A Study on Performance and Internal Variability of the RegCM-4.1 Regional Climate Simulation Model

Tamirat Bekele Jimma
Supervisors: Dr.Stefano Cozzini and Dr.Gizaw Mengistu

Addis Ababa University

May 17, 2012

atomictamirat@{yahoo,gmail}.com

1. Introduction
   - High Performance Computing (HPC)
   - Objective and Significance of this study
2. Theoretical backgrounds
   - Modern software re-engineering and benchmarking
   - Metrics used for performance tests
3. Experimental setup
   - Domains, Compilers and HPC infrastructures tested
   - Software and hardware stacks
4. Performance and IV results
   - Performance results
   - Internal Variability results
5. Conclusion
6. Recommendation

Collaborators

Collaborator institutes

This research conducted in-collaboration with the support of two known Italian laboratories, DEMOCRITOS National Simulation Center of IOM - istituto Officina dei Materiali Consiglio Nationale delle Ricerche and The Abdus Salam International Center for Theoretical Physics (ICTP).

ICTP

CNR-IOM Democritos

HPC...

High Performance Computing (HPC):

- is most commonly associated with computing need used for scientific research
- it refers computational system which utilizes supercomputers and aggregated large computer infrastructure
- HPC environment encompasses a collection of powerful:
  - hardware systems
  - software tools
  - programming languages
  - parallel programming paradigms

RegCM regional climate model package:

RegCM has been the first limited area model developed for long term regional climate simulation. It has evolved from:

- the first version developed in the late eighties RegCM-1
- to later versions in the early nineties RegCM-2
- late nineties RegCM-2.5
- in early 2000s RegCM-3
- in 2010 RegCM-4.0
- and in 2011 RegCM-4.1

RegCM becomes a community model, and it was designed for use by a varied community composed by scientists in industrialized countries as well as developing nations.

Statement of problem

Motivations for this work

RegCM package went under a large restructuring phase, therefore,

- there is a need to address the pro and cons of model transition from RegCM-3 to RegCM-4.1 with regard to its computational point of view.
- there were no sufficient benchmarking assessment with respect to compilers, libraries, computational platforms, etc.
- there were also no clear idea about the role of optimizations to improve performance.
- there was also a concern about the absence of the verification and validation procedures for the model.
- and the need to provide some of performance metrics for the RegCM-4.1.1 motivated the work done in this study.

Objective

Objectives of this study are:

- first to make a detailed benchmarking analysis in order to asses which are the best combination of
  - compilers
  - libraries
  - platform to run a simulation.
- second to compute the Internal Variability (IV) for the model
  - to understand better which kind of optimization can be performed without any danger on the code.

Modern software re-engineering and benchmarking

Software re-engineering:

Primary target of software re-engineering is to formulate a re-engineering roadmap towards

- enhancing
- upgrading
- migrating or recovering legacy software packages

Benchmarking:

- is a test, or set of tests, designed to compare the performance of one computer system against the performance of others.
- term is widely used in computer science community to measure performance of hardware and software packages.

Speedup:

Speedup:

Definition

Performance assessment in HPC is one way of understanding the behavior, pro and cons of any hardware or software with regard to its usage.

Among the most simple metric of performance is measuring execution time. A comparison of how fast a parallel program run when compared to its serial program part is called *Speedup*. It is defined as:

$$S_p = \frac{T_s}{T_p}$$

where,

- T_s : – is time taken by a serial program code.
- T_p : – is time taken by a parallel program code.

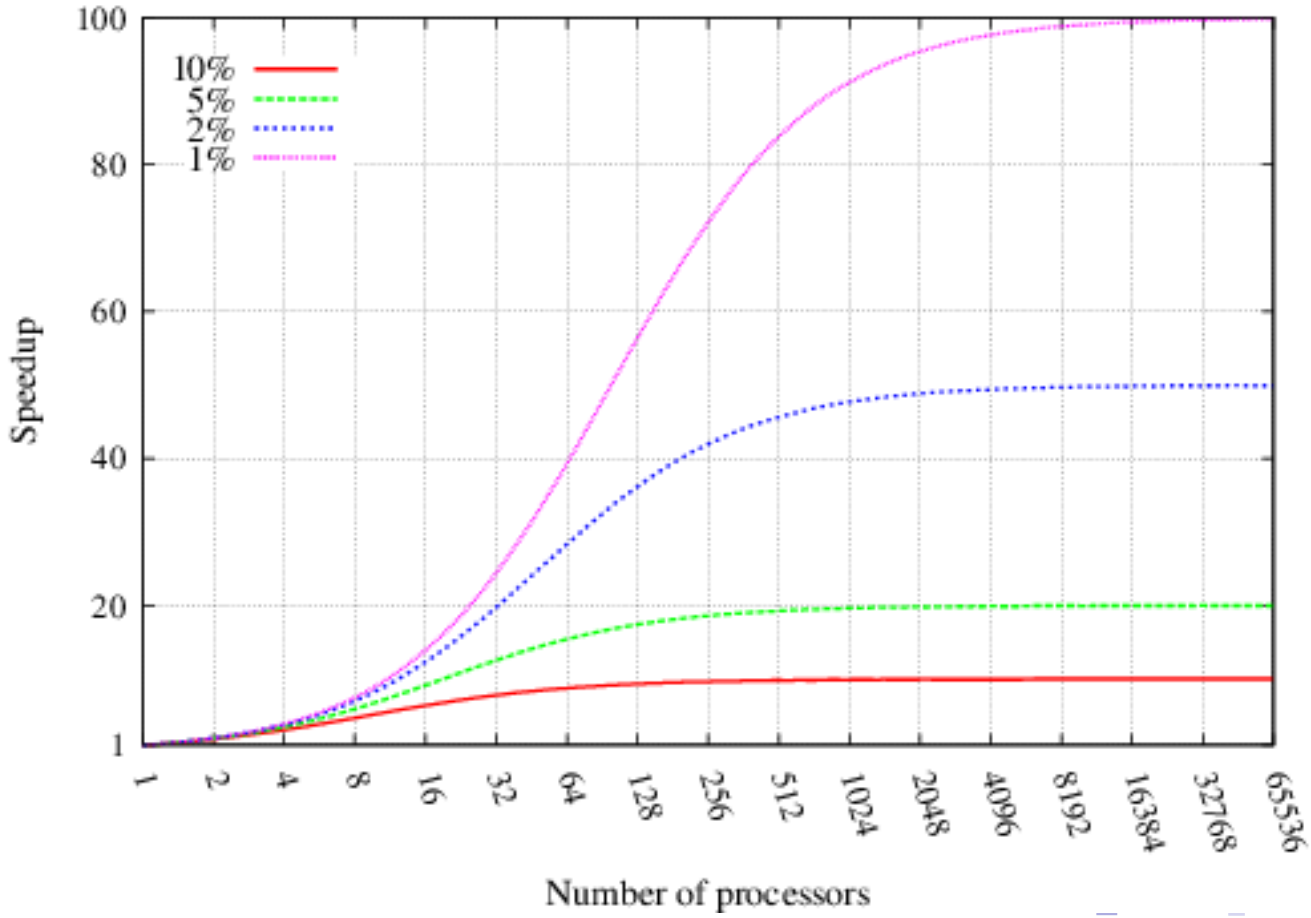
Amdahl's law:

It is one famous formulation to estimate how much speedup one can have for the entire system if a part of the system is made faster.

$$
\begin{aligned}
G &= \frac{t_s + t_{p(single)}}{t_s + \frac{t_{p(single)}}{n}} \\
&= \frac{t_s + t_{p(single)}}{t_s + t_{p(single)} - t_{p(single)}(1 - \frac{1}{n})} \\
&= \frac{1}{1 - f(1 - \frac{1}{n})}, \qquad where \qquad f = \frac{t_{p(single)}}{t_s + t_{p(single)}} \\
&= \frac{1}{(1 - f) + \frac{f}{n}}
\end{aligned}
\tag{1}
$$

Amdahl's law...

Theoretical plot for Amdahl's law for a 10%, 5%, 2% and 1% serial fraction of code in an application.



Efficacy:

Efficacy:

It is measure of power to produce effects, which means it is a gain obtained from adding extra processor out of the cost of adding it into the system.

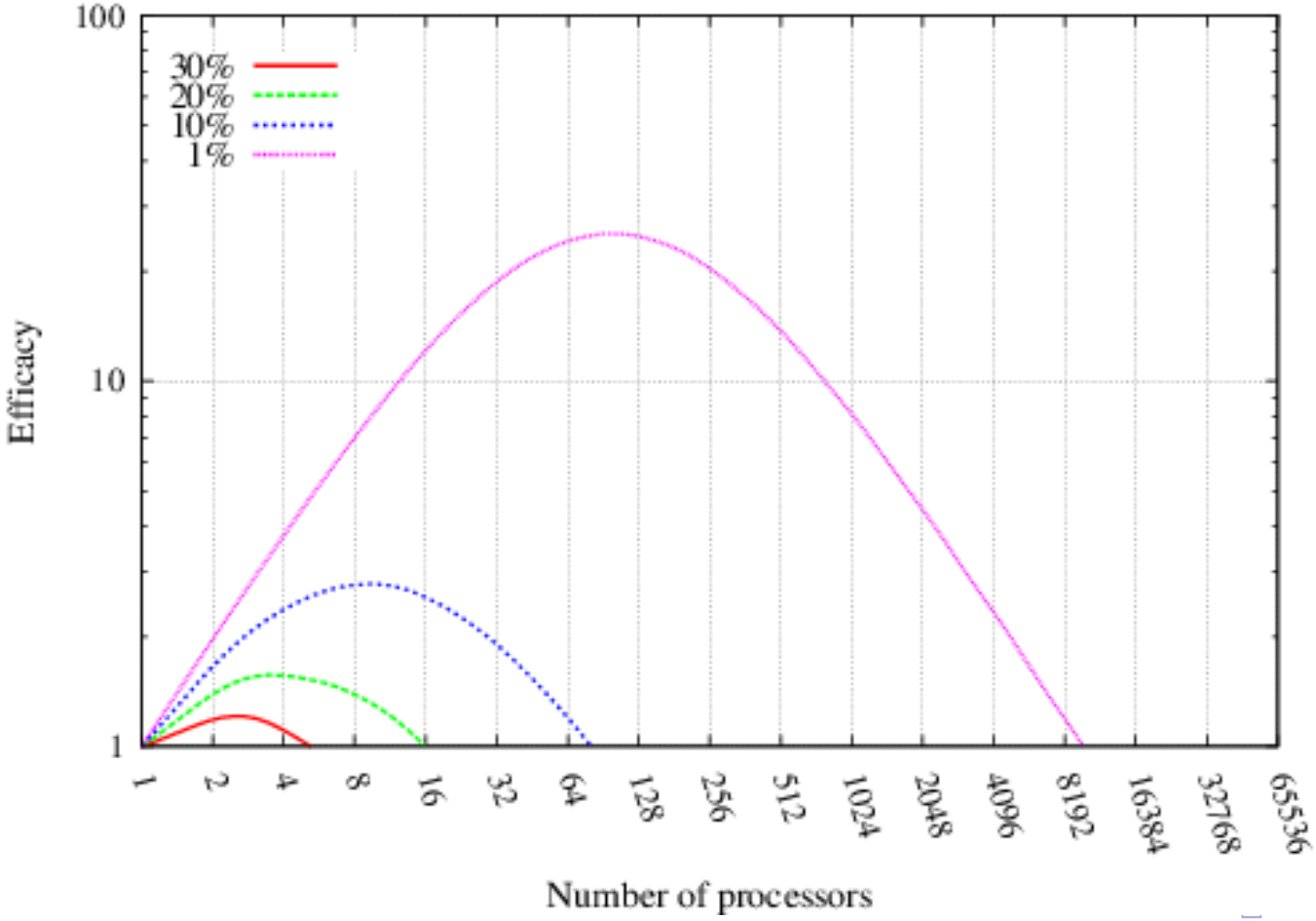
$$\begin{aligned} \Gamma &= S_p \times \eta \\ &= S_p \times \frac{S_p}{p} \\ &= \frac{S_p^2}{p} \end{aligned} \tag{2}$$

where

- S_p : – Speedup
- p : – is number of processors

Efficacy...

Theoretical plot for efficacy using Amdahl's law for 30%, 20%, 10% and 1% serial fraction of code.



Scalability

Definition

Scalability of a scientific code is the measure of the ability of the package to accommodate the increased resources or the enlargement of the input resources.

