Schweizerische Eidgenossenschaft
Confédération suisse
Confederazione Svizzera
Confederaziun svizra

Swiss Confederation

Federal Department of Home Affairs FDHA
Federal Office of Meteorology and Climatology MeteoSwiss

Establishing a baseline for global climate simulations at 1 km on hybrid HPC architectures with COSMO

Carlos Osuna and Oliver Fuhrer, MeteoSwiss

Contributions from T.- Chadha[2], T. Gysi[3], T. Hoefler[3], G. Kwasniewski[3], X. Lapillonne[1], D. Leutwyler[4], D. Lüthi[4], C. Osuna[1], Ch. Schär[4], T. Schulthess[5,6], and H. Vogt[6]

[1]MeteoSwiss, [2]ITS RI ETH, [3]SPCL ETH, [4]IAC ETH, [5]ITP ETH, [6]CSCS

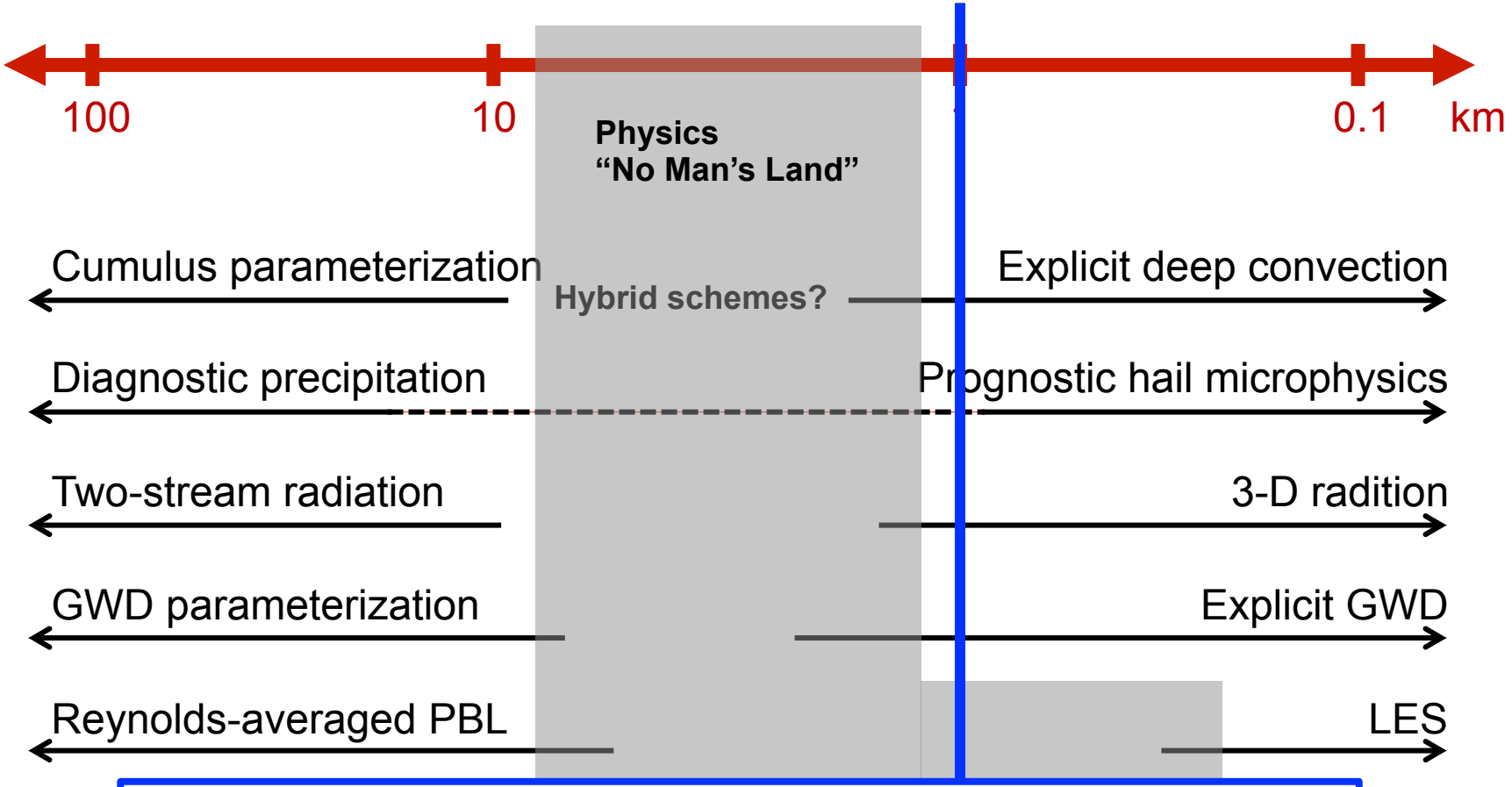
5th HPC Workshop (ESiWACE) - Lecce May 17-18, 2018

Outline

1. Why do we want km-scale global weather and climate simulations?
2. Why is it hard?
3. What can be achieved today using a refactored code on Europe's largest supercomputer?
4. Open questions and challenges

(adapted from Klemp 2007)

The Greyzone



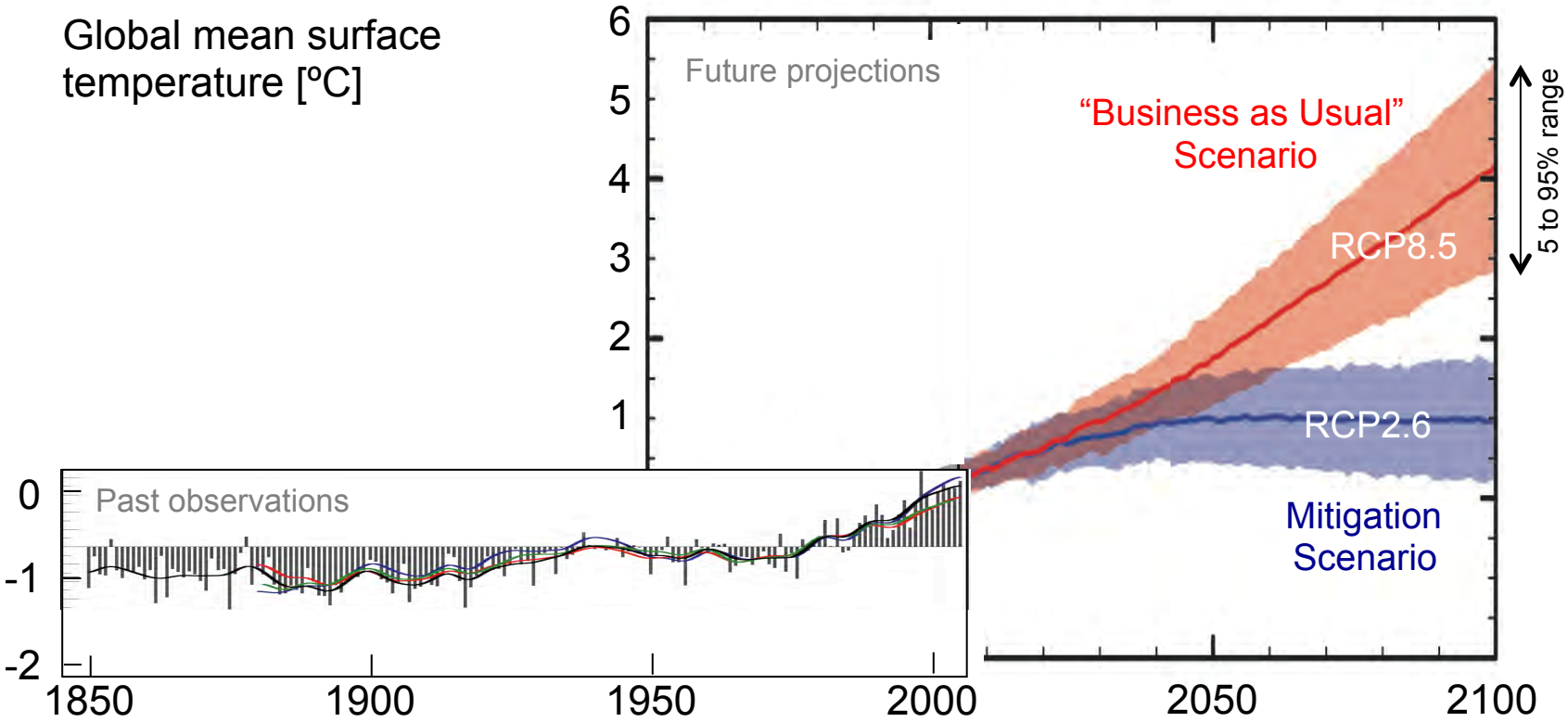
Atmospheric models with $\Delta x \leq 1$ km can steer clear of the greyzone for many processes

(source Ch. Schär, 2017)

Climate change

IPCC AR5, 2013

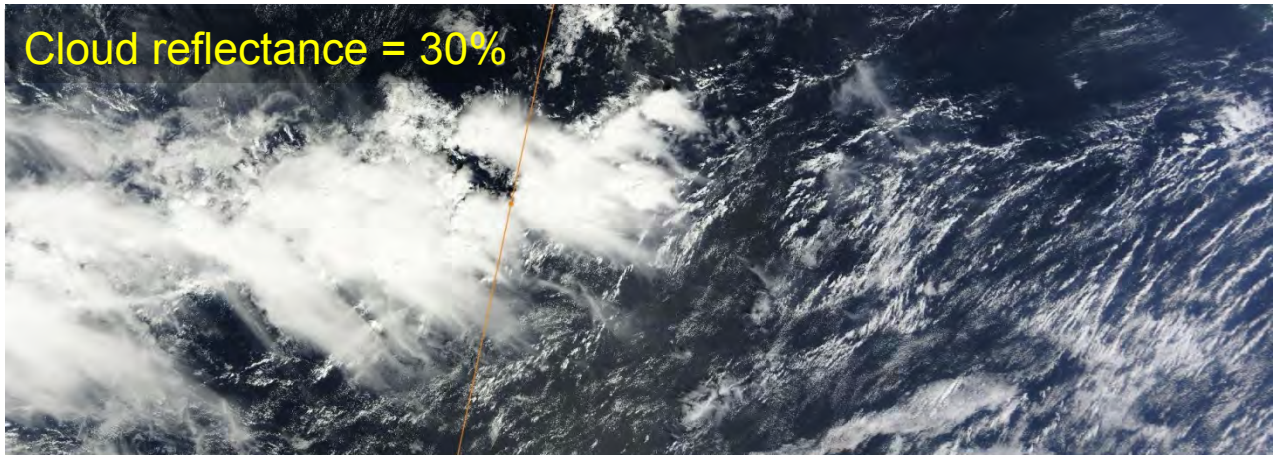
Global mean surface temperature [ºC]

