Available online at www.prace-ri.eu

Partnership for Advanced Computing in Europe

Scaling Coupled Climate Models to Exascale: OpenACC-enabled EC-Earth3 Earth System Model

P. Nolan[a], A. McKinstry [a]

[a]*Irish Centre for High-End Computing (ICHEC), Ireland.*

Abstract

Climate change due to increasing anthropogenic greenhouse gases and land surface change is currently one of the most relevant environmental concerns. It threatens ecosystems and human societies. However, its impact on the economy and our living standards depends largely on our ability to anticipate its effects and take appropriate action. Earth System Models (ESMs), such as EC-Earth, can be used to provide society with information on the future climate. EC-Earth3 generates reliable predictions and projections of global climate change, which are a prerequisite to support the development of national adaptation and mitigation strategies.

This project investigates methods to enhance the parallel capabilities of EC-Earth3 by offloading bottleneck routines to GPUs and Intel Xeon Phi coprocessors. To gain a full understanding of climate change at a regional scale will require EC-Earth3 to be run at a much higher spatial resolution (T3999 ~5km) than is currently feasible. It is envisaged that the work outlined in this project will provide climate scientists with valuable data for simulations planned for future exascale systems.

1. Summary of the Project

The goals of this project are to:

(i) Highlight bottlenecks of the EC-Earth3 earth system model.
(ii) Port EC-Earth3 to new hardware "accelerators" such as general-purpose Graphics Processing Units (GPUs) and the Intel Xeon Phi coprocessor.
(iii) Enhance the parallel capabilities of EC-Earth3 by offloading the corresponding bottleneck routines to GPUs and Intel Xeon Phi coprocessors.
(iv) Assist in addressing the challenges of porting EC-Earth3 to new generation of massively parallel systems composed of millions of heterogeneous cores which will provide multi-Petaflop performances in the next few years and Exaflop performances in 2020.

Goal (i) is fully complete; EC-Earth3 was implemented and scale-tested on the Hermit PRACE Tier-0 system. Bottlenecks were highlighted. Results are presented in section 2.

Goals (ii) and (iii) are partially complete; OpenACC directives were successfully used to offload certain bottleneck routines of EC-Earth3 to GPUs and those sections did achieve some encouraging acceleration. Preliminary results are presented in section 3.3.

1.2. EC-Earth

EC-Earth3 is developed by the EC-Earth consortium [1], gathering a number of national weather services and universities from currently 11 countries in Europe. EC-Earth3 component models are IFS for the atmosphere, NEMO for the ocean, and LIM for the sea-ice, coupled through OASIS3. More components and plans for incorporation are under development. EC-Earth3 current users include the Royal Netherlands Meteorological

Institute (KNMI), the Swedish Meteorological and Hydrological Institute (SMHI), Met Éireann Ireland, the Danish Meteorological Institute (DMI), Meteorologisk Institutt Norway and ETH Zürich.

EC-Earth3 is used in coordinated model intercomparison projects [2] (e.g. CMIP5 and the upcoming CMIP6) to make projections and predictions of near-term and end-of-the-century climate change and variability. The next round of IPCC simulations will address both long-term centennial to millennial climate change, looking in particular for polar ice cap and sea level dynamics, and climate evolution in the next decades.

To gain a full understanding of climate change at a regional scale will require EC-Earth3 to be run at a much higher spatial resolution (T3999 ~5km) than is currently feasible. Although long multi ensemble climate simulations at this resolution are currently not possible, it is envisaged that the work outlined in this project will provide climate scientists with valuable data for simulations planned for future exascale systems.

1.3 Exascale

The sensitivity to the horizontal resolution of the climate, anthropogenic climate change, and seasonal predictive skill of the ECMWF model (IFS) has been studied as part of Project Athena [3]. The resolutions considered in Project Athena for the IFS model are T159 (126 km), T511 (39 km), T1279 (16 km), and T2047 (10 km). Analysis of the Athena integrations indicates the major positive impact of high-resolution on the representation of mean features of climate and its variability. In particular, the simulations have a better representation of blocking in the extratropics and trade winds in the tropics, better spatial and temporal distribution of snow, and better seasonal forecast skill, at high resolution. Several interesting features of climate change, such as annual mean warming over Western Europe, as well as fine-scale features such as changes in alpine snowpack, were also substantially different at high resolution compared to lower resolution.

