Schweizerische Eidgenossenschaft
Confédération suisse
Confederazione Svizzera
Confederaziun svizra

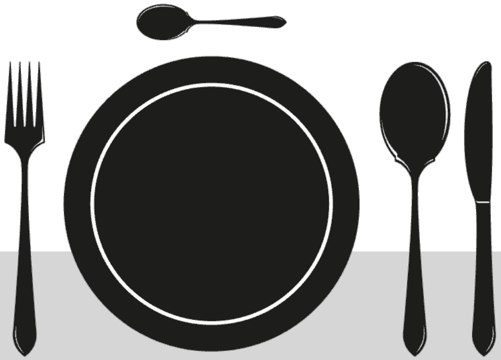
Summary of Priority Project POMPA

Oliver Fuhrer, MeteoSwiss
(for the whole project team)

Colleagues (Thanks!)

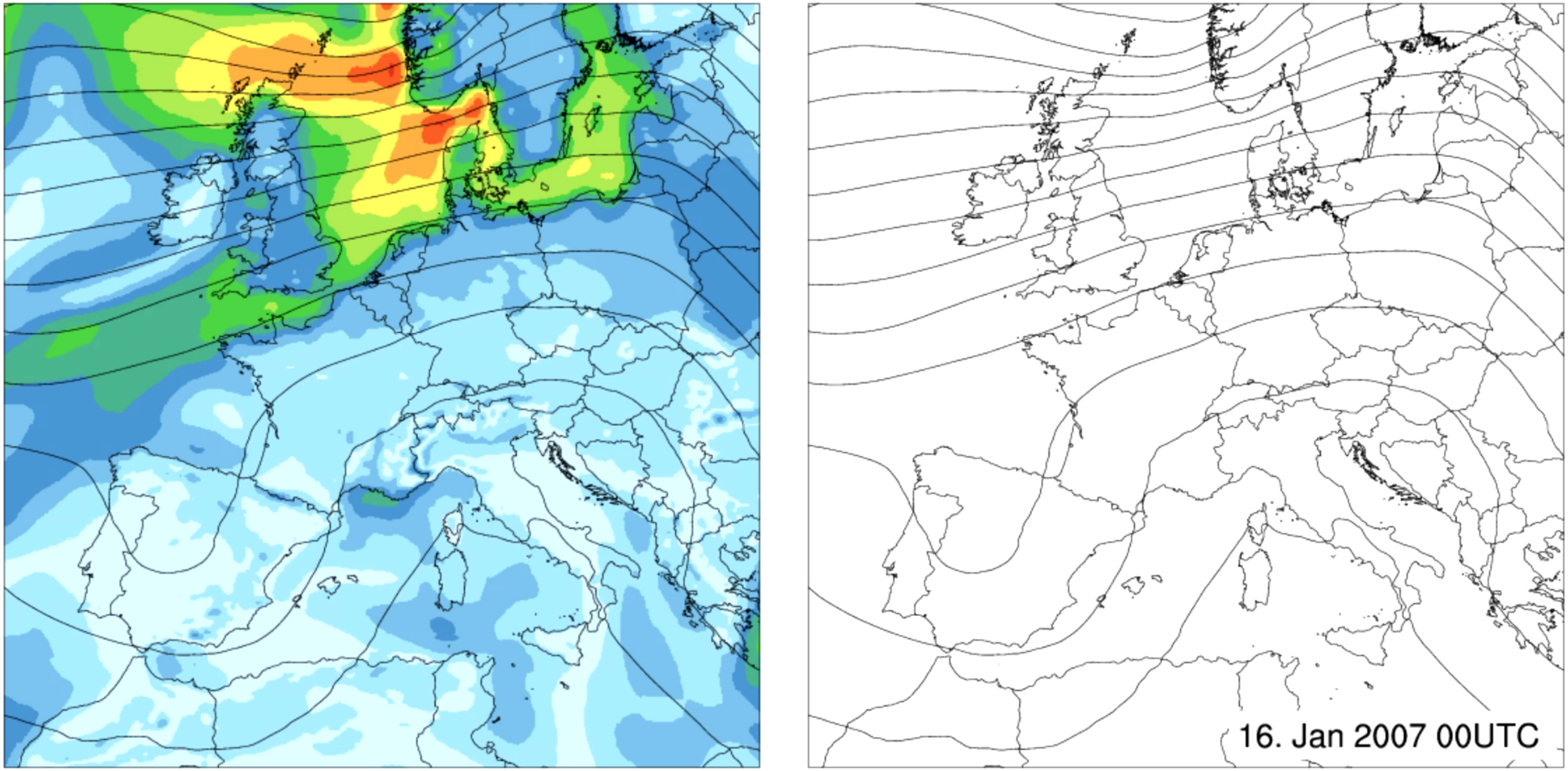
Andrea Arteaga
Anne Roches
Ben Cumming
Carlos Osuna
Christoph Schraff
David Leutwyler
Lucas Benedicic
Mauro Bianco
Michael Baldauf
Peter Messmer
Stefan Rüdisühli
Tobias Gysi
Ulrich Schättler
Xavier Lapillonne

Menu



- Highlight
- Specific tasks
  - Single precision
  - Performance profiling of 5.1
  - COSMO 5.X on GPU
  - STELLA developments & discussion
- Outlook

