Computing the Full Earth System at 1km Resolution

Daniel Klocke
Max Planck Institute for Meteorology
Hamburg, Germany
daniel.klocke@mpimet.mpg.de

Claudia Frauen
Deutsche Klimarechenzentrum
(DKRZ)
Hamburg, Germany
frauen@dkrz.de

Jan Frederik Engels
Deutsche Klimarechenzentrum
(DKRZ)
Hamburg, Germany
engels@dkrz.de

Dmitry Alexeev
NVIDIA
Zurich, Switzerland
dalexeev@nvidia.com

René Redler
Max Planck Institute for Meteorology
Hamburg, Germany
rene.redler@mpimet.mpg.de

Reiner Schnur
Max Planck Institute for Meteorology
Hamburg, Germany
reiner.schnur@mpimet.mpg.de

Helmuth Haak
Max Planck Institute for Meteorology
Hamburg, Germany
helmuth.haak@mpimet.mpg.de

Luis Kornblueh
Max Planck Institute for Meteorology
Hamburg, Germany
luis.kornblueh@mpimet.mpg.de

Nils Brüggemann
Max Planck Institute for Meteorology
Hamburg, Germany
nils.brueggemann@mpimet.mpg.de

Fatemeh Chegini
University of Hamburg
Hamburg, Germany
fatemeh.chegini@uni-hamburg.de

Manoel Römmer
Forschungzentrum Jülich
Jülich, Germany
m.roemmer@fz-juelich.de

Lars Hoffmann
Forschungzentrum Jülich
Jülich, Germany
l.hoffmann@fz-juelich.de

Sabine Griessbach
Forschungzentrum Jülich
Jülich, Germany
s.griessbach@fz-juelich.de

Mathis Bode
Forschungzentrum Jülich
Jülich, Germany
m.bode@fz-juelich.de

Jonathan Coles
Swiss National Supercomputing
Centre
Lugano, Switzerland
jonathan.coles@cscs.ch

Miguel Gila
Swiss National Supercomputing
Centre
Lugano, Switzerland
miguel.gila@cscs.ch

William Sawyer
Swiss National Supercomputing
Centre
Lugano, Switzerland
william.sawyer@cscs.ch

Alexandru Calotoiu
ETH Zurich
Zurich, Switzerland
alexandru.calotoiu@inf.ethz.ch

Yakup Budanaz
ETH Zurich
Zurich, Switzerland
yakupkoray.budanaz@inf.ethz.ch

Pratyai Mazumder
ETH Zurich
Zurich, Switzerland
pratyai.mazumder@inf.ethz.ch

Marcin Copik
ETH Zurich
Zurich, Switzerland
marcin.copik@inf.ethz.ch

Benjamin Weber
ETH Zurich
Zurich, Switzerland
benweber@ethz.ch

Andreas Herten
Forschungzentrum Jülich
Jülich, Germany
a.herten@fz-juelich.de

Hendryk Bockelmann
Deutsche Klimarechenzentrum
(DKRZ)
Hamburg, Germany
bockelmann@dkrz.de

Torsten Hoefler
ETH Zurich
Zurich, Switzerland
htor@ethz.ch

Cathy Hohenegger
Max Planck Institute for Meteorology
Hamburg, Germany
cathy.hohenegger@mpimet.mpg.de

Bjorn Stevens
Max Planck Institute for Meteorology
Hamburg, Germany
bjorn.stevens@mpimet.mpg.de

Abstract

We present the first-ever global simulation of the full Earth system at 1.25 km grid spacing, achieving highest time compression with an unseen number of degrees of freedom. Our model captures the flow of energy, water, and carbon through key components of the Earth system: atmosphere, ocean, and land. To achieve this landmark simulation, we harness the power of 8192 GPUs on Alps and 20 480 GPUs on JUPITER, two of the world's largest GH200 superchip installations. We use both the Grace CPUs and Hopper GPUs by carefully balancing Earth's components in a heterogeneous setup and optimizing acceleration techniques available in ICON's codebase. We show how separation of concerns can reduce the code complexity by half while increasing performance and portability. Our achieved time compression of 145.7 simulated days per day enables long studies including full interactions in the Earth system and even outperforms earlier atmosphere-only simulations at a similar resolution.

CCS Concepts

• **Applied computing → Earth and atmospheric sciences**.

Keywords

Climate and earth system modeling, GPU and CPU hybrid computing, 1 km global grid-spacing

1 JUSTIFICATION FOR ACM GORDON BELL PRIZE FOR CLIMATE MODELLING

We achieve the first-ever simulations of the full Earth system on a 1.25 km global grid at highest time compression with an unseen number of degrees of freedom. Our innovations productively leverage the hybrid architecture of 20 480 GH200 superchips harmonizing and scaling the simulation of diverse physical, chemical, and biological processes.

2 Performance Attributes

Attribute title	Attribute value
Category achievement	Scalability, time-to-solution
Type of method used	Finite volume with explicit implicit two-time-level predictor–corrector time-stepping
Results reported on the basis of	Whole application
Precision report	Double precision
System scale	Near full system
Measurement mechanism	Timers

3 Overview of the Problem

Climate change is one of the greatest challenges facing societies and ecosystems. How human activities influence the climate depends on a complex interplay among diverse physical, biological, and chemical processes within the Earth system, whose non-linear interactions are coupled across a wide range of spatial and temporal scales. The varied processes, time, and spatial scales involved make their computation an extraordinary challenge.

At the heart of the challenge is the ambition to incorporate scales and processes necessary to physically couple the cycles of water, energy, and carbon – the alpha and omega of climate science. Coupling the energy and water cycle requires resolving the gravest modes of convective instability, which most of us experience as weather. The qualitative change in a model able to resolve the dominant modes of moist convection explains why global scale simulations at 1 km have long been the holy-grail for those interested in the physics of the climate and climate change [31, 32, 34].

Coupling the cycles of water and energy to that of carbon results in a manifestly more complex system. Flows of carbon among Earth's spheres depend on biogeochemical and dynamical processes in the ocean, and a cacophony of processes within the land biosphere [6]. This then adds process complexity (biology, chemistry) to the scale complexity of the physical climate system.

Crowning these difficulties is the requirement that they be resolved without overly compromising system throughput. The temporal compression τ, expresses this throughput in units of simulated time versus actual time. Scientifically interesting studies are enabled by a τ of a few tens, to a few hundreds. With $\tau = 30$ it becomes easy to study the diurnal cycle, and starts to become practical to study interannual variability, and hence Earth's main modes of forced variability.

With $\tau = 300$ it becomes easy to study interannual variability, and multi-decadal simulations start to become practical. This enables studies of changes observed over the instrumental record; the exploration of scenarios of future changes to inform adaptation; or the investigation of irreversible changes that can have a bearing on mitigation efforts [30].

In the past, researchers adopted one of two methods to circumvent an inability to explicitly incorporate scales and processes necessary to physically couple the cycles of water, energy, and carbon with sufficient temporal compression. One method was to restrict models to a particular component of the Earth system to capture scale interactions at short time-scales and / or over limited domains [27, 28]. The second method was to forgo scale interactions,

use statistical representations of fast and fine processes and include as many processes as possible of the full Earth system [3, 30]. The former could take advantage of high-performance computing. The latter entailed a computing paradigm that favors simple and smaller computing architectures [4]. From a scientific perspective, by adopting the first approach, researchers decoupled the carbon from the water and energy cycles to understand water and energy. In the second approach, the water cycle was decoupled from the energy and carbon cycles to study energy and carbon. *In this paper we are first to demonstrate how new heterogeneous computing architectures make it now possible to simulate the coupled water, energy, and carbon cycles all on a 1.25-km-scale global mesh and with a temporal compression that allows numerical experimentation on timescales of years to decades.*

The size of the challenge of computing the coupling of water, energy, and carbon with a temporal compression sufficient to answer meaningful scientific questions can be appreciated by considering that a global, km-scale mesh with roughly 100 layers spanning the depth of the atmosphere, respectively the ocean, entails about 5×10^{10} spatial degrees of freedom – or grid cells. When these are combined with prognostic quantities associated with these cells, e.g., the state of the atmosphere and ocean, plus tracers for H_2O, CO_2, and O_3 in the atmosphere, and 19 biogeochemical quantities in the ocean, properties describing the state of soils, the terrestrial biosphere (e.g., plant functional types and various carbon pools), and surface snow and ice, the result is roughly 1×10^{12} physical-spatial degrees of freedom for our largest configuration (Table 2). Storing those degrees of freedom alone requires 8 TiB of main memory in double precision, more than the biggest announced trillion-parameter AI models. The particular challenge of simulating the Earth system at km-scale resolution then becomes one of taming the heterogeneity of a gargantuan system without compromising either the computational weak or strong scaling.