- If the performance of a software system becomes unacceptable when reaching a certain load level with a given environment, but it cannot be improved even with upgrading or additional hardware, then it is said that the software is not scalable

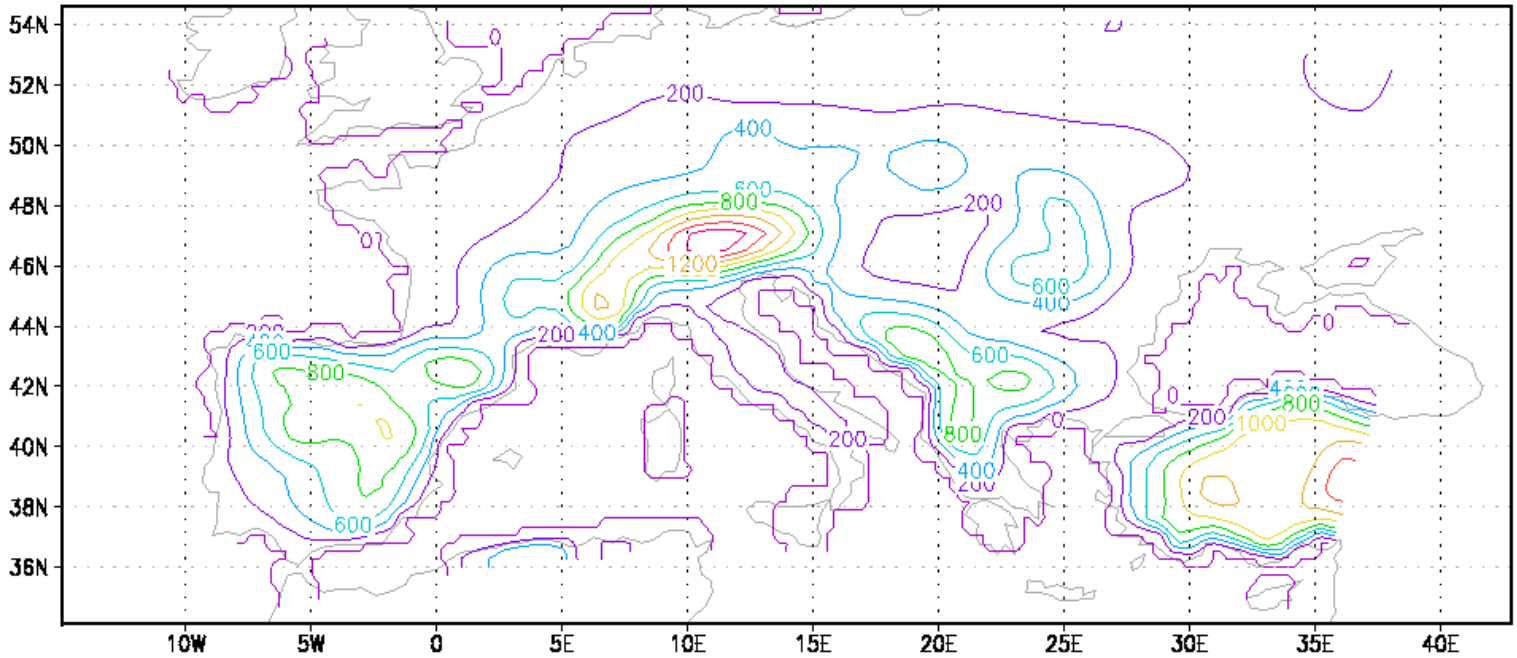
Tested domains, time and global dataset used:

- small European, Ethiopian, big European and African of domain size 34×64, 112×128, 160×192 and 250×256 respectively.
- one month simulation over January 1989 for performance tests and two years January 1990 to December 1991 for IV tests.
- "EIN15" global dataset with $50km$ resolution and 18 vertical levels of a model.
- *svn*[1] trunk version 1928, 1956, 1976 and 2534 have used.
- most of the results have been presented only based on big European domain.

[1] revision control system *see* https://subversion.tigris.org

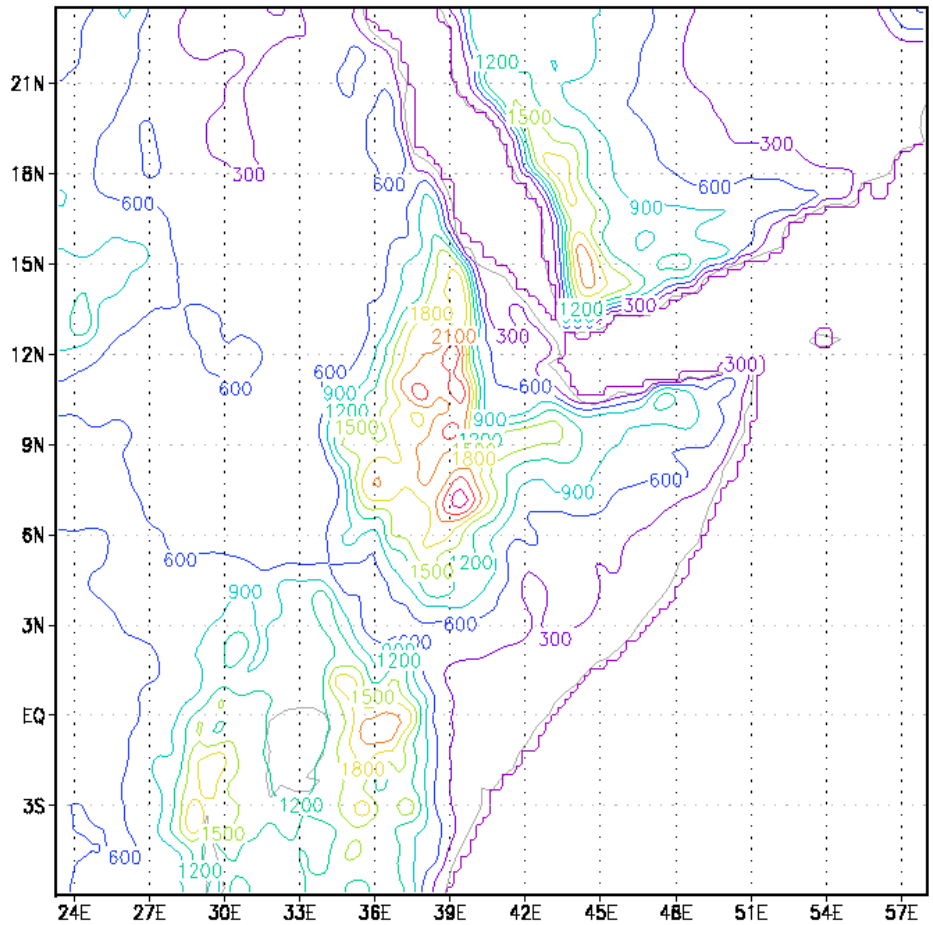
Domains used and topography(m)

Small European domain (34×64)



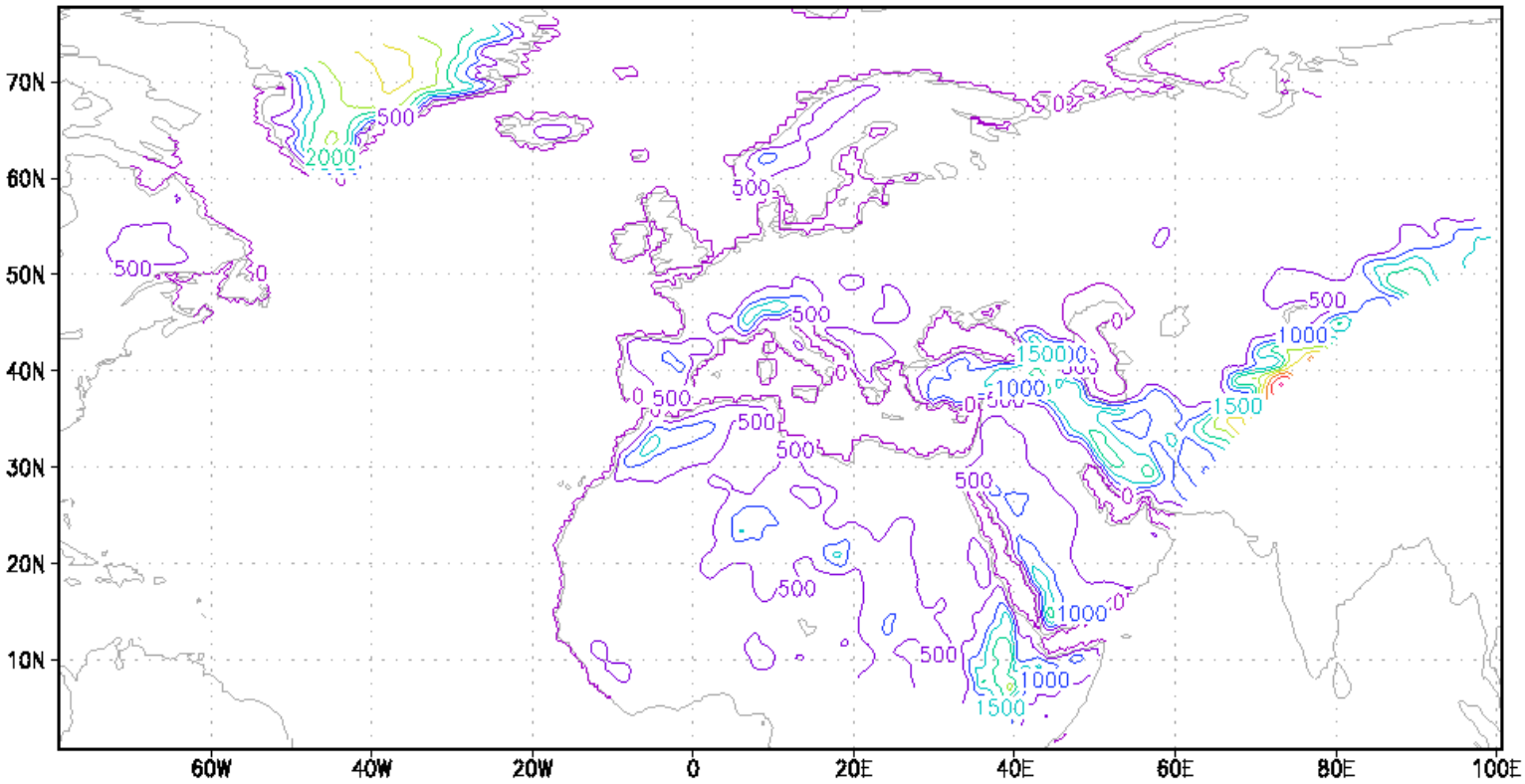
Domains used and topography(m)...

Ethiopian domain (112×128)



Domains used and topography(m)...

Big European domain (160×192)



Compilers and libraries used:

Operating system

- CentOS v5.5 and IBM/AIX

Compilers

- Intel 11.1/12.0
- GNU 4.4.0
- PGI 10.9
- IBM/XLF 13.01

Libraries

- NetCDF 4.1.1/4.1.2/4.1.3
- OpenMPI 1.4.2/1.4.3/1.5.2

Profilers

- IPM/0.983/
- gprof
- pgprof

Compiler's flags used:

Default compiler options provided with the package.

Compilers	F90FLAGS
GNU	-O3 -fconvert=big-endian
PGI	-O3 -byteswapio
IBM/XLF	-O3 -qintlog -qstrict -qmaxmem=-1 -qzerosize
INTEL	-O3 -fp-model precise -convert big-endian -heap-arrays -assume byterecl

Hardware's used:

Computational infrastructures studied in this work.

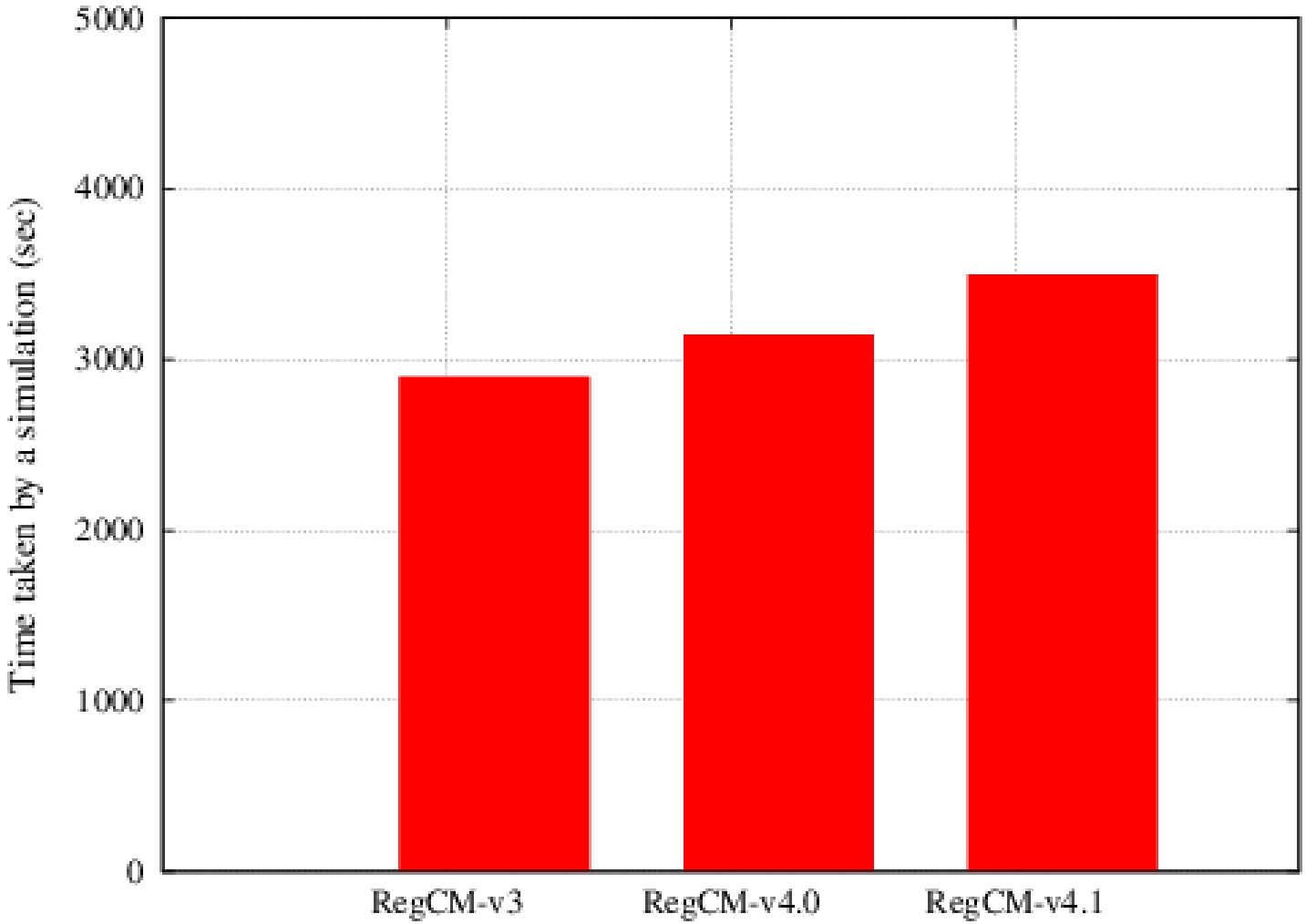
IBM SP6	**ARGO**	**HG1**
IBM Power6 575	Intel E5620 Xeon	Intel E5520 Xeon
16 processors	2 Sockets per node	2 Sockets per node
2 Cores per CPU (32 Cores per node)	4 Cores per CPU (8 Cores per node)	4 Cores per CPU (8 Cores per node)
Clock 4.70 GHz	Clock 2.40 GHz	Clock 2.27 GHz
Infiniband **DDR**	Infiniband **QDR**	Infiniband **DDR**

Performance results.