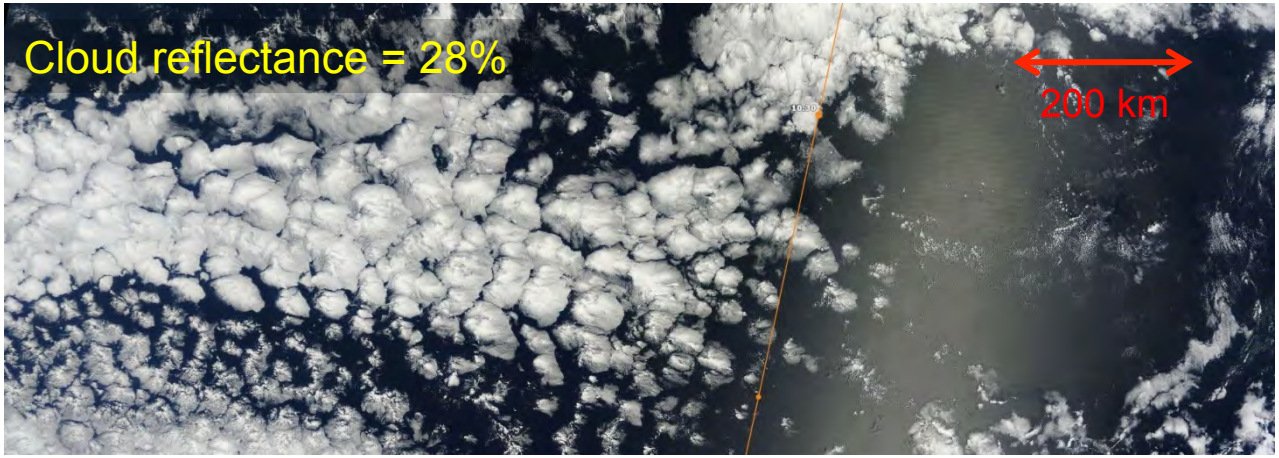


Uncertainties are primarily due to uncertainties in the response of clouds.
(e.g. Schneider et al. 2017, Nature CC)

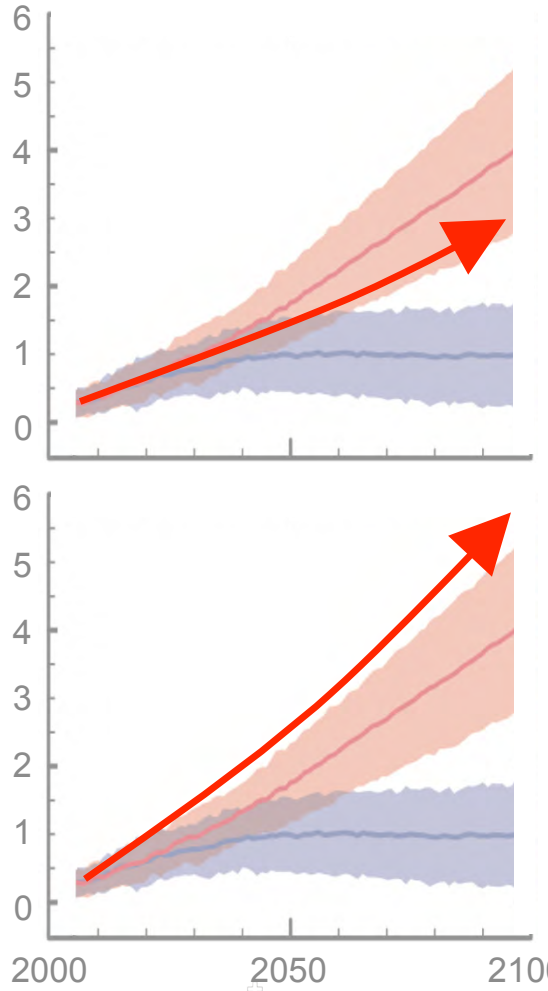
What is the difference?

adapted from Schär, 2017





Low climate sensitivity



High climate sensitivity

1. Why do we want km-scale global weather and climate simulations?

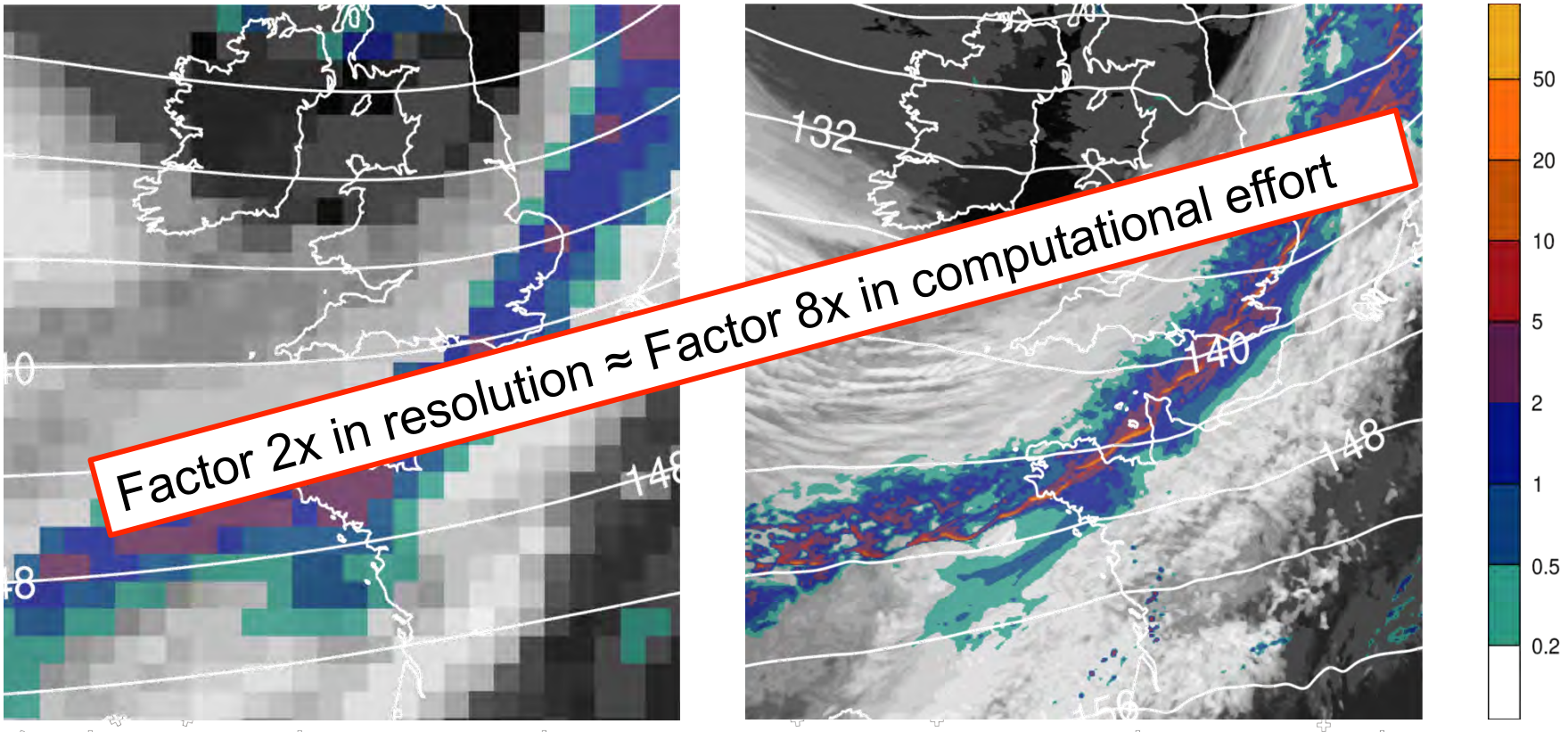
- Explicitly resolve important processes (e.g. deep convection, gravity waves, ocean eddies)
- Reduce uncertainty in cloud response in climate model projections

Computational effort

Current state-of-the-art (CMIP5)

$\Delta x \approx 25$ km

$\Delta x \approx 1$ km

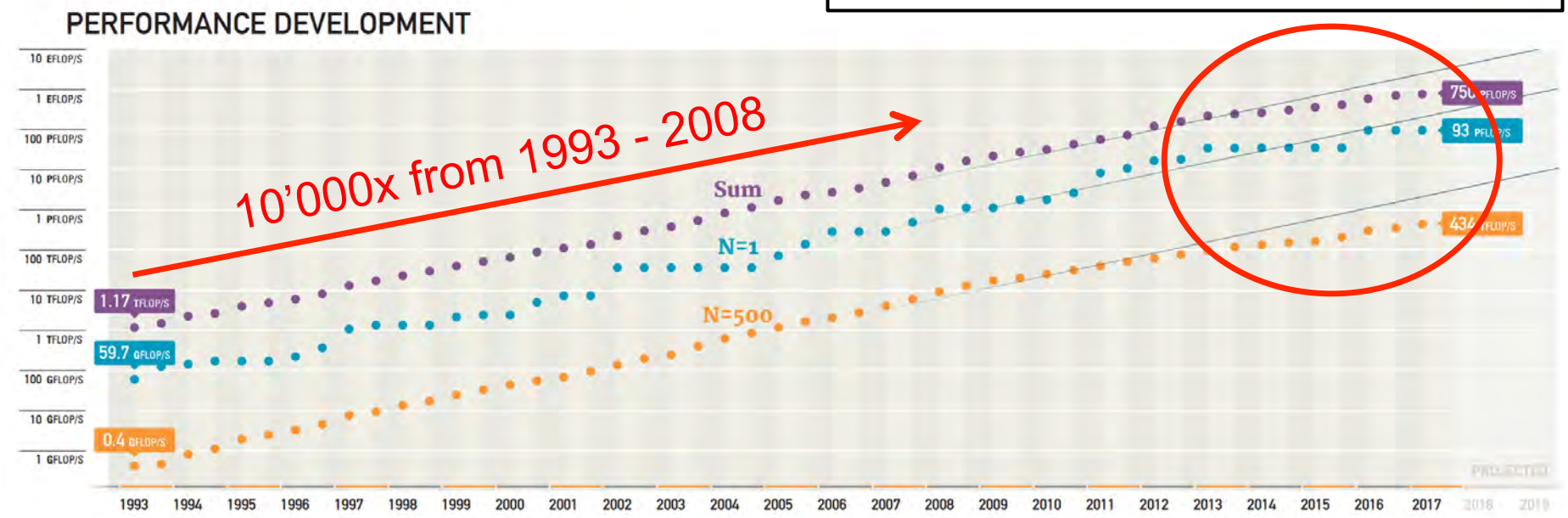


How to achieve a factor 10’000?

