BAMS Article

Kilometer-Scale Climate Models

Prospects and Challenges

Christoph Schär, Oliver Fuhrer, Andrea Arteaga, Nikolina Ban,
Christophe Charpilloz, Salvatore Di Girolamo, Laureline Hentgen,
Torsten Hoefler, Xavier Lapillonne, David Leutwyler, Katherine Osterried,
Davide Panosetti, Stefan Rüdisühli, Linda Schlemmer, Thomas C. Schulthess,
Michael Sprenger, Stefano Ubbiali, and Heini Wernli

ABSTRACT: Currently major efforts are underway toward refining the horizontal resolution (or grid spacing) of climate models to about 1 km, using both global and regional climate models (GCMs and RCMs). Several groups have succeeded in conducting kilometer-scale multiweek GCM simulations and decadelong continental-scale RCM simulations. There is the well-founded hope that this increase in resolution represents a quantum jump in climate modeling, as it enables replacing the parameterization of moist convection by an explicit treatment. It is expected that this will improve the simulation of the water cycle and extreme events and reduce uncertainties in climate change projections. While kilometer-scale resolution is commonly employed in limited-area numerical weather prediction, enabling it on global scales for extended climate simulations requires a concerted effort. In this paper, we exploit an RCM that runs entirely on graphics processing units (GPUs) and show examples that highlight the prospects of this approach. A particular challenge addressed in this paper relates to the growth in output volumes. It is argued that the data avalanche of high-resolution simulations will make it impractical or impossible to store the data. Rather, repeating the simulation and conducting online analysis will become more efficient. A prototype of this methodology is presented. It makes use of a bit-reproducible model version that ensures reproducible simulations across hardware architectures, in conjunction with a data virtualization layer as a common interface for output analyses. An assessment of the potential of these novel approaches will be provided.

https://doi.org/10.1175/BAMS-D-18-0167.1

Corresponding author: Christoph Schär, schaer@env.ethz.ch

Supplemental material: https://doi.org/10.1175/BAMS-D-18-0167.2

In final form 19 July 2019

AFFILIATIONS: Schär, Hentgen, Panosetti, Rüdisühli, Sprenger, Wernli—Atmospheric and Climate Science, ETH Zürich, Zürich, Switzerland; **Ban**—Atmospheric and Climate Science, ETH Zürich, Zürich, Switzerland, and University of Innsbruck, Innsbruck, Austria; **Schlemmer**—Atmospheric and Climate Science, ETH Zürich, Zürich, Switzerland, and German Weather Service (DWD), Offenbach, Germany; **Arteaga, Lapillonne**—Federal Office of Meteorology and Climatology MeteoSwiss, Zürich, Switzerland; **Fuhrer**—Federal Office of Meteorology and Climatology MeteoSwiss, Zürich, Switzerland, and Climate Modeling, Vulcan Inc., Seattle, USA **Charpilloz**—Atmospheric and Climate Science, ETH Zürich, and MeteoSwiss, Zürich, Switzerland; **Di Girolamo, Hoefler**—Computer Science, ETH Zürich, Zürich, Switzerland; **Leutwyler**—Atmospheric and Climate Science, ETH Zürich, Zürich, Switzerland, and Max Planck Institute for Meteorology, Hamburg, Germany; **Osterried**—Center for Climate Systems Modeling, ETH Zürich, Zürich, Switzerland; **Schulthess, Ubbiali**—Institute for Theoretical Physics, ETH Zürich, Zürich, Switzerland, and Swiss National Supercomputing Center, Lugano, Switzerland

While the basic scientific concepts of anthropogenic climate change are now well established, uncertainties in climate projections have remained staggeringly large. For instance, current estimates of the equilibrium climate sensitivity (ECS)—the equilibrium global surface warming in response to a doubling of atmospheric CO_2 concentration—are between 1.5° and 4.5°C. Over the last 40 years, this uncertainty range, covering a probability of 66%, has not narrowed (National Research Council 1979), and according to the most recent IPCC assessment report, even extreme values of the ECS (below 1°C and above 6°C) cannot be excluded (IPCC 2013). This evident uncertainty makes it difficult to plan for adequate response strategies essential to mitigate the anticipated warming. Reducing this uncertainty is also of paramount importance in order to provide more reliable projections of sea level rise, regional climate change, and extreme events, which are essential to climate change adaptation.

The key reason behind the slow progress in reducing the uncertainties of climate projections is likely the lack of adequate computational resolution, together with the importance of small-scale processes in the climate system. In particular, there is evidence that the response of tropical and subtropical clouds may significantly amplify or reduce global warming, depending upon changes in cloud reflectivity with global warming (Bony and Dufresne 2005; Sherwood et al. 2014; Schneider et al. 2017, 2019). Likewise, eddy-resolving ocean models are expected to contribute toward reducing uncertainties in ECS by better representing ocean heat uptake (e.g., Gregory et al. 2002; Ringler et al. 2013; Hewitt et al. 2017), but in the current article we will focus on atmospheric models.

With the advent of emerging supercomputing platforms, and with the progress in high-resolution climate modeling, there are now promising prospects to refine the horizontal resolution[1] of global climate models from today's 50–100 km to 1–2 km, thereby explicitly resolving some of the small-scale convective cloud processes (e.g., thunderstorms and rain showers). There is the well-founded hope that this increase in resolution might lead to a quantum jump in climate modeling, as it enables replacing the parameterizations of moist convection and gravity wave drag by explicit treatments (Palmer 2014). It is also hoped that this will improve the simulation of the water cycle and of extreme events and reduce uncertainties in ECS. However, what resolution will actually be needed for the later purpose is not yet fully understood. On the one hand, convective cloud processes (dynamics, turbulence, and microphysics) occur on scales that are not fully resolved at kilometer resolution (Skamarock 2004; Neumann et al. 2019; Panosetti et al. 2019b). On the other hand, studies have indicated that there is some bulk convergence at grid resolutions around 2 km, that is, the feedbacks between convective clouds and the larger-scale flow are partly captured at resolutions at which the structural details of the cloud field are not yet fully resolved (Langhans et al. 2012; Harvey et al. 2017; Ito et al. 2017; Panosetti et al. 2019, 2020). Following Schulthess et al. (2019) and Neumann

[1] In this paper we are using the terms "resolution" and "grid spacing" synonymously.

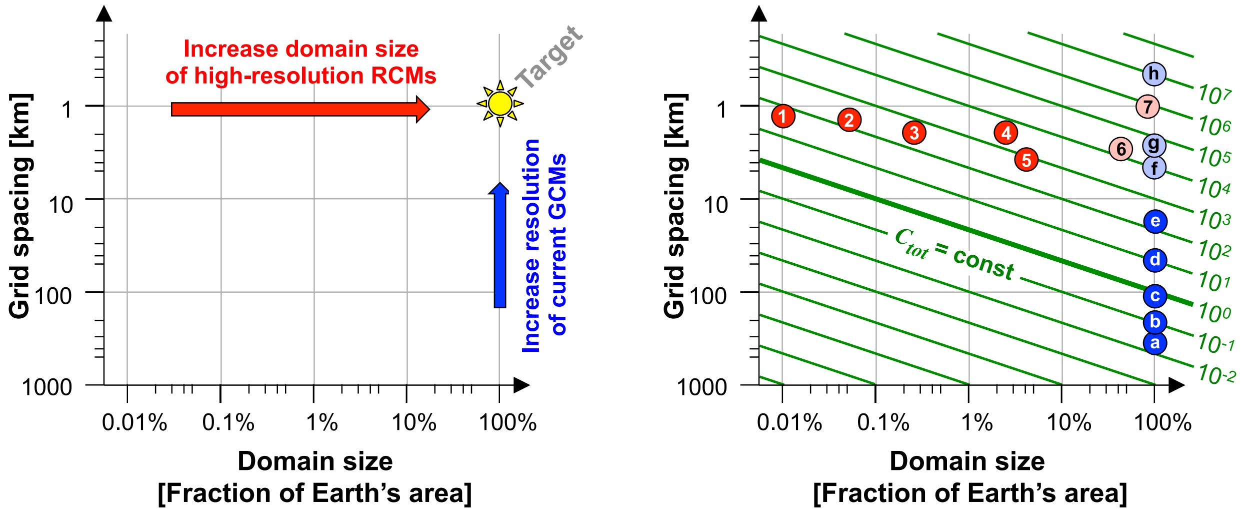


Fig. 1. (left) Approaching the target of global kilometer-scale climate simulations, represented by the sun symbol, either by refining the resolution of GCMs or by expanding the computational domain of high-resolution RCMs. The horizontal axes represent the domain size (fraction of Earth's surface covered by the simulations) and the vertical axes the grid spacing (km). (right) A selection of available simulations are indicated by the data points, showing simulations longer than 10 years in full colors, and short prototype simulations in faint colors. The green contours in the right panel show lines of same computational load, assuming that the time step is refined such as to keep the CFL number constant. Red symbols relate to RCMs: 1 = Knote et al. (2010), 2 = Kendon et al. (2014); 3 = Ban et al. (2014); 4 = Leutwyler et al. (2017); 5 = Liu et al. (2017), Prein et al. (2017); 6 = Bretherton and Khairoutdinov (2015); 7 = Fuhrer et al. (2018). Blue symbols relate to GCMs: a = CMIP1 models (IPCC 1995; average horizontal resolutions of models); b = CMIP3 models (IPCC 2001); c = CMIP5 models (IPCC 2013); d = Sakamoto et al. (2012); e = CMIP6 HighRes MIP (Haarsma et al. 2016), f = Neumann et al. (2019), g = DYAMOND simulations (Satoh et al. 2019), h = Miyamoto et al. (2013).

et al. (2019), we thus assume that a global resolution of 1 km is a suitable near-term target. Thus, further improvements in the parameterizations of the turbulence and microphysical processes appear essential, as these processes will remain poorly resolved.

The development and testing of climate models with horizontal resolutions of around 1 km is already well underway. Both global and regional models have contributed to this development, with the former refining the horizontal resolution on a global domain, and the latter expanding the computational domains of high-resolution limited-area models (Fig. 1). The target (1 km on a global domain) can be approached both ways. The figure also shows an estimate of the relative computational costs (green lines), assuming that the vertical resolution is kept constant, whereupon the number of operations scales with $N_z A \Delta x^{-3}$, with A denoting the horizontal area of the domain, N_z the number of vertical levels, and Δx the horizontal grid spacing. This scaling assumes perfect computational scalability and that the time step is refined together with the horizontal resolution, consistent with maintaining a constant Courant number, a measure for how far information propagates per time step relative to the grid spacing.

Some prototype simulations (e.g., Miyamoto et al. 2013; Fuhrer et al. 2018) are already close to the target (Fig. 1, right-hand panel), but these models have not yet been run over climate time periods, but merely over days to seasons. There are also major initiatives on the further development of these approaches, such as the Energy Exascale Earth System Model (E3SM; https://e3sm.org/) of the U.S. Department of Energy, or the high-resolution modeling activities at many weather and climate centers culminating in simulations of nine atmosphere-only codes at kilometer-scale resolution for a 40-day-long common simulation period (Satoh et al. 2019; DYAMOND: www.esiwace.eu/services-1/dyamond-initiative).

In any case, realizing the potential of global convection-resolving climate simulations requires enormous efforts and innovative solutions at the interface of computer and climate sciences.

Some of these aspects will be addressed in this paper: How can we efficiently leverage the next generations of supercomputers? What programming languages should we use to make our climate codes future-proof? How can we overcome the data avalanche generated by high-resolution models? How can we trade storing the model output with recomputation of model simulations?

We will discuss these aspects by exploiting a version of the Consortium for Small-Scale Modeling (COSMO) limited-area model that has extensively been used at kilometer resolution in the last decade, and that can be run entirely on modern supercomputers at unprecedented speed. While this framework is still far away from the global-domain kilometer-scale target, the main challenges are exposed and potential solutions can be assessed. The second and third sections of the paper outline the main challenges and potential strategies, the fourth section presents some specific applications and results, and the final section presents the conclusions of the study.