Climate Simulations on GPU (ETH)

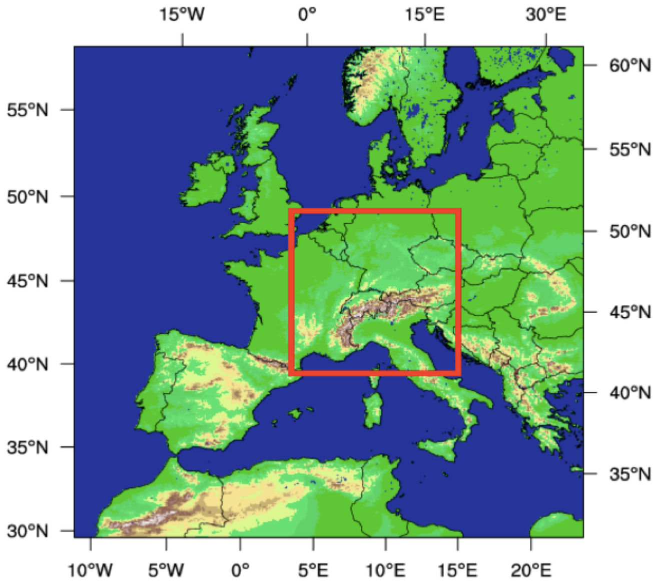


16. Jan 2007 00UTC

(courtesy of David Leutwyler)

Climate Simulations on GPU

- Cray XC30 @ CSCS
  each node has one GPU and one CPU
- COSMO 4.18 + modifs
- European Domain @ 2.2 km
  Size 1536 x 1536 x 60
- 144 nodes (only GPU used)
  40% of DWD's XC30
- Time-to-solution is roughly 0.2 SYPD
  ~2 months for 10 years
- Allocation for ~50 years
  1.1 million nodehours



Third-party funding

Finished

- HP2C COSMO-CLM (June 2010 – June 2013, 1 MCHF)
- HP2C OPCODE (July 2011 – June 2013, 0.5 MCHF)
- HP2C COCoNet (January 2012 – June 2013, 0.2 MCHF)

New

- **PASC GridTools** (January 2014 – June 2017, 0.7 MCHF)
- **SNF Sinergia** (May 2015 – April 2018, 1.5 MCHF)

Planned

- PASC Focused
- H2020 (ECMWF)

Single precision (1/2)

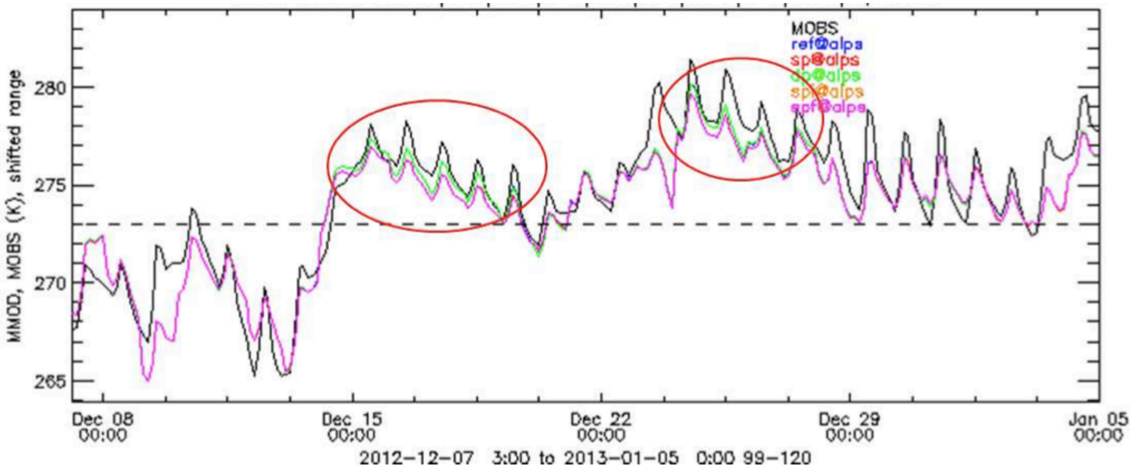
- Will be in official version 5.1, activate with `-DSINGLEPRECISION`
- Runtime & memory consumption decreases significantly (~ 60% of double precision)
- Tested for COSMO-E