To meet this computational challenge requires harmonizing the representation of the system across a code base that incorporates an enormous range of processes operating at very different timescales with different spatial discretizations. For example, by virtue of its much greater mass, the ocean moves slowly, and can be marched forward with time steps five or ten fold longer than used by the atmosphere. Where as the computational effort in the ocean is dominated by a few large computational kernels, over the land (and to a lesser extent in the atmosphere) it is distributed over a large number of processing elements, posing challenges for multi-threading, especially on GPUs. Over the land and the atmosphere, the governing equations allow a local representation of the processes, while over the ocean a global solver is still required, thereby imposing different computational constraints. The relatively simple dynamics of each degree of freedom, and the strong coupling among degrees of freedom, means the final computations are not arithmetically intensive and hence memory bandwidth limited. The required harmonization of the code elements thus requires a meticulous management of an enormous flow of data through a complex computational architecture.
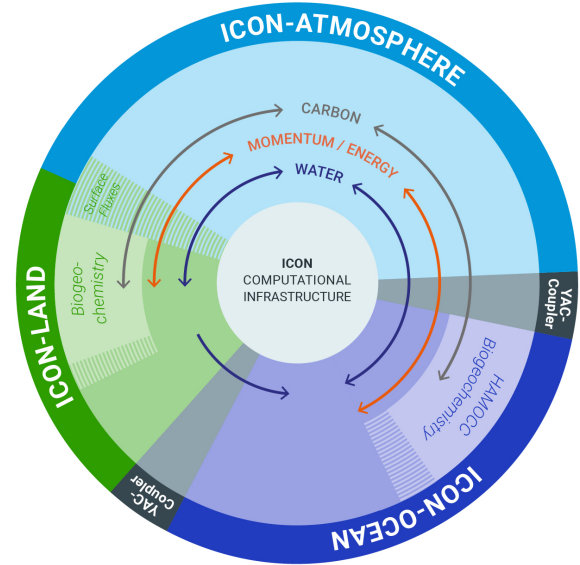

Figure 1: Schematic showing the main components of the Earth system in ICON and the exchange of energy, water, and carbon among them. Table 2 presents the degrees of freedom assigned to each component.

4 Current State of the Art

ICON [23] is a unified global numerical weather forecast and climate modeling framework. It uses a nonhydrostatic formulation of the atmosphere and performs its computations on a icosahedral-triangular C grid as described in [8]. In the vertical it uses a terrain following hybrid sigma height grid for the atmosphere [17]. It includes components for atmosphere, land, the terrestrial biosphere (vegetation), ocean, and sea ice, and ocean biogeochemistry (see Figure 1). It is mostly written in Fortran and uses a hybrid MPI/OpenMP approach for parallelization. Additionally, the atmosphere and land components have been ported to GPUs using OpenACC [9].

The configuration of ICON benchmarked for this study has no counterpart. It unifies two separate approaches (§3) that, until now, have defined the state of the art. These are: (i) simulations that are performed to resolve fine and fast scales globally for single components of the Earth system on state-of-the-art computing facilities; and (ii) complex Earth system models that are run by approximating the behavior of fast and fine scales with statistical models, configurations which then do not benefit from high-performance computing.

Two recent benchmarks define the high-performance frontier for Earth system simulations. The Simple Cloud-Resolving E3SM Atmosphere Model (SCREAM) was implemented from scratch using C++ and Kokkos. Its highest resolution was configured with $\Delta x = 3.25$ km and 128 vertical levels (with a somewhat coarser, $\Delta x = 4.875$ km physics grid). SCREAM achieved a $\tau = 458$ by scaling across 8192 nodes – 32 768 Instinct MI250X GPUs – on Frontier [33], an accomplishment recognized by the inaugural Gordon Bell Prize for Climate Modeling. At a $\Delta x = 3.5$ km, similar to the SCREAM benchmark, a Fortran implementation of the atmosphere

Table 1: Overview of current state of the art km-scale climate simulations, their horizontal grid spacing, included Earth system components (A - Atmosphere, L - Land, V - Vegetation, O - Ocean, B - Biogeochemistry, C - Carbon), the system they ran on, the temporal compression τ (e.g., Simulated Days Per Day) and a rescaled value $\tau^*_{1.25}$ corresponding to the expected performance on the same resource but with $\Delta x = 1.25\,\text{km}$.

Model	Δx / km	Components	Resource	τ	τ^*
SCREAM [33]	3.25/4.875	A L – – – –	≈87 % Frontier GPU	458	26
ICON	1.25	A L – O – –	≈95 % Lumi GPU	69	69
NICAM (H. Yashiro pers.)	3.5	A L – – – –	≈26 % Fugaku CPU	≈ 365	17
this work	**1.25**	**A L V O B C**	≈85 % JUPITER GPU	**145.7**	**145.7**

model NICAM has reported $\tau \approx 365$, for a simulation on 40 960 nodes or 26 % of Fugaku (H. Yashiro personal communication). The different temporal compressions of the applications can be roughly related to each other by calculating a reference τ, which we call τ^*. It rescales the τ from previous applications to estimate its value were the calculations performed with $\Delta x = 1.25$ km on the same resource, i.e., $\tau^* = (1.25/\Delta x)^3 \tau_{\Delta x}$. This yields τ^* of 17 and 26 for both NICAM and SCREAM, respectively (Table 1). At even finer scales, a $\Delta x = 0.22$ km version of NICAM has has been run across half of Fugaku [19], but its performance characteristics are not available.

Other components than the atmosphere of the Earth system have greater throughput and more varied coding paradigms. For instance, global ocean models have been developed using a variety of programming paradigms, including Python with a JAX backend [12] and Julia [35], with $\tau^* \approx 500$ being reported [35]. Over the land surface, continental scale simulations of ground water flow at $\Delta x = 1$ km have been performed and while performance measures are not given, the ability to iterate them to steady state suggests even larger values of compression [21]. For both the ocean and for the atmosphere, high-resolution simulations over regional domains are common [10, 25], but because τ is determined by the finest scales, regional models do not introduce new computing challenges. With the advent of machines like Alps, Fugaku, Frontier, or the emerging JUPITER, τ has ceased to be restricted by domain size for global domains with $\Delta x > 500$ m. Hence looking across component models it appears that the fundamental roadblock is the scaling of the global atmosphere component, for which the state of the art is $\tau^* \approx 20$, and the important challenge of tying all the components together in a heterogeneous computing environment.

Fully-coupled Earth system models are typically run at much coarser resolution and with less computational power. For the carbon cycle the consensus has been that even simulations at $\Delta x = 10$ km are out of reach [26]. One reason for this has been the interest in using them to explore centennial scale variability, which only becomes practical with $\tau \approx 3000$. To compare GPU and CPU performance and to also measure the energy consumption for running on GPUs vs running on CPUs we did additional simulations with a coupled atmosphere-ocean 10 km configuration on the Levante supercomputer (Figure 2). For ICON we achieve about a factor of 2 less throughput on the A100 nodes of Levante compared to the GH200 nodes. On GPUs, strong scaling begins to decline at $\tau \approx 798$ on 40 GH200 nodes. Reducing resolution by a factor of

ten could, with perfect weak scaling, increase τ by a factor of ten but would not provide enough work for a GPU node. This puts a practical limit of about $\tau = 3192$ and $\Delta x = 40$ km, which could effectively utilize 2.5 GH200 nodes. In comparison, on the CPU nodes (2×AMD 7763) strong scaling starts leveling off at a larger τ albeit at a much higher node count (see also Mauritsen et al. [20]), and less dramatically as increased cache efficiency partially offsets the lack of computation. This makes it possible to dial the resolution back further and achieve a larger τ, without making the computation too small. This however comes at a considerable cost in power, with time to solution demanding 4.4 times as much power on CPUs (Figure 2, right). Which explains why, until now, Earth system models tend to be run at coarser resolutions on older computational architectures with smaller node counts. An additional factor is that a model with $\Delta x = 40$ km must parameterize (statistically represent) many processes that a model running at finer resolution can represent explicitly. This makes the model much more complex and its individual components difficult to separate and unit test. The model becomes more challenging to adapt to different computing paradigms even if doing so were computationally advantageous.