Challenges of kilometer-scale resolution

Exploiting next generation hardware architectures. While high-performance computing (HPC) system performance has continued to increase year after year (https://top500.org), a series of fundamental technology transitions had profound impacts on programming models and simulation software. After transistor power efficiency has grown exponentially for decades, the energy required to move data has become the dominant performance constraint (e.g., Kestor et al. 2013). Figure 2 presents the energy consumption for elementary store and compute operations. It illustrates the fact that for common operations (reading two double precision floating point numbers from system memory, performing an addition, and storing the result back into system memory) the energy required for the data transfers is approximately 100 times larger than that required to execute the actual arithmetic operation. Finally, energy constraints for large HPC systems have led to heterogeneous node designs with accelerators such as graphics processing units (GPUs). With the end of exponential scaling of transistor size toward the end of the last decade (often referred to as Moore's law), disruptive architectural changes and architectural diversity and complexity are expected to continue to increase. To take advantage of the computational power of the largest HPC systems, climate models have to be able to run on these emerging hardware architectures.

Lacking proper programming abstractions, details of these novel hardware architectures are exposed to the application developer via software libraries (e.g., MPI to handle data movement between remote memories), extensions to programming languages (e.g., OpenACC

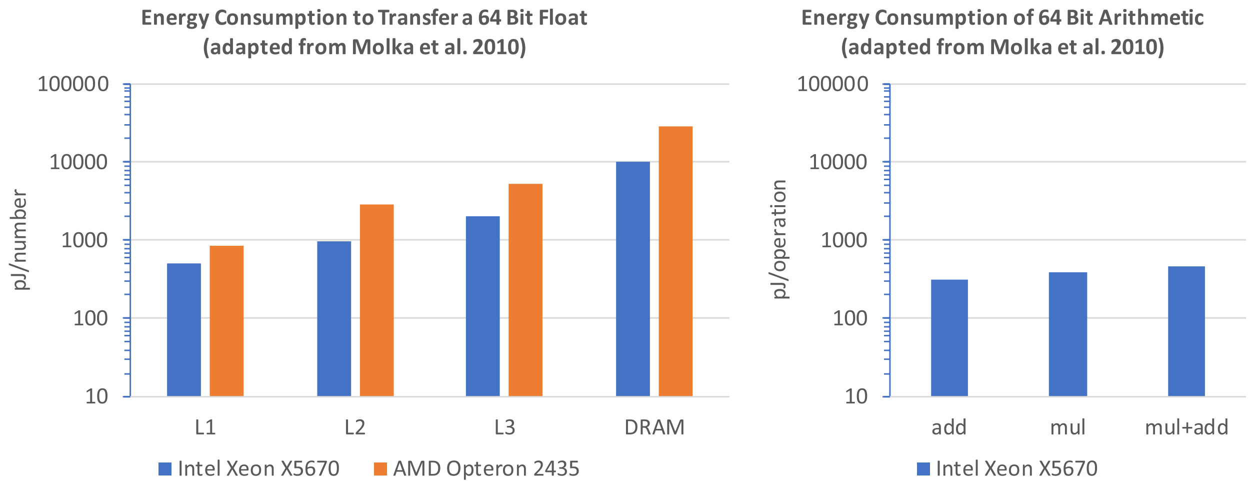
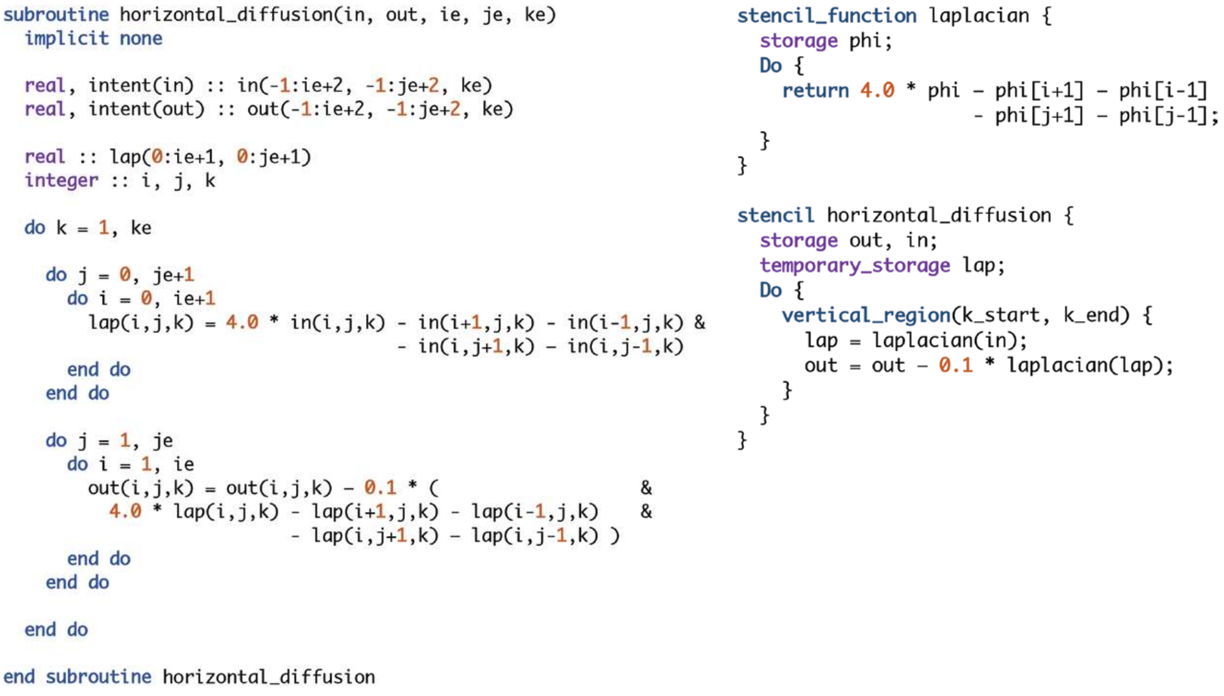


Fig. 2. Comparison of the energy consumption to transfer a single 64-bit floating point number from different levels of cache (L1, L2, L3) and system memory (DRAM), and the energy consumption to execute a 64-bit arithmetic operation (addition, multiplication, and fused multiply add). Data are for Intel Xeon X5670 and AMD Opteron 2435 processors, adapted from Molka et al. (2010).

Domain-specific languages explained

A domain-specific language (DSL) is a language specialized to a specific application domain, in our case the dynamical cores of weather and climate models. To illustrate the power of DSLs, two implementations of a simple fourth-order horizontal diffusion operator are given below. The code on the left is an abridged FORTRAN implementation extracted from a climate model. The original optimized version entails significantly more code to specify parallelism, data placement, and data movement. The code on the right shows an implementation in the gtclang (https://github.com/MeteoSwiss-APN/gtclang) high-level DSL, which is part of the GridTools Framework. The code shown corresponds to the complete code implemented by the domain (climate) scientist. Details of how data are stored in memory and order of iteration over the computational grid are no longer visible. The responsibility to generate optimized, parallel code for a specific hardware architecture is delegated to the DSL compiler. As a result, the DSL implementation is very concise and maintainable. DSLs vary in the level of abstraction. In the example shown, the responsibility to choose an appropriate numerical scheme for the Laplacian remains with the domain scientist. A DSL with a higher level of abstraction may hide the choice of numerics as well as computational grid from the user.



```
subroutine horizontal_diffusion(in, out, ie, je, ke)
  implicit none

  real, intent(in) :: in(-1:ie+2, -1:je+2, ke)
  real, intent(out) :: out(-1:ie+2, -1:je+2, ke)

  real :: lap(0:ie+1, 0:je+1)
  integer :: i, j, k

  do k = 1, ke

    do j = 0, je+1
      do i = 0, ie+1
        lap(i,j,k) = 4.0 * in(i,j,k) - in(i+1,j,k) - in(i-1,j,k) &
                                     - in(i,j+1,k) - in(i,j-1,k)
      end do
    end do

    do j = 1, je
      do i = 1, ie
        out(i,j,k) = out(i,j,k) - 0.1 * (                   &
          4.0 * lap(i,j,k) - lap(i+1,j,k) - lap(i-1,j,k)    &
                           - lap(i,j+1,k) - lap(i,j-1,k) )
      end do
    end do

  end do

end subroutine horizontal_diffusion
```

```
stencil_function laplacian {
  storage phi;
  Do {
    return 4.0 * phi - phi[i+1] - phi[i-1]
                     - phi[j+1] - phi[j-1];
  }
}

stencil horizontal_diffusion {
  storage out, in;
  temporary_storage lap;
  Do {
    vertical_region(k_start, k_end) {
      lap = laplacian(in);
      out = out - 0.1 * laplacian(lap);
    }
  }
}
```

Fig. SB1. Comparison of a second-order Laplacian in (left) FORTRAN and (right) gtclang.

compiler directives for GPU programming) or entirely new programming languages (e.g., CUDA, a language for GPU programming). The climate modeling community has begun to realize the enormity of the challenge facing them. A climate model typically has on the order of one million lines of source code, rendering the traditional programming paradigms and development process unsustainable. As a consequence, global fully coupled climate models are not capable of efficiently leveraging current leadership class HPC systems. The effort required for the maintenance, validation, and migration of climate models has increased drastically. This has become known as the software productivity gap (Lawrence et al. 2018).

One important design principle of modern software engineering is the separation of concerns. It means splitting a computer program into different parts, where each part deals with a separate concern. To this end, there has been an increased interest in the development of higher-level abstractions for weather and climate models (e.g., Bertagna et al. 2019; Adams et al. 2019; Fuhrer et al. 2014; Clement et al. 2018). For example, domain-specific languages (DSLs) can help separate hardware architecture-dependent details from the source code written by the climate scientists (see "Domain-specific languages explained" sidebar). As a result, the source code of a global climate model (GCM) or regional climate model (RCM) implemented using a DSL is more concise and more easily maintainable.

Choice of numerical methods. Weather and climate models consist of a dynamical core and physical parameterizations. For large-scale atmospheric simulations at resolutions explicitly resolving deep convection, choosing a fully compressible, nonhydrostatic set of primitive equations is essential (Davies et al. 2003). The optimal (fastest for a given accuracy) numerical approach for solving these equations depends on the hardware architecture and the underlying numerical method. In particular, the exchange of data across the computational mesh (and thus data movement across compute nodes) is strongly influenced by the numerical method employed. Some schemes avoid global communication (i.e., data are moved only between neighboring grid points), but have rigorous time step restrictions (e.g., horizontally explicit, vertically implicit methods; see Lock et al. 2014). Others require iterative solvers and/or global communication at each time step, but allow for much longer time steps (e.g., semi-implicit semi-Lagrangian or pseudo-spectral methods; see Tanguay et al. 1990; Temperton et al. 2001).

In the real atmosphere, the speed of sound is the fastest velocity in the system. Thus, the temporal evolution of the atmosphere at a given location is influenced by a neighborhood determined approximately by sound propagation (Fig. 3, left). This neighborhood is referred to as the physical domain of dependence. Any numerical scheme must respect this principle, and the numerical domain of dependence must be identical to or larger than its physical counterpart. However, in order to minimize data communication, the numerical domain of dependence should also be as small as allowable. For some implementations (Zängl et al. 2015; Skamarock et al. 2012; Baldauf et al. 2011; Kühnlein et al. 2019) data are exchanged at about twice the minimum rate as determined by sound propagation (Fig. 3, middle), while the spectral approach requires global communication at each time step (Fig. 3, right). It is thus evident that data communication requirements are strongly affected by the underlying numerical approach, and the implied computational costs are influenced in turn by the hardware configuration of the employed supercomputer (e.g., its node-to-node network topology). With higher computational resolution (when more compute nodes become involved), or with current hardware trends (when data movement become more costly), numerical methods with little across-node communication will often have a faster performance.