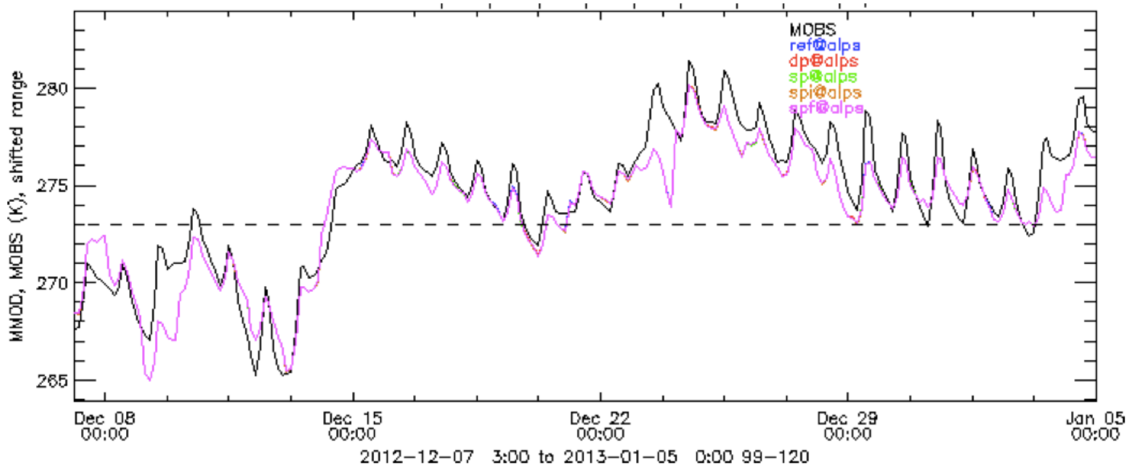
- **But…**
  - Some parts don't work yet (e.g. assimilation) or haven't been tested (e.g. seaice)
  - Developer behavior has to change
  - Developers currently don't run single precision

- **Recommendation**
  - Read CNL!
  - Validate your setup before using SP!
  - Talk to us!

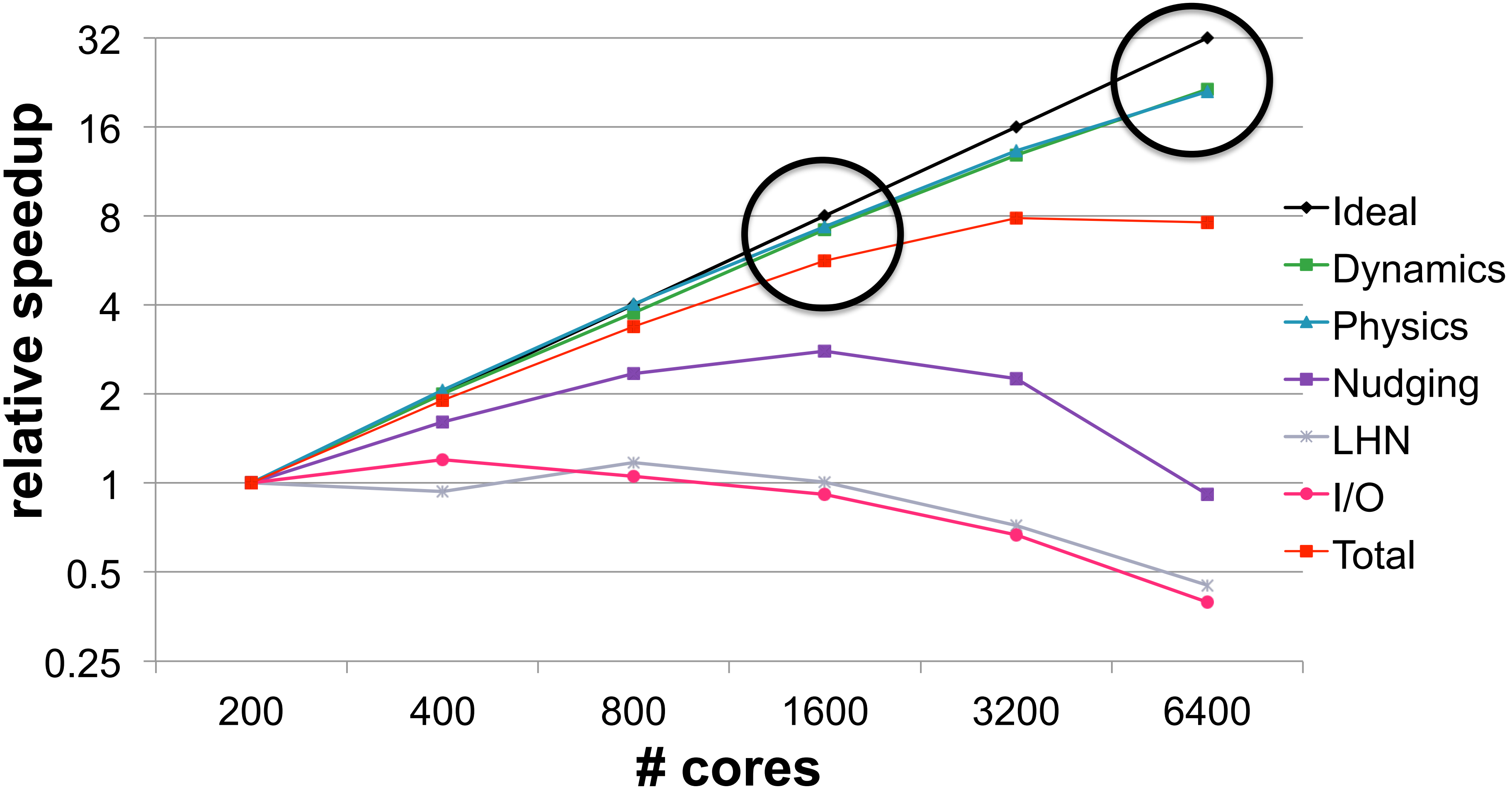
Single precision (2/2)

- Verification of T2m at +5 days (1 month COSMO-E)





