
Build/write output stream			
Boundary updates (read)			

Table H1. File sizes of input (I) and output (O) files for all meshes (uncompressed netCDF3/4). For a given mesh with n grid cells, an estimate of the required file sizes is calculated from the other runs. The given file sizes for the comprehensive output file `output.nc` correspond to a set of 20 atmospheric variables (on 41 geometric height levels), 3 soil variables (on 4 soil levels), 32 single-level variables (surface, top of atmosphere, ...) and 40 static variables, required for the mesh geometry. Likewise, the smaller diagnostic output files `diag.nc` contain 24 variables on single levels (surface, pre-defined pressure levels, ...). It should be noted that the number of atmospheric, soil and single-level variables can be customised for both `output.nc` and `diag.nc` at runtime using simple ASCII files. Further, in an attempt to speed up the I/O and reduce the file sizes, the output of the static variables could be suppressed for all but a single `output.nc` file.

Mesh	Cells	`init.nc` (I)	`diag.nc` (O)	`restart.nc` (O)	`output.nc` (O)
120 km	40 962	758 MB	8.7 MB	1.6 GB	153 MB
100–25 km	163 842	3.0 GB	33 MB	6.0 GB	597 GB
60–12 km	535 554	9.7 GB	107 MB	20 GB	2.0 GB
3 km	65 536 002	1.2 TB	13 GB	2.4 TB	250 GB
	n	19.7 kB $\times n$	0.22 kB $\times n$	38.8 kB $\times n$	4.0 kB $\times n$

Table H2. Summary of the scaling tests: cheapest and fastest model run configurations for a 24 h model integration with disk I/O enabled (rt = realtime).

HPC site	Nodes cheapest run	Tasks cheapest run	CPUh cheapest run	Nodes fastest run	Tasks fastest run	Speedup fastest run
Regular 120 km mesh						
Curie	2	32	5.6	32	512	477 × rt
ForHLR1	1	20	4.8	16	320	680 × rt
Juqueen	32	512	94.0	64	1024	181 × rt
Jtest-half	1	14	9.1	20	280	993 × rt
Jtest-full	1	28	5.1	15	420	949 × rt
Variable 100–25 km mesh						
Curie	1	16	27.1	48	768	387 × rt
ForHLR1	1	20	24.5	45	900	441 × rt
Juqueen	32	512	320	384	6144	156 × rt
Jtest-half	1	14	42.0	35	490	430 × rt
Jtest-full	4	112	25.1	30	840	514 × rt
Variable 60–12 km mesh						
Curie	1	16	122	128	2048	179 × rt
ForHLR1	2	40	122	120	2400	188 × rt
Juqueen	32	512	1678	512	8192	35 × rt
Jtest-half	10	140	220	55	770	139 × rt
Jtest-full	15	420	130	55	1540	202 × rt
Regular 3 km mesh						
Juqueen	4096	65 536	1.3 Mio	24 576	393 216	6.3 × rt