- Option 1: Money
- Option 2: Wait 15 years

Gordon Moore (1965, Intel co-founder)

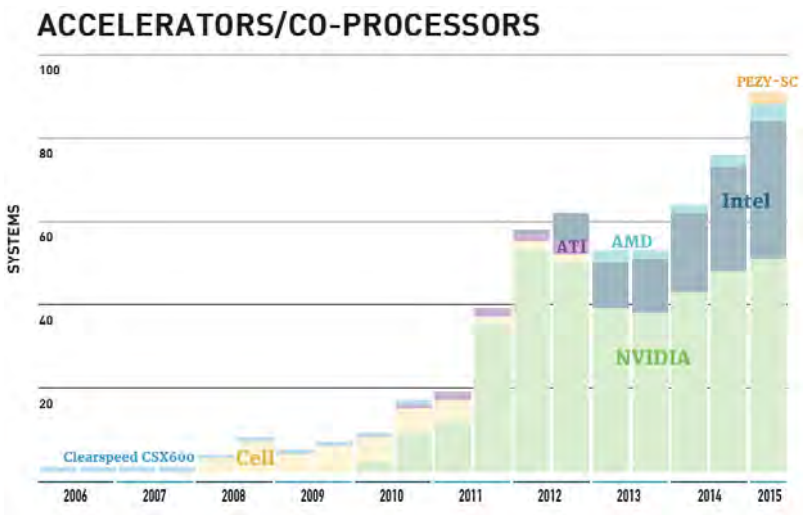
“Number of transistors in per inch^2 in a CPU doubles every 18-24 months at constant cost”

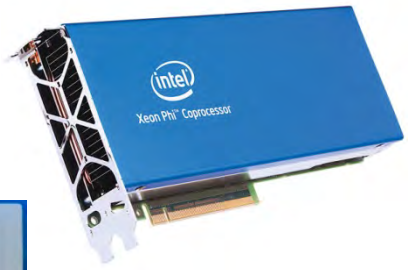


Source: top500.org

Specialized Hardware

20% Systems on Top 500





Many-core processors

GPUs

FPGAs

AI Accelerators
(e.g. Google Tensor Flow, Intel Nervana)

Koenig et al. 2017, UCB
(accelerator for exact dot product)

ARM
(e.g. Mont Blanc)

Supercomputer: Piz Daint

- 5320 hybrid nodes (Intel Haswell + NVIDIA Tesla P100)
- 1431 ”traditional” nodes (Intel Broadwell)
- Currently #3 on Top500 and #1 on Green500
- 90% of compute power from accelerator (GPUs)!!!

2. Why is it hard?

- Requires a massive increase in compute power
- Moore's law is dead!
- Requires continually (!) adapting our models to emerging hardware architectures and programming paradigms

Where are we today? COSMO on GPUs

- Large investement into software (MeteoSwiss, CSCS, ETH/C2SM)
- Adapted to run on GPUs
- Operational for weather forecast on a GPU-based system (Piz Kesch)
- Regional climate simulations on Piz Daint

HPC wire

Since 1986 - Covering the Fastest
in the World and the People Who

Search this site

Home News Technolog

April 1, 2016

**Swiss Weather Foreca
Kesch'**

John Russell

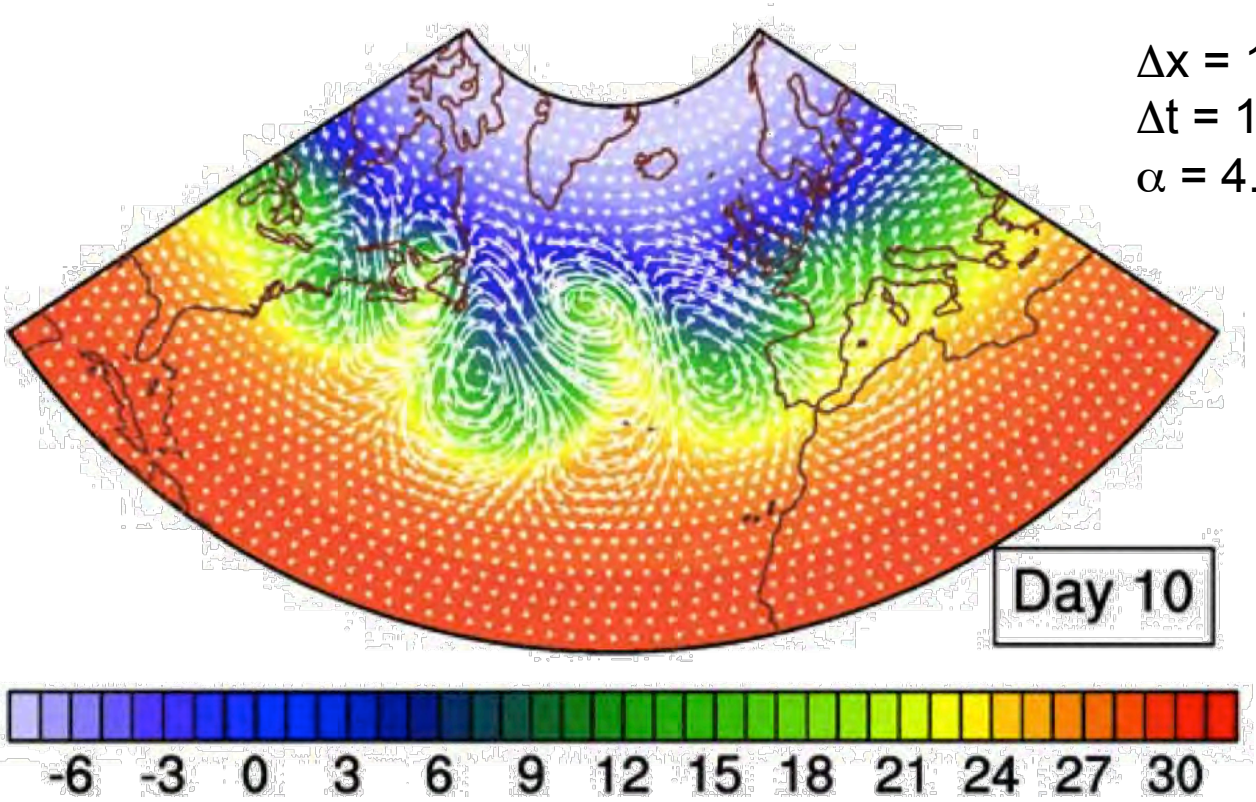
After six months of tweaking – producing a 20 percent reduction in time-to-solution for weather forecasting – MeteoSwiss, the Federal Office of Meteorology and Climatology, today reported its next generation COSMO-1 forecasting system is now operational. COSMO-1 requires 20 times the computing power of COSMO-2 and runs on the hybrid CPU-GPU supercomputer, Piz Kesch, operated by the Swiss National Supercomputing Centre (CSCS) and custom built in collaboration with Cray and NVIDIA.

COSMO-1 was put into service last September (see, Today's

Bibliotheca Alexandri
Solution to Build Mas
ASRock Rack to Exhit

Visit additional Tabor

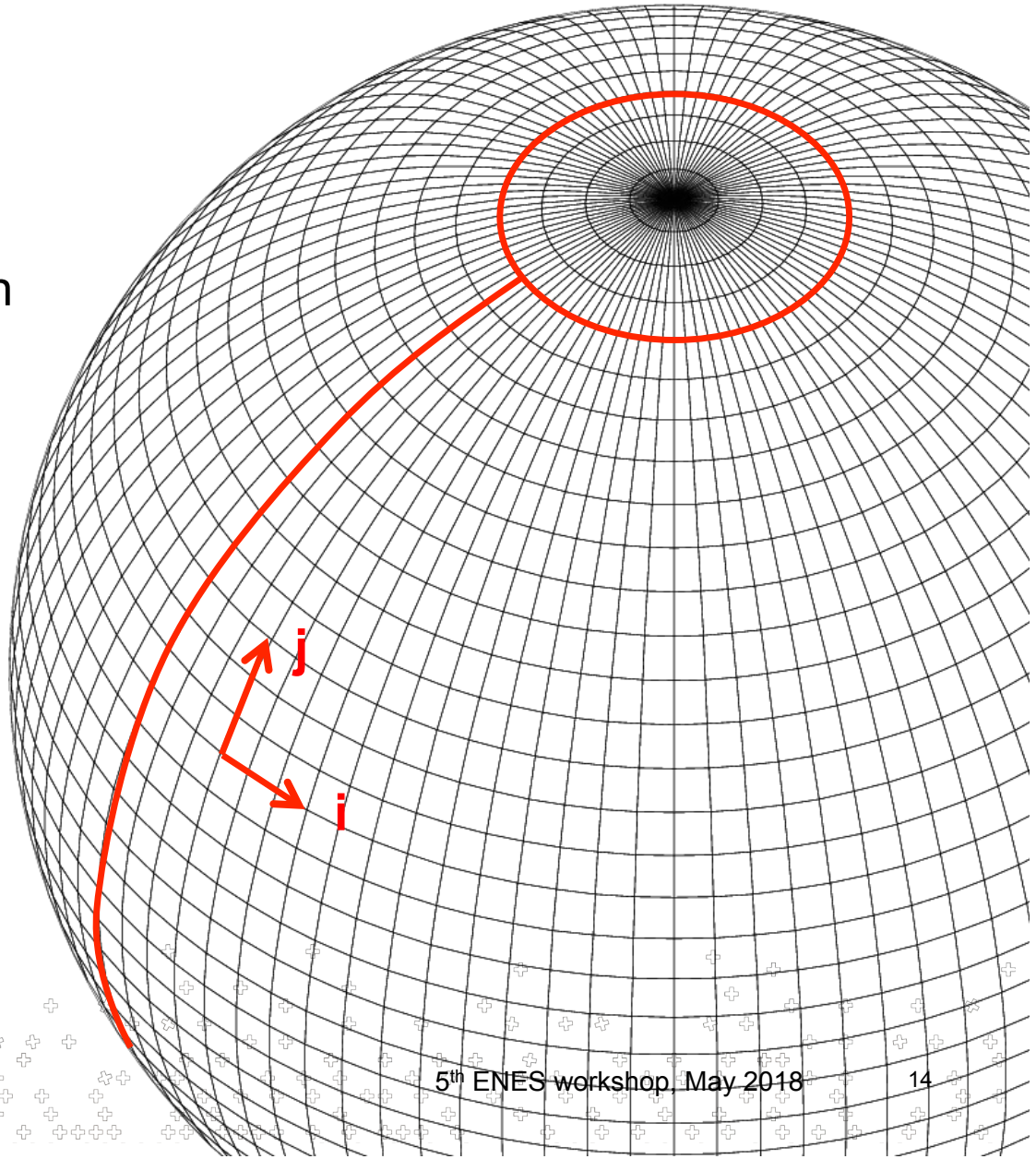
Baroclinic wave (Jablonowski 2006)