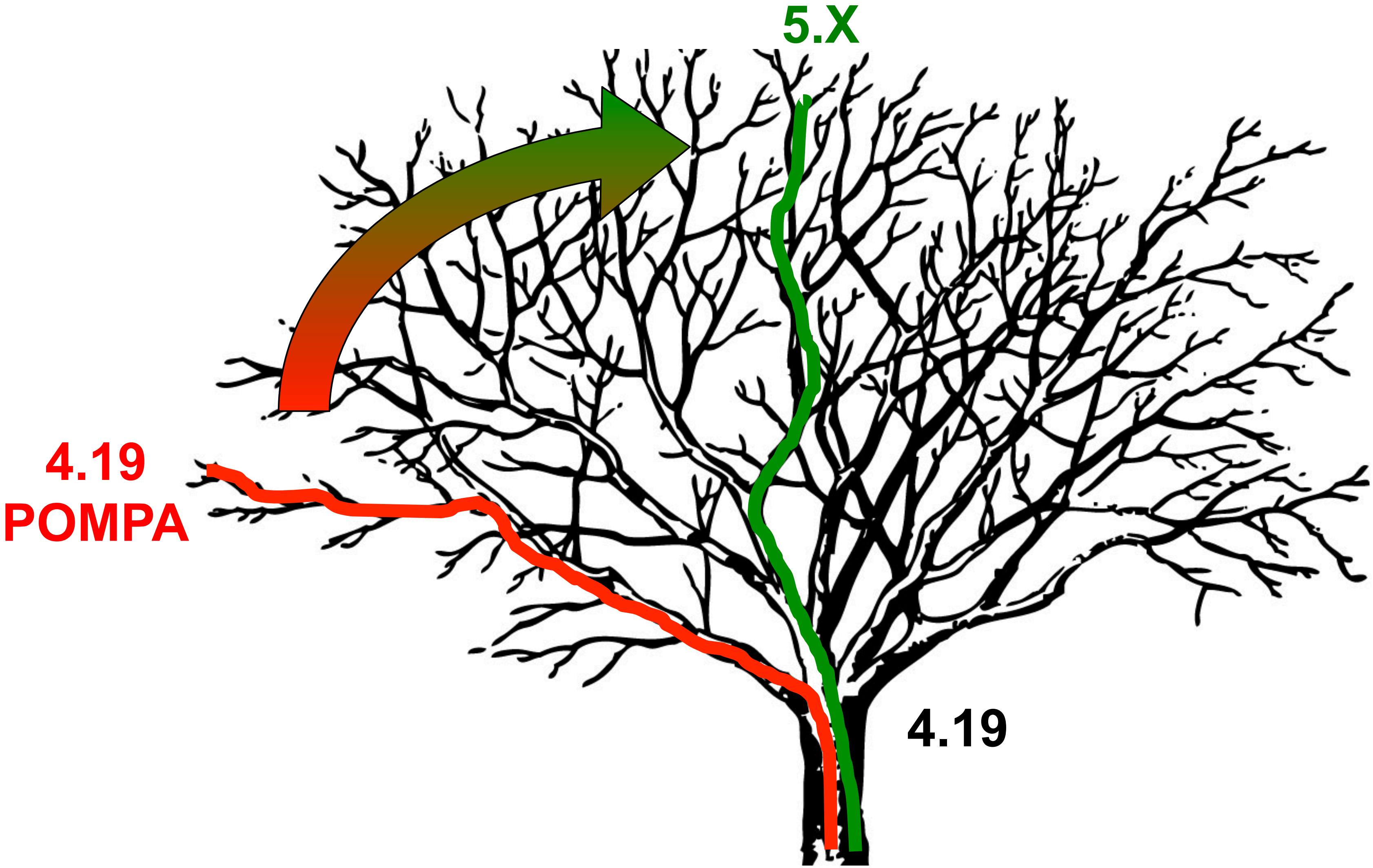
Scalability of COSMO Components (incl. Comm.)



First Conclusions

- ➔ Scalability of COSMO-Model for COSMO-DE65 domain size is reasonably well up to 1600 cores. Dynamics and Physics also scale beyond up to 6400 cores.
- ➔ Operational requirements for COSMO-DE65 ensemble can currently not be met
- ➔ This is not a problem of the scalability, but of some expensive components!
- ➔ Expensive Components:
  - ➔ New fast-waves solver is more expensive than old one (40-50% of dynamics time; but not investigated further up to now)
  - ➔ Communication in the Latent Heat Nudging
  - ➔ Additional Computations: is almost only in RTTOV10
    - ➔ factor of about 10-15 compared to RTTOV7

Summary of POMPA



POMPA developments

- C++ Dycore
- Changes and bugfixes in Fortran dycore
- Static memory allocations
- **Block module**
- **Bl**ock physics
- Serialization
- **Single precision**
- New communication interfaces
- New BC module
- Changes in BCs inside and after dynamics

- Code refactorings
- OpenACC directives
- Tracking and copying of boundary fields
- Re-ordering of microphysics
- Re-ordering of assimilation / relaxation
- Change of application domain in relaxation
- **NetCDF I/O**
- …

Strategy

- **Goal** GPU capable COSMO version 5.X delivered to SCA by December 2014
- **Guideline** COSMO Coding Standards
- **Path** WG chairs → SMC → SCA → Trunk
- Many changes to a large part of the code
- Keeping in sync with the latest repository head is an effort

Conclusion

- In order to make this happen…

  We are dependent on code owners, SCA, WG chairs and SMC for their time and support!
- Bring changes back step-by-step
- Thanks to Uli (block, microph) and Christoph (assimilation)!