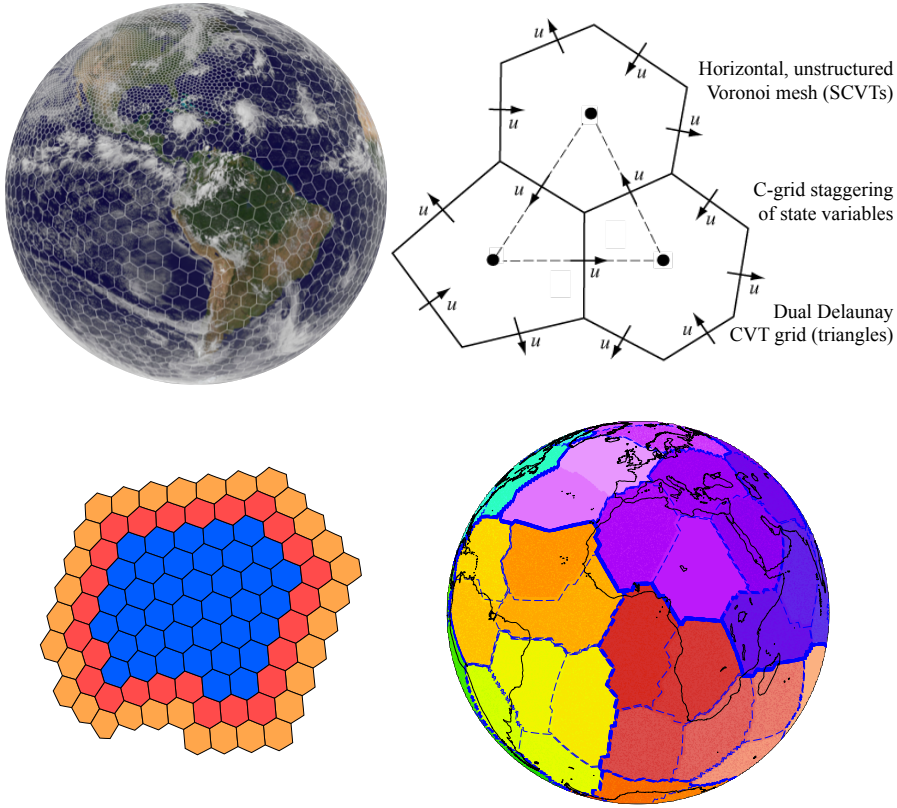

Figure 1. (upper left) Sketch of a variable-resolution Voronoi mesh used for the horizontal grid; (upper right) C-grid staggering of state variables (adapted from Skamarock et al., 2012); (lower left) a block of owned cells (blue) assigned to an MPI task, along with two layers of halo cells (red, orange); (lower right) partitioning of the regular 120 km mesh with 40 962 grid cells for 64 tasks.

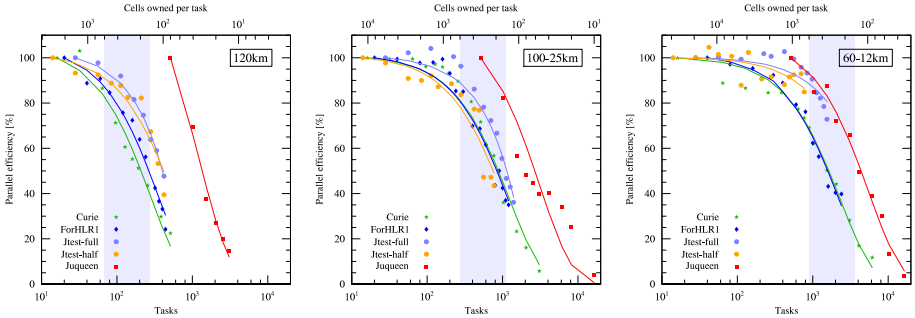

Figure 2. Parallel efficiency [%] for the 120 km regular mesh with 40 962 grid cells, the variable 100–25 km mesh with 163 842 grid cells, and the variable 60–12 km mesh with 535 554 grid cells.

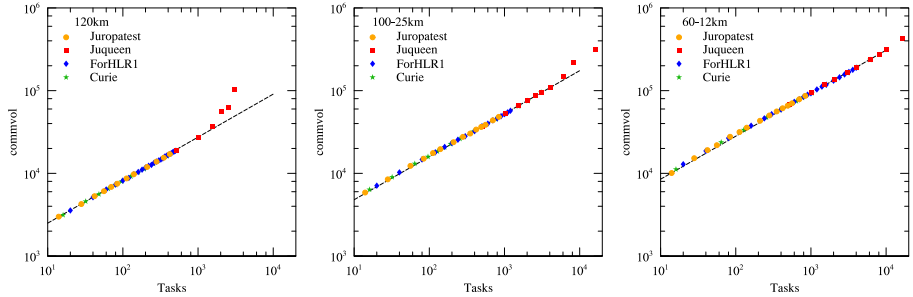

Figure 3. Communication volume as function of number of tasks for the three test cases (left: 120 km; middle: 100–25 km; right: 60–12 km). The dashed lines correspond to an exponential relation with power law index 0.52. For large numbers of tasks, a runaway growth can be seen for the 120 km and to some extent also for the 100–25 km mesh.