Climate change projections are subject to uncertainty which limits their utility. To address this issue, a Multi-Model Ensemble (MME) approach is used to simulate climate change. The ensemble method uses multiple Earth System simulations (e.g. EC-Earth3), to provide climate change projections. Through the MME approach, the uncertainty in the EC-Earth3 projections can be quantified, enabling us to estimate the probability density function of predicted changes, and providing a measure of confidence in the predictions. A large ensemble of very high resolution (<10km) multi-decadal EC-Earth3 simulations will require the use of exascale computing resources.

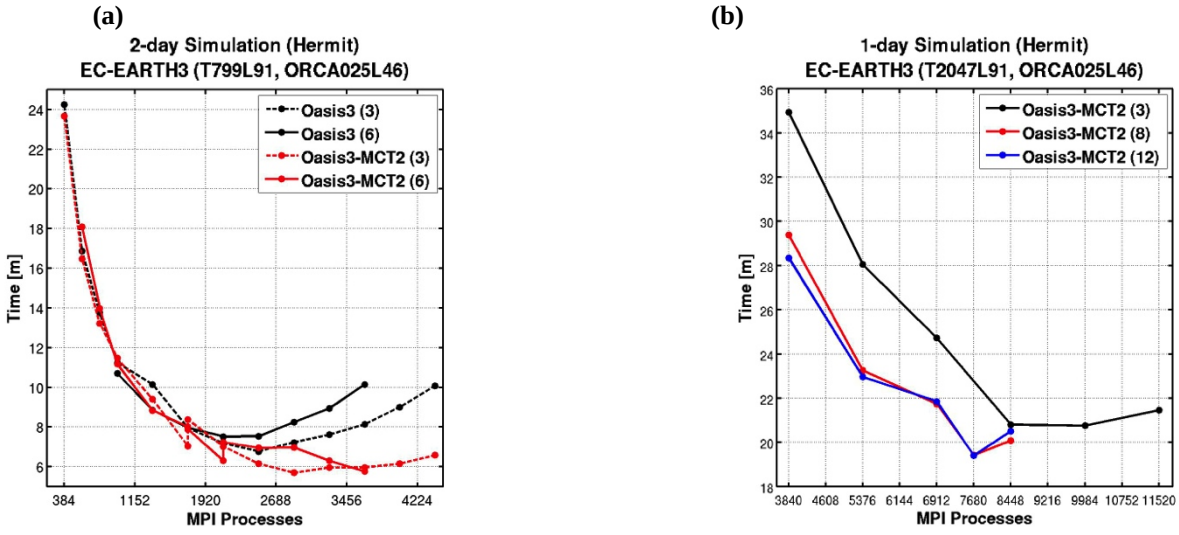

Figure 1. (a) *T799 EC-Earth3 Scaling Results on Hermit using oasis3 (black lines) and oasis3-mct2.0 (red lines). The dashed lines show scaling results using a total processes per NEMO process ratio of 3. Similarly, the full lines present results using a ratio of 6.* ***(b)*** *T2047 EC-Earth3 Scaling Results on Hermit using oasis3-mct2. Results are presented using a total processes per NEMO process ratio of 3, 8 and 12.*

2. Scale Testing and Bottlenecks

2.1. Implement and scale-test EC-Earth3 with Oasis3-MCT2.0 (in place of Oasis3)

EC-Earth3 was implemented and run on the Tier-0 PRACE system, Hermit. To improve the scalability of EC-Earth3, the Oasis3 coupler was updated to Oasis3-MCT2.0. The new setup was validated by running simulations at high resolution and comparing the results with EC-Earth3-Oasis3 output. Differences were found to be insignificant. Scaling results for a 2-day high-resolution T799 simulation are presented in Figure 1a. Oasis3-MCT2.0 shows a clear speedup over Oasis3 for large core counts. The T799 simulations ran using 4500 cores and scaled well to ~2000 cores (Figure 1a).

2.2. Simulations of EC-Earth3 at very high resolution

An investigation was made on the scalability of EC-Earth3 at very high resolution (T1279 & T2047). The simulations ran using 11500 cores and scaled well to ~8000 cores (Figure 1b). Note that the simulation period is one day. To produce a multi-decadal ensemble of such high resolution simulations will require exascale systems and advanced accelerating methods such as offloading bottleneck routines to GPUs and Intel Xeon Phi coprocessors.