GPU Acceleration (1/2)

- On track
- Not all namelist options will be supported for 5.X version (current focus COSMO-1)
- Not all output fields will be supported for 5.X version (e.g. CAPE)
- You require C++ dynamical core based on STELLA in order to run on GPU

Conclusion

- Tell us your requirements!
- Send us your YUSPECIF!
- Talk to us!

GPU Acceleration (2/2)

Parts	Status	Delivery / Required work	Remark
Physics	On-going	18/09/2014	Only turbulence and radiation still on-going.
Fortran-C++ interface	On-going	05/09/2014	First version working. Modifications on-going.
Dynamical core	On-going	18/09/2014	Working. Including new FW solver. Some features for C-1 still missing.
Assimilation	Ready to merge	1 day. On-demand	Tested with Cray, problem with PGI No LHN
Communication	Ready		Use GCL for GPU
Structure code (e.g. initialization, lmorg.f90, …)	On-going	18/09/2014	Mostly in lmorg.f90 + some utility functions
Diagnostics	Not started	2 days (for minimal set)	Minimal set sufficient for standard verification (also for CALMO)
Output	Not started	30/09/2014	Port already available, only need to be merged into 5.0
Single precision	On hold		Doesn't work for assimilation

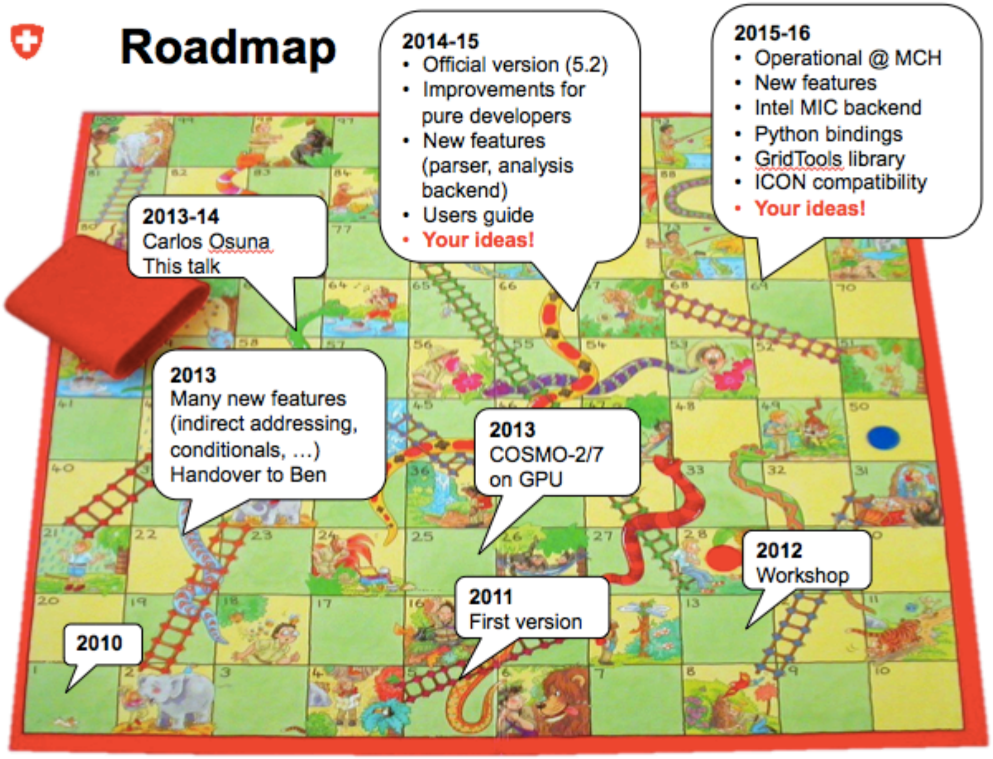
Documentation

- **Existing**
  - Stencil library workshop material
  - Stencil library (implementation)
  - GCL documentation
  - Communication framework
  - Serialization framework
  - C++ style-guide
  - Single precision CNL
  - Block structure API + users guide
  - OpenACC (implementation)

- **Incomplete or Missing**
  - Stencil library (users guide)
  - Parallel NetCDF I/O (users guide)

STELLA developments

- STELLA = Stencil Loop Language
- Generic C++ library for stencils on structured grids
- Still young, but evolving rapidly…



Roadmap

2010

2011
First version

2012
Workshop

2013
COSMO-2/7 on GPU

2013
Many new features (indirect addressing, conditionals, …)
Handover to Ben

2013-14
Carlos Osuna
This talk

2014-15
- Official version (5.2)
- Improvements for pure developers
- New features (parser, analysis backend)
- Users guide
- Your ideas!

2015-16
- Operational @ MCH
- New features
- Intel MIC backend
- Python bindings
- GridTools library
- ICON compatibility
- Your ideas!