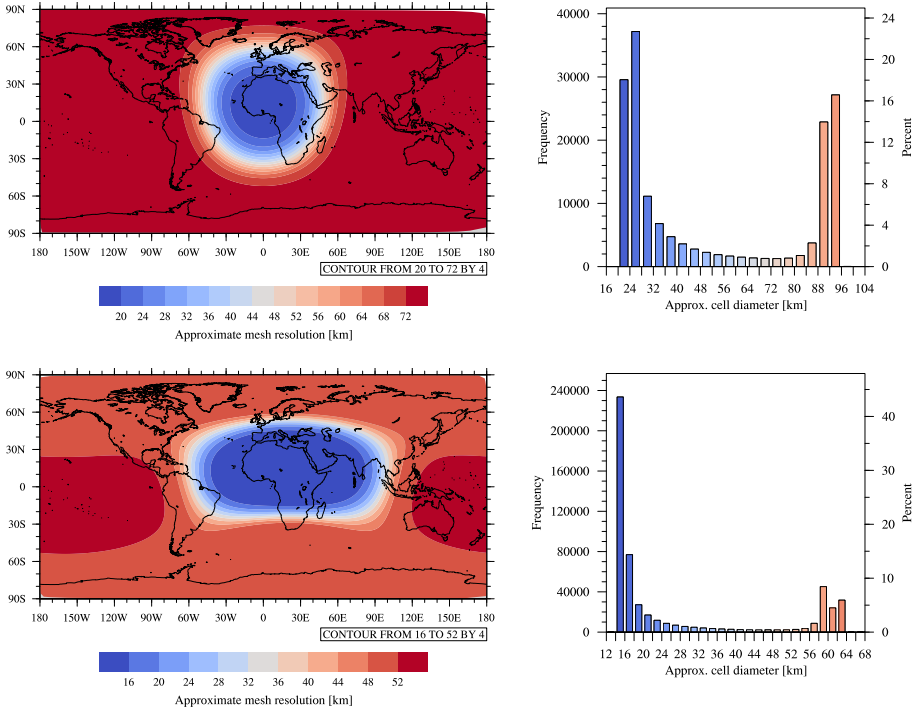

Figure 4. (left) Approximate mesh cell sizes and (right) distribution of mesh cell sizes for (top) the variable 100–25 km mesh with 163 842 grid cells and (bottom) the variable 60–12 km mesh with 535 554 grid cells.

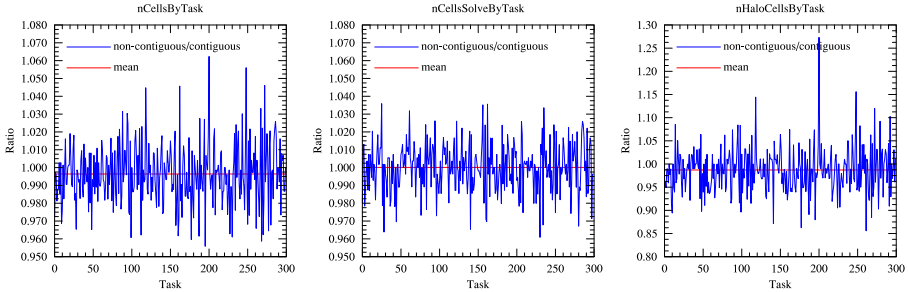

Figure 5. Ratio of (left) total number of cells, (mid) number of owned cells, and (right) number of halo cells per task between the non-contiguous and the contiguous graph partition of the variable 100–25 km mesh for 300 tasks.

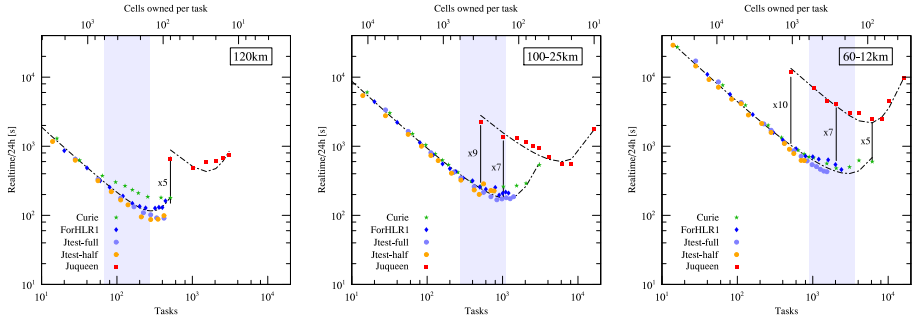

Figure 6. Total runtimes [s] for the 120 km regular mesh with 40 962 grid cells, the variable 100–25 km mesh with 163 842 grid cells, and the variable 60–12 km mesh with 535 554 grid cells. Indicated are time ratios between the Linux-cluster systems Curie, ForHLR1 and Juropatest, and the Blue Gene system Juqueen. Fits to the total runtimes are obtained separately for the Linux-cluster systems and the Blue Gene system, using a modified version of Amdahl's law (see Equation (2) and Table 2), and are displayed as dashed black lines.