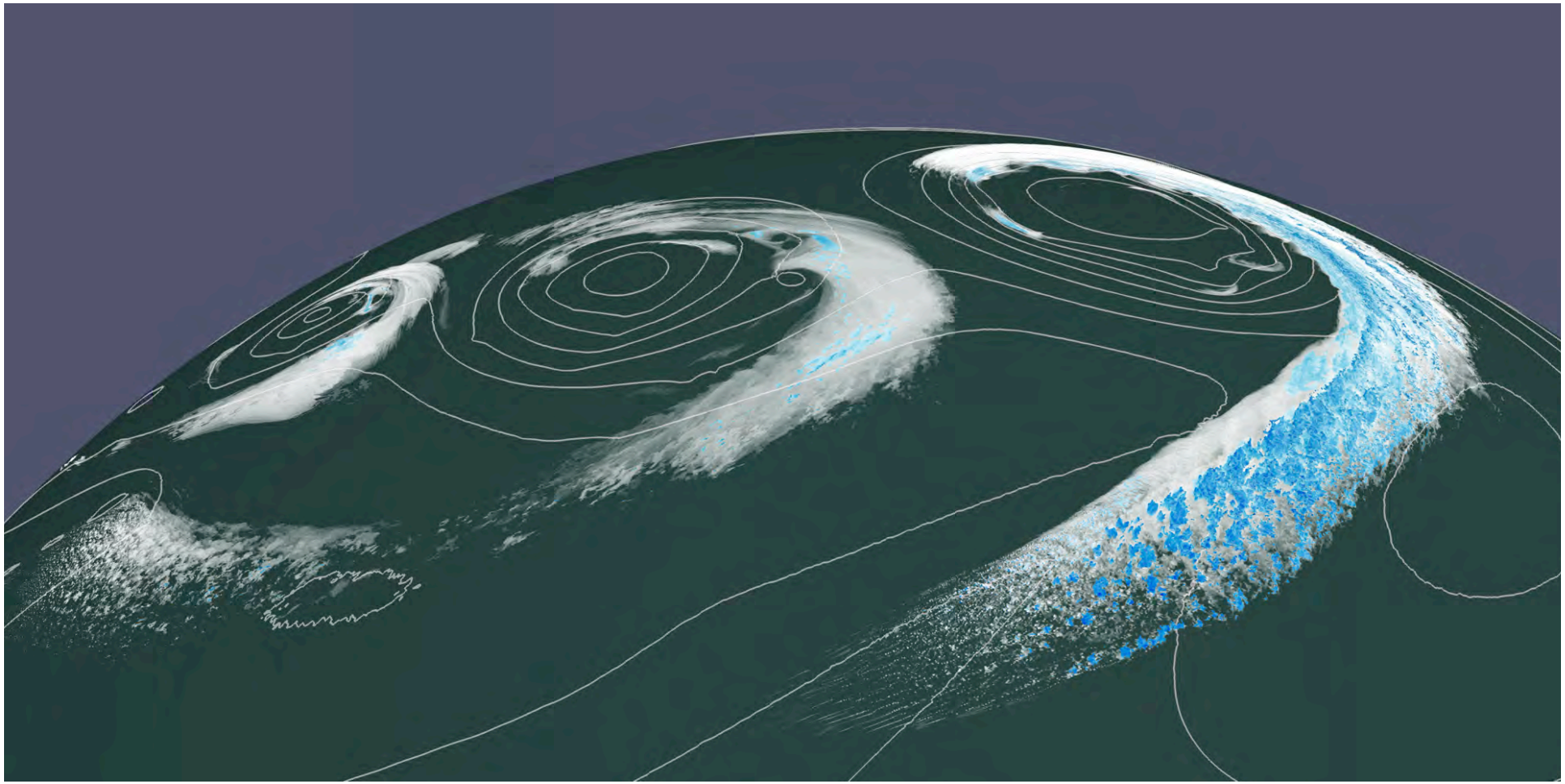
Temperature (°C) and wind field at 500 m at day 10 of the simulation.

Simulation Setup

- Single precision
- Regular lat/lon grid
- Periodic in i-direction
- 80°S to 80°N
- Covering 98.4% of Earth's surface
- Analytical initial condition
- 10 day simulation
- Minimal I/O



Visualization



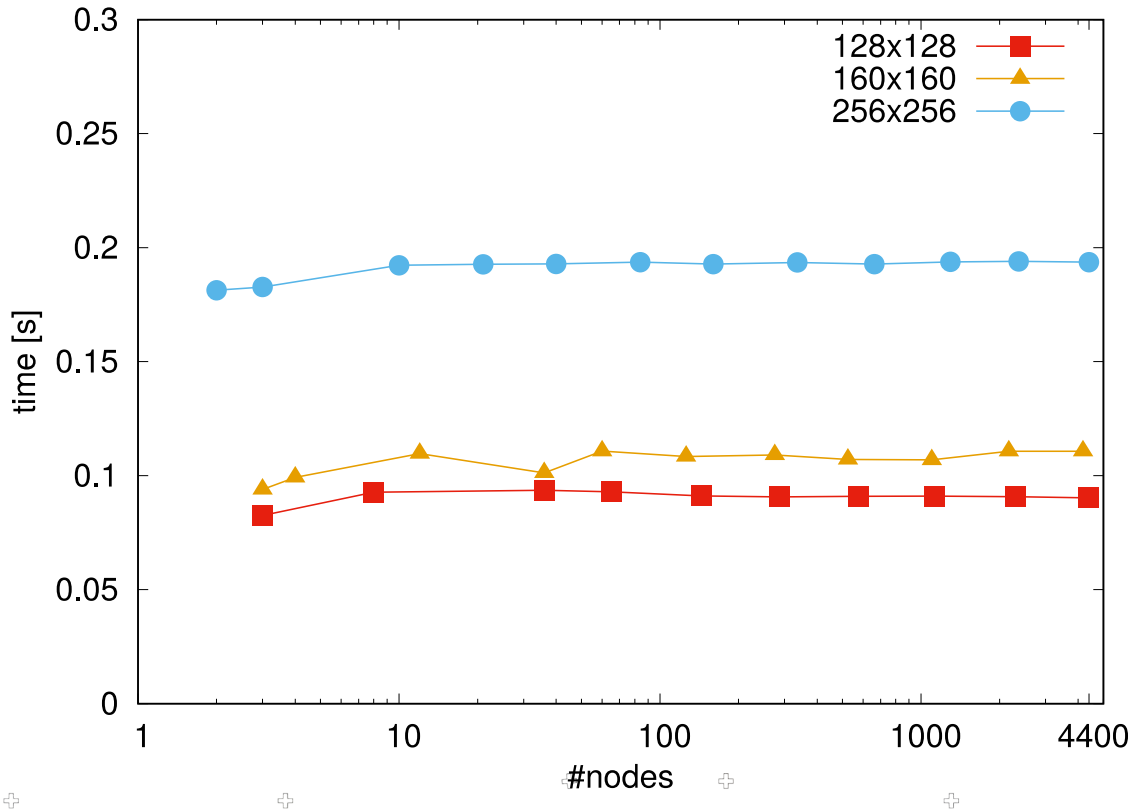
Visualization by Tarun Chadha (C2SM): clouds > 10^{-3} g/kg (white) and precipitation > $4\ 10^{-2}$ g/kg (blue)

Fuhrer et al., 2017, GMD, in rev.

Weak scaling

- Increase problem size and computational resources at the same time

Perfect weak scalability



Metric: time-to-solution

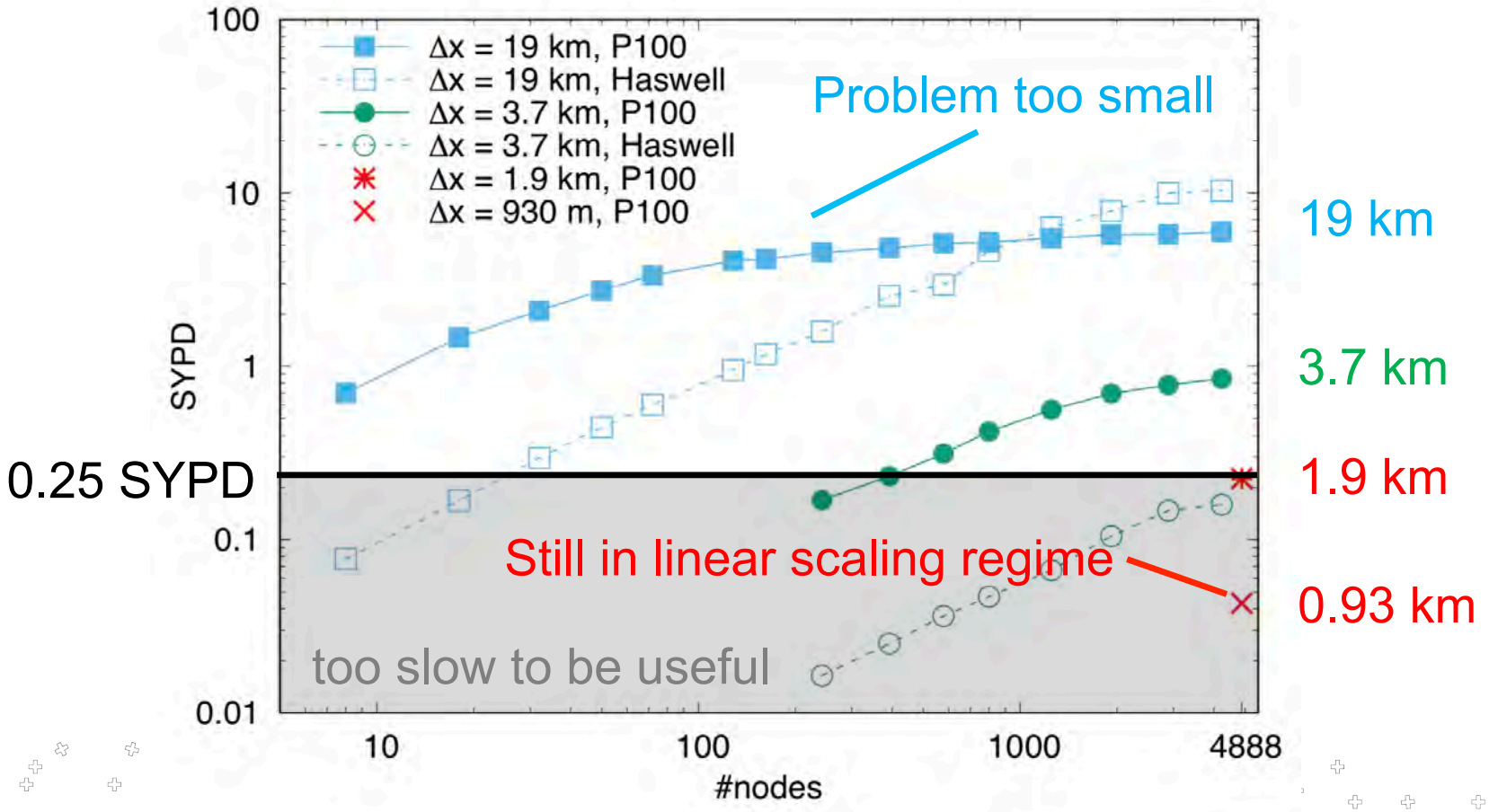
- SYPD = Simulated years per wallclock day

- E.g. AMIP (CMIP6)
  - 1979 – 2014 (36 years)
  - 4 – 6 months
  - 0.2-0.3 SYPD required

Fuhrer et al., 2017, GMD, in rev.