STELLA as a language

Language elements

- Declaration
- Loops (k, ij)
- Conditions (switch/ case, if)

Keywords

- **Base language**
  StencilCompiler, Param, StencilConfiguration, define_loops, define_sweep, define_stages, StencilStage, IJRange, Krange, define_switch, define_case, define_if

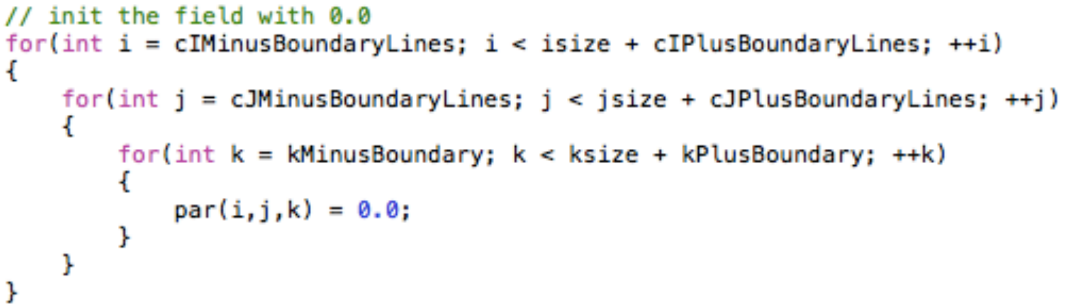
- **Optimization**
  define_temporaries, StencilBuffer, StageVariable, define_caches, IJCache, KCache, IJKCache, KWindow

- **Qualifiers**
  - FullDomain, FlatCoordinates, TerrainCoordinates
  - KMinimumCenter, KMaximumCenter, …
  - cKIncrement, cKDecrement, cKParallel
  - cFillAndFlush, cFill, cFlush, cLocal

STELLA / C++ interoperability

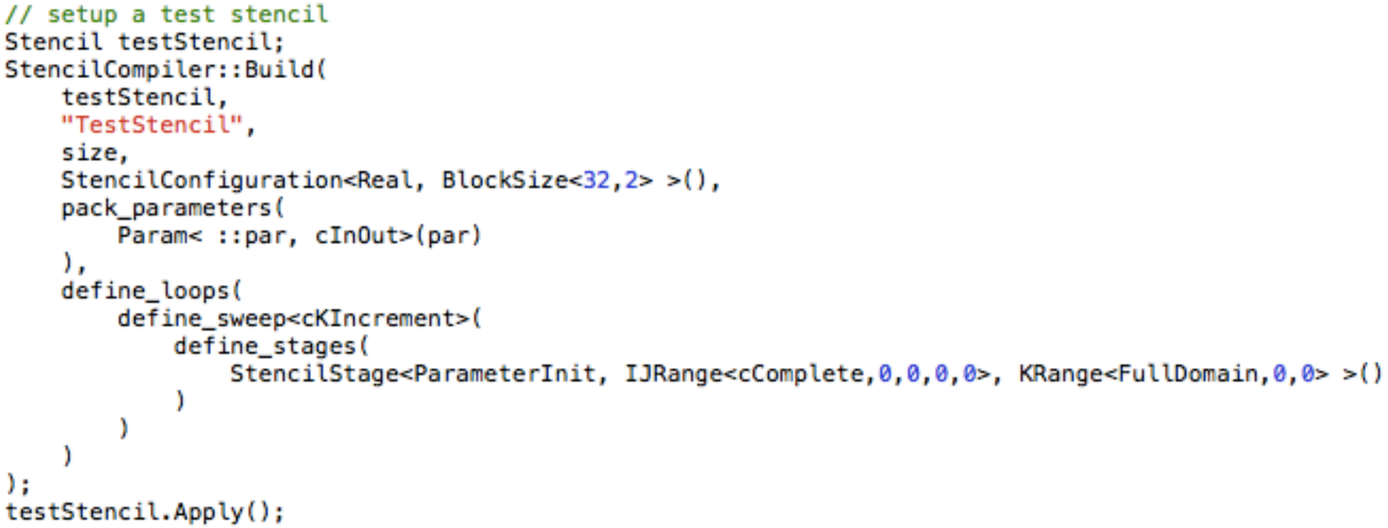
- We do it all the time (for testing)!

C++ code



```
// init the field with 0.0
for(int i = cIMinusBoundaryLines; i < isize + cIPlusBoundaryLines; ++i)
{
    for(int j = cJMinusBoundaryLines; j < jsize + cJPlusBoundaryLines; ++j)
    {
        for(int k = kMinusBoundary; k < ksize + kPlusBoundary; ++k)
        {
            par(i,j,k) = 0.0;
        }
    }
}
```

“STELLA code”