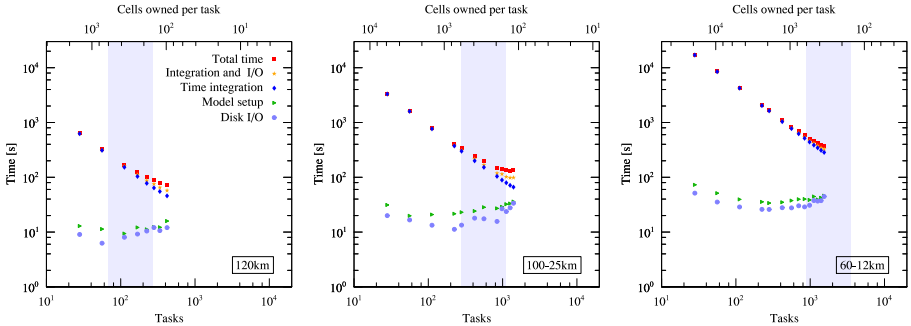

Figure 7. Required time for a 24 h integration [s] split into time integration, model setup and disk I/O for the three test cases on Jtest-full. Also displayed are the total time and a combination of time integration and disk I/O to reflect the parallel performance of MPAS-A for longer model runs, for which the initial setup costs are amortised.

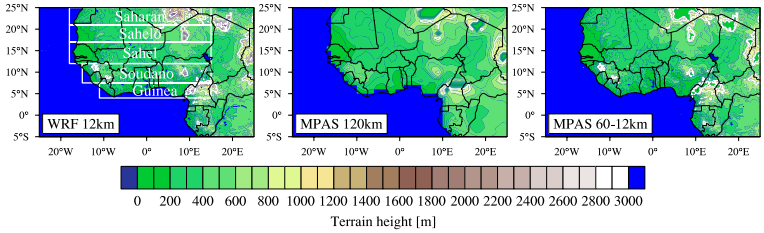

Figure 8. Model topography (terrain height in m) for the WRF reference and the two MPAS model runs for the West African WRF domain. The left panel also indicates five distinct agro-climatical zones, following a gradient of decreasing annual precipitation from South to North.