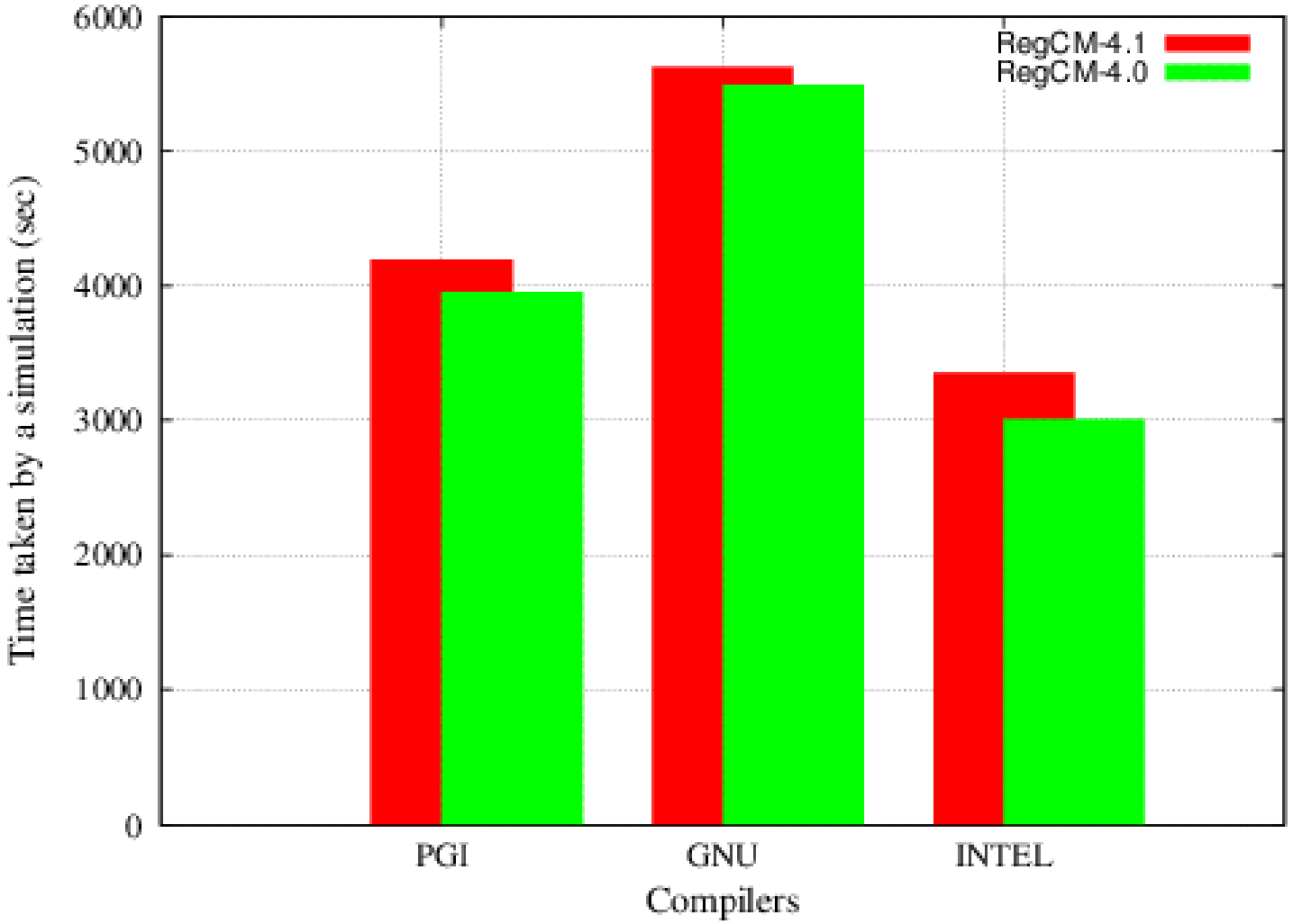
Time taken in seconds by one month simulation in different domains and compilers.

Compilers	European	Ethiopian	European	African
	(34x64)	(112x128)	(160x192)	(250x256)
INTEL	256.27	1676.03	3348.86	10030.78
PGI	330.49	1938.76	4189.08	12630.96
GNU	477.02	2653.21	5617.39	16747.22

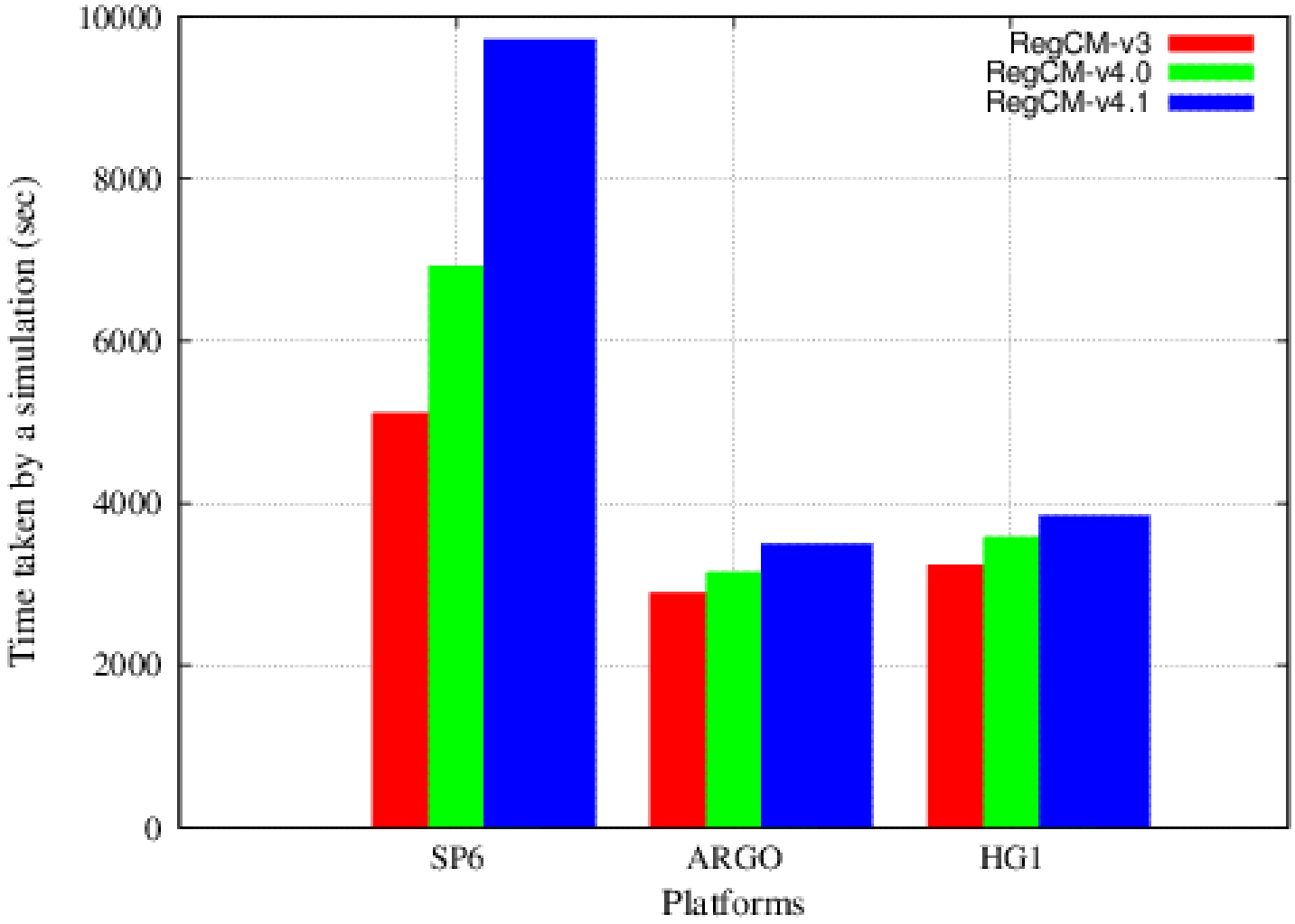
Performance difference between RegCM versions



Performance difference between compilers



Performance difference between platforms

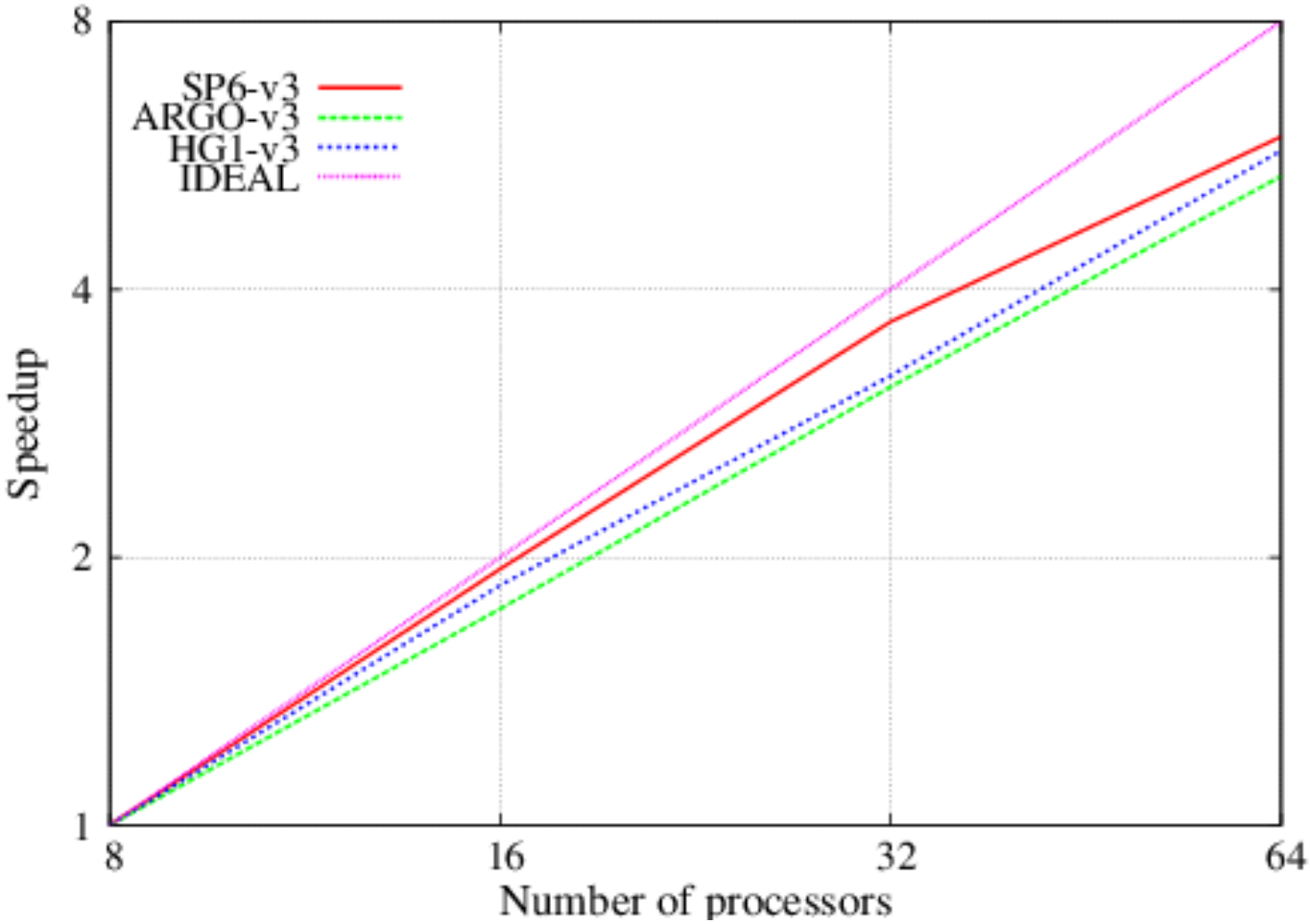


Performance analysis shows:

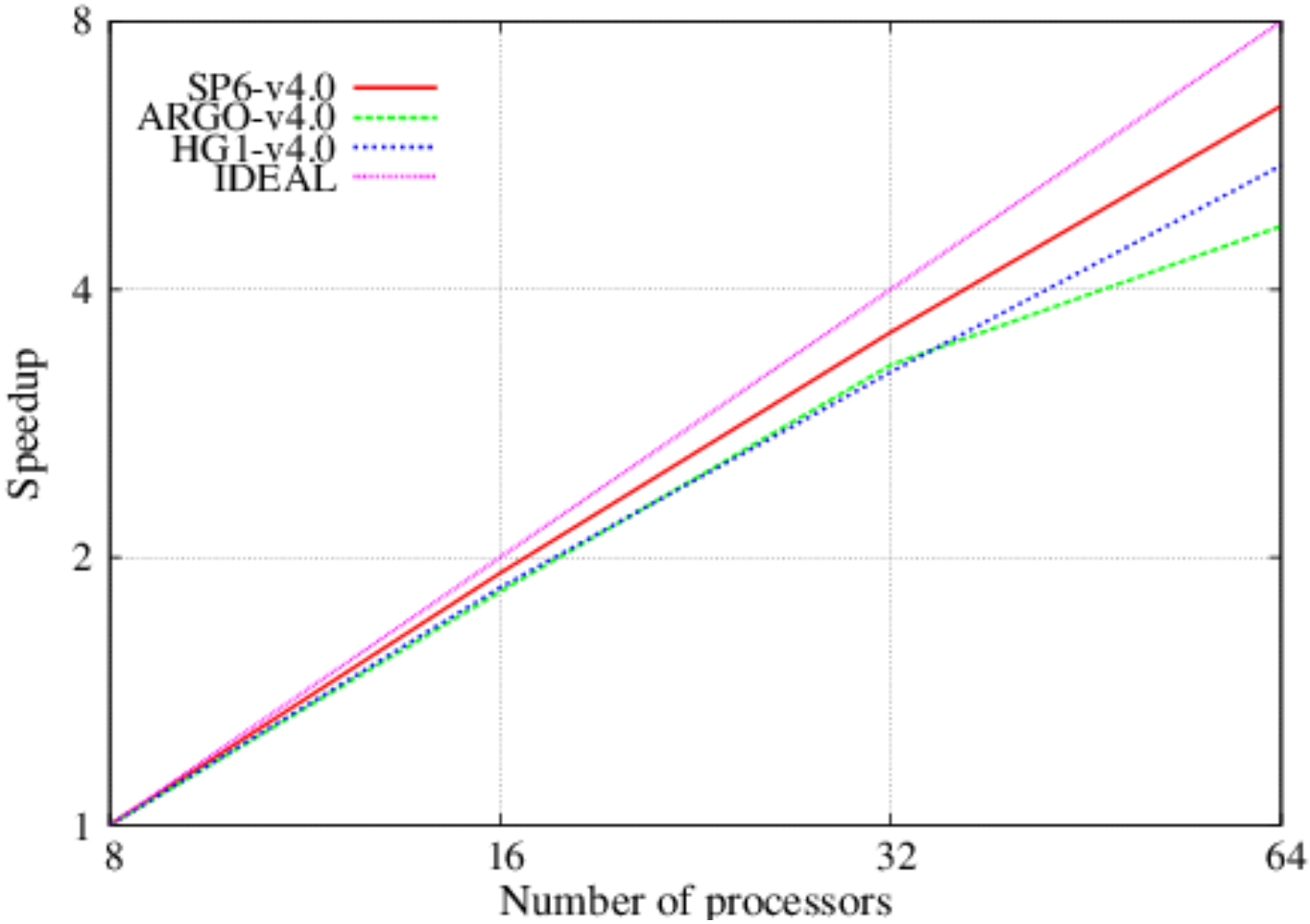
Performance analysis shows:

- RegCM-4.1 takes 600 seconds more than RegCM-3, while RegCM-4.0 takes 350 seconds.
- Intel compiler is 40% faster than the slowest GNU compiler.
- PGI compiler performs 20% slower than the fastest compiler.
- performance among platforms goes up to 64%, code performed best on *argo*, slightly worse on *quicksilver* and definitely worse on *sp6*.

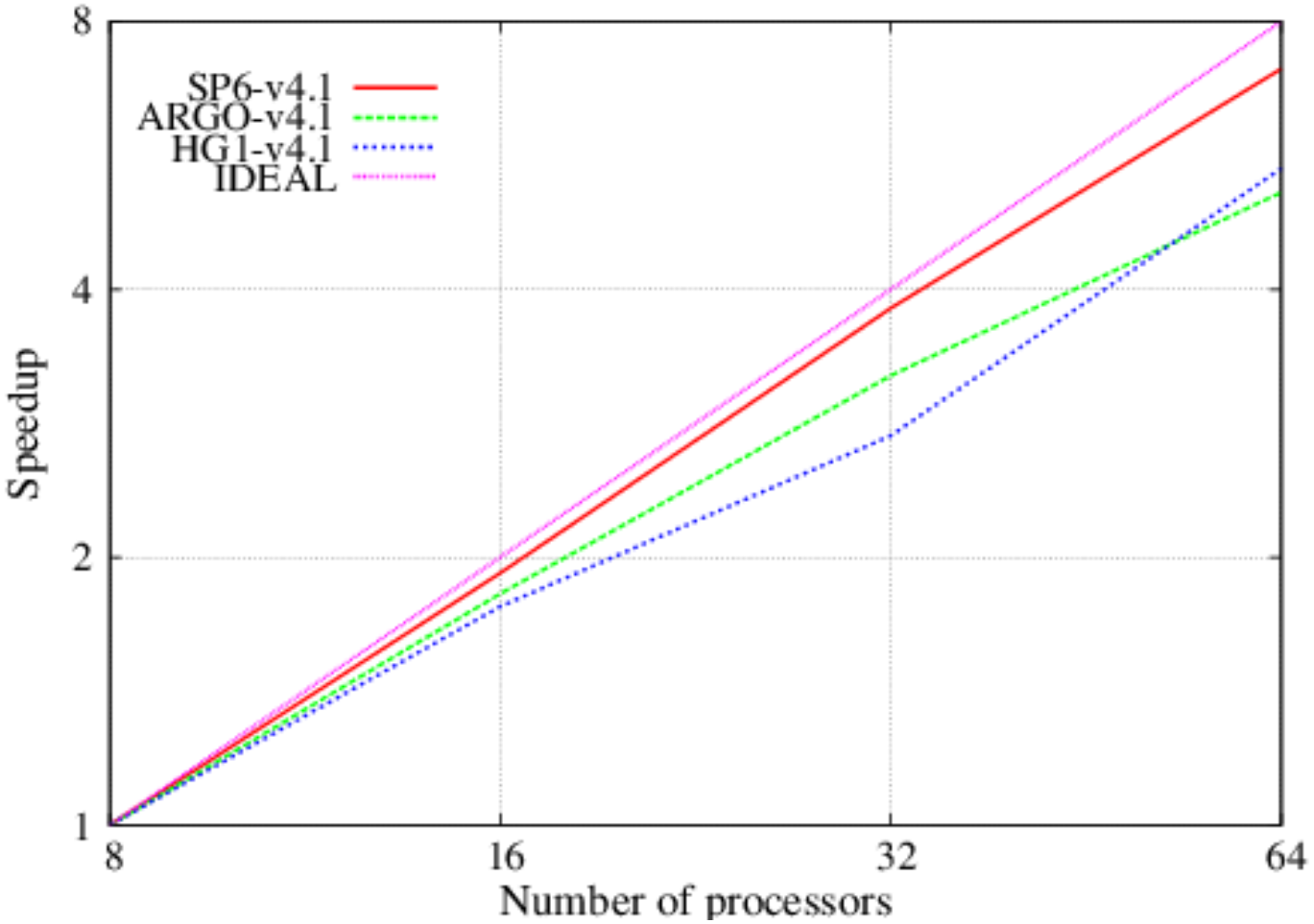
RegCM-3 scalability



RegCM-4.0 scalability



RegCM-4.1 scalability

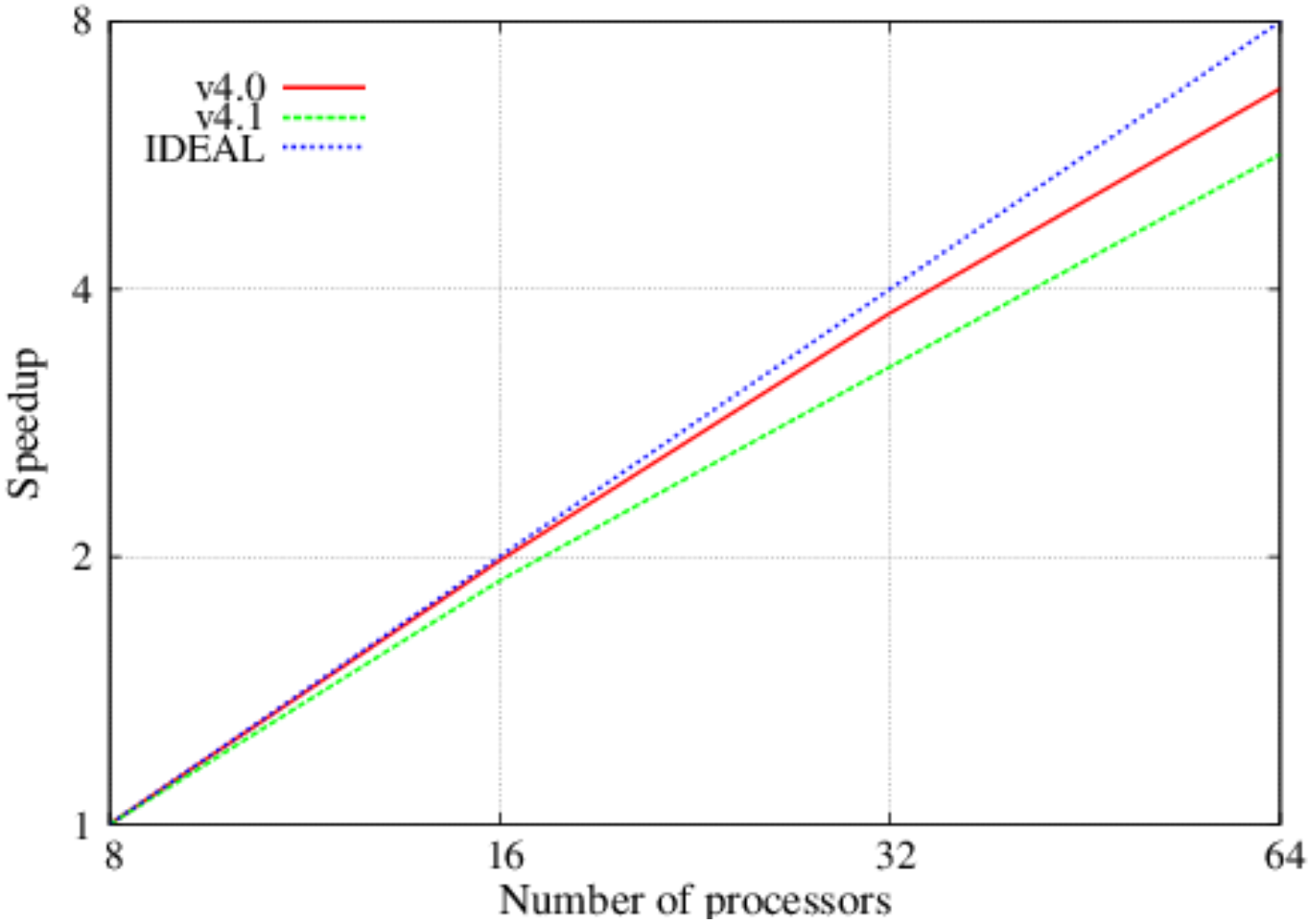


Scalability analysis

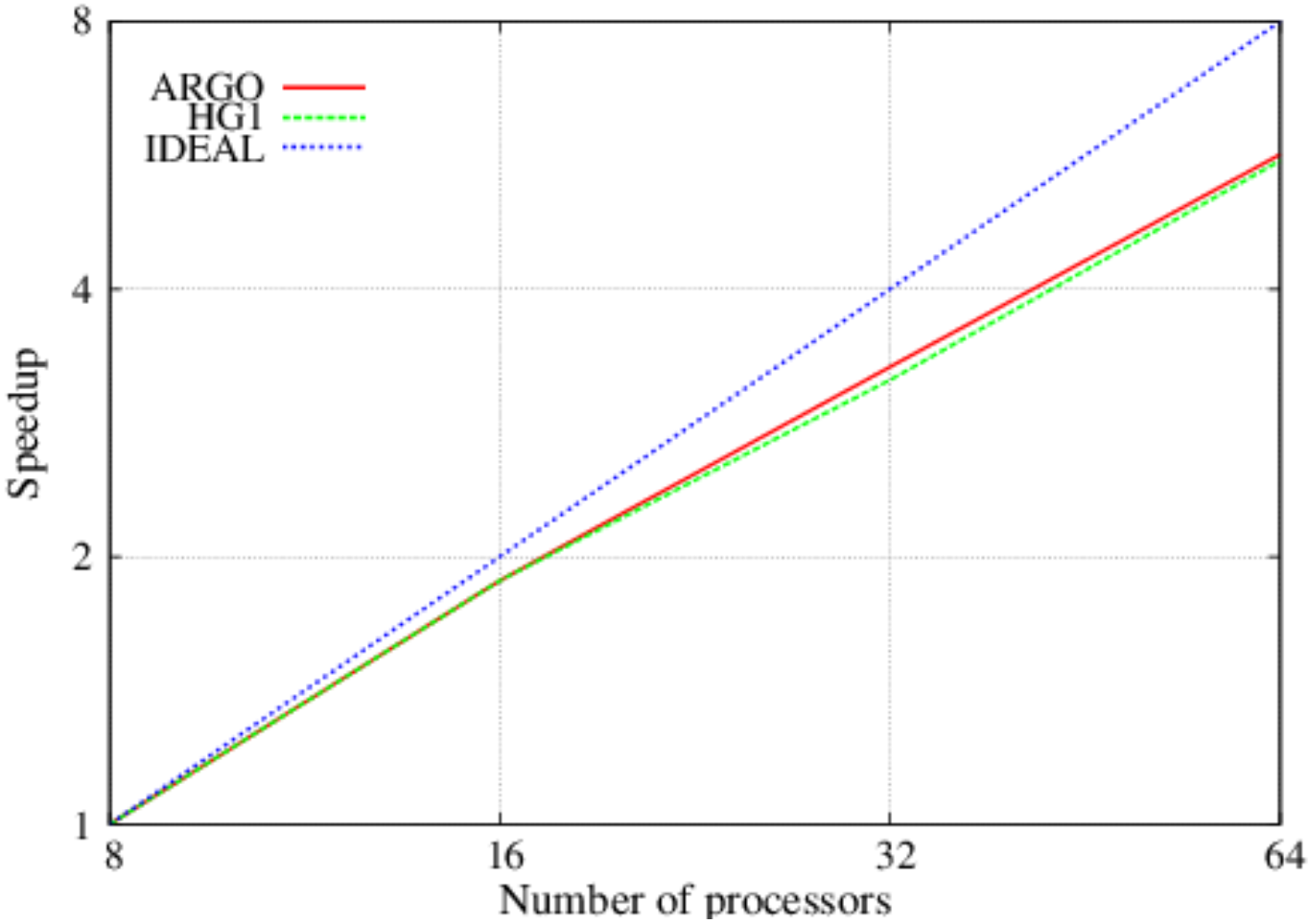
based on results from scalability analysis:

- there is no relevant difference in speedup among the different platforms at our disposal.
- speedup falls beyond optimal when using more than 32 processors and that there is no significant difference among versions in this case.
  - the cause of limitation with regard to scalability relies in the way the code was originally parallelized.
  - the domain under study (160×192) is divided in slices that are becoming too small and therefore communication costs are dominant.