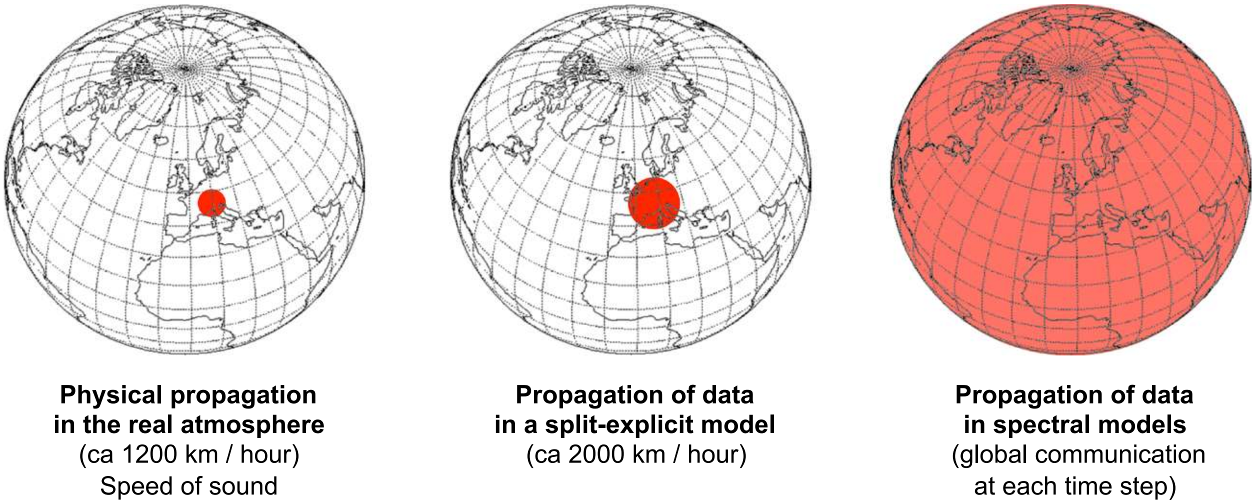

Physical propagation in the real atmosphere
(ca 1200 km / hour)
Speed of sound
Propagation of data in a split-explicit model
(ca 2000 km / hour)
Propagation of data in spectral models
(global communication at each time step)

Fig. 3. Data exchange in atmospheric models. To ensure numerical stability, the exchange of data in an atmospheric model must exceed that of the physical propagation in the real atmosphere. (left) In the real atmosphere information travels approximately with the speed of sound. Within an hour, an air parcel over Zurich will thus exchange information within a radius of about 1,200 km. The data exchange in numerical models strongly depends upon the numerical formulation. For instance, (middle) in a split explicit scheme, data will be exchanged within a radius about twice that size, while (right) in a spectral model, data will be exchanged globally at each time step.

Among other methodologies, the split-explicit approach, as employed in our work-horse COSMO model, is well suited for this challenge, as it restricts communication to near-neighbors and provides perfect weak scaling (Fuhrer et al. 2018). Perfect weak scaling means that the computational domain of a simulation can be expanded in parallel with the number of computational nodes employed, without increasing the wall-clock time required to run the simulation.

Coping with the data avalanche. The climate modeling community is already struggling to cope with the data volumes produced by the current simulation efforts. For instance, performing all the simulations considered for phase 6 of the Coupled Model Intercomparison Project (CMIP6; Eyring et al. 2016) would amount to about 800 TB of output for each of the 100 participating models (Balaji et al. 2018). While it is impossible to foresee all the experiments envisioned in future editions, projecting the output volume of the compulsory Diagnostic, Evaluation and Characterization of Klima (DECK) simulations (Table 1) seems like an illustrative exercise. The DECK consist of four simulations, which every model participating in CMIP6 needs to complete (see Table 1 for details). Performing these simulations at kilometer-scale resolution would exceed the expected overall data volume of CMIP6 by about three orders of magnitude (Table 1, fourth column). This assumes that only a small fraction of the total data is written to disk, while for some applications higher output frequency is needed (see, e.g., examples in “Sophisticated analysis using the virtualization layer” section). A more recent development are DECK simulations with up to 100–1,000 ensemble members (large/grand ensembles; e.g., Maher et al. 2019). While these simulations would be particularly useful to address rare and extreme events, the expected data volume typically prevents storing data at subdaily intervals, which would be essential for, for example, the analysis of diurnal cycles, weather system dynamics, precipitation, and wind extremes.

One possibility to overcome the output avalanche is to merely store the simulation setup, initial conditions and restart files, and rerun the simulation on demand when needed to perform a specific analysis. A more sophisticated scheme would restart the simulation in parallel from a series of restart files. This, in principle, enables us to arbitrarily trade off storage for computation. Depending upon the available hardware resources, an optimized design of a resimulation (in terms of cost and time) might employ an alternate software configuration (e.g., using a different number of compute nodes), or even an alternate hardware platform.

Table 1. Data volumes of the CMIP6-DECK simulations. The third column shows the estimate by the Centre for Environmental Data Analysis for a simulation employing a grid spacing of 0.5°, 40 model levels in the atmosphere, and 60 levels in the ocean (Juckes et al. 2015; CEDA 2018). The fourth column shows the same output list projected to an R2B11 mesh of the ICON model, employing 1.25 km grid spacing, 180 levels in the atmosphere, and 200 levels in the ocean. The fifth column shows total data volume available for analysis for the 1.25 km simulation (footprint of 2.9 TB in single-precision floating-point format), accounting for all 3D prognostic variables (8 in the atmosphere and 5 in the ocean) at each model time step (10 s). Adding all the available 2D fields (e.g., sea ice, soil, vegetation) would amount to about an additional 3D variable. The CMIP6 DECK simulations (first two columns, from top to bottom) include a preindustrial control simulation, an atmospheric model intercomparison simulation, a simulation forced by a 1% yr–1 CO2 increase, and a simulation with abrupt quadrupling of CO2.

Simulation	Length (yr)	CMIP6 at 0.5° (TB)	CMIP6 at 1.25 km (PB)	Data at 1.25 km (ZB)
piControl	500	5	16.2	4.5
amip	36	1.7	1.3	0.4
1pctCO_2	150	1.6	5.3	1.4
abrupt-4 × CO_2	150	8	22.2	1.4

To ensure exactly the same results when resimulating the chaotic dynamical system, we must ensure that the simulation code itself is bitwise reproducible, that is, produces exactly the same output, bit by bit, when rerun with the same input. Bitwise reproducibility is potentially also required across different hardware architectures, depending on the setup of the resimulation. Whether bitwise reproducibility is required will depend upon the targeted analysis. Consider for instance an analysis focusing on a few major hurricanes in an extended simulation, then the lack of bitwise reproducibility presents a serious hurdle (as hurricanes might disappear or change with the chaotic dynamics). Alternatively, for the statistical evaluation of short-term precipitation events, bitwise reproducibility might not be needed, provided the simulation considered is sufficiently long.

It is often assumed that bitwise reproducibility comes at a significant performance cost. However, recently, various approaches to ensure bitwise reproducibility with small performance overheads have been demonstrated by Demmel and Nguyen (2013). Arteaga et al. (2014) demonstrated how to integrate such approaches into full scientific applications. These developments enable efficient resimulation and will be discussed later in this paper.

Compliance with data policies, FAIR principles. In recent years the issue of data sharing and data accessibility has received growing attention (National Academies of Sciences, Engineering, and Medicine 2018a,b; Schuster et al. 2019). To make maximum use and reuse of scientific data, it should be Findable, Accessible, Interoperable und Reusable (FAIR; Wilkinson et al. 2016). Publishers have taken action and their data policies address data accessibility. For example, the American Meteorological Society (AMS) issued a policy statement that "the AMS encourages the Earth System Science community to provide full, open, and timely access to environmental data and derived data products, as well as all associated information necessary to fully understand and properly use the data (metadata)" (www.ametsoc.org/ams/index.cfm/about-ams/ams-statements/statements-of-the-ams-in-force/full-and-open-access-to-data). Moreover, many journals require that the storage archive for the underlying data are documented in the article upon publication. Organizations such as the Coalition for Publishing Data in the Earth and Space Sciences (COPDESS) have been founded to facilitate FAIR data.

It is not clear yet how the FAIR principles can be extended to include the workflow proposed in this study, namely, resimulating data once it is required for further analysis. The aspect of timely access to the data is especially challenging, and often the required source code is subject to some license agreement. It is clear that these emerging strategies will also require updates of data policies. In particular, guaranteeing bitwise reproducibility over extended time periods (say, 5–10 years) should become a central element of the FAIR principles, as for some applications, recomputation will become more cost effective than storing the output.

Strategies toward kilometer-scale resolution

The target model. In this study we use the COSMO model (Steppeler et al. 2003; Baldauf et al. 2011). COSMO is a community model used by many national weather services worldwide as well as research groups at over 100 universities. The COSMO model is a limited-area model used for both numerical weather prediction and climate modeling by the CLM community (www.clm-community.eu). The findings and results presented in this paper have all been carried out using a version of the COSMO model refactored for heterogeneous computing architectures (Fuhrer et al. 2014). This version also supports execution in single precision (Düben and Palmer 2014). The overall effort to refactor COSMO is approximately 20 man-years. We expect that the learnings presented in this article from COSMO carry over to many other models.

Domain-specific languages. Dynamical cores of atmospheric or ocean models such as COSMO typically do not contain singular performance hot spots that can simply be replaced with an efficient implementation.[2] Rather, the program code often contains a series of iterations over all grid points (e.g., applying a fourth-order diffusion filter as in the sidebar "Domain-specific languages explained"). As mentioned in the "Exploiting next generation hardware architectures" section, achieving good

[2] Spectral transforms are a notable exception.

performance on current high-performance computing systems requires decorating the code with hardware dependent compiler directives to specify parallelism and the schedule of how the loop iterations will be executed (see "Use of OpenACC" section). Further, optimizations often entail changes in the looping structure (e.g., blocking), the data structures, and typically also the fusion of consecutive iterations over all grid points. The consequences of the above changes are loss of performance portability, significant decrease in maintainability of the code and often suboptimal performance.

Choosing an alternative route, the dynamical core of the COSMO model has been rewritten using the GridTools DSL (Gysi et al. 2015; Fuhrer et al. 2014). GridTools is a domain-specific language that eases the burden of the application developer by separating the architecture dependent implementation strategy from the user code. GridTools is currently implemented in C++ by using template metaprogramming; thus an application based on GridTools needs to be implemented in C++. GridTools has become publicly available under a permissive open-source license in March 2019 (www.github.com/GridTools/).

Use of OpenACC. While code rewriting using DSLs offers many advantages in terms of performance and maintainability, it may not be applicable to the entire code base. In addition, some parts like the physical parameterizations have been developed by a large and active community, which may not be ready for changing their programming paradigm. However, in order to avoid costly CPU to GPU data transfers, most parts of the code need to run on the GPU. To achieve this, an OpenACC compiler directive porting approach was used for all components of the COSMO model that had not been rewritten using DSLs (Fuhrer et al. 2014; Lapillonne and Fuhrer 2014).

The OpenACC compiler directives can be added to existing code to tell the compiler which part should run on the GPU, offering the possibility to incrementally adapt the code for GPUs. While the directive approach does not offer a hardware optimization comparable to DSLs, it allows us to achieve reasonable performance. Some parts of the code have been further optimized and restructured to achieve a better performance on GPUs. In some cases these changes are not performance portable, that is, they have a negative impact on the CPU execution time, such that two code paths—one for CPU and one for GPU—need to be maintained. Although this approach has proven successful to port large legacy codes, the OpenACC compiler directives have limitations and the long-term support of OpenACC compilers is not guaranteed at this stage. Thus our approach requires reevaluation in the future as new programming paradigms emerge.

Overall the COSMO model with the rewritten GridTools dynamical core and with the other components ported with OpenACC directives runs about 3–4 times faster on GPUs than the original code on CPUs when comparing hardware of the same generation (Fuhrer et al. 2014; Leutwyler et al. 2016). Similar speedups have been reported by other studies (e.g., Govett et al. 2017).

Emerging programming paradigms for climate models. The complexity of climate models is already challenging at current resolutions. However, with further resolution increases, and with the need to account for newly emerging hardware architectures, these challenges become even more significant. In practice there is a high compartmentalization of the model development, with dynamical cores and physics packages mostly developed in isolation (Donahue and Caldwell 2018). The immediate downside of this approach is the proliferation of model components with incompatible structures. Transferring such components to other models often requires a large amount of work (Randall 1996).