The first atmosphere-ocean-coupled km-scale models running at km-scale were developed by the group at the Max Planck Institute for Meteorology, working with the German Climate Computing Center in Hamburg [16, 20]. They initially reported $\tau = 20$ for $\Delta x = 2.5$ km on 600 Levante CPU nodes. First simulations were performed for a few simulated months, later multi-decadal simulations have been conducted with a Δx of 5 km and 10 km [29]. In parallel, a version of the ECMWF's IFS atmosphere model has been coupled to two different ocean models at horizontal grid-spacing of up to 4.4 km [24] for which they achieved a $\tau \approx 100$ on 100 nodes of Levante.

Recently a 5 km configuration of ICON was used to simulate the annual cycle with ocean biogeochemistry included [22]. While still being considerable simpler, this version comes closest to the configuration we describe here. Because it used the same code base as used here and its performance characteristics were not explored, it is not included in Table 1.

5 Innovations Realized

Our main innovations that enable this work are arranged along two main axes: (1) exploiting functional parallelism by efficiently mapping components to specialized heterogeneous systems and (2) simplifying the implementation and optimization of an important
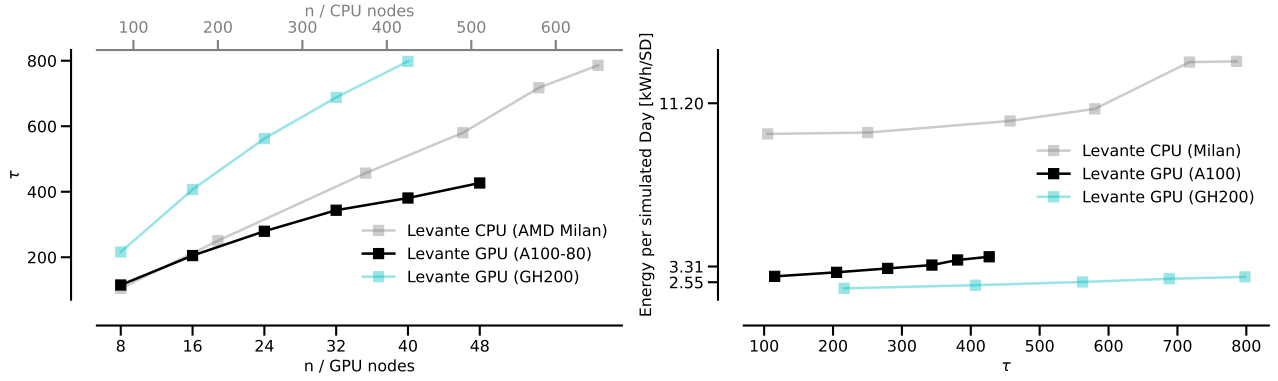
Figure 2: Strong scaling on CPUs and GPUs of the coupled 10 km ICON (without biogeochemistry submodel) on Levante (left). Nearly four-fold improved energy efficiency on Levante GPUs vs CPUs (right).

component by separating its implementation in Fortran from the optimization details of the target architecture.

5.1 Exploiting the Earth System's Complexity to Flexibly Utilize System Parallelism

The additional challenge of simulating the Earth system (as opposed to just the coupled atmosphere and ocean) is that a great number of additional components need to be modeled and their computation needs to be harmonized (see Table 2). What makes the problem challenging is that these components have different performance characteristics and none of them dominates the runtime. This leads to the well-known flat performance profile of Earth system models which makes optimization a painstaking task. To address this, we adopted different strategies, as illustrated schematically in Figure 3.
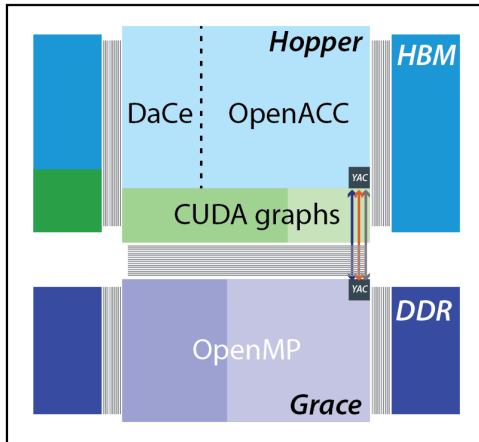

Figure 3: Mapping of the Earth system components (from Figure 1) to the GH200 superchip with indicated innovations. On the platforms tested, four superchips are on each node. For our benchmarking experiments, we utilized 10 MPI processes with each 6 OpenMP threads for the ocean on the CPUs and 1 MPI process for the GPU of each GH200 superchip.

For the **atmosphere component**, two approaches were tested. Originally, the entire atmosphere was GPU-accelerated with OpenACC [9]. For this project we also replaced the OpenACC implementation of one of the largest kernels with a DaCe [1] implementation, as described further below. Both implementations allow us to keep the data resident on the GPU throughout the whole simulation and avoid memory transfers between the GPU and the CPU, which can be prohibitively expensive on some architectures. The boundary exchange is done via MPI point-to-point communications, which is implemented in a way that it can employ direct GPU-GPU transfers with GPUDirect RDMA. *The atmosphere component is highly optimized for parallel GPU execution as we will elaborate later.*

For the **land component** (JSBach), the introduction of an interactive biosphere model introduced a very large number of additional small GPU kernels. The small kernel problem was exacerbated by the small number of soil levels (5), which results in little computational intensity. Launching many small kernels on the GPU incurs a significant latency overhead and becomes very expensive. An innovative way to reduce this cost is to employ CUDA Graphs for the OpenACC kernels. CUDA Graphs record the kernel call flow of a component (like JSBach) and replay it exactly the same way in subsequent calls to this component. Recording causes a slightly increased latency for the first invocation, however, all the subsequent replays are virtually latency-free. Moreover, due to the JSBach model implementation operating on multiple independent plant functional types, many of the executed kernels may run simultaneously using CUDA graphs. *Overall, we observe a speedup for the land and vegetation parts of the model on the order of 8-10× depending on the grid-spacing.*

While the **ocean component** has been ported with OpenACC to run on GPUs, it poses a special challenge for efficient scaling, especially on GPUs. This arises because filtering of fast wind-driven surface waves introduces a tightly-coupled 2d-equation-system distributed over all ranks to solve for its barotropic mode. The computational characteristic of this solver is dominated by global communication, while the computations in between communication are very small. In contrast to the land model, which directly exchanges fluxes with the atmospheric component on the atmospheric timestep, and therefore needs to run on GPUs (for most

architectures), the slow evolution of the ocean allows it to be much more loosely coupled to the atmosphere and thus run concurrently to the atmosphere and land components. Only energy, water and carbon are exchanged between the atmosphere and the ocean at a coupling timestep every 10 simulated minutes through the coupler YAC [13]. These properties – a global solver, loose coupling, and a much larger τ – make the ocean very well suited to exploit CPU resources. Hence we take advantage of this flexibility to utilize the hybrid computational power of both CPUs and GPUs on the same nodes when available. Specifically, for the GH200 superchip we run the ocean on the Grace CPUs which are a powerful resource that would otherwise be underutilized. *Given that most other simulations rely mostly on GPUs, by mapping the Earth system workload smartly, we essentially run the ocean component for free.*

Table 2: Earth system model global grid configurations for ICON, showing the number of grid cells, vertical level and prognostic variables for each component model used in the scaling runs. Nominal resolution for ICON is given as the square root of the mean cell area. Velocity components normal to the triangle edges are counted as 1.5 prognostic variables. Land has four physical state variables on five levels, 21 additional carbon pools, plus the leaf area index as predicted variables, associated with up to 11 plant functional types (reported in the levels column for vegetation). Degrees of freedom are calculated as the product of the spatial degrees of freedom times their number of prognostic variables.