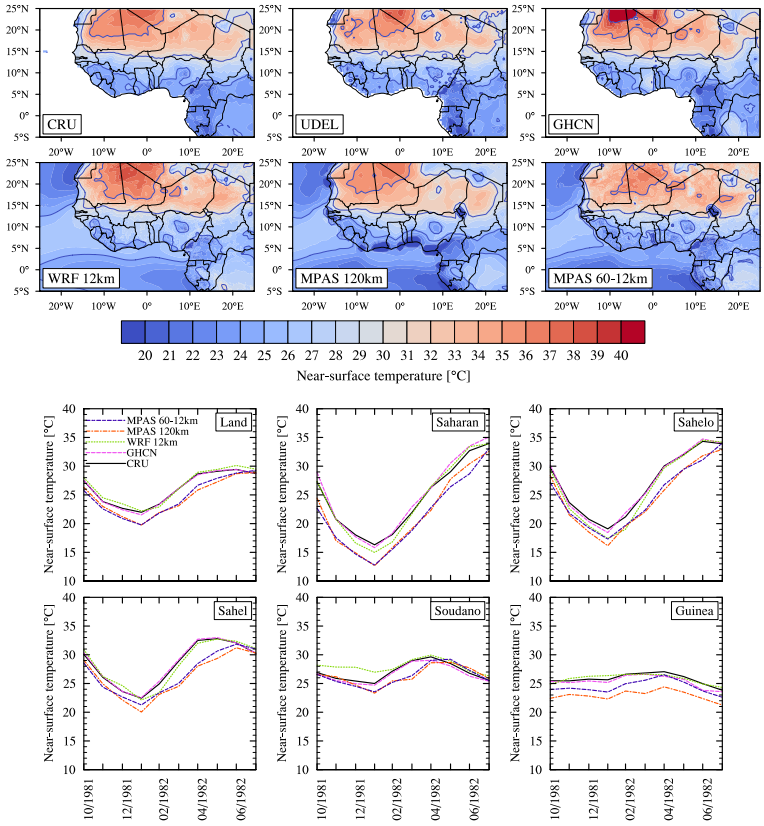

Figure 9. (top) Mean near-surface temperature (over land) in °C for July 1982 for the three observational data sets CRU, UDEL, GHCN, the WRF reference and the two MPAS model runs; (bottom) annual cycle of mean near-surface temperature over the entire land area and the five agro-climatical zones depicted in Fig. 8.

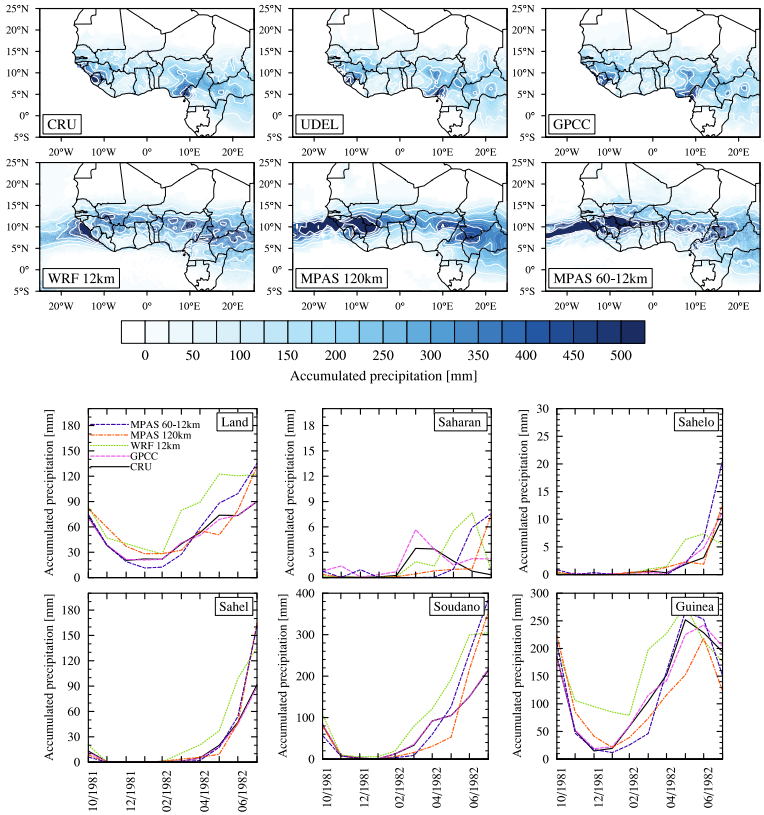

Figure 10. (top) Monthly precipitation (over land) in mm for July 1982 for the three observational data sets CRU, UDEL, GPCC, the WRF reference and the two MPAS model runs; (bottom) annual cycle of monthly precipitation over the entire land area and the five agro-climatical zones depicted in Fig. 8.