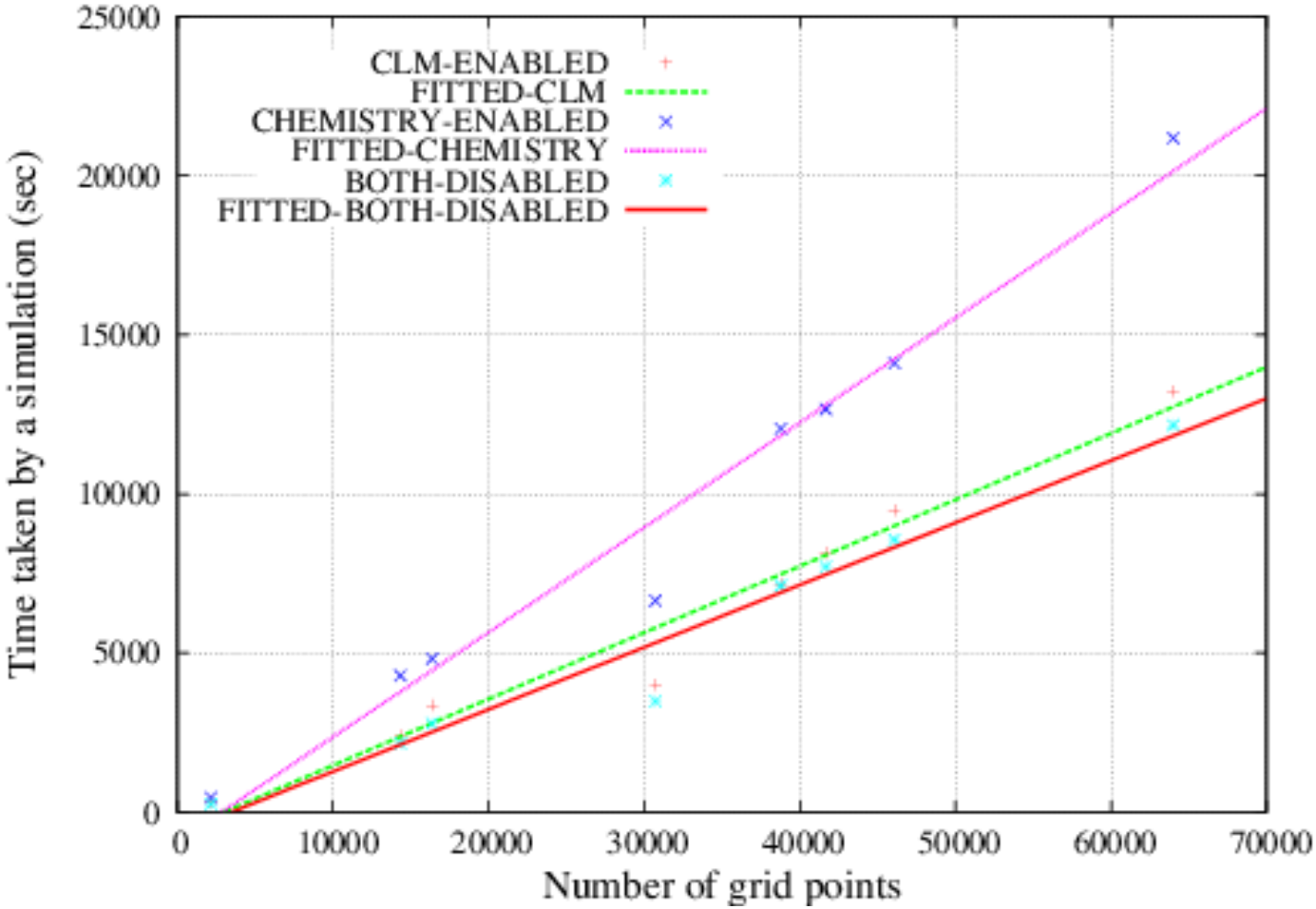
Scalability in African domain (250×256)



Scalability in African domain (250×256)...



Performance cost due to physical parametrizations



Performance cost due to physical parametrizations...

- Time taken by chemistry enabled simulations is significantly higher than CLM enabled and standard simulations. Results show us the performance difference between chemistry enabled and disabled simulation can go up to 43%. CLM enabled and disabled simulation differs in 17%.
- We expect that the effects are additive, so that a CLM+CHEM simulation overhead is the sum of both observed overheads.

Message Passing Interface (MPI) analysis

MPI communication measurement has been done using Integrated Performance Monitoring (IPM)[2] profiler tool.

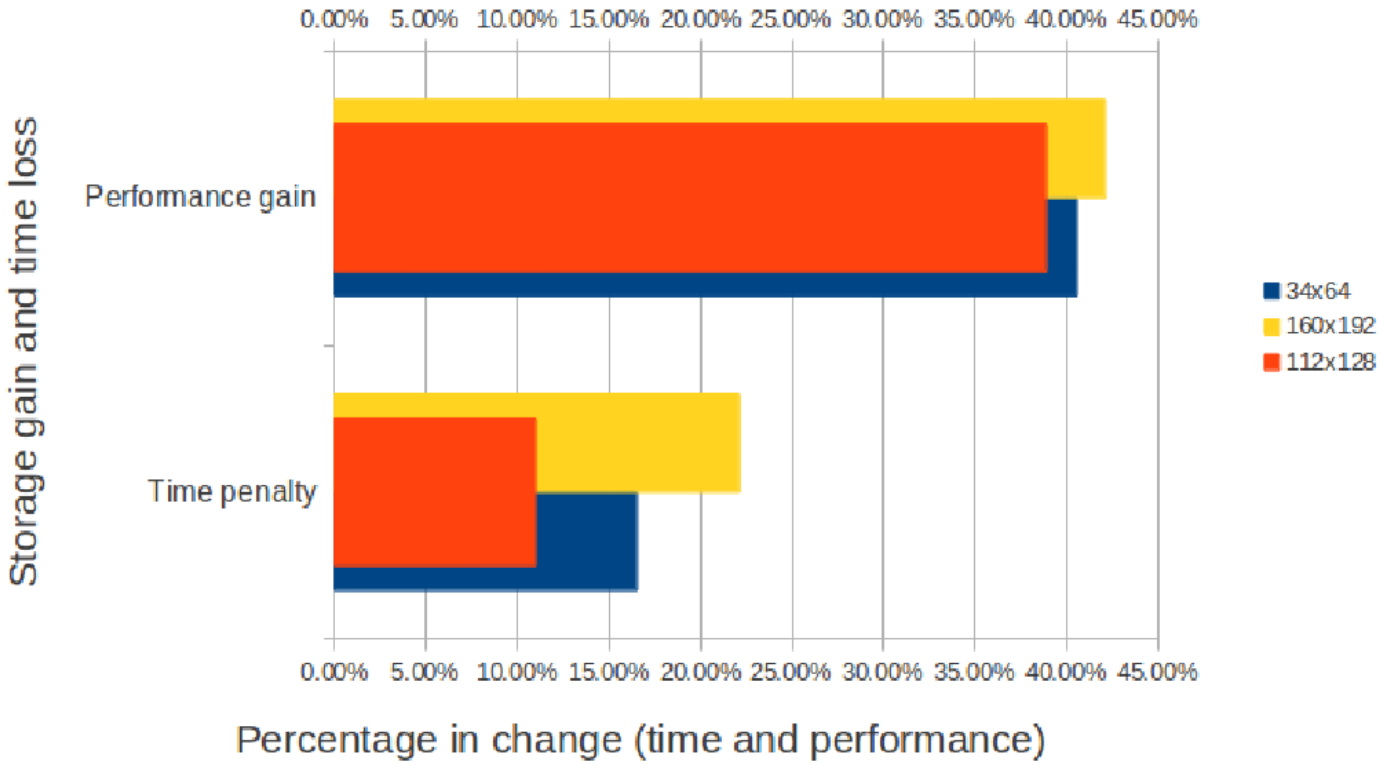
Compilers	RegCM-4.0	RegCM-4.1
GNU	14.47%	13.76%
INTEL	17.81%	17.45%
PGI	14.17%	17.47%

Based on results obtained we can say that there is no significant amount of difference between v4.0 and v4.1 RegCM versions.

[2] http://ipm-hpc.sourceforge.net/

Data compression capability of RegCM-4.1

through the use of NetCDF and HDF5[3] libraries there is a possibility to use data compression for both the input and output of the model.



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

Data compression capability...

Caution

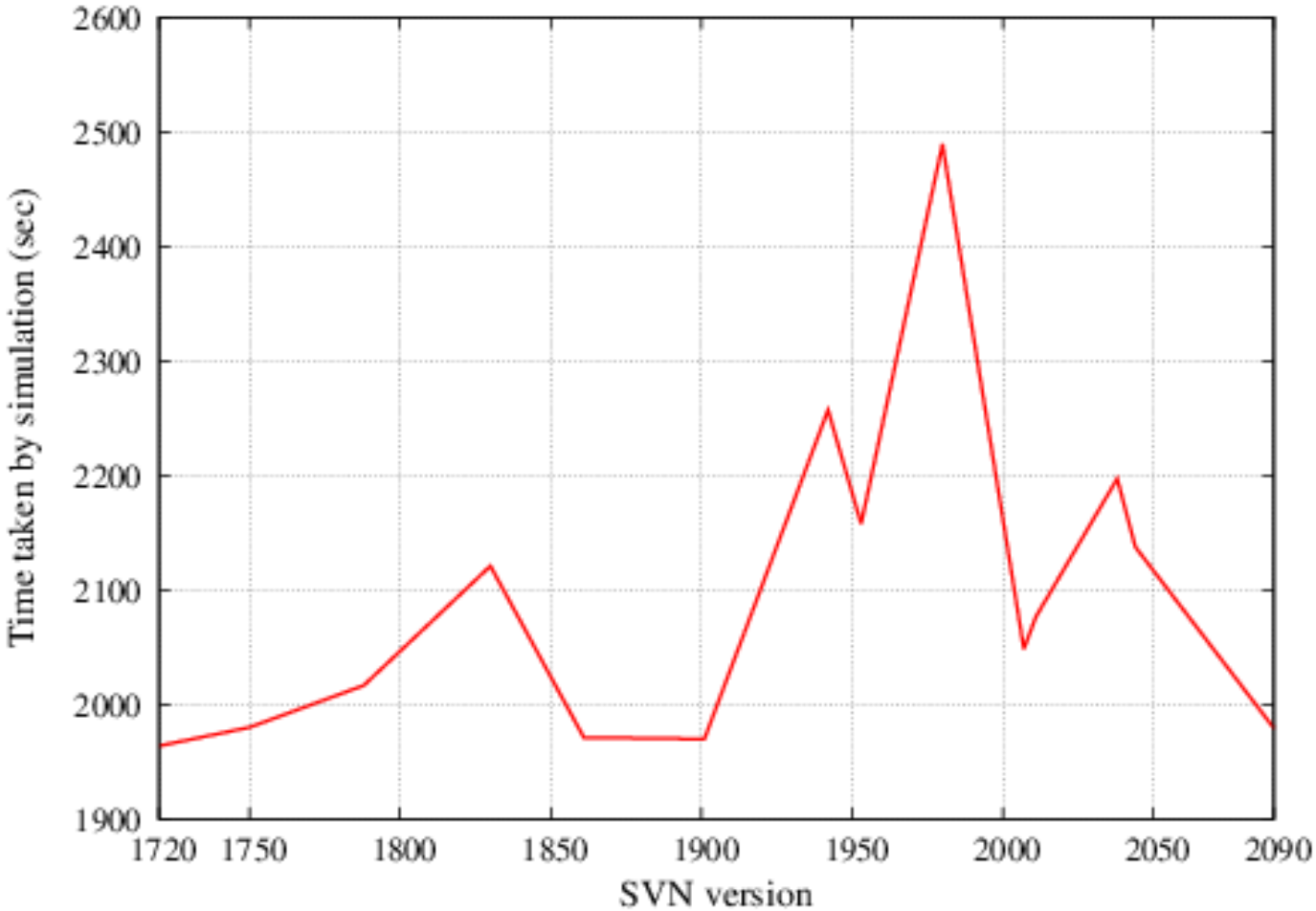
We advise compression should be used with careful considerations. Unless there is a storage space limitation one does not advised to enable compression capability.

In this test we have investigated that by enabling compression capability

- one can save about 40% storage space
- but he/she will loss 16% in performance

Changes in development phase

During the development of v4.1 we had measured a lot of performance tests to investigate the impact of code changes.



Changes in development phase...

Performance changes among revisions:

- could be due to physical parameters adjustment.
- code rearrangements and other important variable replacements.

Important!

Most of the data collected and presented in this study are based on different *svn* revisions between 1950 and 2000. One might get a maximum performance difference of about 500 seconds compared to results presented here with the official release of v4.1, which was released at *svn* version 2094.