	Δx / km	cells $\times 10^8$	levels	vars	Δt / s
10 km: 1.2×10^{10} deg. of freedom					
Atmosphere	10	0.05	90	12.5	75
Land	"	0.015	5	4	"
Vegetation	"	"	≤ 11	22	"
Ocean & sea-ice	"	0.037	72	5	600
Biogeochemistry in ocean	"	"	"	19	"
1.25 km: 7.9×10^{11} deg. of freedom					
Atmosphere	1.25	3.36	90	12.5	10
Land	"	0.98	5	4	"
Vegetation	"	"	≤ 11	22	"
Ocean & sea-ice	"	2.38	72	5	60
Biogeochemistry in ocean	"	"	"	19	"

The **biogeochemistry component** of the ocean, HAMOCC, involves a large number of tracers (prognostic variables in Table 2) that interact with one another and are transported through the ocean. HAMOCC does not have a global solver, but shares the large τ of the ocean (compared to the atmosphere) due to the slow transport, and the loose coupling with the fast atmosphere. This makes HAMOCC better adapted for GPUs than the ocean dynamical core. To exploit more parallelism and gain flexibility in its implementation, a concurrent version of HAMOCC has been developed [18], which allows it to be run on GPUs through OpenACC instrumentation of the Fortran code. A downside of this approach is that large three-dimensional fields need to be exchanged between the ocean dynamical core and HAMOCC at every ocean timestep. Therefore exploiting concurrent GPU parallelism in HAMOCC is not beneficial in all cases. For a powerful enough CPU like on the GH200 superchip it is also possible to include the biogeochemistry together

with the ocean on the CPU, as it shares two of the features (loose coupling and larger τ) and thus can more efficiently use a resource that would otherwise be idle. *In our setup, we run HAMOCC together with the ocean component on the CPU and also essentially get it for free, concurrent to the GPU execution.*

5.1.1 Final Mapping. Summing up, our setup is as follows: For the atmosphere and land models we use four MPI ranks per node, each assigned to a GPU and pinned to the CPU cores of the same superchip. These ranks launch kernels on the GPU and can use a number of helper threads. In addition, for the ocean, sea ice, and biogeochemistry, we use 40 MPI ranks with six OpenMP threads each per node, ten ranks each pinned to its respective NUMA domain. Some of these ranks could also be utilized for asynchronous output. In total, we use 244 of the 288 available cores per node leaving the remaining cores available to support GPU and operating system services and reduce noise [15].

To maximize throughput with our approach of utilizing all components of the system we require a delicate load balancing. Since the GPUs are significantly more powerful than the CPUs, we aim to keep the GPUs running at full capacity at all times. This requires the ocean component to consistently run faster than the atmosphere, ensuring that the ocean component rather than the atmosphere waits at the synchronization points where the coupler exchanges fluxes between the components. However, assigning too many CPU resources to the ocean to make it arbitrarily fast can actually slow down the atmosphere. An important advantage of the GH200 superchips is their ability to flexibly partition power consumption between CPUs and GPUs, which enables further performance optimization of heterogeneous configurations for specific node counts.

This is due to a key characteristic of the systems targeted by this project: CPU and GPU components share a common power and thermal budget – 660 W per superchip on Alps and 680 W per superchip on JUPITER (see Table 3) – which is substantially lower than the maximum combined power capacity of both components (1000 W, according to NVIDIA). As a result, power distribution must also be carefully balanced, adding even more complexity to the setup. Fortunately, climate models – and especially the ICON atmosphere component – are predominantly memory-bandwidth bound, meaning the GPU does not require its full theoretical power budget, which is largely used by the compute facilities of the device.

5.2 Separation of Concerns: From Fortran to GPU

To achieve performance and scalability on today's heterogeneous systems, ICON's code base has evolved to support a wide range of configurations: from pragmas enabling OpenMP and OpenACC parallelization, to Intel, Cray, GCC, and NEC vector annotations to macros that duplicate code to reorder loops. For example the dynamical core, the most compute-intensive part of the most expensive atmosphere component, contains 2728 non-empty Fortran source lines of code, of which less than 50 % are actually describing the computation. The remaining lines of code are optimizations for specific architectures implemented using OpenACC (20 %) and other directives (12 %) as well as duplicated code reordering loops (6 %). The following figure shows a small code excerpt from ICON's dynamical core illustrating three different families of pragmas and

how macros and duplicate loop headers are used to specialize for different architectures.

```
!$ACC PARALLEL DEFAULT(PRESENT) ASYNC(1)          OpenACC GPU pragmas
!$ACC LOOP GANG VECTOR TILE(32, 4)
#ifdef __LOOP_EXCHANGE                             CPU vectorization pragma
    DO jc = i_startidx, i_endidx                   duplicate loop header code
!DIR$ IVDEP
      DO jk = 1, nlev
      z_ekinh(jk,jc,jb) = &
#else
!$NEC outerloop_unroll(4)                          NEC-specific pragma
    DO jk = 1, nlev
      DO jc = i_startidx, i_endidx                 duplicate loop header code
      z_ekinh(jc,jk,jb) = &
#endif
        p_int%e_bln_c_s(jc,1,jb)*z_kin_hor_e(ieidx(jc,jb,1),jk,ieblk(jc,jb,1)) + &
        p_int%e_bln_c_s(jc,2,jb)*z_kin_hor_e(ieidx(jc,jb,2),jk,ieblk(jc,jb,2)) + &
        p_int%e_bln_c_s(jc,3,jb)*z_kin_hor_e(ieidx(jc,jb,3),jk,ieblk(jc,jb,3))

    ENDDO                          some lines down, more OpenMP pragmas
    ENDDO
    !$ACC END PARALLEL                                    ↓
```

The resulting duplication makes not only the initial development costly but makes extending or maintaining the code excessively complicated. For example, changing a loop would require updating all copies consistently. Updating any source code would need to also update the describing pragmas accordingly. Yet, the systems described by those annotations may not even be available to the programmer at the time of the change.

We recognize that the Fortran language itself specifies the computations in a way that allows us to extract their dataflow dependencies. We extend DaCe [1], a parallel datacentric compilation framework, with a Fortran language parser specialized to the dynamical core code. We then remove all pragmas and macros from the code to create the cleanest form with only about 1400 lines of code (less than 50 %).
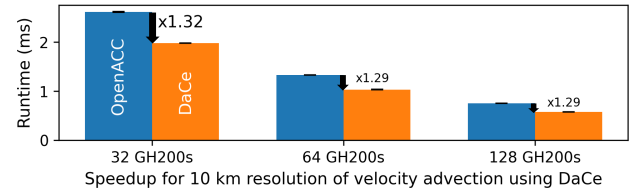
We use our new parser to read this sequential Fortran code into DaCe's internal "Stateful Dataflow Graph" (SDFG) representation [1], maintaining all of the code's semantics while representing data movement explicitly. This representation naturally enables us to analyze independent pieces of code and loops and extract the maximum parallelism, which we can then map again to different target architectures such as GPUs or CPUs.

DaCe has been demonstrated to optimize the established weather and climate code FV3 after it has been rewritten in a Python dialect [2]. The innovation in this work is that *we use the unmodified sequential Fortran code of ICON to achieve a similar result. This demonstrates a clear separation of concerns between the application scientist writing her Fortran code and the performance engineer using transformations and code-generation to accelerate it on various architectures.*
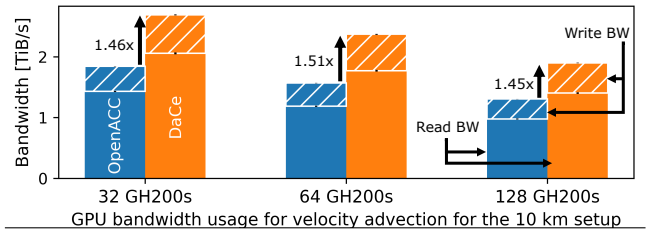
To achieve this separation of concerns, DaCe allows the performance engineers to write "performance metaprograms" that transform a piece of a SDFG into a new representation targeted at specific devices. We designed and wrote a series of such metaprograms to transform the dynamical core code into a fast representation for GH200 GPUs as well as CPUs. We note that all the optimizations are without changes to the original Fortran code and thus invisible to the climate scientist. If the Fortran code is changed but maintains a similar computational structure, then the transformations continue to apply as they are matched to dataflow structures on the SDFG, see Ben-Nun et al. [1] for details.