The recognition of the need for standardizing Earth system models dates back to the 1980s (Pielke and Arritt 1984). Kalnay et al. (1989) suggested a list of basic programming rules to design *plug-compatible* physics packages, enabling a high degree of scientific code exchange. This led to the idea of a common software infrastructure that couples different components while enhancing interoperability, usability, software reuse, and performance portability (Dickinson et al. 2002). Notable examples of such coupling frameworks include the Earth System Modeling Framework (ESMF) combined with the National Unified Operational Prediction Capability (NUOPC) layer (Hill et al. 2004; Theurich et al. 2016), the Flexible Modeling System (FMS) (Balaji 2012), the Program for Integrated Earth System Modeling (PRISM) framework (Guilyardi et al. 2003), and the Weather Research and Forecasting (WRF) Model code infrastructure (Michalakes et al. 2005).

All these frameworks are coded in FORTRAN, which remains the preferred programming language for software development in climate models. However, the new generations of atmospheric and computer scientists are more familiar and proficient with higher-level languages, for example, Python. Python has been increasingly used by academics and scientists due to its clean syntax, great expressiveness and a powerful ecosystem of open source packages, making it ideal for fast prototyping (Millman and Aivazis 2011). Yet, its direct application in high-performance computing has historically been limited by the inherent execution slowness of the Python interpreter. Solutions to overcome the interpreter overhead exist, including DSLs endowed with lower-level and optimized back ends.

In most of the traditional frameworks, the calling sequence of parameterizations (or components like ocean, land, and sea ice) is hard-coded for efficiency reasons. *sympl* (System for Modeling Planets; Monteiro et al. 2018) attempts to circumvent this and other limitations by providing a toolset of Python objects to build hierarchies of Earth system models, in which each component represents a physical process. A model is thus conceived as a chain of computing blocks, which act on and interact through the *state*, that is, the set of variables describing the model state at any point in time. The state is encoded as a dictionary of multidimensional arrays which enables metadata-aware operations (Hoyer and Hamman 2017). To illustrate how this dictionary works, consider a scientist who intends to develop a new parameterization. In doing so, he/she requires access to specific variables of the model state. In current climate modeling frameworks, this requires specific knowledge about how the data are stored and how it can be accessed. In contrast, *sympl* provides a transparent set of tools for accessing the data in the model state dictionary. The tools take care of some of the annoying issues, such as the transformation of data between different units. In doing so, it hides the complexities of the data storage in the respective parent model, and can in principle provide a general approach across many different models.

Currently several research groups are exploring *sympl*. In our own work, we are using it to investigate the physics time stepping. Although it appears to be of similar importance as the choice of the spatial discretization (Knoll et al. 2003), in the majority of the current weather and climate codes the time stepping is merely accurate to first order, and the results and sensitivity of models depend upon the choice of the calling sequence (e.g., Donahue and Caldwell 2018; Gross et al. 2018). It is not only the lack of a common interface, but also the simplified time stepping, that hinders the exchange of parameterizations. With the help of an idealized hydrostatic model in isentropic coordinates, we are currently conducting numerical experiments to quantify the impact of the employed coupling strategy on the solution. We find that *sympl* is a suitable prototype framework for building flexible, modular and interoperable codes, and believe that such frameworks could aid the development of future climate codes.

Bit-reproducible code. A bit-reproducible climate model produces the exact same numerical results for a given precision, regardless of its execution setup—which includes the choice of domain decomposition, the type of simulation (continuous or restarted), compilers, and the architectures executing the model (here, CPU or GPU).

One source of nonreproducibility stems from the way arithmetic operators are evaluated on a computer. A floating-point arithmetic operation is equivalent to the application of the operator on the operands, followed by a rounding of the result: $r(a + b) \neq a + b$, where $r(\cdot)$ denotes the rounding function. The latter function produces a representable floating-point value in the computer's memory from a real number. For simple operations (addition, subtraction, multiplication, division and square root), the IEEE-754 standard ensures bitwise reproducibility across hardware architectures (IEEE 2008; Arteaga et al. 2014). However, the associativity property of arithmetic operators is broken. This means that $(a + b) + c = a + (b + c)$ is not preserved, as $r[r(a + b) + c] \neq r[a + r(b + c)]$. Although the rearrangements are equivalent in their mathematical form, they are not equal in a floating point computation.

Achieving reproducibility across architectures is a challenge, as compilers do not produce the same executable code when targeting different hardware architectures (i.e., GPU or CPU). Mathematical expressions can be rewritten (contraction, reassociation, fast mathematics) in different manners to ensure best performance on the targeted architecture, leading to potentially different results due to the aforementioned properties of floating-point arithmetic. The key points to achieve bit reproducibility with COSMO are to (i) forbid the reassociation of mathematical expression, (ii) forbid the creation of alternative execution strategies for a given computation, (iii) forbid the usage of mathematical approximation or contraction operators, and (iv) provide portable transcendental functions (i.e., logarithm, exponential function, or the trigonometric functions) to ensure reproducibility of their evaluation.

Compilers can be more or less aggressive with the level of optimization they apply. By using execution flags, the user can have some control over the optimizations applied during compilation. We used a set of flags that limits instructions rearrangement as much as possible [see Table ES1 in the online supplemental material (https://doi.org/10.1175/BAMS-D-18-0167.2)]. This increases the probability that compilers targeting different architectures produce identical mathematical expressions. Finally, we wrote a preprocessor to automatically add parentheses to every mathematical expression of the model, ensuring a unique way to evaluate these expressions. The preprocessor also replaces all intrinsic function calls with our custom version of portable transcendental functions.

In our work with COSMO, reproducibility between the CPU (Intel Xeon E5-2690) and GPU (Nvidia Tesla K80) versions of the model has been achieved, although at the time of writing discrepancies remain in some modules relevant for long simulations and with restarted simulations. These challenges still need to be addressed. The performance penalty of making the code bit reproducible is acceptable (Fig. 4). On the CPU the bit-reproducible version is 37% slower than the original version of the program code, and on the GPU it is 13% slower. Overall this demonstrates that the overhead associated with bit reproducibility may be smaller than previously thought.

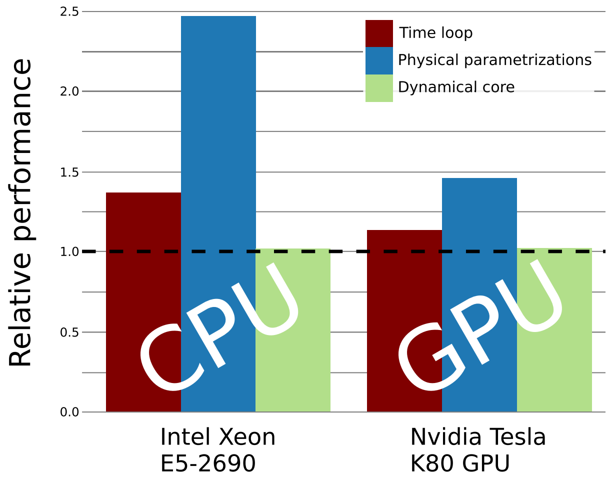


Fig. 4. Performance penalty of a bit-reproducible COSMO version (providing reproducibility across an Intel Xeon E5–2690 CPU and a Nvidia Tesla K80 GPU). The dynamics section (green) hardly shows any penalty. The physics (blue) suffers from a large penalty (almost 2.5 times slower) due to the constraints imposed upon the compiler to avoid instruction rearrangement when the CPU is targeted. Nevertheless, the entire time loop (red) containing both sections displays only a moderate performance penalty.

Virtualization layer. Data produced by high-resolution simulations are expected to be potentially valuable for a large number of climate and impact scientists over the course of decades. The way these data are commonly analyzed today is by storing them on disk and letting the analysis applications access them. This solution enables the analyses to access the data with arbitrary access patterns (e.g., forward or backward in time) and guarantees that the exact same data can be reanalyzed to produce the same results. However, high-resolution simulations produce petabytes of data today, and may produce exabytes in the near future (Table 1): storing this amount of data for long periods of time is not cost effective and, in some cases, not possible at all. This issue can be addressed by employing online (or in situ) analyses. Online analysis provides a solution to this problem by not storing data and by coupling analyses and simulations. However, this approach leads to a loss of flexibility (e.g., the data access pattern of the analysis must follow the simulation), and most of the times it requires one to instrument the model code with analysis software (Zhang et al. 2012) that runs as the data are produced by the model. While this alleviates the storage issues (for our European-scale simulations, storage for the monthly restart files amounts to only 38 GB in comparison to the standard output per month of 0.4 TB), this approach makes the analysis less flexible.

We developed and tested SimFS (Di Girolamo et al. 2019), a virtualization layer that is in between the analysis applications and the simulation data (https://github.com/spcl/SimFS). SimFS exposes a virtualized view of the simulation data: the data are seen by the analysis as if they were on disk, while they may not be stored there. SimFS is responsible to recreate data that is being accessed by an analysis but not present on disk (i.e., on demand).

Analysis applications can be transparently interfaced to the virtualization layer: calls to standard I/O libraries (e.g., NetCDF, HDF5) are intercepted by a SimFS client library that can be loaded at runtime into the analysis application, without requiring any changes of the analysis code. To guide optimizations and gain control and information about the virtualized environment, the analysis can also interface SimFS through a set of specialized application programming interfaces (APIs).

Virtualizing the simulation data means enabling the analysis of multipetabytes datasets on terabytes storage systems. As a consequence, SimFS may need to evict data when the given storage share becomes full. To select which files to evict, SimFS tracks the analyses access patterns and employs caching and prefetching strategies to (i) identify the most relevant (i.e., most accessed) parts of simulation data and keep them on disk, avoiding their resimulation, and (ii) minimize the time to recover missing data.

Figure 5 sketches the SimFS workflow. The scientists set up the initial simulation that runs to completion (top left) and produces the restart files (black files in top right) that are stored. Later, analysis tools access the simulation data through the virtualization layer (bottom left). SimFS intercepts these accesses and manages/restarts simulations to recreate the requested output data if not already present (bottom right). The system can be configured to cache the simulation data on a hierarchy of data storage mediums (e.g., fast flash memories, mechanical disks, magnetic tapes).

SimFS requires that simulations can be restarted and deliver bitwise-identical output (see "Bit-reproducible code" section). If bitwise reproducibility is not provided, analyses should be able to operate on data that can differ from the one produced by the initial simulation.

Results and applications

Near-global benchmarking. As stated in the introduction, there is significant thrust in the modeling community to decrease the grid spacing of global climate simulations to the kilometer-scale in order to address some of the most pressing deficiencies in understanding and projections of climate change. Figure 1 summarizes some of the pioneering simulations that have been reported in the literature, notably the prototype simulations of Miyamoto et al. (2013) and Fuhrer et al. (2018). But how far are we from actually achieving kilometer-scale simulations on leadership class HPC facilities?

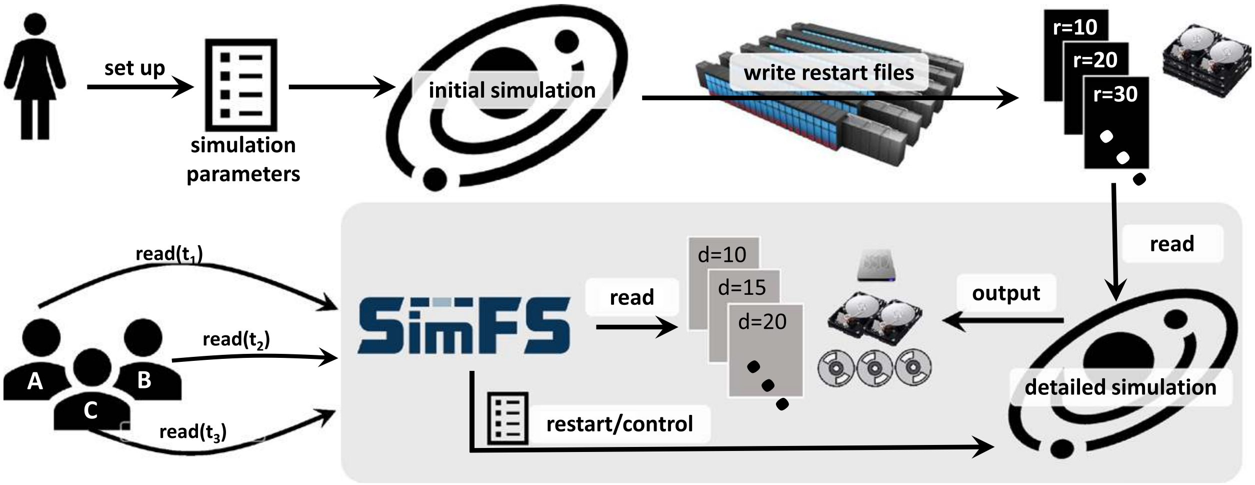


Fig. 5. Overview of the rerun (vs store) approach using SimFS. (top left) The scientist runs the initial simulation that produces the restart files and during which a first online analysis can be performed. The restart files are made available to SimFS, which can use them to restart the model. (bottom left) Later, analysis applications are transparently interfaced to SimFS via common I/O libraries (e.g., NetCDF, HDF-5) or by using the SimFS API. SimFS checks if a simulation output file requested by an application is available on the configured storage mediums (e.g., fast flash memories, mechanic disks, magnetic tapes). If the file is not available, SimFS runs the model in order to recreate the file, otherwise it lets the requesting application open it.