Internal Variability (IV)

Definition

Internal variability (IV) is the difference in results from the same simulation due to a small change on the initial and boundary conditions of a climate model.

Our aim here is to understand and estimate how large is the IV and its role in validating the model. Before making IV analysis we made solution verification.

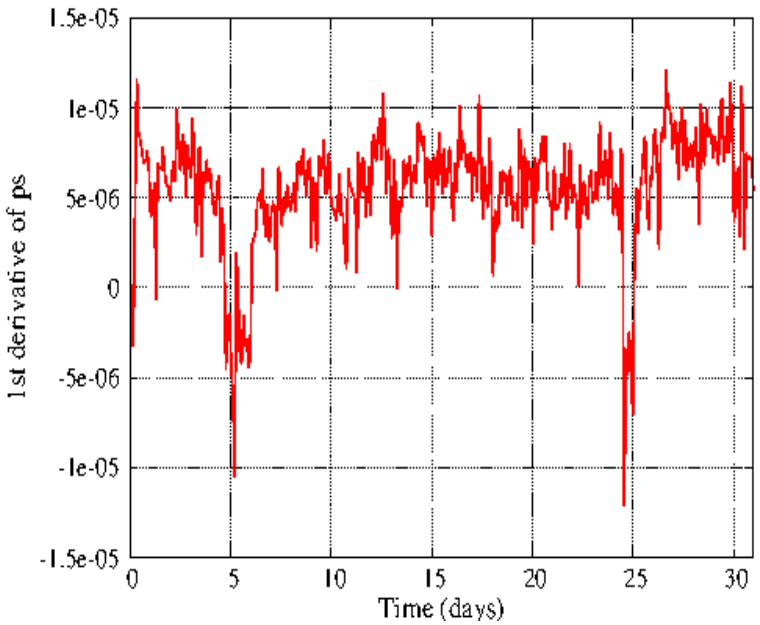
Verification

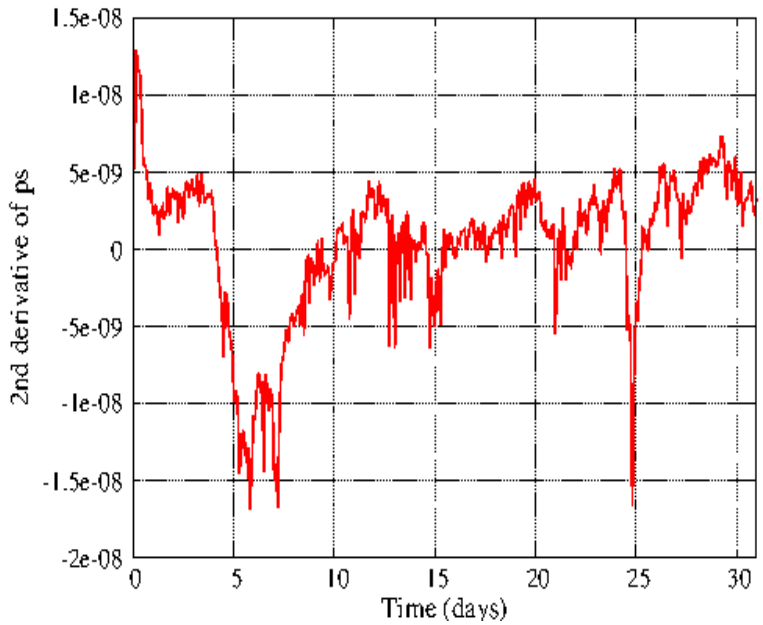
Verification is a question of how well the conceptual model has implemented, which means it deals with mathematics and addresses the correctness of the numerical solution to a given model.

-
-
-

First and second derivative output control variable

The verification procedure we implemented is to compare standard output of the model between two small different runs line by line.





First and second derivative output control variable...

These output difference comparison between the reference and a new run show us:

- a difference in the order of 10^{-06} for the first derivative of ps.
- and order of 10^{-08} in second derivative of ps

Important

This could be an acceptable result and an indication that the new simulation is not far from the reference run. It still arguable however if this change is just within the internal variability of the model or it is due to the net the effect of the change in the code. For this reason it is important to estimate the variability within the model.

-
-
-
-

Theoretical backgrounds and experimental setup

- In each experiment we applied some different kind of perturbation on some variables in the initial (IC) and lateral boundary conditions (LBC) dataset of the model as described by *Giorgi and Bi (2000)*.
- A new variable V_{new} after applying a random perturbation on the old variable V_{old} is given by:

$$V_{new} = V_{old} + 0.5 \times frand \tag{3}$$

Where $frand$ is a random number between -1 and $+1$.

- GNU Scientific Library (GSL) random number generator *ranlux* is used with different random number generator *seed* for each perturbation experiment.

Theoretical backgrounds...

The spatial Root-Mean-Square-Difference (RMSD) can be used as an indicator of internal variability of the model *(Rinke et al. 2004)*. The RMSD between the base and perturbed runs have been calculated by equation given in *(Giorgi and Bi 2000)*:

$$RMSD = \sqrt{\sum_{i} \frac{\left(V^{i}_{new} - V^{i}_{base}\right)^{2}}{N}} \tag{4}$$

Where i runs overall grid points and N is the total number of grid points.

Theoretical backgrounds...

The inter-member variance among the samples are calculated by equation given in *(Lucas-Picher and Daniel Caya 2007)*. First the average among the members of the ensemble for each variable V^j is calculated by equation:

$$V^j_{average} = \sum_j^n \frac{(V^j_{sample})}{n} \tag{5}$$

$$\sigma^2 = \sum_j^n \frac{(V^j_{sample} - V^j_{average})^2}{n} \tag{6}$$

- Where n is the number of members in ensemble
- σ^2 is the variance among the member of ensemble
- V^j_{sample} is individual sample value in specific sample j
- where j runs over the size of ensembles.

Experimental setup...

Experiment type	Perturbation area	Perturbation maximum magnitude	Experiment description
Expt-1	LBC	$T^o = 0.5$	1-year fully perturbed run
Expt-2	LBC	$T^o = 0.5$ $U^o = 0.5$ $V^o = 0.5$ $ps^o = 0.5$	1-year fully perturbed run

-
-
-

Experimental setup...

Experiment type	Perturbation area	Perturbation maximum magnitude	Experiment description
Expt-3	LBC	$T^o = 0.5$	1^{st} year perturbed and then 2^{nd} year unperturbed run
Expt-4	LBC	$T^o = 0.5$	1^{st} month perturbed and then 23 months' unperturbed run
Expt-5	LBC	$T^o = 0.5$	A total of two year perturbed run

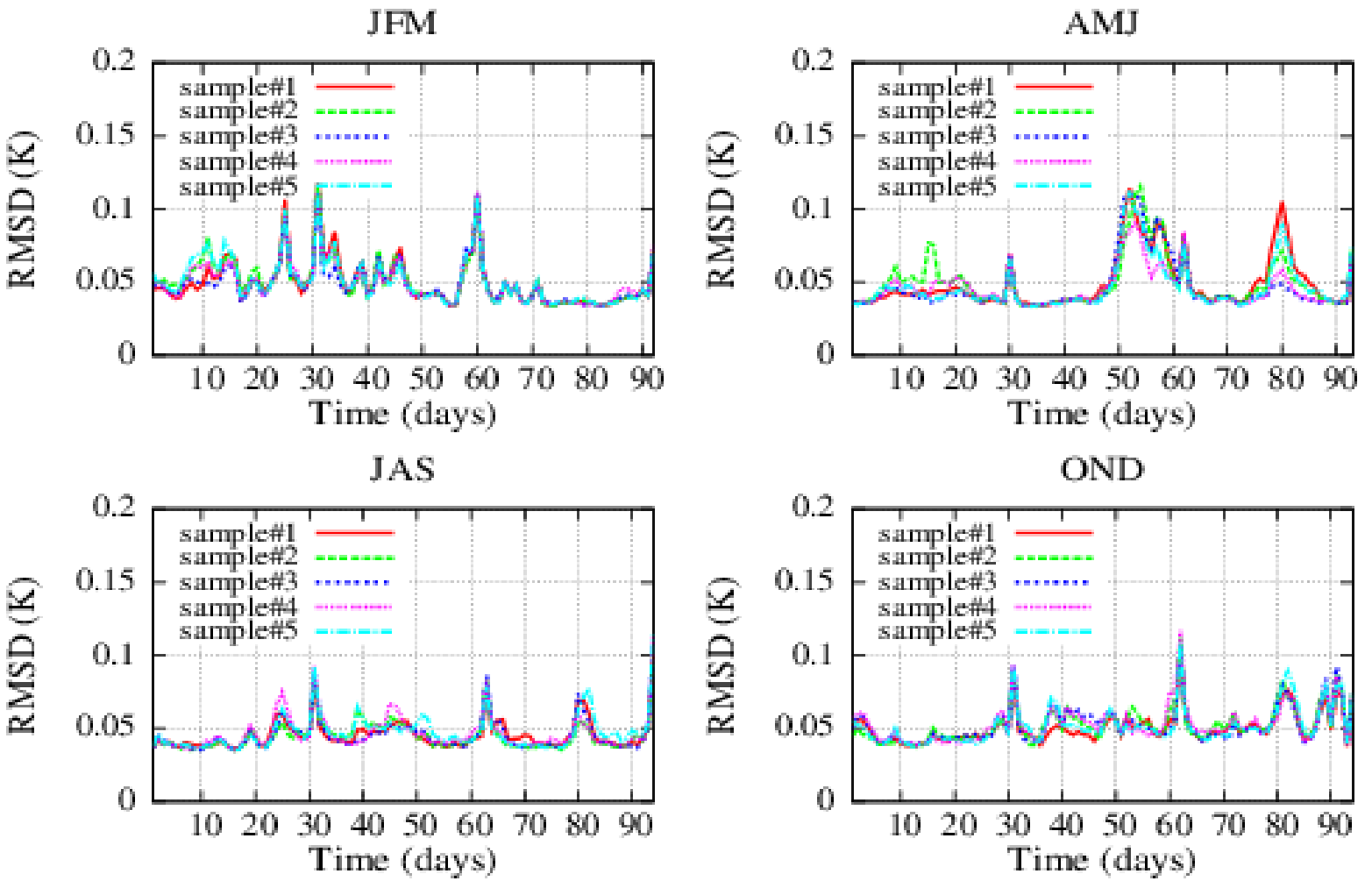
-
-
-
-
-

Optimization flag options:

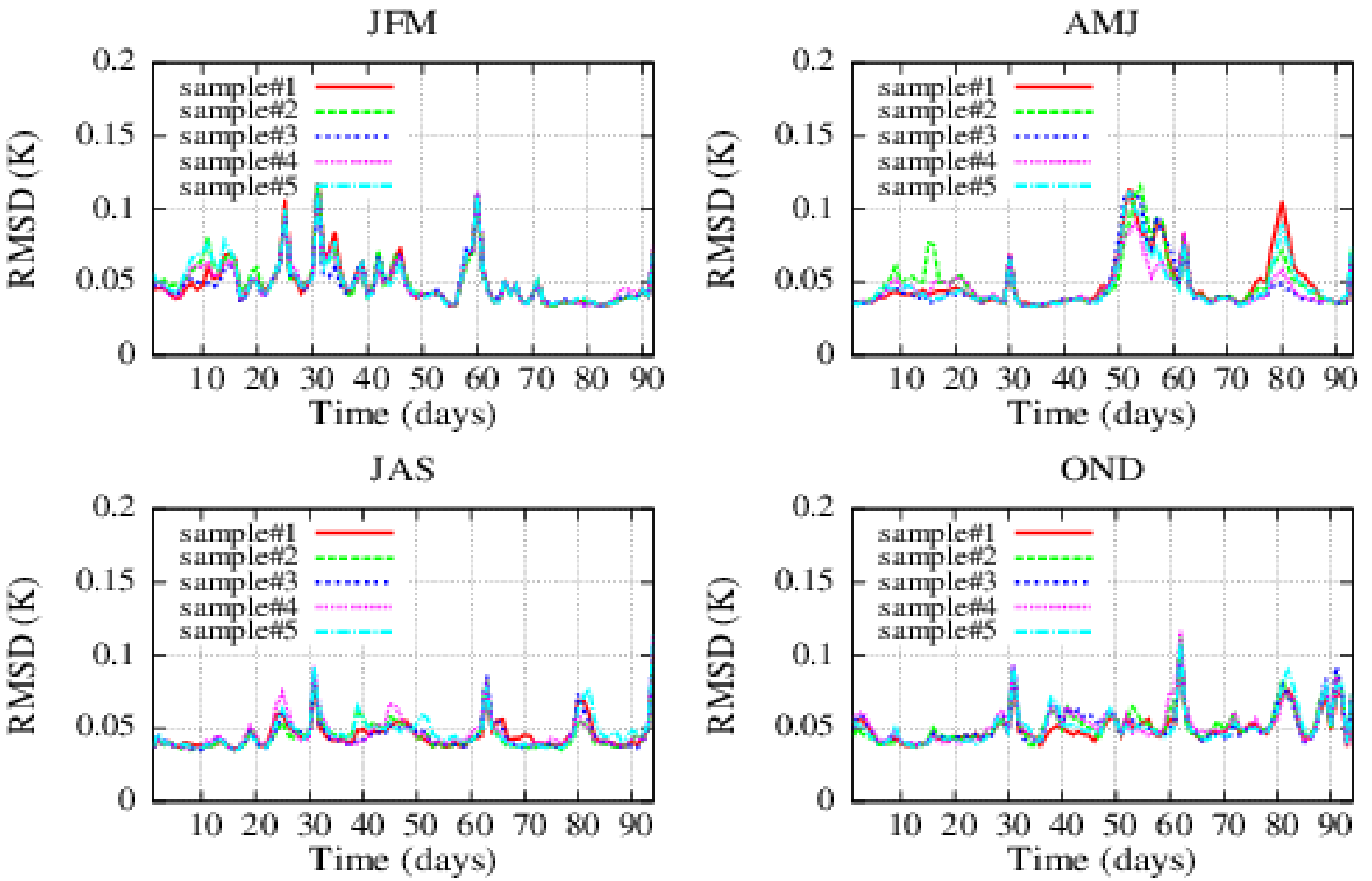
Compiler flag	Options	Description
-fp-model	precise	enables value safe optimizations on floating point data and rounds intermediate results to source defined precision.
	fast	enables more aggressive optimizations when implementing floating-point calculations.
	except	floating point exception semantics are used.
	strict	enables precise and except.