Strong scaling

- Near-global simulations at a fixed horizontal resolution



Source: Vogt, 2017

Results (and beyond)

Δx	#nodes	Δt [s]	SYPD	MWh/SY
1.9 km	4,888	12	0.23	97.8
930 m	4,888	6	0.043	596

What would it take to do a 36 year AMIP simulation?

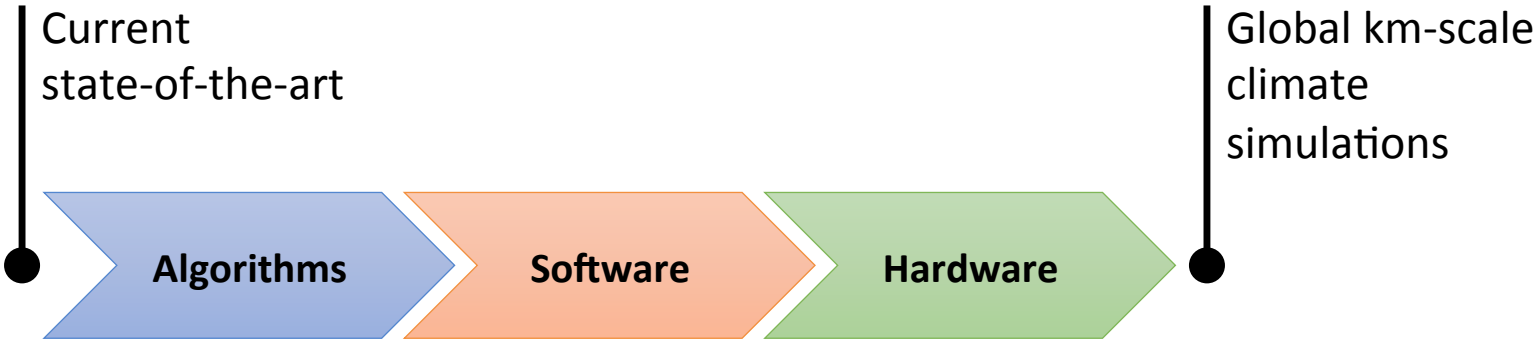
	Δx = 1.9 km	Δx = 930 m
Time to solution	156 days	840 days
Energy to solution	3.5 GWh (150 kCHF)	22 GWh (940 kCHF)
CO2eq* to solution	640 tons	3'800 tons
CHF to solution (2go.cscs.ch)	18 M node hours → 12 MCHF	97 M node hours → 68 MCHF

* See https://www.bafu.admin.ch/bafu/de/home/themen/klima/klimawandel--fragen-und-antworten.html

3. Where are we today?

- No global weather and climate model ready for production on state-of-the-art hardware
- Global km-scale AMIP-type simulations are feasible (0.23 SYPD at Δx = 1.9 km)
- Cost of simulation is significant

How do we get there?



- Many opportunities and challenges on the way, here are some examples…

Software (Domain-specific languages)

- Our models comprise 0.1 – 1 M lines of legacy code
- Effort to adapt is huge!
- Domain-specific languages (DSLs) can reduce code by 10x
- E.g. “Towards a performance portable, architecture agnostic implementation strategy for weather and climate models ” (Fuhrer et al. 2010, Superfri)

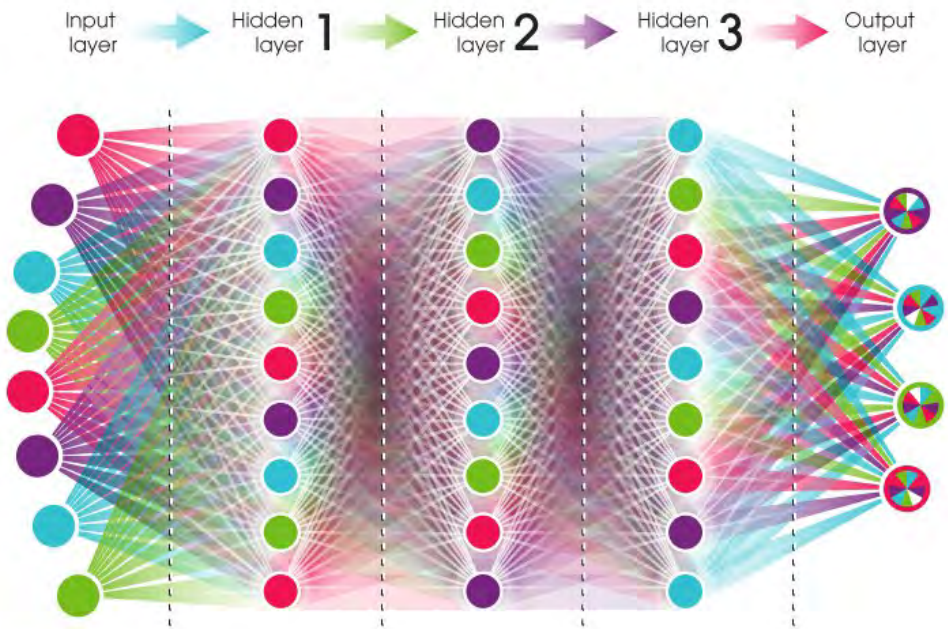
```
function avg {
  offset off
  storage in

  avg = 0.5 * ( in(off) + in() )
}

function coriolis_force {
  storage fc, in

  coriolis_force = fc() * in()
}

operator coriolis {
  storage u_tend, u, v_tend, v, fc

  vertical_region ( k_start , k_end ) {
      u_tend += avg(j-1, coriolis_force(fc, avg(...
      v_tend -= avg(i-1, coriolis_force(fc, avg(...
    }
  }
}
```

Example: Coriolis force

- No loops
- No data structures
- No halo-updates
- Different hardware backends (x86 multi-core, NVIDIA Tesla, Xeon Phi)

Algorithms (Artificial Neural Networks)

- Accelerators for ANNs
- Physical parametrizations can be 50% of computational cost
- Inherent uncertainties can be large
- E.g. “Accurate and Fast Neural Network Emulations of Model Radiation for the NCEP Coupled Climate Forecast System: Climate Simulations and Seasonal Predictions” (Krasnopolsky et al. 2010, MWR)



4. Open questions and challenges

- We need to rethink how we formulate and implement our models
- Many research opportunities on the way

Summary

1. Why do we want km-scale global weather and climate simulations?
2. Why is it hard?
3. What can be achieved today using a refactored code on Europe's largest supercomputer?
4. Open questions and challenges

Thank you! Questions?

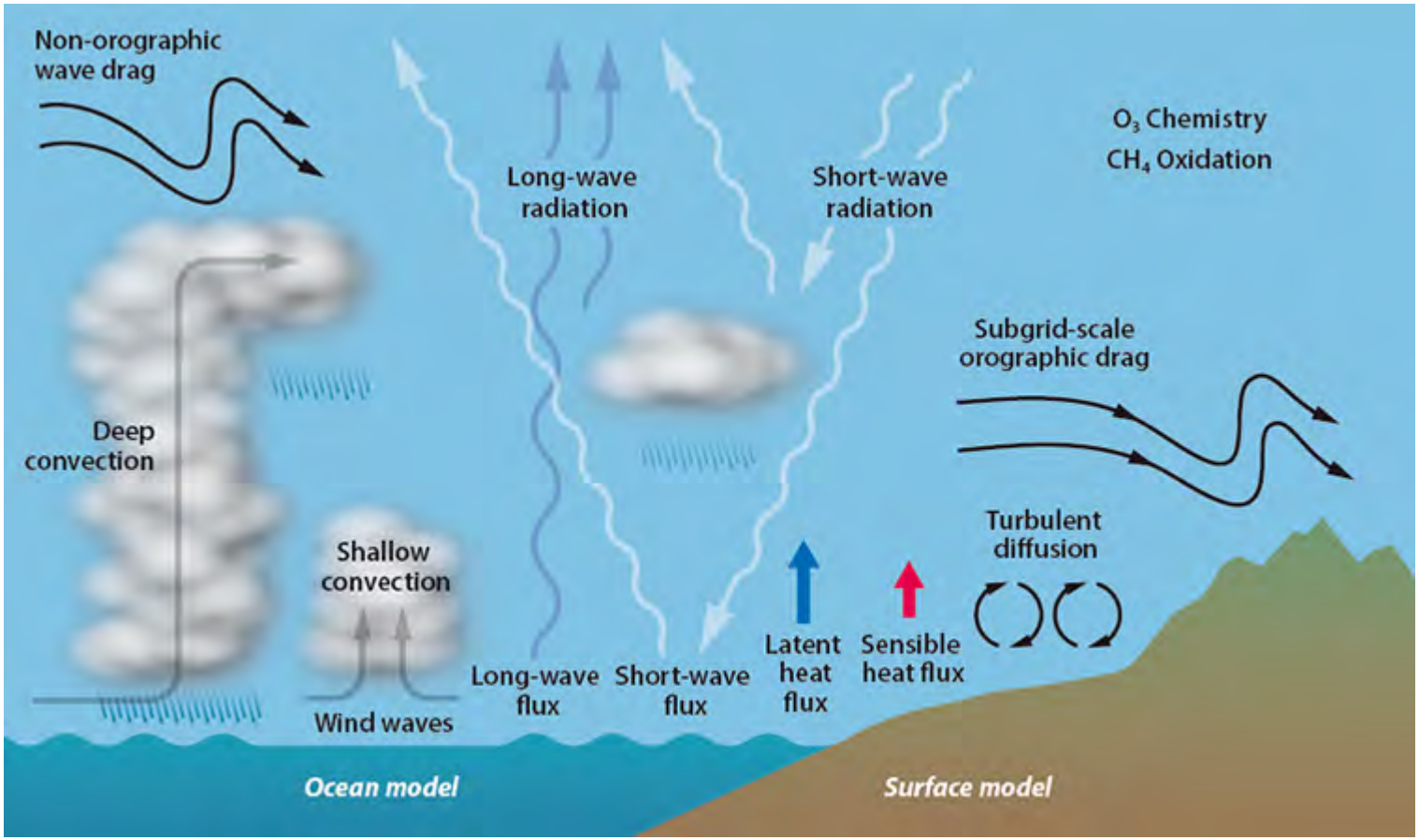
MeteoSwiss
Operation Center 1
CH-8058 Zurich-Airport
T +41 58 460 91 11
www.meteoswiss.ch