2.3 In-depth profiling on Hermit to identify routines suitable for GPU acceleration

EC-Earth3 (implemented with OASIS3-MCT2) T799 and T1279 resolution runs were profiled, using CrayPat, to highlight bottlenecks at low, medium and high core counts. Specifically, the T799 simulations were profiled using 1536, 3840 and 4608 cores while the T1279 simulations were profiled using 2304, 3840 and 5376 cores. The CrayPat performance analysis tool is the primary high-level tool for identifying opportunities for optimization on Cray X1 systems. CrayPat provides access to a wide variety of experiments and reports that measure how fast the profiled code is running and where the performance bottlenecks lie. Results for resolution T799, using 1536 MPI processes, are presented in Figure 2. Note that, since a large number of cores are used, a large fraction of runtime is spent on MPI overhead.

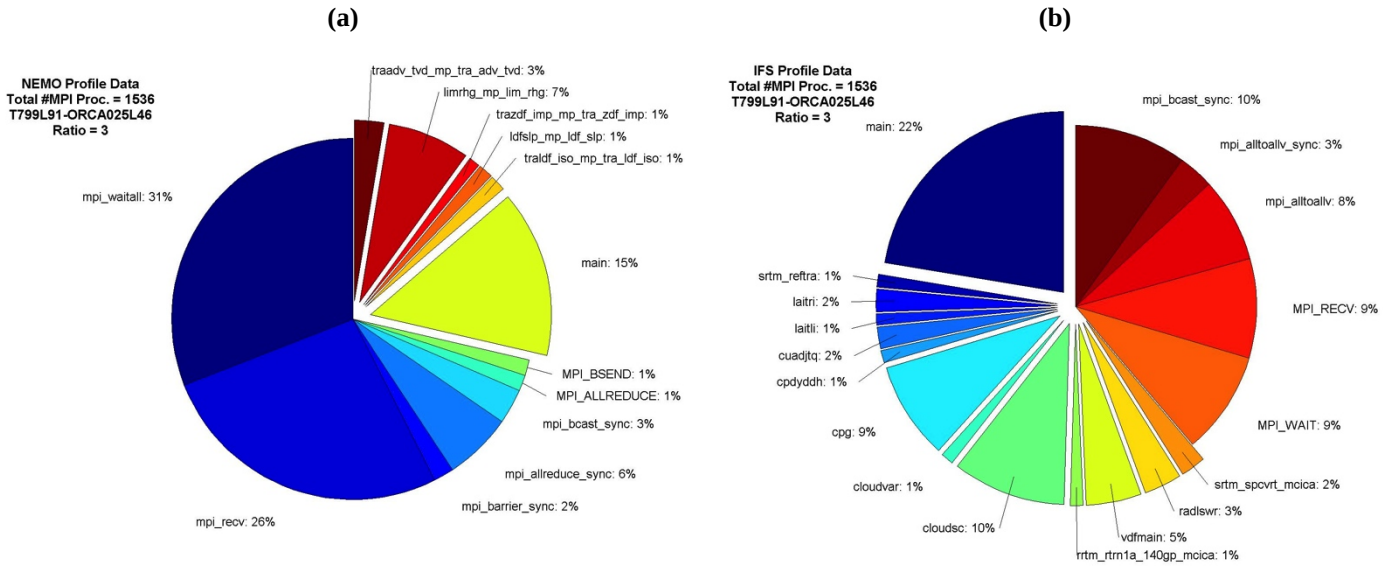

Figure 2. *Profiling of EC-Earth3 (T799L91-ORCA025L46). The simulation was run using 1536 cores. Routines are presented only of they account for greater than 1% of total run time.* ***(a)*** *NEMO Routines,* ***(b)*** *IFS Routines.*

The profiling data were analysed to determine bottleneck routines common to the profiling simulations described above. This analysis highlighted routines such as the traadv.f90, trazdf_imp.F90 & limrhg.F90 for NEMO and cloudsc.F90, cpg.F90 & laitri.F90 for IFS. An additional analysis, using loop-profiling tools, was carried out to further highlight suitable routines for offloading to GPUs. This analysis involved compiling and running EC-Earth3 using the Intel compilers with the following compile options:

```
mpiifort -profile-loops=all -profile-loops-report=1
```

This option inserts instrumentation calls at a function's entry and exit points. When the instrumented application is run, output files are generated containing loop-timing statistics. In practice, routines were chosen to offload to GPUs which contained large “do” loops and a lot of calculations during each trip through the loop. In contrast, while routines with a lot of branches (“if” conditions) or short loops with little computational work were deemed not suitable for GPU offloading.