-
-
-
-

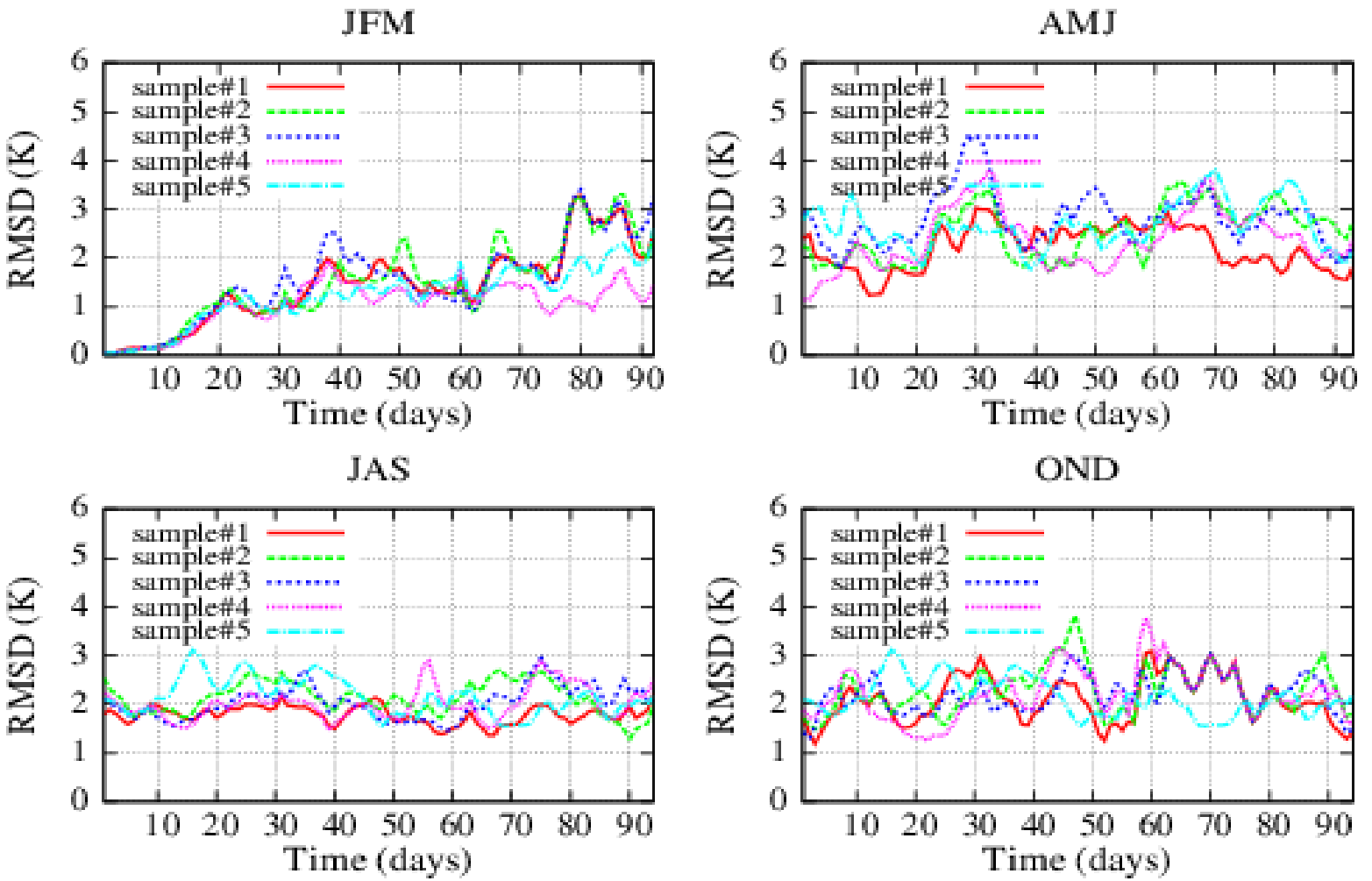
Standard simulation (small European domain)



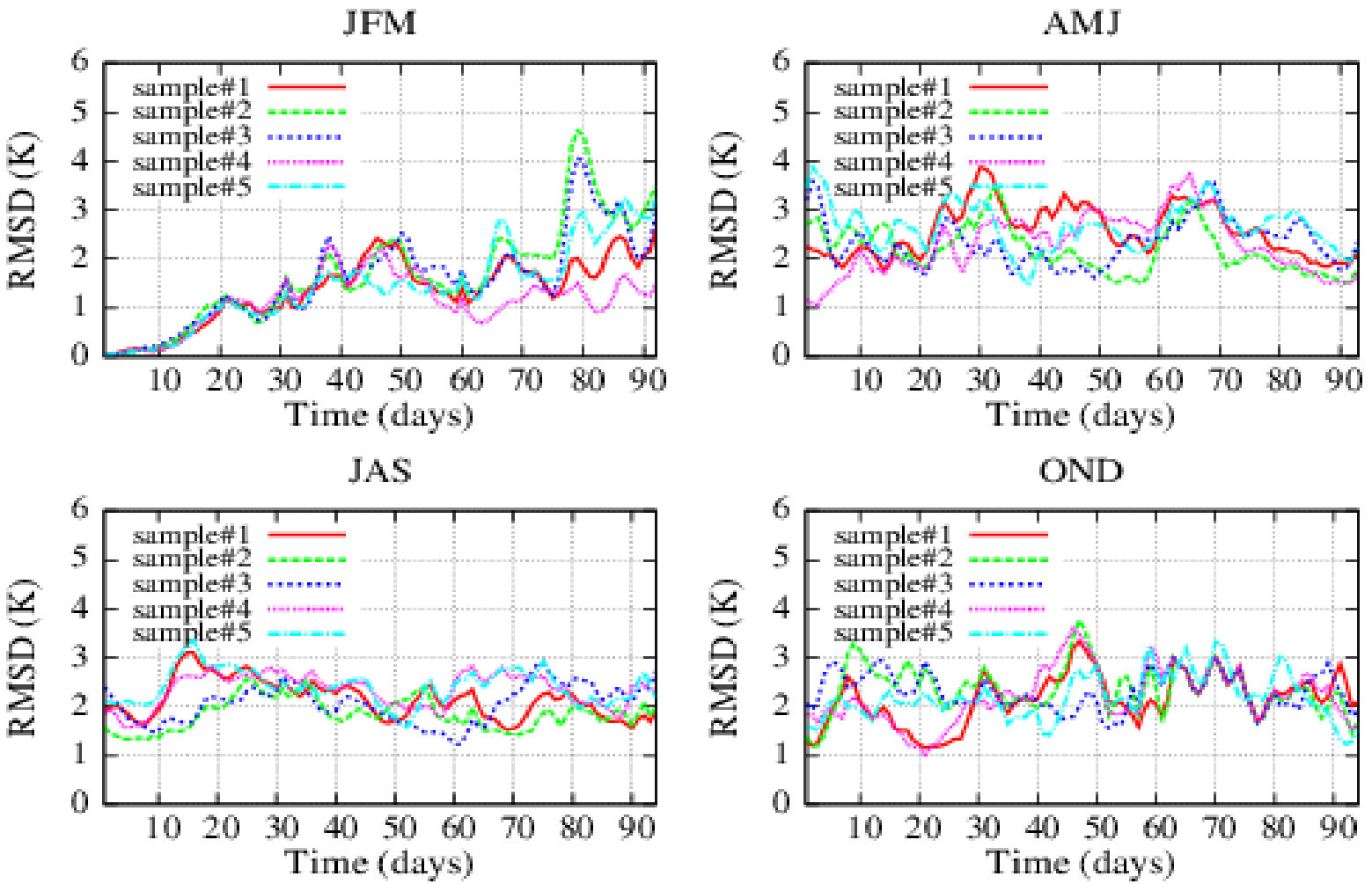
Optimized simulation (small European domain)



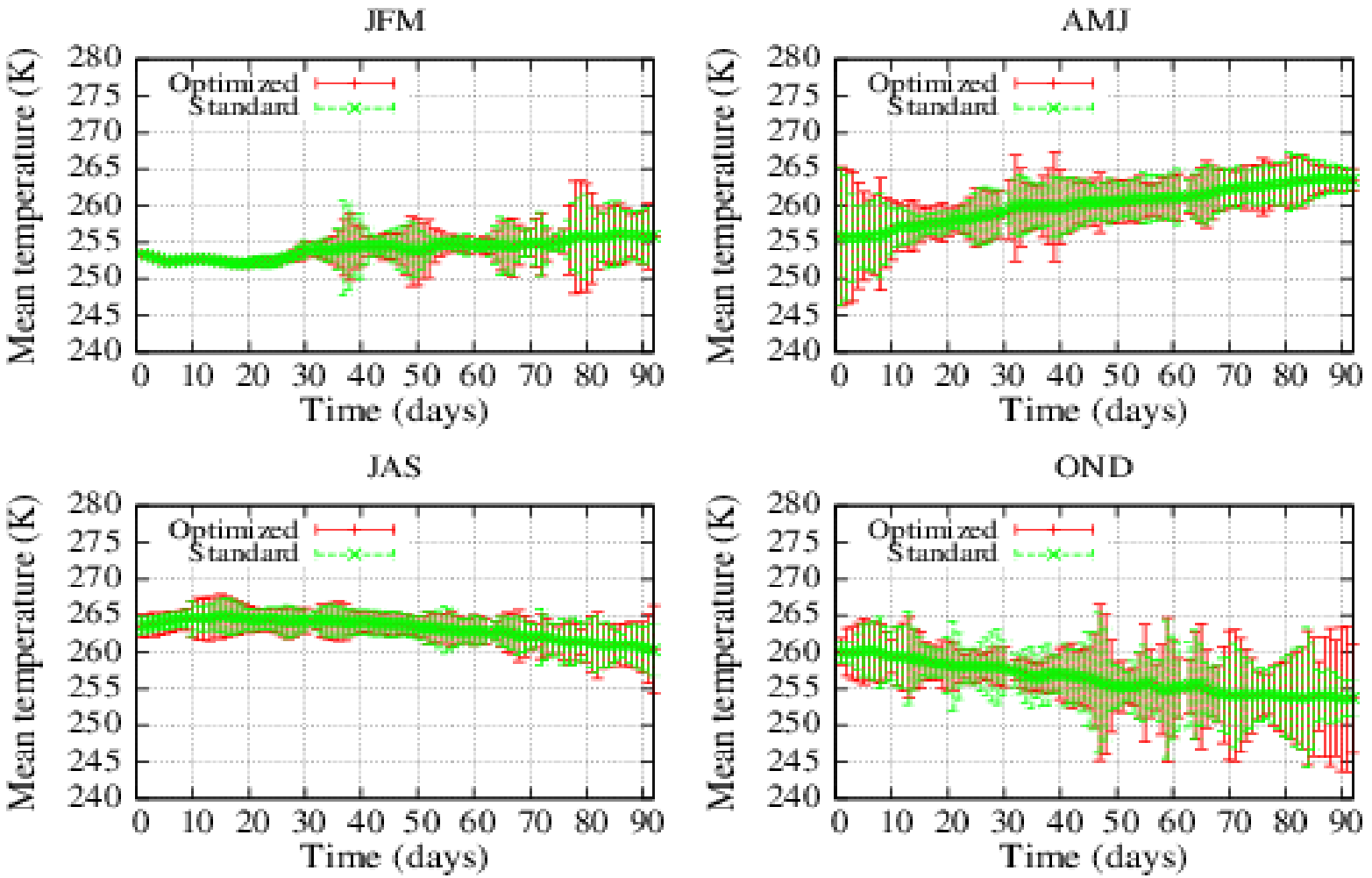
Standard simulation (big European domain)



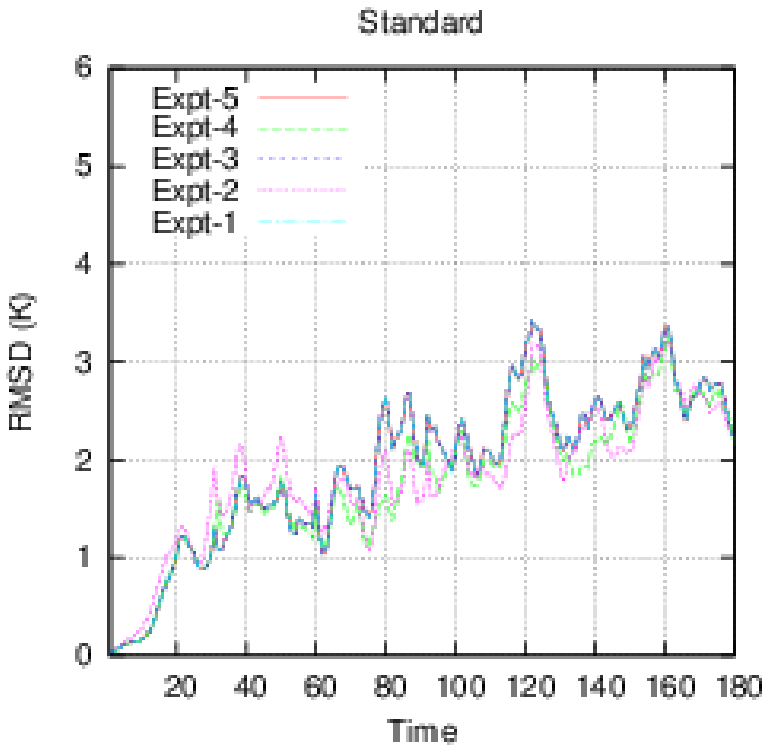
Optimized simulation (big European domain)