MeteoSvizzera
Via ai Monti 146
CH-6605 Locarno-Monti
T +41 58 460 92 22
www.meteosvizzera.ch

MétéoSuisse
7bis, av. de la Paix
CH-1211 Genève 2
T +41 58 460 98 88
www.meteosuisse.ch

MétéoSuisse
Chemin de l'Aérologie
CH-1530 Payerne
T +41 58 460 94 44
www.meteosuisse.ch

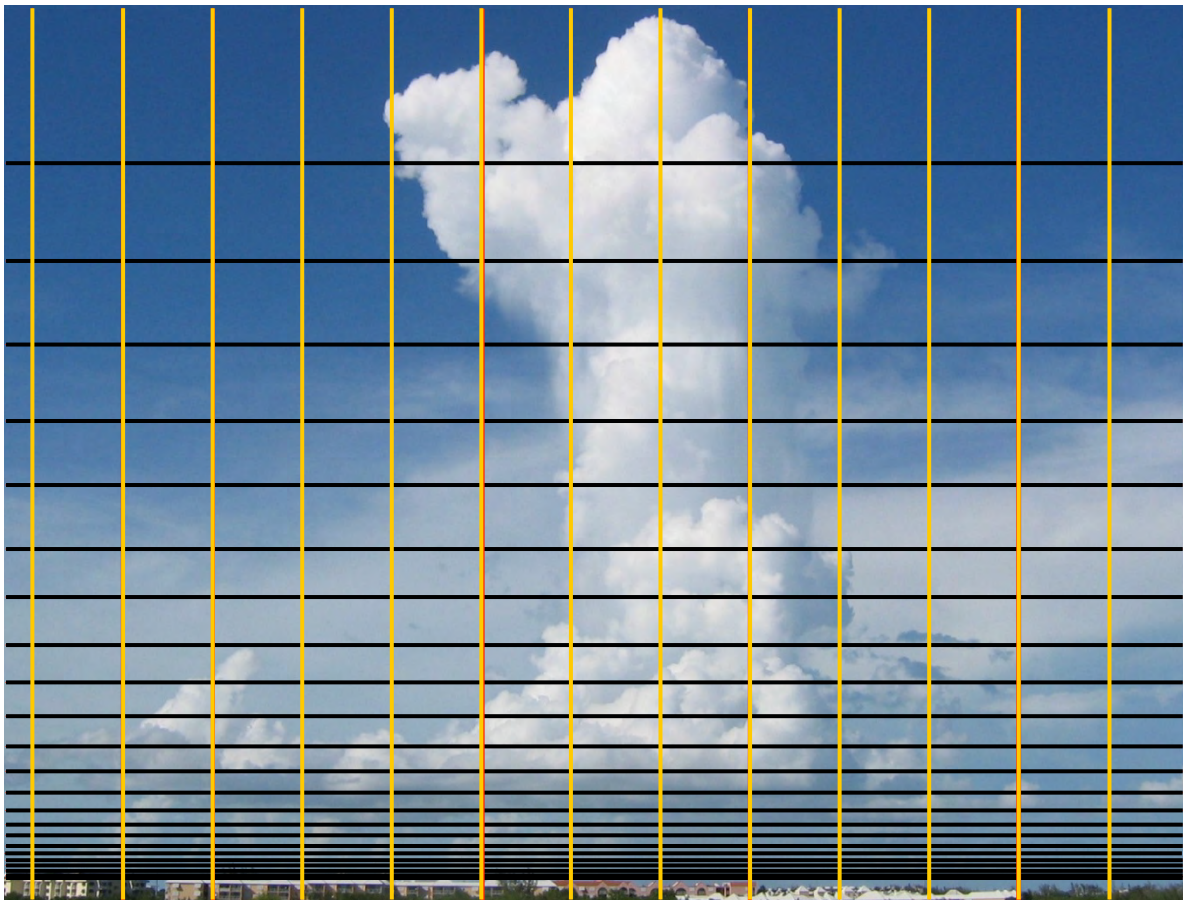
Modeling the Earth system



(source https://www.ecmwf.int/sites/default/files/medialibrary/2017-09/atmospheric-physics-754px.jpg)

(source https://www.weather.gov/images/key/Cloud_Chart/Low/Large/L2d.jpg)

Horizontal resolution



not resolved
→ parametrization

partially resolved
→ ?

resolved

Δx = 10.0 km

Δx = 3.0 km

Δx = 1.0 km

(Marvel 2017, doi:10.1038/scientificamerican1217-72)

Cloud response to climate change

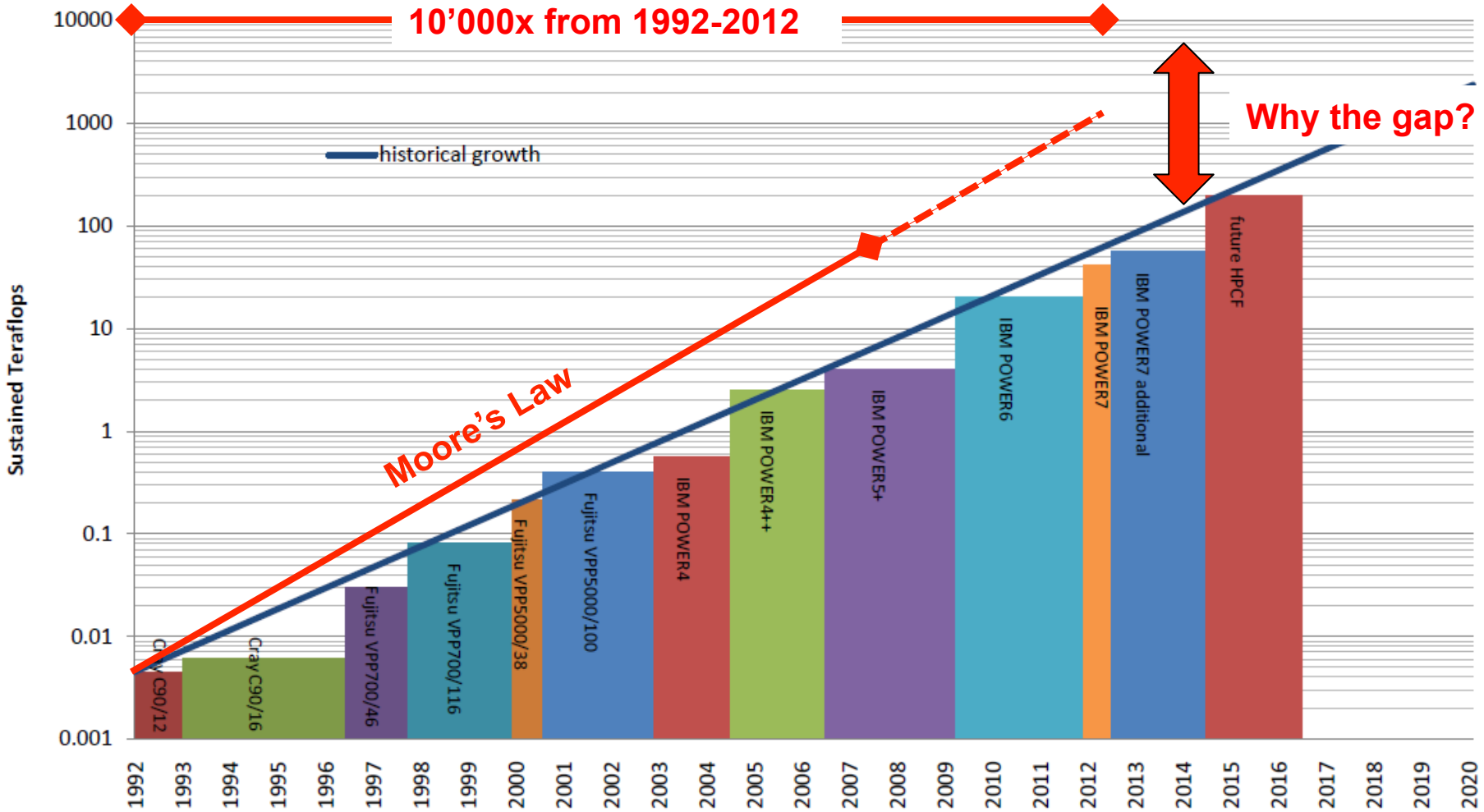
...

likely, observational evidence (e.g. Norris et al. 2016, Nature)

- Small changes in cloud reflectance can have a large impact
- State-of-the-art climate models use Δx = 25 km and do not explicitly resolve these clouds

Source: ECMWF, 2016

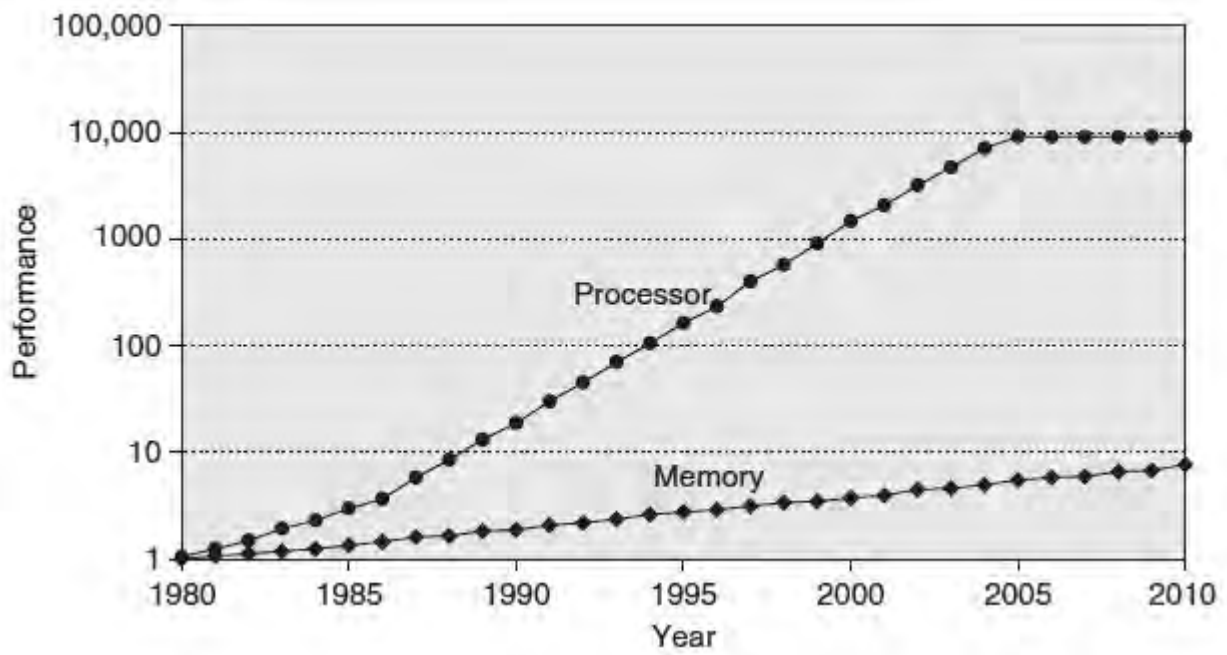
But wait a minute...