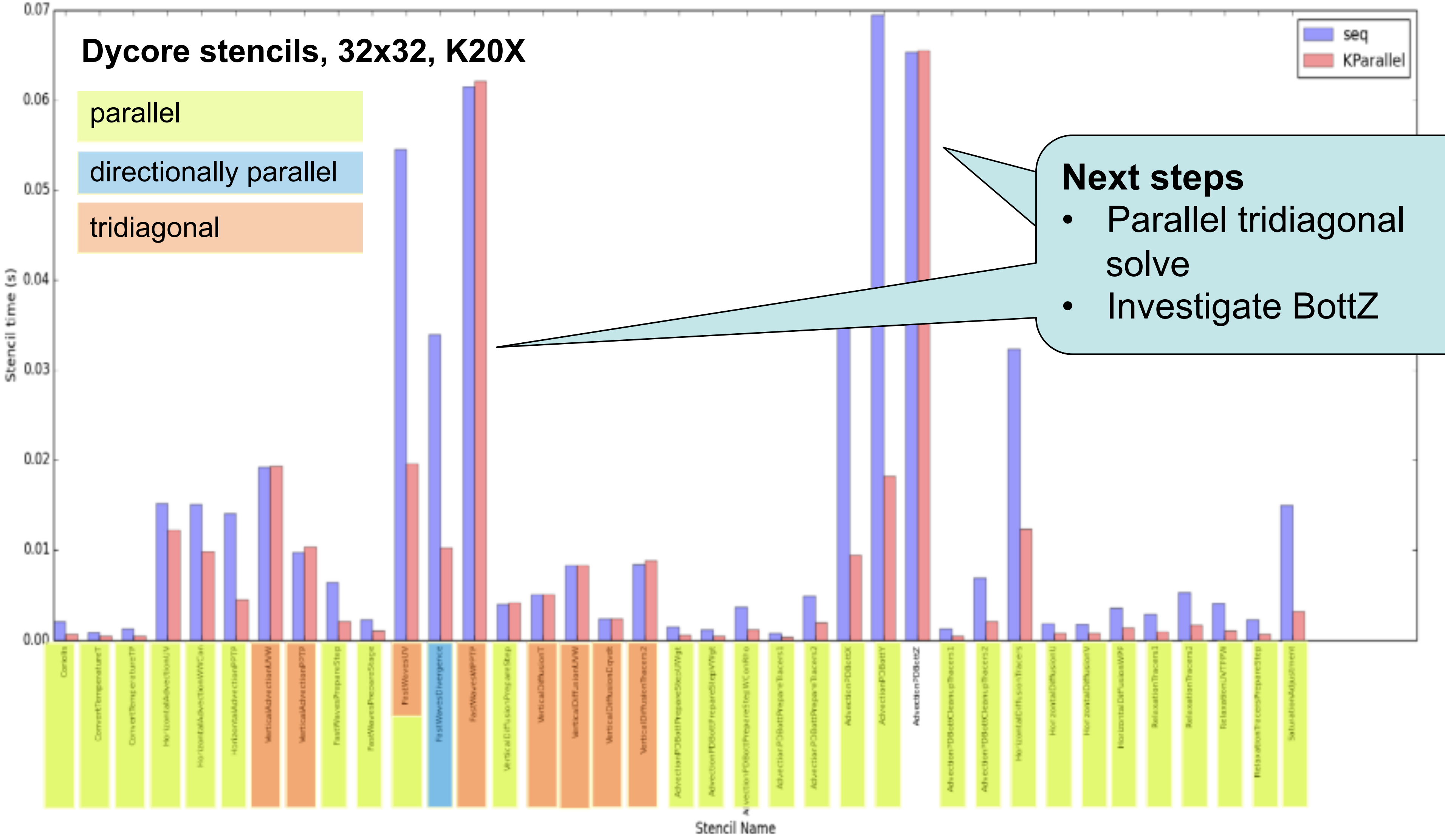


```
// setup a test stencil
Stencil testStencil;
StencilCompiler::Build(
    testStencil,
    "TestStencil",
    size,
    StencilConfiguration<Real, BlockSize<32,2> >(),
    pack_parameters(
        Param< ::par, cInOut>(par)
    ),
    define_loops(
        define_sweep<cKIncrement>(
            define_stages(
                StencilStage<ParameterInit, IJRange<cComplete,0,0,0,0>, KRange<FullDomain,0,0> >()
            )
        )
    )
);
testStencil.Apply();
```

STELLA developments

- **Syntax features**
  - flexible runtime / compile time options
  - flexible if and switch/case statements
- **Performance features**
  - vertical parallelization
  - improved caching on GPU
- **Debugging features**
  - Unified compile time errors
  - Parser
- **Standalone usage features**
  - Debugging features
  - Logging / verbose mode
  - Python interface

Vertical parallelization



Next steps

- Parallel tridiagonal solve
- Investigate BottZ

Review by Michael Baldauf

- We take the feedback **very** seriously

- Summary of (negative) feedback
  - It is hard to learn a new language (→ help, docu)
    - switch imperative → declarative is hard
    - STELLA is harder than C++
  - Productivity is low (factor 5-10)
    - it get's better over time
    - performance portable code
    - coding is typically fraction of working time
  - No advantages of DSEL, use source-to-source translator? (→ evaluation)