3.1 Porting EC-Earth3 routines to GPU accelerators

There are two main options for porting a code like EC-Earth3 to GPUs: one is to translate it to C, and then to CUDA (or OpenCL); the other is to insert accelerator directives into the source, similar to OpenMP directives. There is now a standard for accelerator directives called OpenACC (see *www.openacc.org* for details). The main drawback to using directives is that they are more restrictive than CUDA and do not support the full CUDA feature-set. Moreover, currently only 3 compilers support such directives, namely: PGI, CAPS/HMPP, and Cray. The main advantages of using OpenACC directives are

- They are very straightforward to use.
- Directives can be added incrementally, one loop or one subroutine at a time. So it should be possible to make, and evaluate, initial progress relatively quickly.
- OpenACC directives are emerging as the standard for GPUs, and in future may well merge with OpenMP into a single standard directives package. Intel has its own set of directives to support its Xeon Phi coprocessor, and eventually these too may be absorbed into a single standard with OpenACC and OpenMP.

```
!$acc kernels
  DO jk = 1, jpkm1
     DO jj = 1, jpj
        DO ji = 1, jpi
           zt = pts (ji,jj,jk,jp_tem)
           zs = pts (ji,jj,jk,jp_sal)
           zh = gdept(ji,jj,jk) ! depth
           zsr= zws (ji,jj,jk) ! square root salinity
           zr1= ( ( ( ( 6.536332e-9_wp *zt - 1.120083e-6_wp )*zt + 1.001685e-4_wp )*zt &
                & -9.095290e-3_wp )*zt + 6.793952e-2_wp )*zt + 999.842594_wp
           zr2= ( ( ( 5.3875e-9_wp*zt-8.2467e-7_wp ) *zt+7.6438e-5_wp ) *zt &
                & -4.0899e-3_wp ) *zt+0.824493_wp
           zr3= ( -1.6546e-6_wp*zt+1.0227e-4_wp ) *zt-5.72466e-3_wp
           zr4= 4.8314e-4_wp
           zrhop= ( zr4*zs + zr3*zsr + zr2 ) *zs + zr1
           ze = ( -3.508914e-8_wp*zt-1.248266e-8_wp ) *zt-2.595994e-6_wp
           zbw= ( 1.296821e-6_wp*zt-5.782165e-9_wp ) *zt+1.045941e-4_wp
           zb = zbw + ze * zs
           zd = -2.042967e-2_wp
           zc = (-7.267926e-5_wp*zt+2.598241e-3_wp ) *zt+0.1571896_wp
           zaw= ( ( 5.939910e-6_wp*zt+2.512549e-3_wp ) *zt-0.1028859_wp ) *zt - 4.721788_wp
           za = ( zd*zsr + zc ) *zs + zaw
           zb1= (-0.1909078_wp*zt+7.390729_wp ) *zt-55.87545_wp
           za1= ( ( 2.326469e-3_wp*zt+1.553190_wp) *zt-65.00517_wp ) *zt+1044.077_wp
           zkw= ( ( (-1.361629e-4_wp*zt-1.852732e-2_wp ) *zt-30.41638_wp ) * &
                zt + 2098.925_wp ) *zt+190925.6_wp
           zk0= ( zb1*zsr + za1 )*zs + zkw
           prd(ji,jj,jk) = ( zrhop / ( 1.0_wp - zh / ( zk0 - zh * ( za - zh * zb ) ) ) &
                & - rau0 ) * zrau0r * tmask(ji,jj,jk)
        END DO
     END DO
  END DO
!$acc end kernels
```

Figure 3. *A simple example of offloading loops to the GPU using OpenACC directives. This snippet of code is from the NEMO “eosbn2.F90” routine.*

3.2 Porting EC-Earth3 routines to GPU accelerators using OpenACC directives

For the reasons outlined in the above section, it was decided to use OpenACC directives to offload bottleneck routines of EC-Earth3 to GPU accelerators. Future work will focus on translating the code to CUDA.

EC-Earth3 was compiled and run on the ICHEC system, Stoney using the PGI compilers. Stoney is a Bull Novascale R422-E2 cluster with 64 compute nodes. Each compute node has two 2.8GHz Intel (Nehalem EP) Xeon X5560 quad-core processors and 48GB of RAM. This results in a total of 496 cores and 2976GB of RAM available for jobs. In addition, 24 of the compute nodes reserved for GPGPU computing. The nodes have two NVIDIA Tesla M2090 cards installed, with each card providing 512 GPU cores, 6GB of local GDDR5 memory and a theoretical peak double-precision performance of 665GFlops.