One of the most important metrics for assessing the usability of climate simulations is the simulation throughput measured in simulated years per wall-clock day (SYPD). Different applications of global climate models require different minimal simulation throughput in order to be feasible. For example, a global climate model achieving 1 SYPD on a given HPC system can be considered useful for simulations spanning several decades. While not sufficient for all applications, 1 SYPD can be considered a reasonable first target for global kilometer-scale climate simulations.

Since COSMO is one of the few models which has been systematically adapted to run on modern supercomputer architectures with GPU-accelerated node designs, it is an interesting benchmark to consider. Fuhrer et al. (2018) report a simulation throughput of 0.043 SYPD for idealized, near-global simulations using the COSMO model on 4,888 nodes of the Piz Daint supercomputer at CSCS with a grid spacing of 0.93 km. In a detailed analysis, Schulthess et al. (2019) conclude, that this result corresponds to a shortfall of about a factor 100 with respect to the defined goal.

Summit, the system currently leading the TOP500 ranking of supercomputers, has approximately 5 times more GPUs than Piz Daint and a more recent generation of GPUs (NVIDIA Tesla V100 16 GB) which execute COSMO 1.5 times faster than the GPUs in Piz Daint (NVIDIA Tesla P100 16 GB). We cannot expect to be able to scale COSMO to the full Summit system, but results from Fuhrer et al. (2018) indicate that further linear strong scalability by a factor of 3 is possible. Taking these factors into account, we find that running a global climate simulation with a realistic setup (cf. Table 1 of Schulthess et al. 2019) and a horizontal grid spacing of 1 km on the currently largest supercomputer available would fall short of the 1 SYPD target by approximately a factor of 20 (Schulthess et al. 2019). A recent study by Neumann et al. (2019) reports a shortfall of a factor of 30, extrapolating results from the ICON model at 5 km grid spacing and assuming perfect weak scaling.

Addressing the remaining shortfall will likely require a combination of several strategies, including algorithmic, software, and hardware improvements. Addressing the challenge of I/O for global kilometer-scale simulations will require fundamental changes in our simulation and analysis workflow such as SimFS.

However, at a resolution of 2 km, the simulation throughput of COSMO on Piz Daint for a near-global climate simulation setup already reaches 0.23 SYPD, thus the model can in principle already be used for decadelong simulations at such a resolution. Some examples for regional climate simulations shown in the next section.

Regional climate simulations. There are three areas where kilometer-scale resolution is raising hopes for significant benefits. First, there is a better representation of the underlying surface—complex topography, coast lines, and land surface properties. Second, higher resolution allows us to better represent mesoscale processes and the associated feedbacks to the larger scale, such as fronts, orographic wind systems, boundary layer processes, and soil moisture–atmosphere feedbacks. Third, and likely most importantly, kilometer-scale resolution allows switching off two of the most critical parameterizations in climate models, namely, moist convection and gravity wave drag, which constitute critical sources of uncertainties in climate change projections. Explicit simulation of convection has led to significant improvements in simulations of the diurnal cycle of precipitation, addressing aspects of frequency and intensity of heavy hourly precipitation (e.g., Kendon et al. 2012; Ban et al. 2014, 2015; Prein et al. 2015; Leutwyler et al. 2017; Berthou et al. 2020), which can potentially lead to hydrological impacts like flash floods, floods, and landslides. An example of this is shown in Fig. 6 for hourly precipitation over Europe on a summer day. The 12 km model produces widespread low-intensity precipitation (a long-standing problem of convective parameterizations), while a more realistic representation of intense summer precipitation is obtained in the 2 km model. Furthermore, kilometer-scale resolution is needed for resolving local-scale wind systems, like sea breeze and orographic circulations (e.g., Belušić et al. 2018), and for a better representation of clouds and their vertical profiles (e.g., Hentgen et al. 2019).

A comparison of cloud cover at different resolutions over the tropical Atlantic is shown in Fig. 7. In comparison with Moderate Resolution Imaging Spectroradiometer (MODIS; https://terra.nasa.gov/about/terra-instruments/modis) imagery observations, convection-parameterized simulations at 50 and 12 km show an overestimation of clouds and do not reproduce the organized cloud structures visible in observations. In contrast, the 2 km simulation with explicit convection can qualitatively reproduce the characteristic cloud structures known as mesoscale cloud flowers (e.g., Bony et al. 2017). More detailed analysis demonstrates that the use of explicit convection also significantly reduces top-of-the-atmosphere radiation biases. The simulations suggest that the organization of the subtropical clouds considered does not overly depend upon small-scale processes truncated at kilometer-scale resolution. Animations of these simulations are shown in the online supplement.

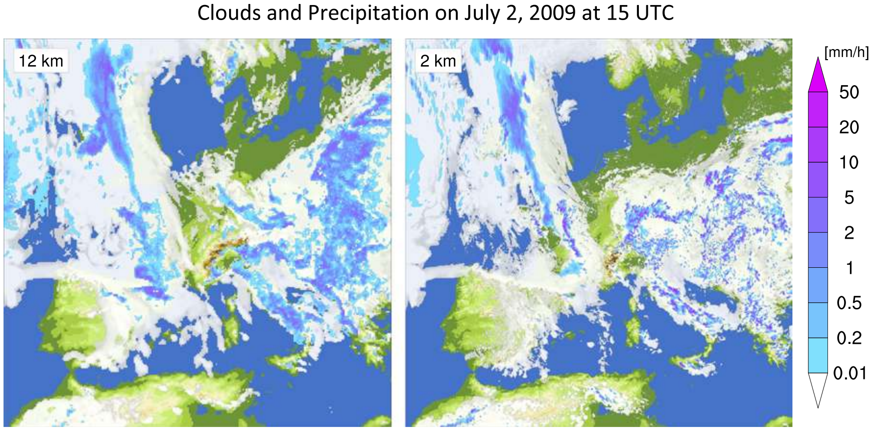


Fig. 6. Total cloud cover and precipitation over Europe obtained from convection-parameterizing (12 km horizontal grid spacing) and convection-resolving model simulations (2 km horizontal grid spacing) at 1500 UTC 2 Jul 2009. The simulation snapshots demonstrate major differences in the simulation of clouds and precipitation. Namely, the 12 km model shows widespread precipitation with low intensities and more clouds, while the 2 km model simulates summer convection over Europe more realistically with more intense precipitation cells. The results are from a decadelong continental-scale simulation (Leutwyler et al. 2017).

In addition to a better representation of the present-day climate, convection-resolving climate models provide modified climate change signals. Although changes in mean

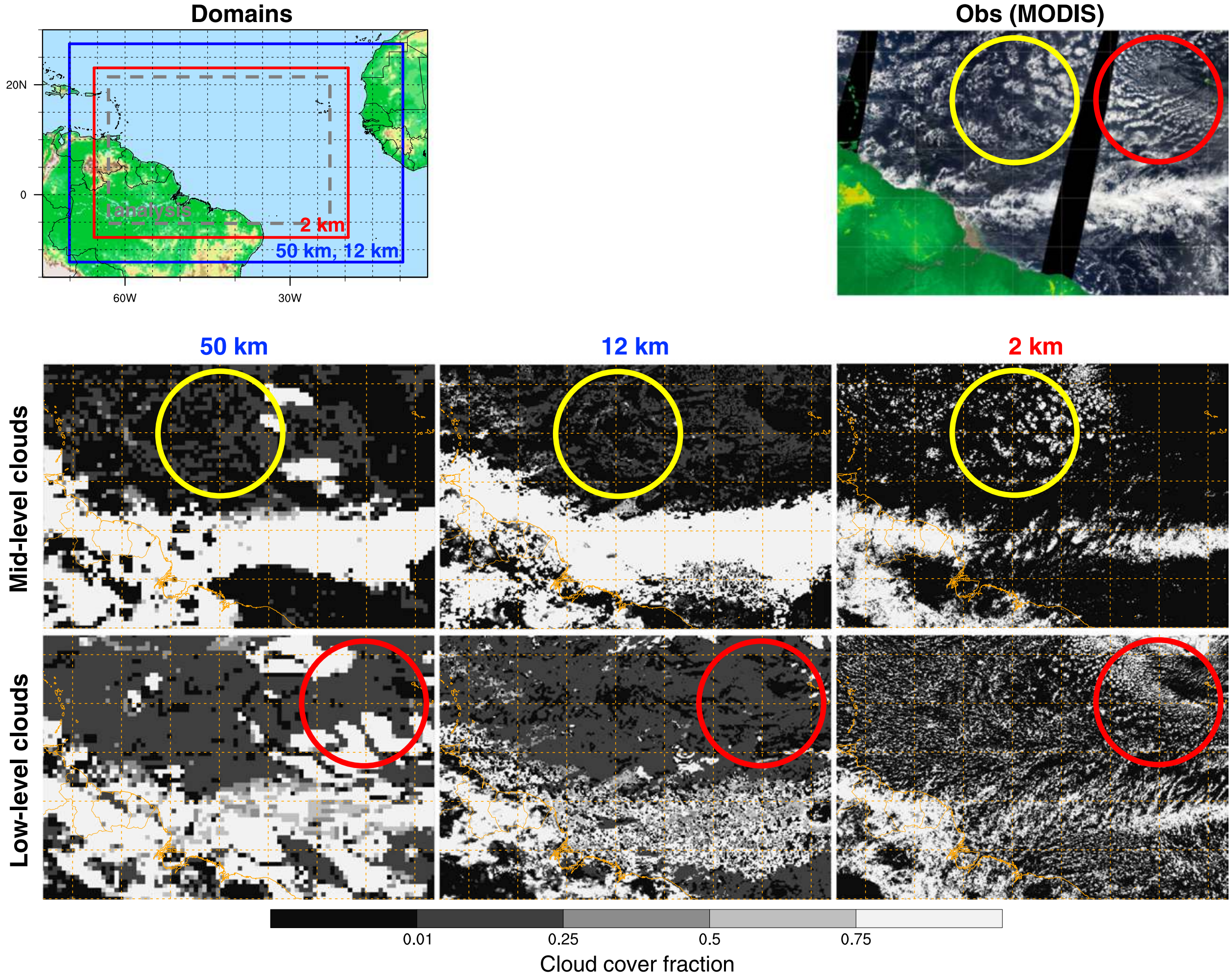


Fig. 7. (top right) Cloudiness in MODIS shortwave satellite observations, compared against (middle) mid- and (bottom) low-level cloudiness in simulations at different horizontal resolutions on 15 Dec 2013. The simulation snapshots show the cloud cover fractions from convection-parameterizing simulations at 50 and 12 km resolutions, and a convection-resolving simulation at 2 km resolution. The results are from monthlong simulations driven by the ERA-Interim reanalysis initialized on 25 Nov 2013. Red and yellow circles pinpoint regions with large differences between simulations. (top left) The geographical characteristics of the considered computational domains.

seasonal precipitation are generally robust between convection-resolving and convection-parameterizing models, significant differences occur for projections of heavy hourly precipitation events (Ban et al. 2015; Kendon et al. 2017) and for changes in the vertical structure of clouds (Hentgen et al. 2019).