Source: Hennessy & Patterson, 2011

The memory wall

- Off-chip memory bandwidth is not increasing at the same rate as FLOP performance



Consequence for atmospheric models

- "Dynamics" code (niter = 48, nwork = 4096000)

```
do j = 1, niter
  do i = 1, nwork
    c(i) = a(i) + b(i) * ( a(i+1) - 2.0d0*a(i) + a(i-1) )
  end do
end do
```

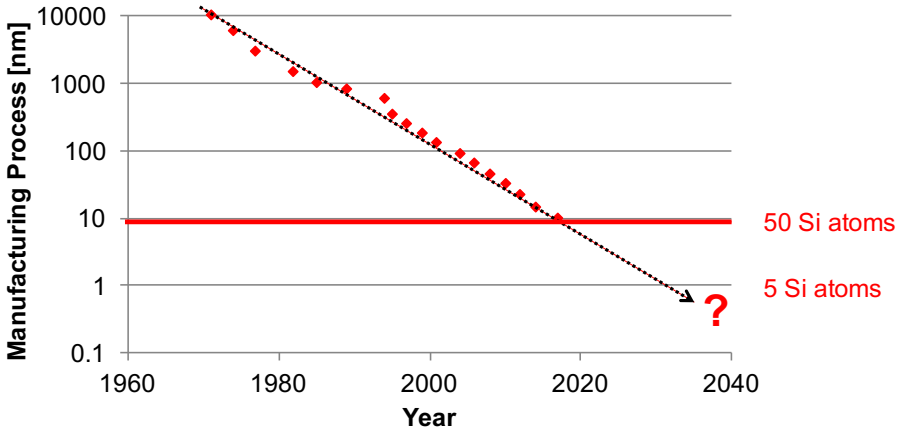
$$\frac{\partial a}{\partial t} = k\frac{\partial^2 a}{\partial x^2}$$

Machine	Cray XT4	Cray XT5	Cray XE6	Cray XK6
# cores	4	6	12	16
Single core	0.80 s	0.84 s	0.63 s	0.65 s
All cores	0.56 s	0.46 s	0.18 s	0.16 s
Speedup	**1.4**	**1.8**	**3.4**	**4.0**

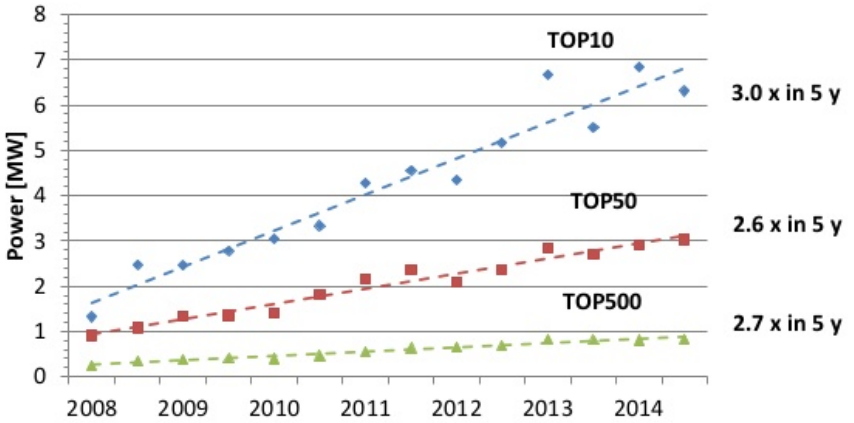
Runtime is dominated by data-movement and not FLOPs

Moore's law is dead!

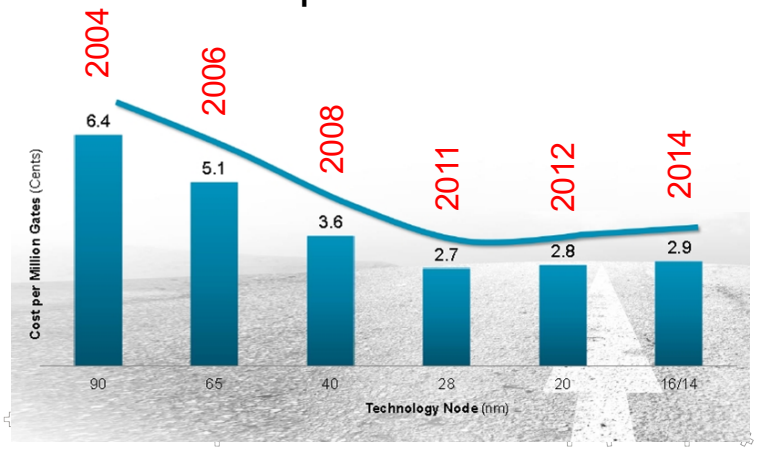
Decreasing feature size



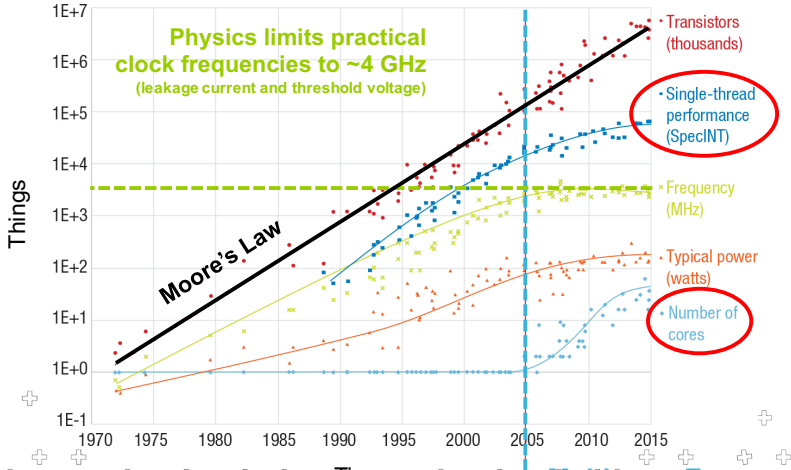
Increasing power consumption



Price per transistor



Stagnating clock frequencies

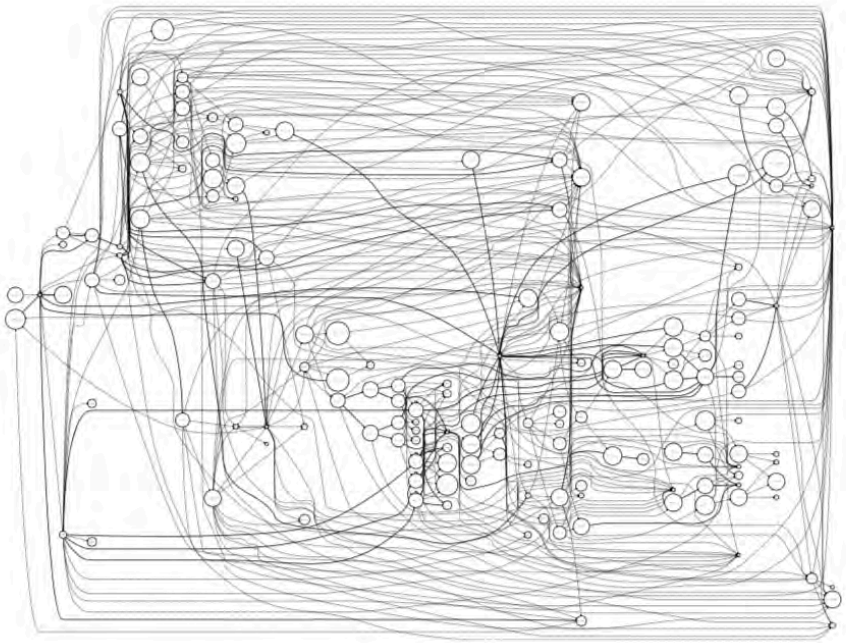


HPC Challenges

- **Rapid change**
  - Timescale of HPC system is 3-4 years → lagging behind
- **New design constraints (not FLOPs!)** → not efficient → wrong algorithms
  - Maximize parallelism
  - Minimize data movement and energy consumption
  - Minimize synchronizations
- **New and disruptive programming models**
  - Emerging “exotic” HW architectures → cannot run
  - E.g. OpenMP 4.5, Coarray Fortran, CUDA, OpenACC

Software (Automatic optimization)

- Compilers will not solve the problem!
- DSL-based code can be automatically optimized for a specific hardware target
- E.g. “Design of a Compiler Framework for Domain Specific Languages for Geophysical Fluid Dynamics Models” (Fabian Thüring, MSc thesis)

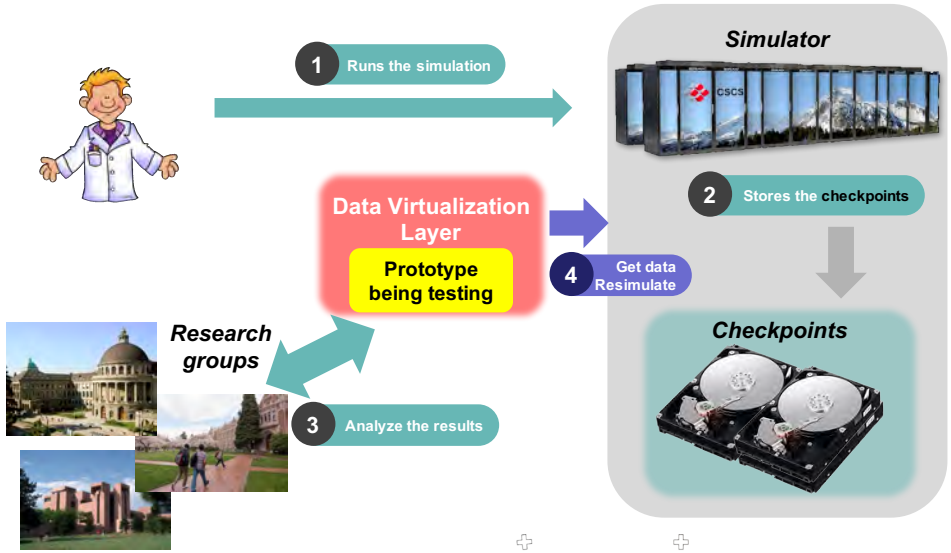


Example: Fast-waves solver

- Graph representation of code
- Rearrange for data-locality
- Run independent computations in parallel

Software (I/O, data compression)

- We expect data volumes 1-2 PB/year for climate simulations at $\Delta x = 1$ km
- Traditional workflow (compute → store → analyze) will break!
- E.g. “Data compression for climate data” (Kuhn et al. 2016, Superfri) and “Convection-resolving climate modeling on future supercomputing platforms (crCLIM)” (SNF Sinergia project, Lead: Ch. Schär)



Up or down?