The following flags were used to compile EC-Earth3:

```
mpif90 –O1 -acclibs -ta=nvidia,cuda4.2,fastmath,time -Mipa=inline -
                                Minfo=accel
```

Once a working version of EC-Earth3 was built, it was straightforward to insert OpenACC directives and to accelerate the bottleneck routines identified in Section 2. A code section, or "kernel", suitable for offload to a GPU can be identified with a "!$acc kernels" directive at the beginning of the section and a closing "!$acc end kernels" directive at the end. An example of using simple OpenACC directives to accelerate the NEMO routine "eosbn2.F90" is presented in Figure 3.

Given that data movement between the host and device over the PCIe bus is relatively slow, one basic principle of programming for GPUs is to minimize such traffic. To this end, clauses can be added to the basic directives to identify variables that need to be transferred in one direction or another between the host and the device, or are not required to be transferred at all (i.e., are purely local to the device). More complicated OpenACC directives were used to accelerate the NEMO routine "trazdf_imp.F90" while limiting the data transfer between host and device. A snippet of the code is presented in Figure 4.

```
! Copy required data to device outside the tracer do loop
!$acc data pcopyin(ptb,p2dt(:jpkm1),e3t(2:jpim1,2:jpjm1,1:jpkm1),tmask(2:jpim1,2:jpjm1,:),pta)
  DO jn = 1, kjpt ! tracer loop !
! code cut form here
!.....................
! to here

! Do not copy this data inside the tracer loop as it is already present on the device
!$acc kernels present(ptb(2:jpim1,2:jpjm1,:jpkm1,jn),p2dt(:jpkm1),e3t(2:jpim1,2:jpjm1,1:jpkm1),&
!$acc& tmask(2:jpim1,2:jpjm1,:),pta(2:jpim1,2:jpjm1,:,jn))
     DO jj = 2, jpjm1
        DO ji = 2, jpim1
           ze3tb = ( 1. - r_vvl ) + r_vvl * e3t(ji,jj,1)
           ze3tn = ( 1. - r_vvl ) + r_vvl * e3t(ji,jj,1)
           pta(ji,jj,1,jn) = ze3tb * ptb(ji,jj,1,jn) + p2dt(1) * ze3tn * pta(ji,jj,1,jn)
        END DO
     END DO
     DO jk = 2, jpkm1
        DO jj = 2, jpjm1
           DO ji = 2, jpim1
              ze3tb = ( 1. - r_vvl ) + r_vvl * e3t(ji,jj,jk)
              ze3tn = ( 1. - r_vvl ) + r_vvl * e3t(ji,jj,jk)
              zrhs = ze3tb * ptb(ji,jj,jk,jn) + p2dt(jk) * ze3tn * pta(ji,jj,jk,jn)
              pta(ji,jj,jk,jn) = zrhs - zwi(ji,jj,jk) / zwt(ji,jj,jk-1) * pta(ji,jj,jk-1,jn)
           END DO
        END DO
     END DO
     DO jj = 2, jpjm1
        DO ji = 2, jpim1
           pta(ji,jj,jpkm1,jn) = pta(ji,jj,jpkm1,jn) / zwt(ji,jj,jpkm1) * tmask(ji,jj,jpkm1)
        END DO
     END DO
     DO jk = jpk-2, 1, -1
        DO jj = 2, jpjm1
           DO ji = 2, jpim1
              pta(ji,jj,jk,jn) = ( pta(ji,jj,jk,jn) - zws(ji,jj,jk) * pta(ji,jj,jk+1,jn) ) &
                   & / zwt(ji,jj,jk) * tmask(ji,jj,jk)
           END DO
        END DO
     END DO
!$acc end kernels
  END DO ! end tracer loop !
!$acc update host(pta)  ! Update pta once outside the tracer loop
!$acc end data
```

Figure 4. *A simple example minimizing data transfer when using OpenACC directives. This snippet of code is from the NEMO "trazdf_imp.F90" routine.*