Convection-resolving and convection-parameterizing models often exhibit important differences for subdaily variables, or when feedback effects are considered. Most of the analysis in current climate studies is done using two-dimensional daily and/or hourly output fields, which are currently feasible to store. Three-dimensional fields are usually not available over extended time periods, which limits detailed investigations of the flow dynamics. Convective clouds can grow, mature, and dissipate within an hour, and thus it is difficult to gain deeper understanding of convection and its characteristics in current and future climates if restricted to hourly output fields.

Refining the horizontal resolution of regional climate models is a key focus in a number of internationally coordinated projects, like the Coordinated Regional Downscaling Experiment (CORDEX; www.cordex.org) and the European Climate Prediction System (EUCP; www.eucp-project.eu). Within these two projects, several groups across Europe are conducting regional climate simulations in common domains with horizontal resolutions around 3 km, with the aim of producing a multimodel ensemble of climate simulations (Coppola et al. 2020). Similar initiatives are also underway within GEWEX (https://ral.ucar.edu/events/2018/cpcm). The availability of long-term high-resolution simulations would also enable to link to short-term case studies (e.g., Dauhut et al. 2015) and idealized simulations of convective events (e.g., Loriaux et al. 2017).

Sophisticated analysis using the virtualization layer. This section presents online analysis applications of convection-resolving COSMO simulations with SimFS, and briefly discusses the limitations of offline and online analyses. An offline analysis would follow the traditional approach of saving all necessary fields on disk (e.g., with a temporal resolution of 1 h) and then running the diagnostic. In contrast, an online analysis would be run as part of the main model forward integration, allowing for an almost arbitrary temporal resolution of input fields—for example, online forward trajectory calculations (Miltenberger et al. 2016). In the following, two applications are considered, with differing requirements in terms of the temporal resolution and data volume of the input fields. The results are based on a week-long COSMO simulation, starting at 0000 UTC 10 April 2000. The first application tracks precipitation cells, and the second uses backward trajectories to investigate the foehn flow in an Alpine valley.

Precipitation cells are identified every 6 min using a threshold of 2 mm h^{-1} and tracked in time with a criterion considering feature overlap and size (Rüdisühli 2018). Access to the data are provided through SimFS, that is, without storing it on disk. To speed up the analysis, the grid resolution is reduced by averaging the surface precipitation field over 3 × 3 grid points, and a minimum feature size of two coarse grid points is required. To facilitate the tracking, the overlap of features in consecutive steps is increased temporarily by 3 coarse grid points in all directions. Results are shown in Fig. 8. At 1000 UTC 12 April 2000, precipitation occurs over large areas, extending along a frontal band extending from the British Isles over Germany to the Alps, and in the form of small shower cells in the Bay of Biscay and adjacent regions (Fig. 8a). The cell tracking reveals the strongly differing lifetimes of the various cells, ranging from minutes to days (Fig. 8b). While short-lived cells produce less precipitation than longer-lived cells, they are more frequent. An animation of this figure over an extended period is provided in the online supplement. SimFS allows us to use this approach for tracking precipitation cells at temporal resolutions of a few minutes in long climate simulations without storing the fields on disk.

The second application is based on air-parcel trajectories, which implies considerable computational challenges for SimFS: the trajectories are run 12 h backward in time and hence do not follow the forward integration of the COSMO simulation (backward trajectories prohibit a standard online implementation). The trajectories are released in a narrow (2–5 km wide) Alpine valley and therefore the temporal resolution of the wind fields must be high in order to capture the spatial and temporal variability of the winds as the air parcels descend into the valley. The backward trajectories are initialized in the upper Rhine valley—a classical Alpine foehn valley (e.g., Würsch and Sprenger 2015) (see Fig. ES1). Trajectory computations use wind fields at different update intervals from 1 to 60 min. Results show that depending upon the case, there is considerable sensitivity to the temporal resolution, pinpointing different origins of the air parcels. This illustrates the importance of using input fields with very high temporal resolution (1–5 min). This example further emphasizes the value of SimFS: it allows computing backward trajectories (which would be difficult with a standard online implementation) with winds at very high temporal resolution (which would not be possible with an offline implementation).

The two applications differ substantially in terms of their computational requirements. For the foehn flow the bottleneck is I/O, due to the demand of 3D wind fields at high

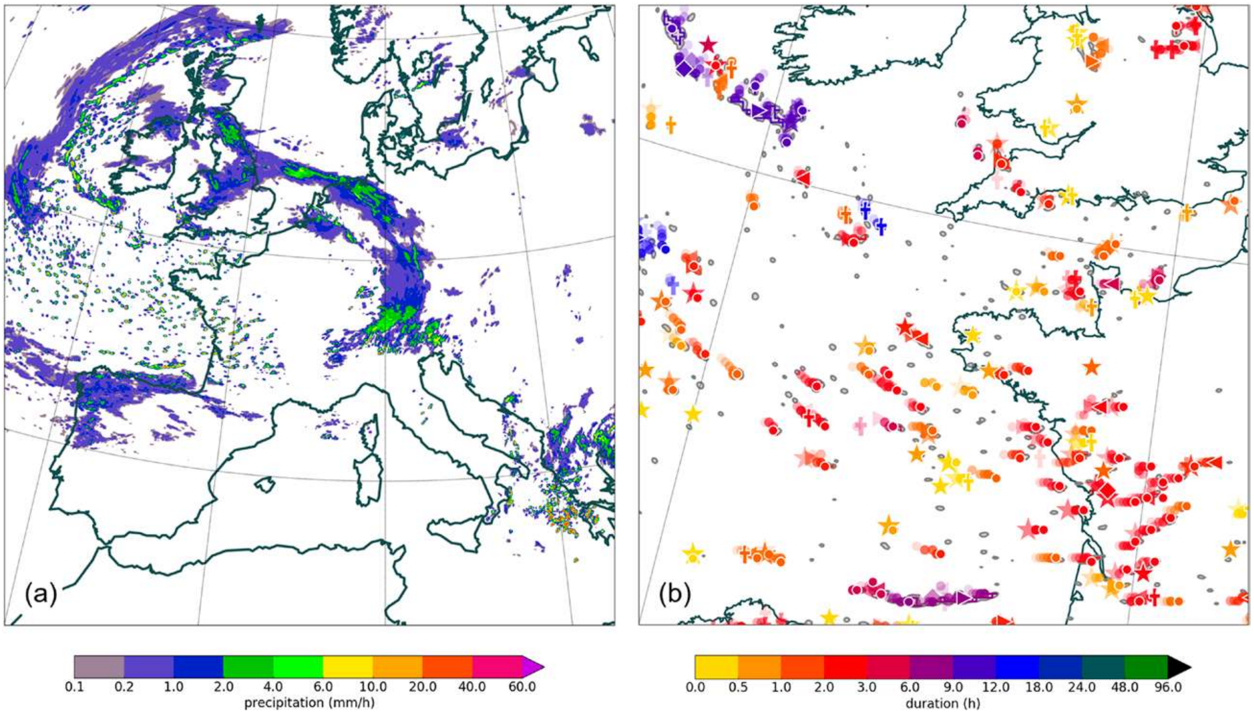


Fig. 8. (a) Six-minute surface precipitation (mm h^{-1}) at 1000 UTC 12 Apr 2000 in the entire domain, and (b) tracked precipitation cells at the same time over the Bay of Biscay. The symbols depict the tracked events (star: genesis; cross: lysis; circle: continuation; right-pointing triangle: merging; left-pointing triangle: splitting; diamond: merging–splitting). The symbols and feature outlines are colored with the total cell lifetime (i.e., track duration). To indicate recent cell movement, the previous six positions (every 6 minutes) of the track center are also shown in fading colors.

temporal resolution. The calculation of the trajectories is then rather cheap. In contrast, the precipitation cell tracking relies on 2D fields only. Therefore, it is not restricted by I/O but rather by the cell tracking algorithm itself. Both requirements are relevant when using SimFS to analyze long climate simulations. SimFS provides a lot of flexibility. For instance, an analysis may be designed conditional upon the occurrence of a particular weather event, such as the occurrence of a hurricane or in our example the occurrence of foehn flow at a particular location.

Conclusions and outlook

In this article we have explored the use of a high-resolution modeling system for extended simulations over a large computational domain, and discussed potential challenges associated with the further development of climate models. A series of fundamental technology transitions are having a profound impact on the development of models, simulation software, and modeling workflows:

1) Moving data has become more expensive than arithmetic operations. While in the past compute performance has commonly been expressed in floating point operations per second, the energy and runtime footprints of high-resolution atmospheric models are dominated by accessing system memory.
2) Energy costs of large compute centers have increased by a factor of 10–20 relative to hardware costs over the last two decades (Schulthess et al. 2019) and are becoming a dominant factor in design and implementation strategies of major supercomputing centers.
3) While early supercomputers used chips that were specifically designed for science applications, today's supercomputers are commonly based on commodity hardware that is produced in large quantities for a wide range of markets.

4) The common climate modeling workflow—that is, run the model on a supercomputer, store the results on a mass-storage system, and run analysis software on the stored results—increasingly approaches a bottleneck. The bandwidth of mass-storage systems does not keep up with the speed at which high-resolution models produce data, and the cost of storage increases faster than that of compute power.

The high cost of data movement favors hardware architectures with deep memory hierarchies having multiple layers of cache that have to be managed explicitly. Further, power constraints lead to heterogeneous node designs where accelerators such as graphics processing units deliver the bulk of the compute capacity. Current atmospheric models are unable to fully exploit such hardware. One hindrance is currently used programming languages, which impose the burden of leveraging the hardware architecture on the model developer.

In this article we have used the limited-area model COSMO and have explored a range of options to address these challenges. In particular, we have accomplished the following:

- We further developed and used a model version that uses the domain-specific language (DSL) GridTools. These languages enable a high-level abstraction to stencil operations and allow for a separation of concerns, that is, the model source code is less contaminated by hardware-specific implementation details and optimizations.
- We developed and tested a novel modeling workflow that is based on recomputation and online analyses (rather than storing the results). This exploits a virtualization environment (SimFS), which transparently provides data access in a similar fashion as used today for the analysis of climate data on mass-storage systems.
- We explored a bit-reproducible version of the model code, to enable bitwise reproducible simulations across two different hardware architectures and different compilers.
- We tested new programming paradigms such as the *sympl* framework to ease the work with complex codes and parameterizations in a Python environment.

Some of the new developments (the GPU-enabled COSMO model) have been used operationally at MeteoSwiss for several years, others (i.e., SimFS) have been developed and tested in extended regional climate model integrations, and still others will require further development before becoming applicable in full climate simulations (e.g., the use of *sympl* and bit-reproducible code versions). Results demonstrate the functionality of the approach, and also provide a look into future capabilities of climate models at high spatial resolution.

We discussed our experience with COSMO as background material for future model developments, but we are aware that additional challenges will emerge if applied to other numerical approaches and to global model applications. It is worth mentioning that the GridTools DSL is currently being extended for applications with some global meshes. However, we have not yet started to work on addressing the complexities of efficiently coupling atmosphere and ocean models in full-blown Earth system models.

Acknowledgments. Some of this work was supported by the Swiss National Science Foundation under Sinergia Grant CRSII2 154486/1 crCLIM, and several projects under the Swiss Platform for Advanced Scientific Computing (PASC). In addition we acknowledge PRACE for awarding us access to Piz Daint at CSCS, Switzerland. To estimate the current and future CMIP data volume in Table 1, version 01.00.28 of the Data Request Python API written by Martin Juckes from the Centre for Environmental Data Analysis (CEDA) has been used. Additional input and suggestions on the topic have been provided by a number of individuals, among these Peter Düben, Carlos Osuna, Bjorn Stevens, Thomas Stocker, Pier Luigi Vidale, and two anonymous reviewers.

References

Adams, S. V., and Coauthors, 2019: LFRic: Meeting the challenges of scalability and performance portability in weather and climate models. *J. Parallel Distrib. Comput.*, **132**, 383–396, https://doi.org/10.1016/j.jpdc.2019.02.007.