We illustrate one code-specific optimization out of many that we applied: The ICON dynamical core performs its computations on an icosahedral grid composed of triangles and its dual grid composed of mostly hexagons and 12 pentagons. This structure requires a significant amount of index lookups in arrays to determine grid indices. Some of these indices can be reused by carefully reordering computations [11]. By leveraging the freedom offered by our maximally parallel representation, we can reduce the number of integer index lookups required per grid point by an average factor of 8×.

After our transformations, the whole dynamical core code transformed via DaCe consistently outperforms the manually tuned OpenACC implementation - starting from the sequential Fortran implementation and requiring no pragmas or manual data layout transformations. The following figure shows an overview of the achieved performance for the 10 km setup comparing the version with manual pragmas (blue) to the DaCe implementation (orange) for the GPU execution on GH200.

Speedup for 10 km resolution of velocity advection using DaCe

As the dynamical core is memory-bound, we evaluate the sustained bandwidth during execution for both the OpenACC and DaCe implementations. Assuming that 100 % busy DRAM would yield a bandwidth of 4 TiB/s on GH200 GPUs, we convert the DRAM utilization percentages reported by Nsight for these kernels to average bandwidth in different experiment configurations as is shown in the following figure.

GPU bandwidth usage for velocity advection for the 10 km setup

As the 128 GH200 configuration reflects the same work per chip as our hero run (see Figure 4), it would achieve more than 15 PiB/s sustained memory bandwidth, which is about 50 % peak.

6 How Performance Was Measured

6.1 Earth System Model Configuration

Computational benchmarks were performed using the coupled Earth system model ICON including the atmosphere, land (with dynamic vegetation), ocean, sea-ice, ocean biogeochemistry, hydrological discharge from land to ocean, and the carbon exchange between the components. The model was initialized with the Earth system state on the 1st of January 2020. For the ocean and its biogeochemical state a separate spin-up simulation was performed

using prescribed surface conditions from reanalysis of the atmosphere. The initial atmospheric state was interpolated from reanalysis onto the computational grid and carbon-pools were initialized from separately spun-up states. For our simulations, we consider a development configuration with 10 km horizontal resolution and a configuration to run at scale with 1.25 km horizontal resolution. Both configurations only differ in their horizontal grid and the timestep length.

6.2 System Details

The multi-component setup of ICON presented in this work is enabled by NVIDIA's Grace Hopper GH200 superchip in the Alps and JUPITER systems. The superchip combines in one package an ARM CPU, *Grace*, with 72 cores and 120 GB LPDDR5 memory, and a *Hopper* GPU with 96 GB HBM3 memory. Both parts are connected by the *NVLink-C2C* bus with 900 GB/s bandwidth, allowing high-bandwidth, cache-coherent access to the two memories [7]. The CPU and the GPU share one combined Thermal Design Power (TDP), in which power is dynamically distributed first to the CPU and the remainder to the GPU. The TDP value is system-dependent.

	JUPITER	Alps
# Nodes	5884	2688
# Superchips per Node	4	4
# Superchips	23 536	10 752
Superchip TDP	680 W	660 W
Interconnect	InfiniBand NDR200	Slingshot-11
Inj. Bandwidth / Node	4×200 Gbit/s	4×200 Gbit/s

Table 3: High-performance computing systems used.

JEDI/JUPITER. JUPITER, which is currently being installed at the Jülich Supercomputing Centre in Germany, is expected to be the first European (HPL) exascale computer. The system currently consists of 5884 nodes (23 536 GH200 superchips), connected with NDR200 InfiniBand. ICON is one of the driving applications for the design of JUPITER [14]. Through early access we used JEDI, a single rack (48 node) development platform, to prepare larger runs on Alps and then JUPITER.

The code and setup we demonstrate here will be used in production to simulate 13 months at 1.25 km, an unprecedented scale for the simulation of the fully coupled Earth systemx.

Alps. Alps is a supercomputer operated by the Swiss National Supercomputing Centre CSCS that deploys 2688 nodes with the total of 10 752 GH200 superchips to reach 435 PFLOP/s HPL performance on the TOP500 list (place 7; November 2024). The interconnect is Slingshot-11. An overview of both systems is presented in Table 3.

Software Infrastructure. On JUPITER, CUDA 12.6, NVIDIA HPC SDK 25.5, and OpenMPI 5.0.5 with support for CUDA-aware MPI were used. Alps supports a very similar environment with Cray-MPICH instead of OpenMPI. The final runs on Alps used CUDA 12.8, NVIDIA HPC SDK 25.3 and Cray-MPICH 8.1.32. Because ICON runs so close to the machine limits, it is sensitive to the balance of many low-level system parameters at different node counts. Moving from

JEDI, where the initial development happened, to Alps, involved the close cooperation of the operators to harmonize these settings and maintain good performance across the entire system.

6.3 Performance Metrics

The most relevant performance metric for climate simulations is the temporal compression τ, which describes the model throughput in units of simulated time versus actual time. We derive τ from the time it takes to simulate 3 hours in the 1.25 km configuration (24 hours in the 10 km configuration) excluding the initialization time. Initialization takes a measurable amount of time in our short benchmark runs. However, in a production setup when the model is run for hours its contribution to the overall run time is relatively small, which is why we exclude it here. The simulation time is measured independently for the atmosphere/land and ocean/sea-ice/biogeochemistry components. Included in timings is the coupling time, i.e., the amount of time atmosphere/land have to wait for ocean/sea-ice/biogeochemistry and vice versa. Ideally, the coupling time should be close to zero for the computational expensive atmosphere/land components running on the GPUs, meaning that they do not have to wait for the ocean/sea-ice/biogeochemistry components to finish their calculations. Timings are measured based on the CPUs time stamp counter. Measurements are taken across all ranks and the maximum value is presented.

6.4 I/O Performance Metrics

ICON has a number of different I/O schemes for various use-cases. Most notable are asynchronous schemes for model output, output via the coupling interface, writing and reading of restarts, and distributed reading of initial and boundary conditions.

Checkpoint/restart means that the model dumps the complete state from memory to disk. Since the state is huge, efficient and fast I/O is necessary. Checkpoint/restart I/O in ICON has a number of modes, from rank-0-I/O to asynchronous multi-file-I/O. We employ the synchronous multi-file variant, where a configurable subset of ranks collects the variables and writes them to one file each. Reading, in turn, can be done with a different subset of ranks, where each rank reads parts of the files and distributes the data to the corresponding ranks.

With the asynchronous output scheme, additional MPI tasks are assigned to output servers. Output fields are communicated to these servers via MPI one-sided remote memory access in configurable intervals, additional operations (averaging, accumulating, interpolation to different output grid in vertical and/or horizontal) can be applied. Disk I/O takes place concurrently to the model integration. The overhead in computational resources depends on the number of fields and desired storage frequency. The required bandwidth to the storage is low and impact on neighbor exchange is small, so that I/O does not appreciably impact τ. Moreover, this is something that is aided by increased temporal sparsity output as Δx, and hence the timestep, becomes small, while the output frequency remains constant.