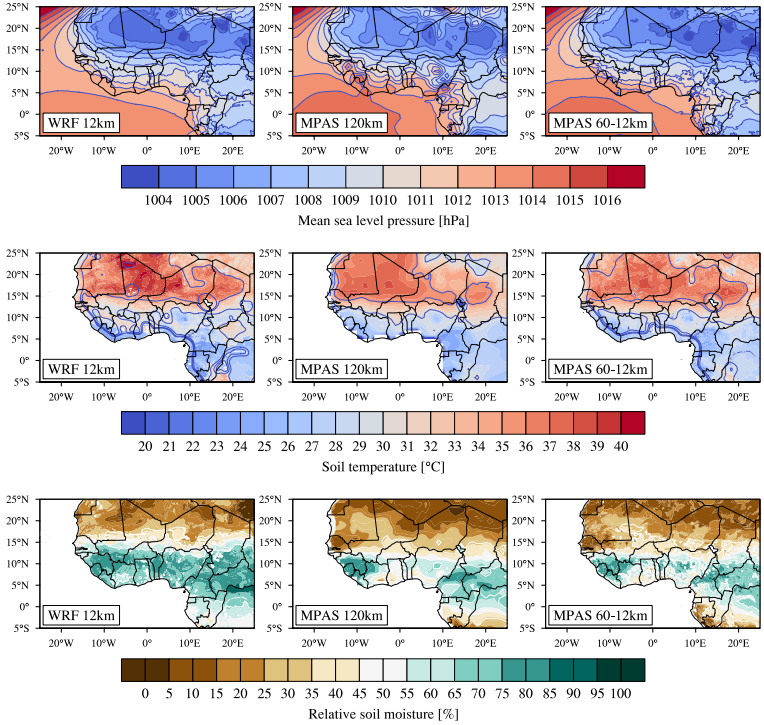

Figure 11. (top) Mean sea level pressure, (middle) mean soil temperature, and (bottom) mean relative soil moisture over land for July 1982 for the WRF reference and the MPAS model runs.

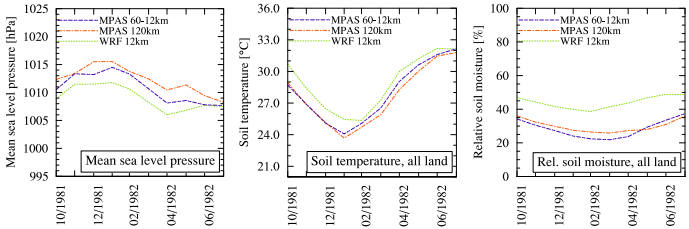

Figure 12. Annual cycle of (left) mean sea level pressure in hPa, (middle) mean soil temperature in °C, and (right) mean relative soil moisture in % over land for the WRF reference and the MPAS model runs.

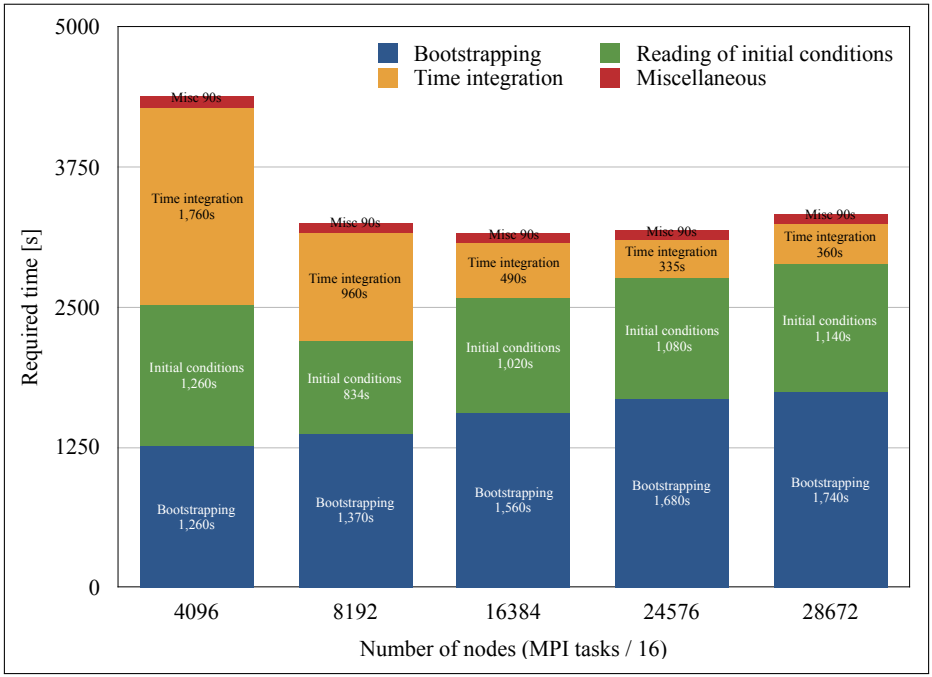

Figure 13. Required times for individual steps of the 3 km test runs on Juqueen (18 s time step).