Arteaga, A., O. Fuhrer, and T. Hoefler, 2014: Designing bit-reproducible portable high-performance applications. *28th IEEE Int. Parallel and Distributed Processing Symp.*, Phoenix, AZ, IEEE, https://doi.org/10.1109/IPDPS.2014.127.

Balaji, V., 2012: The flexible modeling system. *Earth System Modelling*, Vol. 3, Springer, 33–41, https://doi.org/10.1007/978-3-642-23360-9_5.

——, and Coauthors, 2018: Requirements for a global data infrastructure in support of CMIP6. *Geosci. Model Dev.*, **11**, 3659–3680, https://doi.org/10.5194/gmd-11-3659-2018.

Baldauf, M., A. Seifert, J. Förstner, D. Majewski, M. Raschendorfer, and T. Reinhardt, 2011: Operational convective-scale numerical weather prediction with the COSMO model: Description and sensitivities. *Mon. Wea. Rev.*, **139**, 3887–3905, https://doi.org/10.1175/MWR-D-10-05013.1.

Ban, N., J. Schmidli, and C. Schär, 2014: Evaluation of the convection-resolving regional climate modeling approach in decade-long simulations. *J. Geophys. Res. Atmos.*, **119**, 7889–7907, https://doi.org/10.1002/2014JD021478.

Ban, N., J. Schmidli, and C. Schär, 2015: Heavy precipitation in a changing climate: Does short-term summer precipitation increase faster? *Geophys. Res. Lett.*, **42**, 1165–1172, https://doi.org/10.1002/2014GL062588.

Belušić, A., M. T. Prtenjak, I. Güttler, N. Ban, D. Leutwyler, and C. Schär, 2018: Near-surface wind variability over the broader Adriatic region: Insights from an ensemble of regional climate models. *Climate Dyn.*, **50**, 4455–4480, https://doi.org/10.1007/s00382-017-3885-5.

Bertagna, L., M. Deakin, O. Guba, D. Sunderland, A. M. Bradley, I. K. Tezaur, M. A. Taylor, and A. G. Salinger, 2019: HOMMEXX 1.0: A performance portable atmospheric dynamical core for the Energy Exascale Earth System Model. *Geosci. Model Dev.*, **12**, 1423–1441, https://doi.org/10.5194/gmd-12-1423-2019.

Berthou, S., E. J. Kendon, S. C. Chan, N. Ban, D. Leutwyler, C. Schär, and G. Fosser, 2020: Pan-European climate at convection-permitting scale: A model intercomparison study. *Climate Dyn.*, https://doi.org/10.1007/s00382-018-4114-6, in press.

Bony, S., and J.-L. Dufresne, 2005: Marine boundary layer clouds at the heart of tropical cloud feedback uncertainties in climate models. *Geophys. Res. Lett.*, **32**, L20806, https://doi.org/10.1029/2005GL023851.

——, and Coauthors, 2017: EUREC4a: A field campaign to elucidate the couplings between clouds, convection and circulation. *Surv. Geophys.*, **38**, 1529–1568, https://doi.org/10.1007/s10712-017-9428-0.

Bretherton, C. S., and M. F. Khairoutdinov, 2015: Convective self-aggregation feedbacks in near-global cloud-resolving simulations of an aquaplanet. *J. Adv. Model. Earth Syst.*, **7**, 1765–1787, https://doi.org/10.1002/2015MS000499.

CEDA, 2018: CMIP6 data request. Centre for Environmental Data Analysis, accessed 19 November 2018, http://clipc-services.ceda.ac.uk/dreq/tab01_1_1.html.

Clement, V., S. Ferrachat, O. Fuhrer, X. Lapillonne, C. E. Osuna, R. Pincus, J. Rood, and W. Sawyer, 2018: The CLAW DSL: Abstractions for performance portable weather and climate models. *Proc. Platform for Advanced Scientific Computing Conf.*, Basel, Switzerland, ACM, 2, https://doi.org/10.1145/3218176.3218226.

Coppola, E., and Coauthors, 2020: A first-of-its-kind multi-model convection permitting ensemble for investigating convective phenomena over Europe and the Mediterranean. *Climate Dyn.*, https://doi.org/10.1007/s00382-018-4521-8, in press.

Dauhut, T., J. P. Chaboureau, J. Escobar, and P. Mascart, 2015: Large-eddy simulations of Hector the convector making the stratosphere wetter. *Atmos. Sci. Lett.*, **16**, 135–140, https://doi.org/10.1002/asl2.534.

Davies, T., A. Staniforth, N. Wood, and J. Thuburn, 2003: Validity of anelastic and other equation sets as inferred from normal-mode analysis. *Quart. J. Roy. Meteor. Soc.*, **129**, 2761–2775, https://doi.org/10.1256/qj.02.1951.

Demmel, J., and H. D. Nguyen, 2013: Fast reproducible floating-point summation. *2013 IEEE 21st Symp. on Computer Arithmetic*, Austin, TX, IEEE, 163–172, https://doi.org/10.1109/ARITH.2013.9.

Dickinson, R. E., and Coauthors, 2002: How can we advance our weather and climate models as a community? *Bull. Amer. Meteor. Soc.*, **83**, 431–436, https://doi.org/10.1175/1520-0477(2002)083<0431:HCWAOW>2.3.CO;2.

Di Girolamo, S., P. Schmid, T. Schulthess, and T. Hoefler, 2019: SimFS: A simulation data virtualizing file system interface. arXiv, https://arxiv.org/abs/1902.03154.

Donahue, A. S., and P. M. Caldwell, 2018: Impact of physics parameterization ordering in a global atmosphere model. *J. Adv. Model. Earth Syst.*, **10**, 481–499, https://doi.org/10.1002/2017MS001067.

Düben, P. D., and T. Palmer, 2014: Benchmark tests for numerical weather forecasts on inexact hardware. *Mon. Wea. Rev.*, **142**, 3809–3829, https://doi.org/10.1175/MWR-D-14-00110.1.

Eyring, V., S. Bony, G. A. Meehl, C. A. Senior, B. Stevens, R. J. Stouffer, and K. E. Taylor, 2016: Overview of the coupled model intercomparison project phase 6 (cmip6) experimental design and organization. *Geosc. Model Dev.*, **9**, 1937–1958, https://doi.org/10.5194/gmd-9-1937-2016.

Fuhrer, O., C. Osuna, X. Lapillonne, T. Gysi, B. Cumming, M. Bianco, A. Arteaga, and T. C. Schulthess, 2014: Towards a performance portable, architecture agnostic implementation strategy for weather and climate models. *Supercomput. Front. Innovations*, **1**, 45–62, https://doi.org/10.14529/jsfi140103.

——, and Coauthors, 2018: Near-global climate simulation at 1 km resolution: establishing a performance baseline on 4888 GPUs with COSMO 5.0. *Geosci. Model Dev.*, **11**, 1665–1681, https://doi.org/10.5194/gmd-11-1665-2018.

Govett, M., and Coauthors, 2017: Parallelization and performance of the NIM weather model on CPU, GPU, and MIC processors. *Bull. Amer. Meteor. Soc.*, **98**, 2201–2213, https://doi.org/10.1175/BAMS-D-15-00278.1.

Gregory, J. M., R. Stouffer, S. Raper, P. Stott, and N. Rayner, 2002: An observationally based estimate of the climate sensitivity. *J. Climate*, **15**, 3117–3121, https://doi.org/10.1175/1520-0442(2002)015<3117:AOBEOT>2.0.CO;2.

Gross, M., and Coauthors, 2018: Physics–dynamics coupling in weather, climate, and earth system models: Challenges and recent progress. *Mon. Wea. Rev.*, **146**, 3505–3544, https://doi.org/10.1175/MWR-D-17-0345.1.

Guilyardi, E., R. Budich, G. Komen, and G. Brasseur, 2003: PRISM system specification handbook, version 1. PRISM Rep., 230 pp., www.prism.enes.org/dev/Publications/Reports/no1.pdf.

Gysi, T., C. Osuna, O. Fuhrer, M. Bianco, and T. C. Schulthess, 2015: STELLA: A domain-specific tool for structured grid methods in weather and climate models. *Proc. Int. Conf. for High Performance Computing, Networking, Storage and Analysis*, Austin, TX, IEEE, https://doi.org/10.1145/2807591.2807627.

Haarsma, R. J., and Coauthors, 2016: High Resolution Model Intercomparison Project (HighResMIP v1. 0) for CMIP6. *Geosci. Model Dev.*, **9**, 4185–4208, https://doi.org/10.5194/gmd-9-4185-2016.

Harvey, B., J. Methven, C. Eagle, and H. Lean, 2017: Does the representation of flow structure and turbulence at a cold front converge on multiscale observations with model resolution? *Mon. Wea. Rev.*, **145**, 4345–4363, https://doi.org/10.1175/MWR-D-16-0479.1.

Hentgen, L., N. Ban, N. Kröner, D. Leutwyler, and C. Schär, 2019: Clouds in convection resolving climate simulations over Europe. *J. Geophys. Res.*, **124**, 3849–3870, https://doi.org/10.1029/2018JD030150.

Hewitt, H. T., and Coauthors, 2017: Will high-resolution global ocean models benefit coupled predictions on short-range to climate timescales? *Ocean Modell.*, **120**, 120–136, https://doi.org/10.1016/j.ocemod.2017.11.002.

Hill, C., C. DeLuca, V. Balaji, M. Suarez, and A. Silva, 2004: The architecture of the earth system modeling framework. *Comput. Sci. Eng.*, **6**, 18–28, https://doi.org/10.1109/MCISE.2004.1255817.

Hoyer, S., and J. Hamman, 2017: xarray: N-D labeled arrays and datasets in Python. *J. Open Res. Software*, **5** (1), 10, https://doi.org/10.5334/jors.148.

IEEE, 2008: 754-2008 - IEEE Standard for Floating-Point Arithmetic. IEEE, 1–70, https://doi.org/10.1109/IEEESTD.2008.4610935.

IPCC, 1995: *Climate Change 1995: The Science of Climate Change*. J. T. Houghton et al., Eds., Cambridge University Press, 586 pp.

——, 2001: *Climate Change 2001: The Scientific Basis*. J. T. Houghton et al., Eds. Cambridge University Press, 881 pp.

——, 2013: *Climate Change 2013: The Physical Science Basis*. Cambridge University Press, 1535 pp., https://doi.org/10.1017/CBO9781107415324.

Ito, J., S. Hayashi, A. Hashimoto, H. Ohtake, F. Uno, H. Yoshimura, T. Kato, and Y. Yamada, 2017: Stalled improvement in a numerical weather prediction model as horizontal resolution increases to the sub-kilometer scale. *SOLA*, **13**, 151–156, https://doi.org/10.2151/sola.2017-028.

Juckes, M., V. Eyring, K. Taylor, V. Balaji, and R. Stouffer, 2015: The CMIP6 data request: The next generation climate archive. *Geophysical Research Abstracts*, Vol. 17, Abstract EGU2015-13112, https://meetingorganizer.copernicus.org/EGU2015/EGU2015-13112.pdf.

Kalnay, E., and Coauthors, 1989: Rules for interchange of physical parameterizations. *Bull. Amer. Meteor. Soc.*, **70**, 620–622, https://doi.org/10.1175/1520-0477(1989)070<0620:RFIOPP>2.0.CO;2.

Kendon, E. J., N. M. Roberts, C. A. Senior, and M. J. Roberts, 2012: Realism of rainfall in a very high-resolution regional climate model. *J. Climate*, **25**, 5791–5806, https://doi.org/10.1175/JCLI-D-11-00562.1.

——, ——, H. J. Fowler, M. J. Roberts, S. C. Chan, and C. A. Senior, 2014: Heavier summer downpours with climate change revealed by weather forecast resolution model. *Nat. Climate Change*, **4**, 570–576, https://doi.org/10.1038/nclimate2258.

——, and Coauthors, 2017: Do convection-permitting regional climate models improve projections of future precipitation change? *Bull. Amer. Meteor. Soc.*, **98**, 79–93, https://doi.org/10.1175/BAMS-D-15-0004.1.