7 Performance Results

We performed four sets of benchmark experiments. With our 10 km development configuration, we ran a series of simulations for one
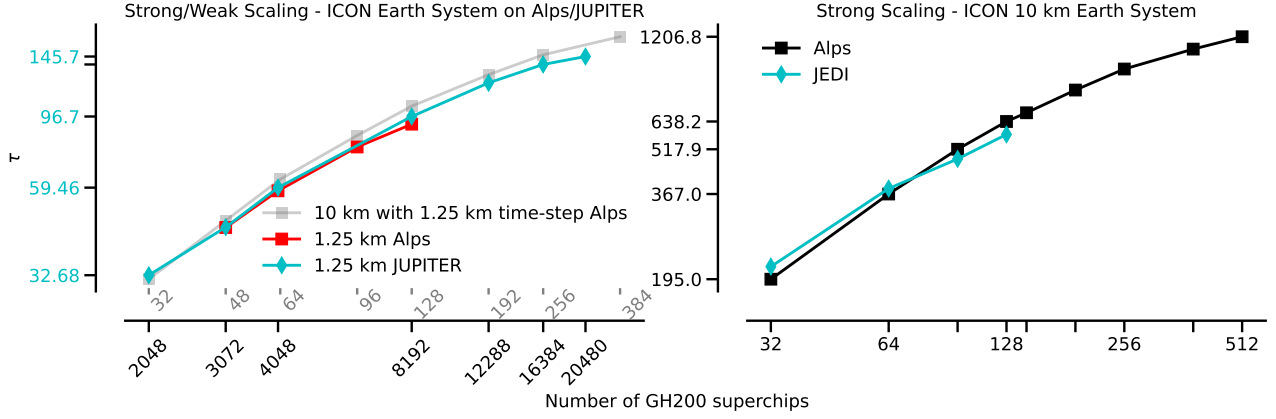
Figure 4: Strong scaling of the full ICON Earth system model at 1.25 km grid-spacing (left). The gray line shows scaling with a configuration with a factor 64 less grid cells (10 km resolution) as the 1.25 km configuration but using the same timestep, serving as a reference line assuming perfect weak-scaling. Strong scaling of the ICON Earth system model with 10 km grid-spacing on Alps and JEDI (right)

simulated day each on 32 to 128 GH200 superchips on the JEDI system to measure strong scaling. The same configuration was also run on Alps, where we extended the strong-scaling measurements to 512 superchips. Results are shown in Figure 4 on the right-hand side. Performance on the two systems is comparable, with slightly better performance on JEDI for low superchip counts, while the performance on Alps is slightly better for higher superchip counts. The scaling curve flattens when approaching to 512 superchips. At this point only about 10 800 horizontal grid cells are distributed to each GPU for calculations, which is too little to fully utilize the GPU compute power in our configuration with 90 levels.

To measure weak-scaling, we performed another set of experiments with the 10 km configuration on Alps, where we use the same time steps of 10 s as needed for the 1.25 km computations. Increasing the grid-spacing of the computational mesh from 10 km to 1.25 km grid-spacing corresponds to a 64 fold increase of grid cells in the horizontal (Table 2). This means, the 10 km configuration has the same computational load on 32 superchips, as the 1.25 km setup on 2048 superchips (Figure 4, left).

Finally, we also performed scaling simulations with the 1.25 km full Earth system configuration of ICON on Alps and JUPITER. The smallest node count we could fit the configuration with its nearly 1× 10^{12} degrees of freedom (see Table 2) is 2048 superchips of JUPITER (≈10 % of the full system) achieving a temporal compression of $\tau = 32.7$. The production setup scales well to 20 480 superchips, the maximum number of nodes to which we had access for the strong scaling experiment, achieving a temporal compression of $\tau = 145.7$ on JUPITER and $\tau = 91.8$ on 8192 superchips of Alps (see Figure 4).

Weak scaling results from the benchmarks with the 10 km configuration with the time step of the 1.25 km configuration (gray curve showing the 10 km ICON Earth system model in the left plot of figure 4), gives a weak scaling efficiency of about 90 % while increasing the size of the computational problem by a factor of 64. For 384 superchips with the 10 km resolution (with the 1.25 km timestep), we get $\tau \approx 167$. Applying this to the 1.25 km configuration, accounting

for the 90 % weak scaling estimated above, gives a $\tau = 150$ for the 1.25 km model on 24 576 superchips (which corresponds to the full JUPITER system).

Our complex Earth system model configuration requires all components to scale similarly well to achieve results up to highest superchip counts of the largest machines. We have configured the model so that the most computationally expensive component dictates the maximum time compression achievable. At superchip counts not yet available to us on Jupiter, global communication in the 2d solver in the ocean model can become a bottleneck. Performance tests at lower node counts give us confidence that a recent refactoring of this global solver will allow us to maintain scaling of the 1.25 km configuration across the entirety of JUPITER.

Since our 1.25 km setup runs for 3 simulated hours, we did not write any output during the time loop but wrote restart files consisting of the full model states at the end of the simulation. We measured I/O performance based on the reading and writing of these restart files. For the 1.25 km setup the restart files have a size of 9265.50 GiB for the atmosphere and 7030.91 GiB for the ocean respectively. To read and write the restart files for the ocean part, we ran up to 2579 processes when running on 8000 superchips. Staggered reading allows us to read the restart at a rate of 615.61 GiB/s. The writing of the ocean restart file at the end of the simulation achieved a rate of 198.19 GiB/s.

8 Implications

Until now it has not been possible to represent the interaction between the fine and fast scales that mediate the interactions between water and energy – for instance the daytime heating over land that causes afternoon thunderstorms – with the large and slow scales of the carbon reservoirs they influence. These reservoirs collectively define the land biosphere, which then go on to influence how the fast and fine scales develop in the future. Figure 5 shows those Earth system component interactions through the flow of carbon between them.
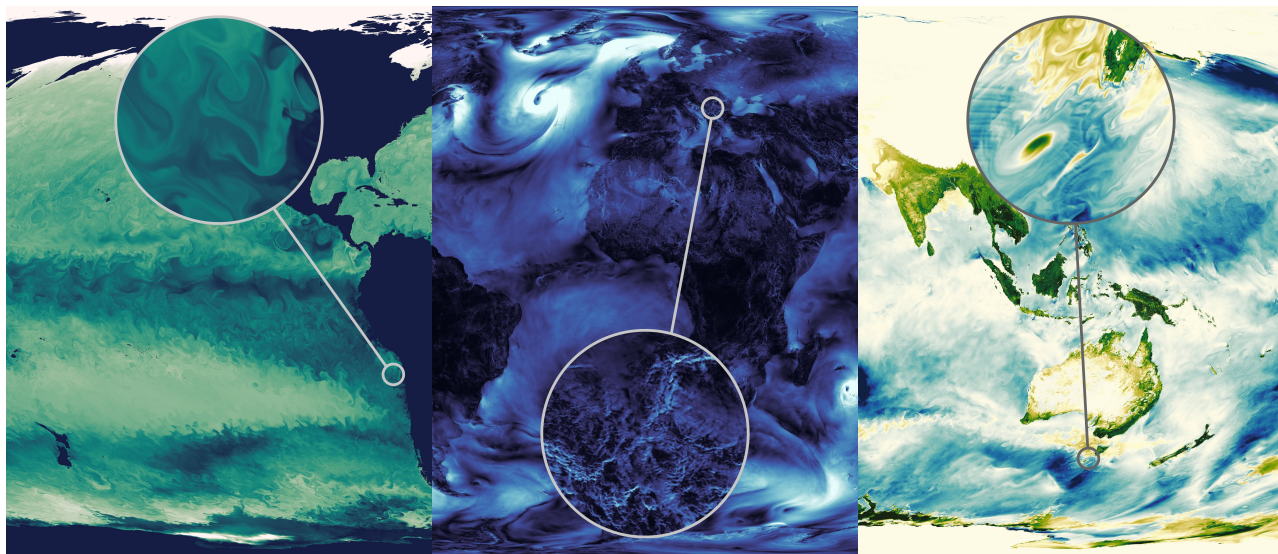
Figure 5: Snapshot of phytoplankton, near-surface wind and air-sea CO_2 flux at 2020-01-01 03:00. On the left, phytoplankton concentration on a logarithmic scale between 10^{-9} kmol Pa/m^3 and 10^{-6} kmol Pa/m^3 are shown. In the center, surface winds are displayed with color limits in the range from 0 m/s to 20 m/s. On the right, the air-sea/land carbon flux is illustrated with the color scale spanning $\pm 4 \cdot 10^{-7}$ kg m^{-2} s^{-1} and green values implying carbon uptake and blue values indicating carbon release for ocean and land (values over the ocean were multiplied by 30 to enhance visibility). Insets highlight specific regions: water mass upwelling off the coast of Chile (left), near-surface wind patterns over the mountains of the Balkans (center), and CO_2 flux off the coast of Tasmania (right).