- **Increase level of abstraction**
  - Hide implementation details
  - Can be disruptive
- **Decrease level of abstraction**
  - Add implementation details
  - Often incremental

DOWN – Decrease level of abstraction

- **Approaches**
  - Fortran + MPI + Directives (OpenMP, OpenACC)
  - Optimize code for a specific hardware
  - Custom implementations (#ifdef) or programming languages

```
!---- Local automatic arrays
!$acc present ( pa1c,pa1f,pa2c,pa2f,pa3c,pa3f                    ) &
!$acc present ( ztu1,ztu2,ztu3,ztu4,ztu5,ztu6,ztu7,ztu8,ztu9     ) &
!---- Module arrays
!$acc present ( cobi,coali,cobti                                  )

!$acc parallel
!$acc loop gang vector(32) collapse(2)
DO j3 = ki3sc, ki3ec+1
  DO j1 = ki1sc, ki1ec
    pflfd(j1,j3) = pbbr(j1,j3)
    pflcd(j1,j3) = 0.0_dp
  ENDDO
ENDDO
!$acc end parallel

#ifdef _OPENACC
!$acc parallel
!$acc loop gang vector(32)
DO j1 = ki1sc, ki1ec
  CALL coe_th_gpu(pduh2oc (j1,ki3sc), pduh2of (j1,ki3sc), &
                  pduco2  (j1,ki3sc), pduo3   (j1,ki3sc), &
                  palogp  (j1,ki3sc), palogt  (j1,ki3sc), &
                  podsc   (j1,ki3sc), podsf   (j1,ki3sc), &
                  podac   (j1,ki3sc), podaf   (j1,ki3sc), &
                  pbsfc   (j1,ki3sc), pbsff   (j1,ki3sc), &
                  kspec   , kh2o    , kco2    , ko3     , &
                  pa1c(j1), pa1f(j1), pa2c(j1),           &
                  pa2f(j1), pa3c(j1), pa3f(j1)            )
ENDDO
!$acc end parallel
#else
  CALL coe_th ( pduh2oc,pduh2of,pduco2 ,pduo3  ,palogp ,palogt , &
                podsc  ,podsf  ,podac  ,podaf  ,pbsfc  ,pbsff  , &
                ki3sc  ,kspec  ,kh2o   ,kco2   ,ko3    ,       &
                ki1sd  ,ki1ed  ,ki3sd  ,ki3ed  ,ki1sc  ,ki1ec  , &
                ldebug_coe_th  ,jindex ,                        &
                pa1c   ,pa1f   ,pa2c   ,pa2f   ,pa3c   ,pa3f)
#endif
```

Example: OpenACC

- More details
  - data movement
  - data structures
- Less encapsulation
- Hardware dependent code
  - #ifdef
- …

- **“Easy”**
- **Incremental**
- **Hard to understand / modify**
- **Hard to maintain**

DOWN – Challenges

- **Is it possible to reach a good compromise?**
  - Multiple hardware architectures
  - Good performance
  - Maintainable / modifiable code

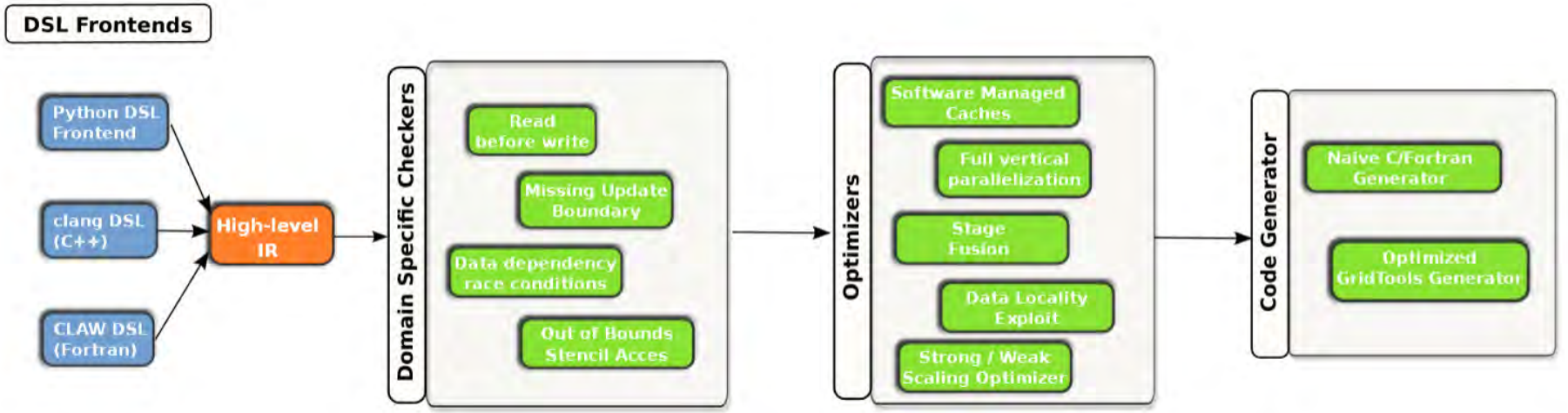
UP – Increase level of abstraction

- **Approaches**
  - Compilers
  - Libraries / Frameworks
  - Code generators and source-to-source translators
  - Domain-specific languages (DSL)

Source: Thuering, 2017

Domain-specific language (DSL)

- DSL compiler compiles into standard programming language

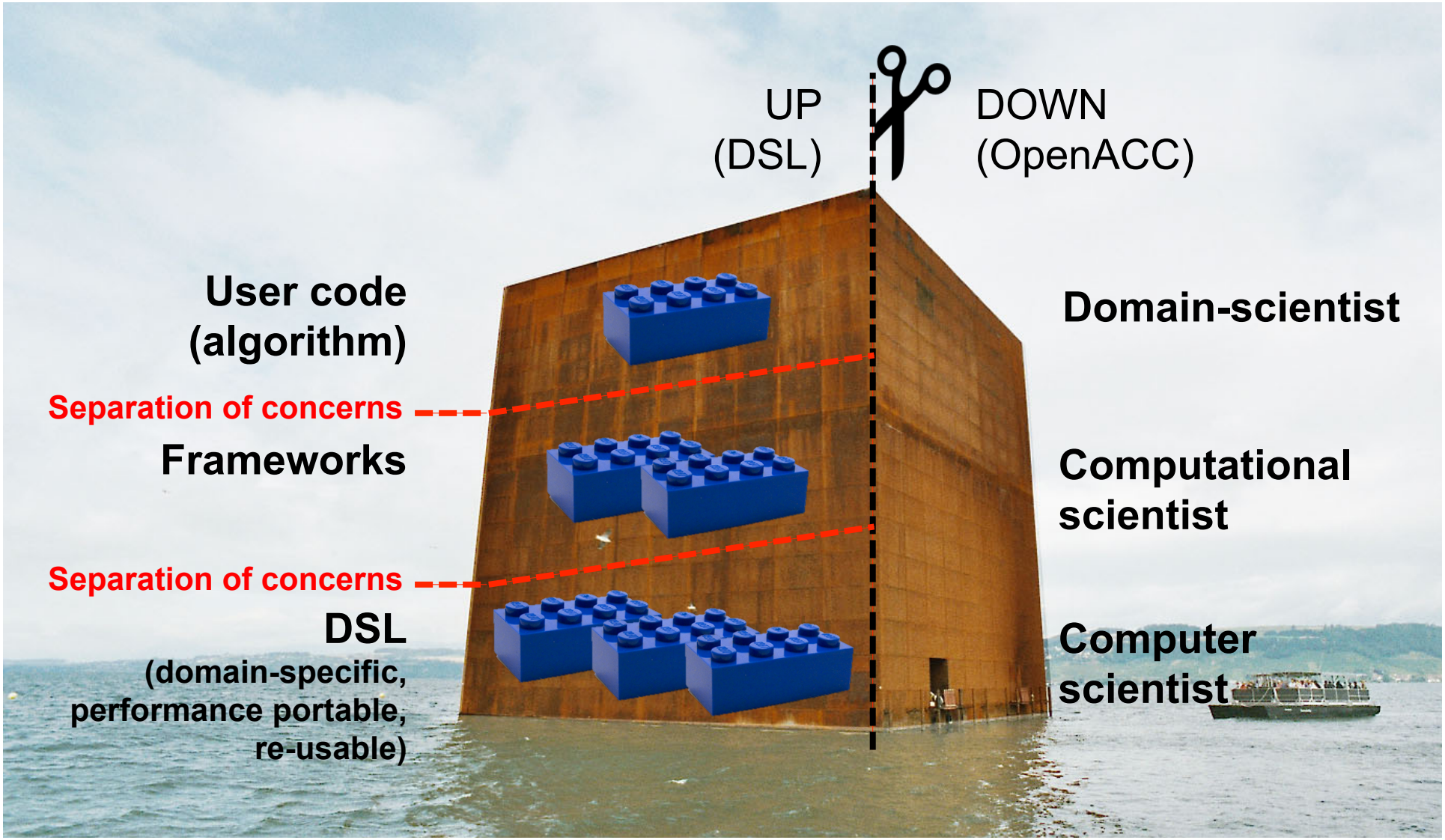


Version	Runtime [ms]	Lines of Code
gtclang	0.460	23
GridTools	0.461	98
STELLA	0.442	116

UP – Challenges

- **Feasibility**
  - No turn key solutions.
  - Can we achieve good performance on different hardware architectures with a high-level specification of our algorithms?
- **Acceptance**
  - Disruptive change
  - Can we achieve a community solution?

Example: COSMO



Traditional vs. next-generation?

	Piz Dora (old code)	Piz Kesch (new code)	Factor
Sockets	~26 CPUs	~7 GPUs	**3.7 x**
Energy	10 kWh	2.1 kWh	**4.8 x**

→ Investment into hardware **and software**!

Hardware	~ 2-3 MCHF
Software (projects, in-kind)	~ 5-7 MCHF