3.3 Preliminary EC-Earth3 GPU accelerating results

3.3.1 NEMO Acceleration

NEMO was run for one month using the 'ORCA1L46' configuration which has a horizontal resolution of approximately 1 degree and 46 ocean levels. Simple OpenACC directives were added to the "eosbn2.F90" and "trazdf_imp.F90" routines as outlined in Figures 3 & 4. These routines were responsible for ~0.6% and 1% respectively of total execution time, when run within EC-Earth3 (T799L91-ORCA025L46) using 1536 MPI processes. These percentages increase when using less MPI processes and when running NEMO in standalone mode. Table 1 shows the performance of NEMO on a single CPU-core and a single GPU. An acceleration of 13 minutes is noted when the two routines are accelerated using GPUs. Typically, climate simulations are run for ~100 years with an ensemble size of over twenty. This corresponds to potential acceleration of over 200 days for a multi-decadal ensemble of EC-Earth3 climate simulations.

	One CPU	One CPU + GPU (trazdf.F90)	One CPU + GPU (eosbn2.F90)
RunTime	2h 51m	2h 43m	2h 46m

Table 1. *Run Times for a one-month NEMO simulation. A clear speedup is noted when the trazdf.F90 and eosbn2.F90 routines are ported to GPUs. In each case, we compare runtimes using a CPU (Xeon X5560) with no GPUs attached to the same CPU chipset with a GPU (NVIDIA Tesla M2090) attached.*

3.3.2 IFS Acceleration

OpenACC directives were added to the main loop in laitri.F90, and also to some loops in rtm_rtrn1a_140gp.F90. Table 2 shows performance of the routines with very small 50 x 50 x 60 point problem size on a single CPU-core and a single GPU. The values in red (10.1s for LAITRI and 3.7s for RRTM_RTRN1A_140GP) are run-times for the sections of those routines offloaded to the GPU. The GPU achieved a speedup factor of about 3 in both cases. It should be noted that this simulation domain is quite small and work remains to be done to achieve acceleration on higher resolution domains.

#	% Time (self)	Cumulated (sec)	Self (sec)	Self (1-CPU, CPU+GPU)	Total	# calls	Routine
1	15.24	108.87	108.87	(33.5,**10.1**)	108.89	59475	LAITRI
2	3.80	206.77	27.18	(12.2,**3.7**)	27.18	325	RRTM_RTRN1A_140GP

Table 2*. DR_HOOK output showing performance of two IFS routines on CPU & GPU. In each case, we compare runtimes using a CPU (Xeon X5560) with no GPUs attached to the same CPU chipset with a GPU (NVIDIA Tesla M2090) attached.*

4. Conclusions

EC-Earth3 was implemented and scale-tested so as to highlight bottleneck routines suitable for offloading to GPU and Intel Xeon Phi accelerators. OpenACC directives were successfully used to offload these routines of EC-Earth3 to GPUs. Preliminary scaling results show promising GPU acceleration.

The results presented in Section 3 show acceleration for two NEMO and two IFS routines. However, a large number of EC-Earth3 routines were unsuccessfully 'accelerated'. This was largely due to the fact that data movement between the host and device over the PCIe bus is relatively slow. We are currently working to minimize this data movement by offloading data only when completely necessary. Another option that can then be used to avoid the transfer becoming a performance bottleneck is to make the transfers *asynchronous.*

Asynchronous transfers, once started, are only required to complete by the time some specified point is reached further along in the algorithm. In this way, either (or both) the host and device can continue calculating while the transfer is taking place. Synchronous transfers, on the other hand, must start and complete before the code can move on to the next instruction.

It is envisaged that, on completion, this project will assist in addressing the challenges of porting EC-Earth3 to new generation of massively parallel systems composed of millions of heterogeneous cores which will provide multi-Petaflop performances in the next few years and Exaflop performances in 2020.

6. References.

[1] http://www.ec-earth.org/

[2] http://pcmdi-cmip.llnl.gov/cmip5/index.html

[3] Jung, T., M. J. Miller, T. N. Palmer, P. Towers, N. Wedi, D. Achuthavarier, J. M. Adams, E. L. Altshuler, B. A. Cash, J. L. Kinter III, L. Marx, C. Stan, K. I. Hodges, 2012: High-Resolution Global Climate Simulations with the ECMWF Model in the Athena Project: Experimental Design, Model Climate and Seasonal Forecast Skill. J. Climate, **25**, 3155-3172.

Acknowledgements

This work was financially supported by the PRACE project funded in part by the EUs 7th Framework Programme (FP7/2007-2013) under grant agreement no. RI-312763.