For the first time, we simulate the impact of small scales on the carbon flows, globally. More tangibly, this means we represent the global fluxes of carbon as the net result of local interactions between vegetation, topographic features, watersheds, local wind systems, ocean eddies and a realistic representation of the spatial and intensity distribution of precipitation. Capturing these types of interactions require a model with local (km-scale) granularity, globally, an explicit representation of a diverse array of Earth system processes, and a throughput (or time-compression) of about 100. The computational challenge was thought to be unachievable, forcing researchers to focus on approaches that would otherwise be suboptimal (e.g. [5]). We show that this thinking was wrong, and km-scale global models of the full Earth system are realizable on today's technology.

Our computations bring Earth system modeling to the domain of high-performance computing. Previous and pioneering studies demonstrated the relevance of exascale computing for studies of the physical climate system [27, 33]. This existing capability is helping to uncover new physics, for instance in how imposed changes to the land-surface influence the atmosphere [36]. By expanding this capability we demonstrate that it can now also be used to study the Earth system in all its fullness. This opens up new and exciting scientific frontiers.

One implication of our work is to show that the complexity of the problem, one of the aspects thought to make it difficult, makes it tractable. This is because it allows us to pair the physical complexity with the complexity of today's computational environments to expose different types of parallelism and use machines more

efficiently than for simpler problems. In our case this means using the powerful CPU on the GH200 superchip to couple the rest of the Earth system to the atmosphere, more or less for free. We also show how a flexible structuring of the solvers, to use GPUs or CPUs or, as in the case of the ocean biogeochemistry, to run concurrently or as part of the main kernels, makes it possible to change the configuration to efficiently adapt to very different architectural profiles.

Our achievement also demonstrates the potential of new approaches to help separate the concerns of domain scientists and computer scientists, which makes it possible for them to work more closely together. Using the Data Centric programming paradigm we optimize a large kernel of a native Fortran code to run more efficiently on a demanding computational architecture than the originally heavily optimized Fortran implementation instrumented with various libraries. Our approach preserves the readability of the code, making it easier to maintain and modify and retaining its familiarity for the domain scientist. Hence, a further implication of our work, beyond showing the relevance of high performance computing to Earth system modeling, is to show the relevance of computer scientists to Earth system scientists.

Looking forward, with the capability already achieved, new machines with architectures similar to those used to perform our benchmarks are coming on line. JUPITER is roughly 2.5 times larger than Alps when at full capacity. With the performance demonstrated for 1024 nodes of JUPITER ($\tau = 59.5$, Figure 4), this amounts to an ability to simultaneously simulate two scenarios of future warming, each for thirty years, with three ensemble members to

sample variability, in a little less than half a year. This has enormous and enduring potential to provide *full global Earth system information on local scales* about the implications of future warming for both people and eco-systems, information that otherwise would not exist.

Acknowledgments

References

[1] Tal Ben-Nun, Johannes de Fine Licht, Alexandros Nikolaos Ziogas, Timo Schneider, and Torsten Hoefler. 2019. Stateful Dataflow Multigraphs: A Data-Centric Model for Performance Portability on Heterogeneous Architectures. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis (SC19)*.

[2] Tal Ben-Nun, Linus Groner, Florian Deconinck, Tobias Wicky, Eddie Davis, Johann Dahm, Oliver Elbert, Rhea George, Jeremy McGibbon, Lukas Trümper, Elynn Wu, Oliver Fuhrer, Thomas Schulthess, and Torsten Hoefler. 2022. Productive Performance Engineering for Weather and Climate Modeling with Python. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis (SC'22)*.

[3] Sandrine Bony, Bjorn Stevens, Isaac H. Held, John F. Mitchell, Jean-Louis Dufresne, Kerry A. Emanuel, Pierre Friedlingstein, Stephen Griffies, and Catherine Senior. 2013. Carbon Dioxide and Climate: Perspectives on a Scientific Assessment. In *Climate Science for Serving Society*, G.R. Asrar and J.W. Hurrell (Eds.). Springer-Verlag, 391–413. doi:10.1007/978-94-007-6692-1_14

[4] D. Chen, M. Rojas, B.H. Samset, K. Cobb, A. Diongue Niang, P. Edwards, S. Emori, S.H. Faria, E. Hawkins, P. Hope, P. Huybrechts, M. Meinshausen, S.K. Mustafa,

[5] G.-K. Plattner, and A.-M. Tréguier. 2021. Framing, Context, and Methods. In *Climate Change 2021: The Physical Science Basis. Contribution of Working Group I to the Sixth Assessment Report of the Intergovernmental Panel on Climate Change*, V. Masson-Delmotte, P. Zhai, A. Pirani, S.L. Connors, C. Péan, S. Berger, N. Caud, Y. Chen, L. Goldfarb, M.I. Gomis, M. Huang, K. Leitzell, E. Lonnoy, J.B.R. Matthews, T.K. Maycock, T. Waterfield, O. Yelekçi, R. Yu, and B. Zhou (Eds.). Cambridge University Press, 147–286. doi:10.1017/9781009157896.003

[5] Veronika Eyring, Pierre Gentine, Gustau Camps-Valls, David M. Lawrence, and Markus Reichstein. 2024. AI-empowered next-generation multiscale climate modelling for mitigation and adaptation. *Nature Geoscience* 17, 10 (Sept. 2024), 963–971. doi:10.1038/s41561-024-01527-w

[6] P. Friedlingstein, J.-L. Dufresne, P. M. Cox, and P. Rayner. 2003. How Positive Is the Feedback between Climate Change and the Carbon Cycle? *Tellus B* 55, 2 (2003), 692–700. doi:10.1034/j.1600-0889.2003.01461.x

[7] Luigi Fusco, Mikhail Khalilov, Marcin Chrapek, Giridhar Chukkapalli, Thomas Schulthess, and Torsten Hoefler. 2024. Understanding Data Movement in Tightly Coupled Heterogeneous Systems: A Case Study with the Grace Hopper Superchip. arXiv:2408.11556 [cs.DC] https://arxiv.org/abs/2408.11556

[8] M. A. Giorgetta, R. Brokopf, T. Crueger, M. Esch, S. Fiedler, J. Helmert, C. Hohenegger, L. Kornblueh, M. Köhler, E. Manzini, T. Mauritsen, C. Nam, T. Raddatz, S. Rast, D. Reinert, M. Sakradzija, H. Schmidt, R. Schneck, R. Schnur, L. Silvers, H. Wan, G. Zängl, and B. Stevens. 2018. ICON-A, the Atmosphere Component of the ICON Earth System Model: I. Model Description. *Journal of Advances in Modeling Earth Systems* 10, 7 (2018), 1613–1637. doi:10.1029/2017MS001242

[9] Marco A Giorgetta, William Sawyer, Xavier Lapillonne, Panagiotis Adamidis, Dmitry Alexeev, Valentin Clément, Remo Dietlicher, Jan Frederik Engels, Monika Esch, Henning Franke, et al. 2022. The ICON-A model for direct QBO simulations on GPUs (version icon-cscs: baf28a514). *Geoscientific Model Development* 15, 18 (2022), 6985–7016.

[10] Jonathan Gula, M. Jeroen Molemaker, and James C. McWilliams. 2016. Submesoscale Dynamics of a Gulf Stream Frontal Eddy in the South Atlantic Bight. *Journal of Physical Oceanography* 46, 1 (2016), 305–325. doi:10.1175/JPO-D-14-0258.1

[11] Tobias Gysi, Christoph Müller, Oleksandr Zinenko, Stephan Herhut, Eddie Davis, Tobias Wicky, Oliver Fuhrer, Torsten Hoefler, and Tobias Grosser. 2021. Domain-Specific Multi-Level IR Rewriting for GPU: The Open Earth Compiler for GPU-accelerated Climate Simulation. *ACM Trans. Archit. Code Optim.* 18, 4, Article 51 (Sept. 2021), 23 pages. doi:10.1145/3469030

[12] Dion Häfner, Roman Nuterman, and Markus Jochum. 2021. Fast, cheap, and turbulent—Global ocean modeling with GPU acceleration in python. *Journal of Advances in Modeling Earth Systems* 13, 12 (2021), e2021MS002717.

[13] Moritz Hanke, René Redler, Teresa Holfeld, and Maxim Yastremsky. 2016. YAC 1.2. 0: new aspects for coupling software in Earth system modelling. *Geoscientific Model Development* 9, 8 (2016), 2755–2769.