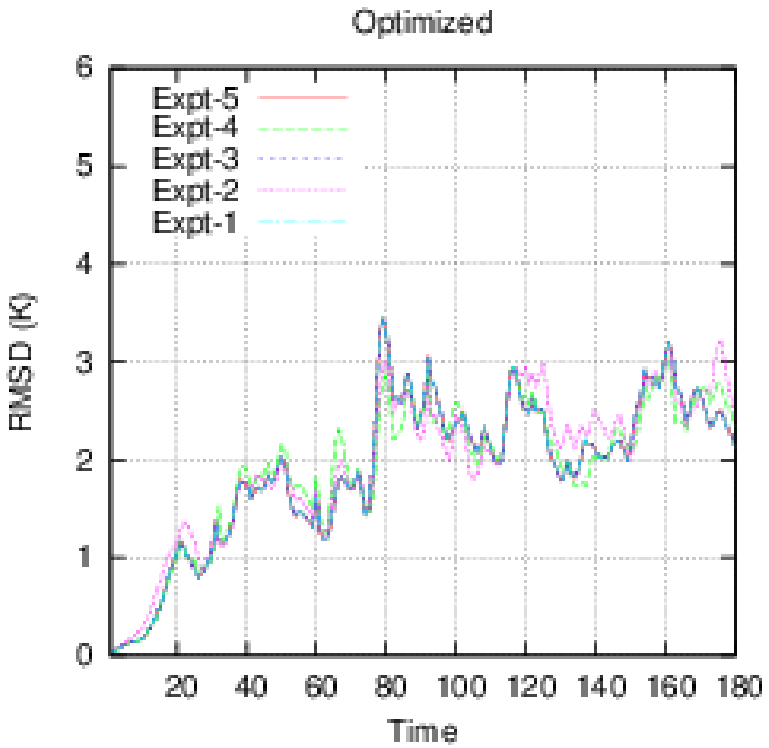


Variance between standard and optimized simulations

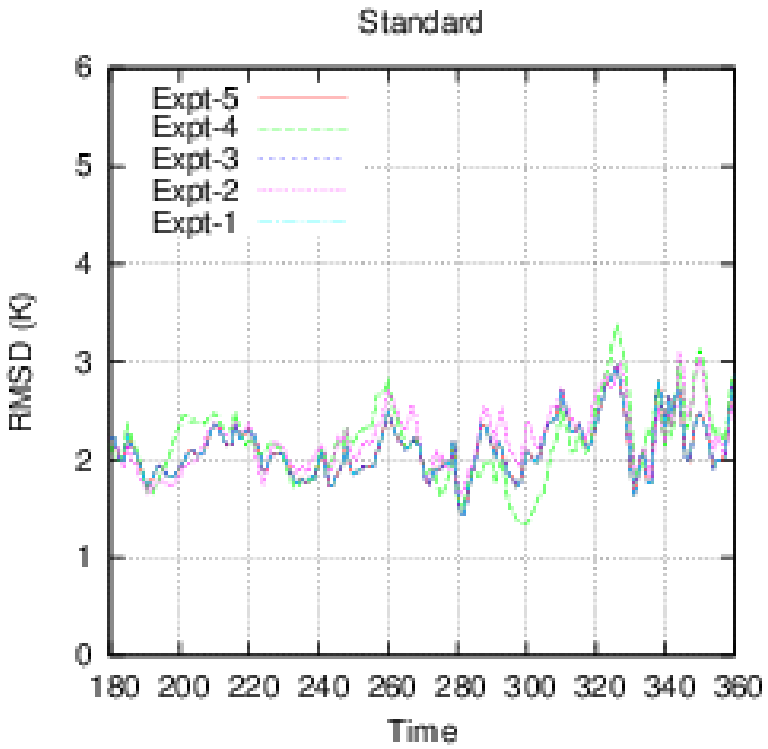


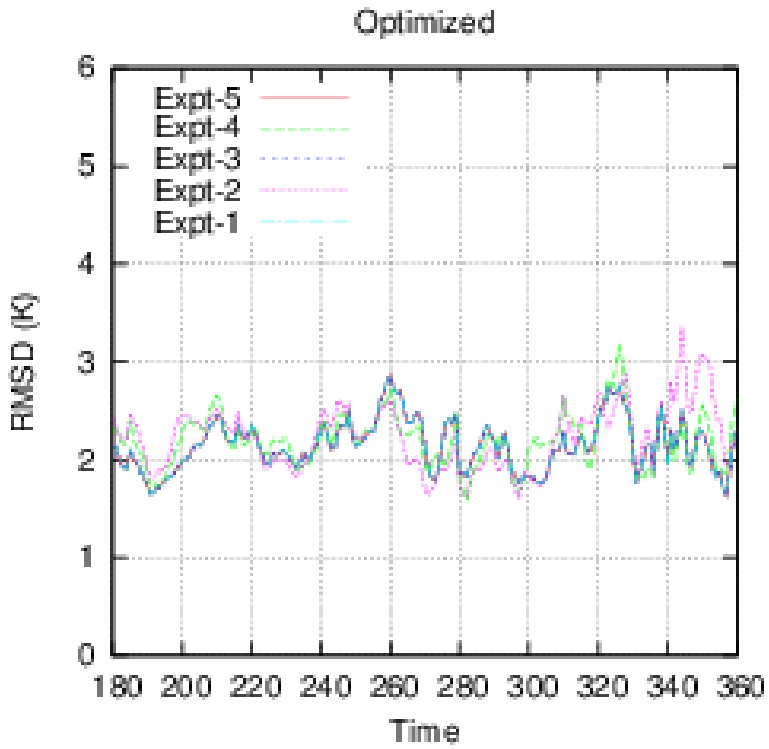
All long run simulations results on the first year



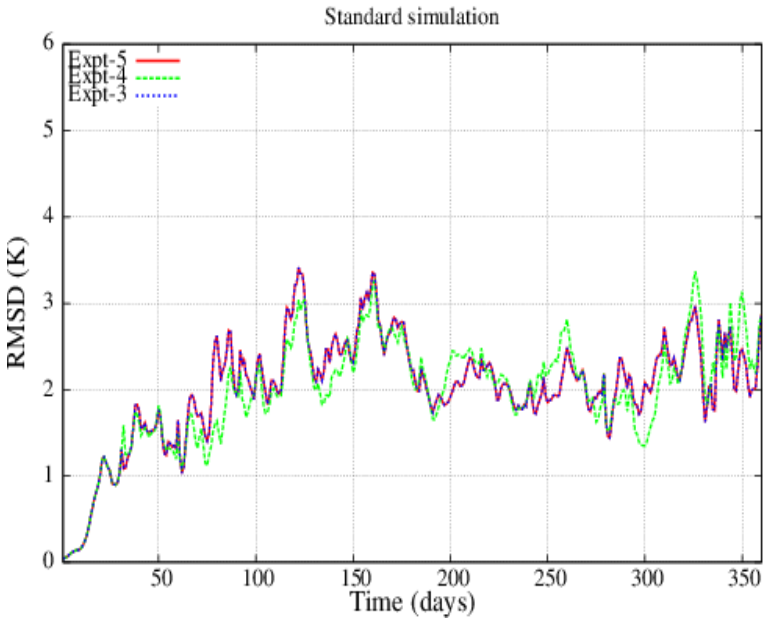


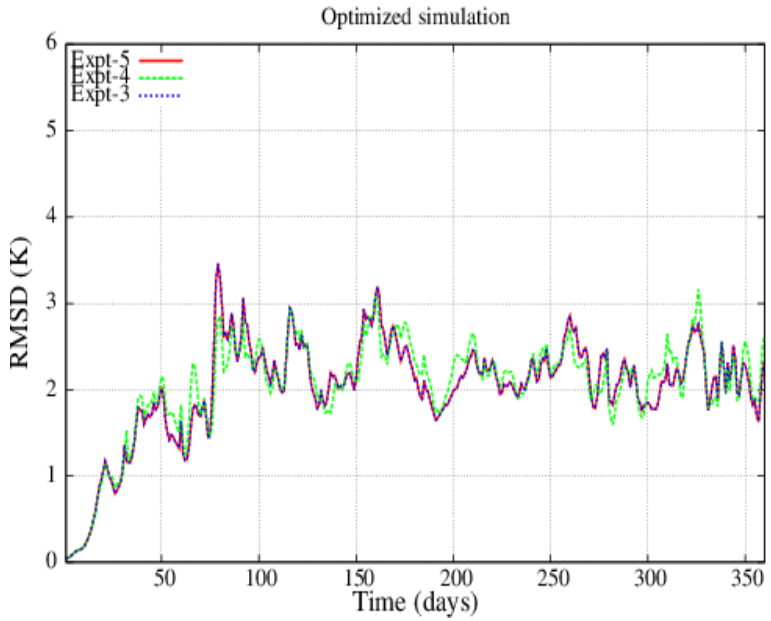
All long run simulations results on the first year...



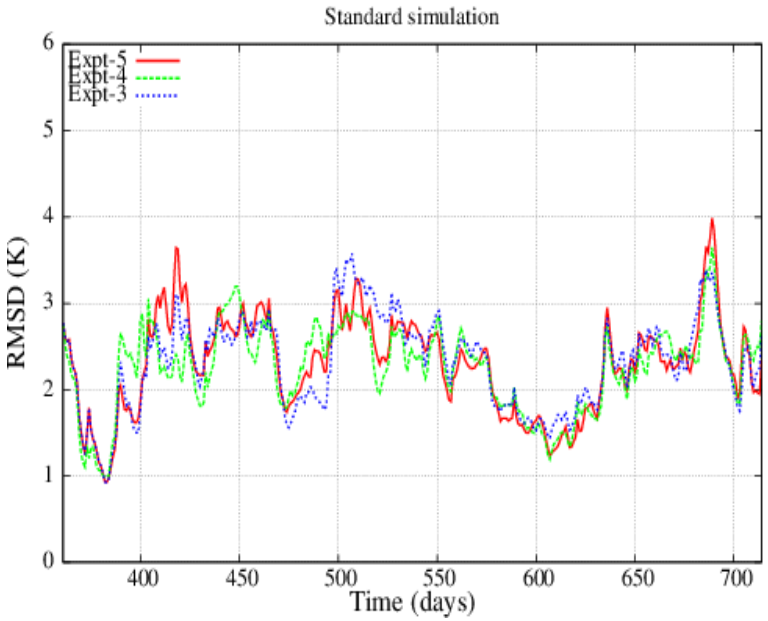


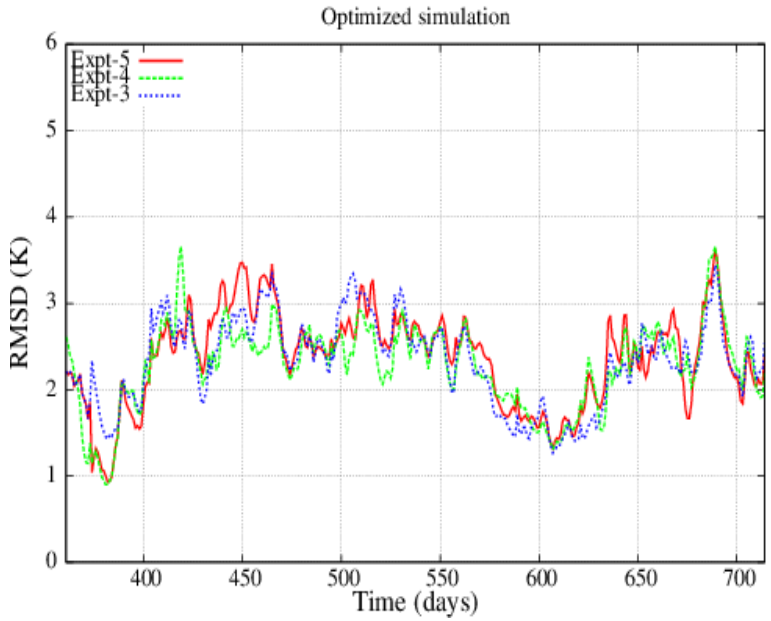
Standard and optimized simulations in long run (1990)





Standard and optimized simulations in long run (1991)





Conclusion

We have found out that:

- Intel is the fastest compiler suitable for RegCM.
- PGI can be used as an alternative.
- ARGO computational platform made from Intel E5620 Xeon performs best relative to other platforms.
- Some of time consuming MPI communications and FORTRAN modules needs to be improved.
- The model can scale up.
- Advantages introduced in this new version in term of easiness of use, portability, scientific stability and accuracy are worth a 15-20% decrease in speed.

Recommendation

- data compression capability could be an option for limited storage spaces.
- currently implemented 1D domain decomposition paralleling technique could be improved by 2D for performance enhancements.
- due to its high performance, we recommend to use Intel compiler to run the model.
- inter node communication using OpenMP could be an option for performance advancement.
- It's advisable to use BATS coupler than CLM coupler unless CLM coupler specifically needed.
- it is better to use real physical processors than that of Simultaneous Multithreading (SMT) enabled processors.
- we recommend to remove -**fp-model** *precise* option and get a performance of about 11%.

-
-
-
-
-

Thank you!

Acknowledgement

Special thanks for all

Stefano Cozzini
Gizaw Mengistu
Graziano Giuliani
Clement Onime
Filippo Giorgi
Moreno Baricevic
Martin Scarcia
Gulilat Tefera
Antonio Messina

and for everyone who has work with us to complete this study.