Kestor, G., R. Gioiosa, D. J. Kerbyson, and A. Hoisie, 2013: Quantifying the energy cost of data movement in scientific applications. *2013 IEEE Int. Symp. on Workload Characterization (IISWC)*, Portland, OR, IEEE, 56–65, https://doi.org/10.1109/IISWC.2013.6704670.

Knoll, D., L. Chacon, L. Margolin, and V. Mousseau, 2003: On balanced approximations for time integration of multiple time scale systems. *J. Comput. Phys.*, **185**, 583–611, https://doi.org/10.1016/S0021-9991(03)00008-1.

Knote, C., G. Heinemann, and B. Rockel, 2010: Changes in weather extremes: Assessment of return values using high-resolution climate simulations at convection-resolving scale. *Meteor. Z.*, **19**, 11–23, https://doi.org/10.1127/0941-2948/2010/0424.

Kühnlein, C., W. Deconinck, R. Klein, S. Malardel, Z. P. Piotrowski, P. K. Smolarkiewicz, J. Szmelter and N. P. Wedi, 2019: FVM 1.0: a nonhydrostatic finite-volume dynamical core for the IFS. *Geosci. Model Dev.*, **12**, 651–676, https://doi.org/10.5194/gmd-12-651-2019.

Langhans, W., J. Schmidli, and C. Schär, 2012: Bulk convergence of cloud-resolving simulations of moist convection over complex terrain. *J. Atmos. Sci.*, **69**, 2207–2228, https://doi.org/10.1175/JAS-D-11-0252.1.

Lapillonne, X., and O. Fuhrer, 2014: Using compiler directives to port large scientific applications to GPUs: An example from atmospheric science. *Parallel Process. Lett.*, **24**, 1450003, https://doi.org/10.1142/s0129626414500030.

Lawrence, B. N., and Coauthors, 2018: Crossing the chasm: How to develop weather and climate models for next generation computers? *Geosci. Model Dev.*, **11**, 1799–1821, https://doi.org/10.5194/gmd-11-1799-2018.

Leutwyler, D., O. Fuhrer, X. Lapillonne, D. Lüthi, and C. Schär, 2016: Towards European-scale convection-resolving climate simulations with GPUs: A study with COSMO 4.19. *Geosci. Model Dev.*, **9**, 3393–3412, https://doi.org/10.5194/gmd-9-3393-2016.

——, D. Lüthi, N. Ban, O. Fuhrer, and C. Schär, 2017: Evaluation of the convection-resolving climate modeling approach on continental scales. *J. Geophys. Res. Atmos.*, **122**, 5237–5258, https://doi.org/10.1002/2016JD026013.

Liu, C., and Coauthors, 2017: Continental-scale convection-permitting modeling of the current and future climate of North America. *Climate Dyn.*, **49**, 71–95, https://doi.org/10.1007/s00382-016-3327-9.

Lock, S.-J., N. Wood, and H. Weller, 2014: Numerical analyses of Runge–Kutta implicit–explicit schemes for horizontally explicit, vertically implicit solutions of atmospheric models. *Quart. J. Roy. Meteor. Soc.*, **140**, 1654–1669, https://doi.org/10.1002/qj.2246.

Loriaux, J. M., G. Lenderink, and A. P. Siebesma, 2017: Large-scale controls on extreme precipitation. *J. Climate*, **30**, 955–968, https://doi.org/10.1175/JCLI-D-16-0381.1.

Maher, N., and Coauthors, 2019: The Max Planck Institute Grand Ensemble: Enabling the exploration of climate system variability. *J. Adv. Model. Earth Syst.*, **11**, 2050–2069, https://doi.org/10.1029/2019MS001639.

Michalakes, J., J. Dudhia, D. Gill, T. Henderson, J. Klemp, W. Skamarock, and W. Wang, 2005: The Weather Research and Forecast Model: Software architecture and performance. *Use of High Performance Computing in Meteorology*, World Scientific, 156–168, https://doi.org/10.1142/9789812701831_0012.

Millman, K. J., and M. Aivazis, 2011: Python for scientists and engineers. *Comput. Sci. Eng.*, **13**, 9–12, https://doi.org/10.1109/MCSE.2011.36.

Miltenberger, A. K., S. Reynolds, and M. Sprenger, 2016: Revisiting the latent heating contribution to foehn warming: Lagrangian analysis of two foehn events over the Swiss Alps. *Quart. J. Roy. Meteor. Soc.*, **142**, 2194–2204, https://doi.org/10.1002/qj.2816.

Miyamoto, Y., Y. Kajikawa, R. Yoshida, T. Yamaura, H. Yashiro, and H. Tomita, 2013: Deep moist atmospheric convection in a subkilometer global simulation. *Geophys. Res. Lett.*, **40**, 4922–4926, https://doi.org/10.1002/grl.50944.

Molka, D., D. Hackenberg, R. Schöne, and M. S. Müller, 2010: Characterizing the energy consumption of data transfers and arithmetic operations on x86–64 processors. *Int. Conf. on Green Computing*, Chicago, IL, IEEE, 123–133, https://doi.org/10.1109/GREENCOMP.2010.5598316.

Monteiro, J. M., J. McGibbon, and R. Caballero, 2018: Sympl (v. 0.4.0) and climt (v. 0.15.3) – Towards a flexible framework for building model hierarchies in Python. *Geosci. Model Dev.*, **11**, 3781–3794, https://doi.org/10.5194/gmd-11-3781-2018.

National Academies of Sciences, Engineering, and Medicine, 2018a: *International Coordination for Science Data Infrastructure: Proceedings of a Workshop–in Brief*. National Academies Press, 8 pp., https://doi.org/10.17226/25015.

——, 2018b: *Data Matters: Ethics, Data, and International Research Collaboration in a Changing World: Proceedings of a Workshop.* National Academies Press, 102 pp., https://doi.org/10.17226/25214.

National Research Council, 1979: *Carbon Dioxide and Climate: A Scientific Assessment*. National Academies Press, 34 pp., https://doi.org/10.17226/12181.

Neumann, P., and Coauthors, 2019: Assessing the scales in numerical weather and climate predictions: Will exascale be the rescue? *Philos. Trans. Roy. Soc.*, **377A**, 20180148, https://doi.org/10.1098/RSTA.2018.0148.

Palmer, T., 2014: Climate forecasting: build high-resolution global climate models. *Nature*, **515**, 338, https://doi.org/10.1038/515338a.

Panosetti, D., L. Schlemmer, and C. Schär, 2019: Bulk and structural convergence at convection-resolving scales in real-case simulations of summertime moist convection over land. *Quart. J. Roy. Meteor. Soc.*, **145**, 1427–1443, https://doi.org/10.1002/qj.3502.

——, ——, and ——, 2020: Convergence behavior of idealized convection-resolving simulations of summertime deep moist convection over land. *Climate Dyn.*, https://doi.org/10.1007/s00382-018-4229-9, in press.

Pielke, R. A., and R. W. Arritt, 1984: A proposal to standardize models. *Bull. Amer. Meteor. Soc.*, **65**, 1082, https://doi.org/10.1175/1520-0477-65.10.1072.

Prein, A. F., and Coauthors, 2015: A review on regional convection-permitting climate modeling: Demonstrations, prospects, and challenges. *Rev. Geophys.*, **53**, 323–361, https://doi.org/10.1002/2014RG000475.

——, R. M. Rasmussen, K. Ikeda, C. Liu, M. P. Clark, and G. J. Holland, 2017: The future intensification of hourly precipitation extremes. *Nat. Climate Change*, **7**, 48–52, https://doi.org/10.1038/nclimate3168.

Randall, D. A., 1996: A university perspective on global climate modeling. *Bull. Amer. Meteor. Soc.*, **77**, 2685–2690, https://doi.org/10.1175/1520-0477(1996)077<2685:AUPOGC>2.0.CO;2.

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

Rüdisühli, S., 2018: Attribution of rain to cyclones and fronts over Europe in a kilometer-scale regional climate simulation. Ph.D. thesis, ETH Zurich, 207 pp., https://doi.org/10.3929/ethz-b-000351234.

Sakamoto, T. T., and Coauthors, 2012: Miroc4h—A new high-resolution atmosphere-ocean coupled general circulation model. *J. Meteor. Soc. Japan*, **90**, 325–359, https://doi.org/10.2151/jmsj.2012-301.

Satoh, M., B. Stevens, F. Judt, M. Khairoutdinov, S.-J. Lin, W. M. Putman, and P. Düben, 2019: Global cloud-resolving models. *Curr. Climate Change Rep.*, **5**, 172–184, https://doi.org/10.1007/s40641-019-00131-0.

Schneider, T., J. Teixeira, C. Bretherton, F. Brient, K. Pressel, C. Schär, and A. Siebesma, 2017: Climate goals and computing the future of clouds. *Nat. Climate Change*, **7**, 3–5, https://doi.org/10.1038/nclimate3190.

——, C. M. Kaul, and K. G. Pressel, 2019: Possible climate transitions from breakup of stratocumulus decks under greenhouse warming. *Nat. Geosci.*, **12**, 163–167, https://doi.org/10.1038/s41561-019-0310-1.

Schulthess, T. C., P. Bauer, N. Wedi, O. Fuhrer, T. Hoefler, and C. Schär, 2019: Reflecting on the goal and baseline for exascale computing: A roadmap based on weather and climate simulations. *Comput. Sci. Eng.*, **21**, 30–41, https://doi.org/10.1109/MCSE.2018.2888788.

Schuster, D. C., and Coauthors, 2019: Challenges and future directions for data management in the geosciences. *Bull. Amer. Meteor. Soc.*, **100**, 909–912, https://doi.org/10.1175/BAMS-D-18-0319.1.

Sherwood, S. C., S. Bony, and J.-L. Dufresne, 2014: Spread in model climate sensitivity traced to atmospheric convective mixing. *Nature*, **505**, 37–42, https://doi.org/10.1038/nature12829.

Skamarock, W. C., 2004: Evaluating mesoscale NWP models using kinetic energy spectra. *Mon. Wea. Rev.*, **132**, 3019–3032, https://doi.org/10.1175/MWR2830.1.

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

Steppeler, J., G. Doms, U. Schättler, H. Bitzer, A. Gassmann, U. Damrath, and G. Gregoric, 2003: Meso-gamma scale forecasts using the nonhydrostatic model LM. *Meteor. Atmos. Phys.*, **82**, 75–96, https://doi.org/10.1007/s00703-001-0592-9.

Tanguay, M., A. Robert, and R. Laprise, 1990: A semi-implicit send-Lagrangian fully compressible regional forecast model. *Mon. Wea. Rev.*, **118**, 1970–1980, https://doi.org/10.1175/1520-0493(1990)118<1970:ASISLF>2.0.CO;2.

Temperton, C., M. Hortal, and A. Simmons, 2001: A two-time-level semi-Lagrangian global spectral model. *Quart. J. Roy. Meteor. Soc.*, **127**, 111–127, https://doi.org/10.1002/qj.49712757107.

Theurich, G., and Coauthors, 2016: The earth system prediction suite: Toward a coordinated us modeling capability. *Bull. Amer. Meteor. Soc.*, **97**, 1229–1247, https://doi.org/10.1175/BAMS-D-14-00164.1.

Wilkinson, M. D., and Coauthors, 2016: The fair guiding principles for scientific data management and stewardship. *Sci. Data*, **3**, 160018, https://doi.org/10.1038/sdata.2016.18.

Würsch, M., and M. Sprenger, 2015: Swiss and Austrian foehn revisited: A Lagrangian-based analysis. *Meteor. Z.*, **24**, 225–242, https://doi.org/10.1127/metz/2015/0647.

Zängl, G., D. Reinert, P. Rípodas, and M. Baldauf, 2015: The icon (icosahedral non-hydrostatic) modelling framework of DWD and MPI-M: Description of the non-hydrostatic dynamical core. *Quart. J. Roy. Meteor. Soc.*, **141**, 563–579, https://doi.org/10.1002/qj.2378.

Zhang, F., C. Docan, M. Parashar, S. Klasky, N. Podhorszki, and H. Abbasi, 2012: Enabling in-situ execution of coupled scientific workflow on multi-core platform. *26th Int. Parallel and Distributed Processing Symp.*, Shanghai, China, IEEE, 1352–1363, https://doi.org/10.1109/IPDPS.2012.122.