[14] Andreas Herten et al. 2024. Application-Driven Exascale: The JUPITER Benchmark Suite . In *SC24: International Conference for High Performance Computing, Networking, Storage and Analysis*. IEEE Computer Society, Los Alamitos, CA, USA, 1–45. doi:10.1109/SC41406.2024.00038

[15] Torsten Hoefler, Timo Schneider, and Andrew Lumsdaine. 2010. Characterizing the Influence of System Noise on Large-Scale Applications by Simulation. In *International Conference for High Performance Computing, Networking, Storage and Analysis (SC'10)*.

[16] Cathy Hohenegger, Peter Korn, Leonidas Linardakis, René Redler, et al. 2023. ICON-Sapphire: simulating the components of the Earth system and their interactions at kilometer and subkilometer scales. *Geoscientific Model Development* 16, 2 (Jan. 2023), 779–811. doi:10.5194/gmd-16-779-2023

[17] Daniel Leuenberger, Marcel Koller, Oliver Fuhrer, and Christoph Schär. 2010. A Generalization of the SLEVE Vertical Coordinate. *Monthly Weather Review* 138, 9 (2010), 3683 – 3689. doi:10.1175/2010MWR3307.1

[18] Leonidas Linardakis, Irene Stemmler, Moritz Hanke, Lennart Ramme, Fatemeh Chegini, Tatiana Ilyina, and Peter Korn. 2022. Improving scalability of Earth system models through coarse-grained component concurrency–a case study with the ICON v2. 6.5 modelling system. *Geoscientific Model Development* 15, 24 (2022), 9157–9176.

[19] S. Matsugishi, T. Ohno, and M. Satoh. 2024. Differences in the cloud, precipitation, and convection representation between the global sub-km mesh simulation and km simulations. In *EGU General Assembly 2024, Vienna, Austria, 14–19 Apr 2024*.

[20] Thorsten Mauritsen, Rene Redler, Monika Esch, Bjorn Stevens, Cathy Hohenegger, Daniel Klocke, Renate Brokopf, Helmut Haak, Leonidas Linardakis, Niklas Röber, and Reiner Schnur. 2022. Early Development and Tuning of a Global Coupled Cloud Resolving Model, and Its Fast Response to Increasing CO2. *Tellus Series A: Dynamic Meteorology and Oceanography* 74 (2022), 346–363. doi:10.16993/tellusa.54

[21] R. M. Maxwell, L. E. Condon, and S. J. Kollet. 2015. A high-resolution simulation of groundwater and surface water over most of the continental US with the integrated hydrologic model ParFlow v3. *Geoscientific Model Development* 8, 3 (2015), 923–937. doi:10.5194/gmd-8-923-2015

[22] D. M. Nielsen, F. Chegini, N. Serra, A. Kumar, N. Brüggemann, C. Hohenegger, and T. Ilyina. 2025. Resolved tropical cyclones trigger CO2 uptake and phytoplankton

bloom in an Earth system model simulation. *Earth ArXiv Preprint* (2025). doi:10.31223/X5KF05

[23] ICON partnership (MPI-M; DWD; DKRZ; KIT; C2SM). 2024. ICON release 2024.10. World Data Center for Climate (WDCC) at DKRZ. doi:10.35089/WDCC/IconRelease2024.10

[24] Thomas Rackow, Tobias Becker, Xabier Pedruzo Bagazgoitia, Irina Sandu, Lorenzo Zampieri, and Florian Ziemen. 2022. Storm-resolving simulations with IFS-NEMO/FESOM in the NextGEMS project. In *EGU General Assembly 2022*. Copernicus GmbH, Vienna, Austria. doi:10.5194/egusphere-egu22-10757

[25] R. M. Rasmussen, F. Chen, C. H. Liu, K. Ikeda, A. Prein, J. Kim, T. Schneider, A. Dai, D. Gochis, A. Dugger, Y. Zhang, A. Jaye, J. Dudhia, C. He, M. Harrold, L. Xue, S. Chen, A. Newman, E. Dougherty, R. Abolafia-Rosenzweig, N. D. Lybarger, R. Viger, D. Lesmes, K. Skalak, J. Brakebill, D. Cline, K. Dunne, K. Rasmussen, and G. Miguez-Macho. 2023. CONUS404: The NCAR–USGS 4-Km Long-Term Regional Hydroclimate Reanalysis over the CONUS. *Bulletin of the American Meteorological Society* 104, 8 (2023), E1382–E1408. doi:10.1175/BAMS-D-21-0326.1

[26] Malcolm J. Roberts, Kevin A. Reed, Qing Bao, Joseph J. Barsugli, Suzana J. Camargo, et al. 2025. High-Resolution Model Intercomparison Project phase 2 (HighResMIP2) towards CMIP7. *Geoscientific Model Development* 18, 4 (March 2025), 1307–1332. doi:10.5194/gmd-18-1307-2025

[27] Masaki Satoh, Bjorn Stevens, Falko Judt, Marat Khairoutdinov, Shian-Jiann Lin, William M. Putman, and Peter Düben. 2019. Global Cloud-Resolving Models. *Current Climate Change Reports* 5, 3 (2019), 172–184. doi:10.1007/s40641-019-00131-0

[28] 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. 2020. Kilometer-Scale Climate Models: Prospects and Challenges. *Bulletin of the American Meteorological Society* 101, 5 (2020), E567–E587. doi:10.1175/BAMS-D-18-0167.1

[29] H. Segura, X. Pedruzo-Bagazgoitia, P. Weiss, S. K. Müller, T. Rackow, J. Lee, E. Dolores-Tesillos, and et al. 2025. nextGEMS: entering the era of kilometer-scale Earth system modeling. *EGUsphere* 2025 (2025), 1–39. doi:10.5194/egusphere-2025-509

[30] Tiffany A. Shaw and Bjorn Stevens. 2025. The Other Climate Crisis. *Nature* 639, 8056 (March 2025), 877–887. doi:10.1038/s41586-025-08680-1

[31] J. Shukla, T. N. Palmer, R. Hagedorn, B. Hoskins, J. Kinter, J. Marotzke, M. Miller, and J. Slingo. 2010. Toward a New Generation of World Climate Research and Computing Facilities. *Bulletin of the American Meteorological Society* 91, 10 (2010), 1407–1412. doi:10.1175/2010BAMS2900.1

[32] Julia Slingo, Paul Bates, Peter Bauer, Stephen Belcher, Tim Palmer, Graeme Stephens, Bjorn Stevens, Thomas Stocker, and Georg Teutsch. 2022. Ambitious partnership needed for reliable climate prediction. *Nature Climate Change* 12, 6 (June 2022), 499–503. doi:10.1038/s41558-022-01384-8

[33] Mark Taylor, Peter M. Caldwell, Luca Bertagna, Conrad Clevenger, Aaron Donahue, et al. 2023. The Simple Cloud-Resolving E3SM Atmosphere Model Running on the Frontier Exascale System. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis* (Denver, CO, USA) *(SC '23)*. Association for Computing Machinery, New York, NY, USA, Article 7, 11 pages. doi:10.1145/3581784.3627044

[34] H. Tomita, H. Miura, S. Iga, T. Nasuno, and M. Satoh. 2005. A Global Cloud-Resolving Simulation: Preliminary Results from an Aqua Planet Experiment. *Geophysical Research Letters* 32, 8 (2005). doi:10.1029/2005GL022459

[35] Gregory L. Wagner, Simone Silvestri, Navid C. Constantinou, Ali Ramadhan, Jean-Michel Campin, Chris Hill, Tomas Chor, Jago Strong-Wright, Xin Kai Lee, Francis Poulin, Andre Souza, Keaton J. Burns, John Marshall, and Raffaele Ferrari. 2025. High-level, high-resolution ocean modeling at all scales with Oceananigans. arXiv:2502.14148 [physics.ao-ph] https://arxiv.org/abs/2502.14148

[36] Arim Yoon and Cathy Hohenegger. [n. d.]. Muted Amazon Rainfall Response to Deforestation in a Global Storm-Resolving Model. 52, 4 ([n. d.]), e2024GL110503. doi:10.1029/2024GL110503