POMPA Conclusions

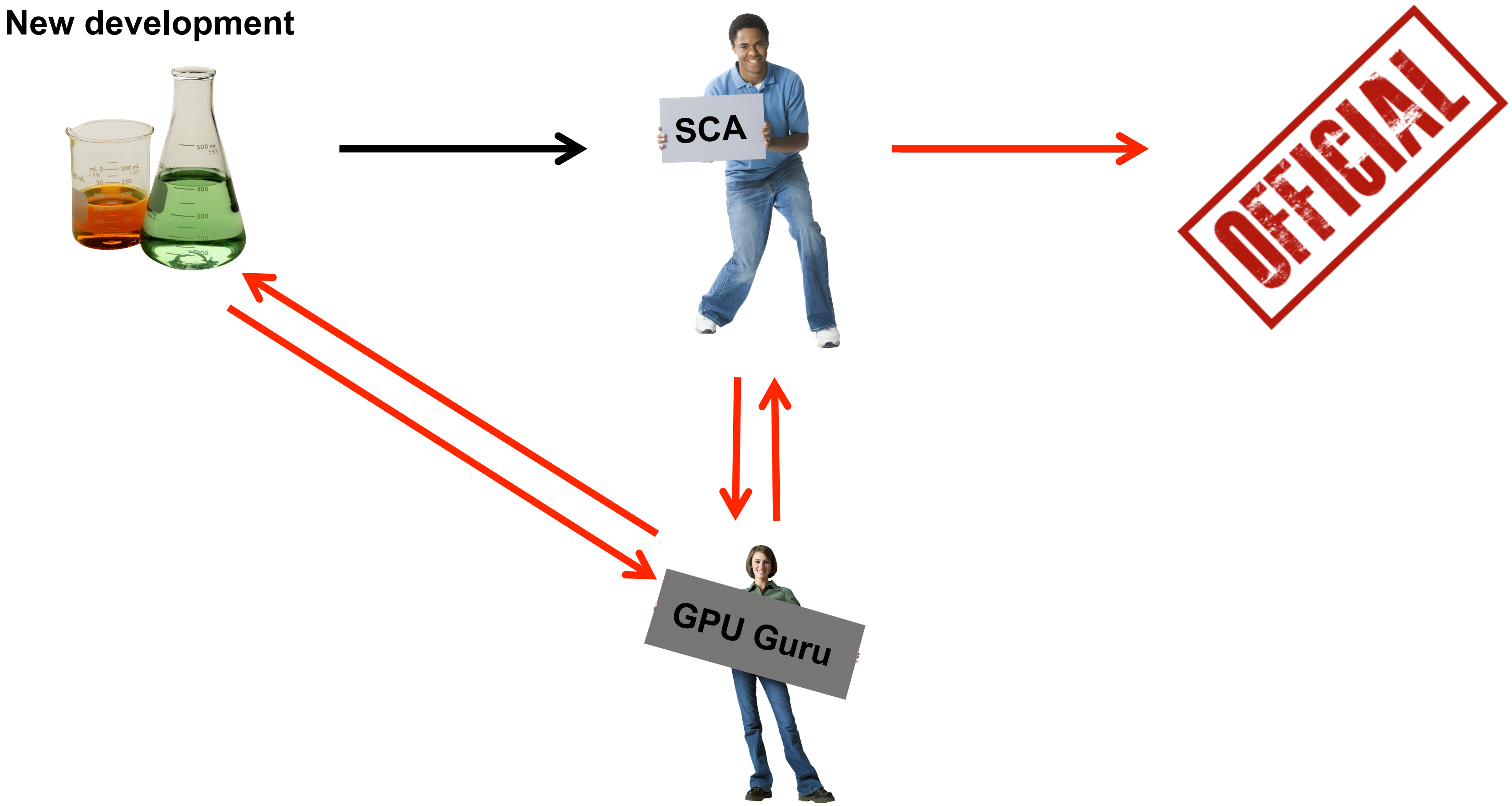
- **Retain the Fortran code, but re-evaluate this decision regularly**
  - Situation is evolving rapidly
  - Extra effort carried by COSMO consortium

- **Synchronization of Fortran with C++ code has to be organized**
  - This can not be **only** done by the **dynamics** developers
  - But, interaction is critical for efficiency and knowhow transfer

- **Involve developers in design and implementation next version of stencil library**
  - Via a joint research project?
  - Especially also from the ICON team

- **Focus more on usability features of C++ code in standalone mode**
  - Try a new development using STELLA

Coordination of new versions



New development

SCA

GPU Guru

We would not recommend delivering a version which is out-of-synch to the users

Coding standards

Coding standards require adaption / extension

- **C++ code**
  - Coding conventions of Fortran don’t apply (e.g. naming)
  - Integrate POMPA project coding conventions?
- **OpenACC / GPU**
  - Changes for good practices
- **Conflicting interests**
  - Performance on CPUs / GPUs / other hardware
  - Memory usage vs. efficiency
- License for STELLA and C++ Dycore?

Knowhow Transfer

- Stencil workshop
- OpenACC tutorial
- Documentation + Presentations + Publications + Newsletters
- **What else? Your suggestions?**

Project extension

- **POMPA project extension proposed until 09.2015 (according to project plan v5.0)**
- **Main reasons**
  - Integration into 5.X will require further work with code responsibles, SCA, and working group chairs
  - Further GPU porting work required/requested (physical parametrizations, LHN)
  - Work to keep C++ version of dycore synchronized
  - Support, training and documentation
  - Assimilation does not work in single precision
  - Open tasks (hybrid OpenMP/MPI, new halo-update, …)
  - Ongoing related activities (e.g. PASC GridTools project)

Thank you!