ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

CSCS
Swiss National Supercomputing Centre

# GPU considerations for next generation weather simulations

Thomas C. Schulthess

# What exactly is High-Performance Computing?

- "... it depends on who you talk to" Richard Dracott (Intel)
- "At IBM we called it scientific computing" Ulla Thiel (Cray)

**HPC = Scientific computing where computer performance matters**

**Supercomputers are by definition the fastest and most powerful general-purpose scientific computing systems available at any given time.** (1)

(1) Dongarra, Duff, Sorensen and van der Vorst in "Numerical Linear Algebra for High-Performance Computers"

# Applications running at scale on Jaguar @ ORNL

Fall 2009

| Domain area | Code name | Institution | # of cores | Performance | Notes |
|---|---|---|---|---|---|
| Materials | DCA++ | ORNL | 213,120 | **1.9 PF** | 2008 Gordon Bell Prize Winner |
| Materials | WL-LSMS | ORNL/ETH | 223,232 | **1.8 PF** | 2009 Gordon Bell Prize Winner |
| Chemistry | NWChem | PNNL/ORNL | 224,196 | 1.4 PF | 2008 Gordon Bell Prize Finalist |
| Materials | OMEN | Duke | 222,720 | 860 TF | |
| Chemistry | MADNESS | UT/ORNL | 140,000 | 550 TF | |
| Materials | LS3DF | LBL | 147,456 | 442 TF | 2008 Gordon Bell Prize Winner |
| Seismology | SPECFEM3D | USA (multiple) | 149,784 | 165 TF | 2008 Gordon Bell Prize Finalist |
| Combustion | S3D | SNL | 147,456 | 83 TF | |
| Weather | WRF | USA (multiple) | 150,000 | 50 TF | |

# Algorithmic motifs and their arithmetic intensity

Arithmetic intensity: number of operations per word of memory transferred

**Finite difference / stencil in S3D and WRF**

**Rank-1 update in HF-QMC** → **Rank-N update in DCA++**

**QMR in WL-LSMS**

**Linpack (Top500)**

Sparse linear algebra

Matrix-Vector

| Vector-Vector | Fast Fourier Transforms | Dense Matrix-Matrix |
|---|---|---|
| BLAS1&2 | FFTW | BLAS3 |
| **O(1)** | **O(log N)** | **O(N)** |

**Fact: supercomputers are designed for certain algorithmic motifs**

# Relationship between simulations and supercomputer system

Simulations + Theory + Experiment

Science

Model & method of solution

Mapping applications to computer system

**> Algorithm re-engineering**

**> Software refactoring**

**> Domain specific libraries/languages, etc.**

Models & methods typically live for decades, codes are part of the technology stack
> requires interdisciplinary effort / team

Runtime system

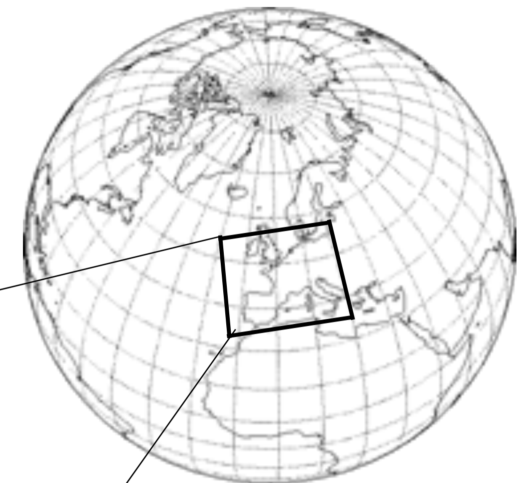
Operating systems

Computer Hardware

Supercomputer

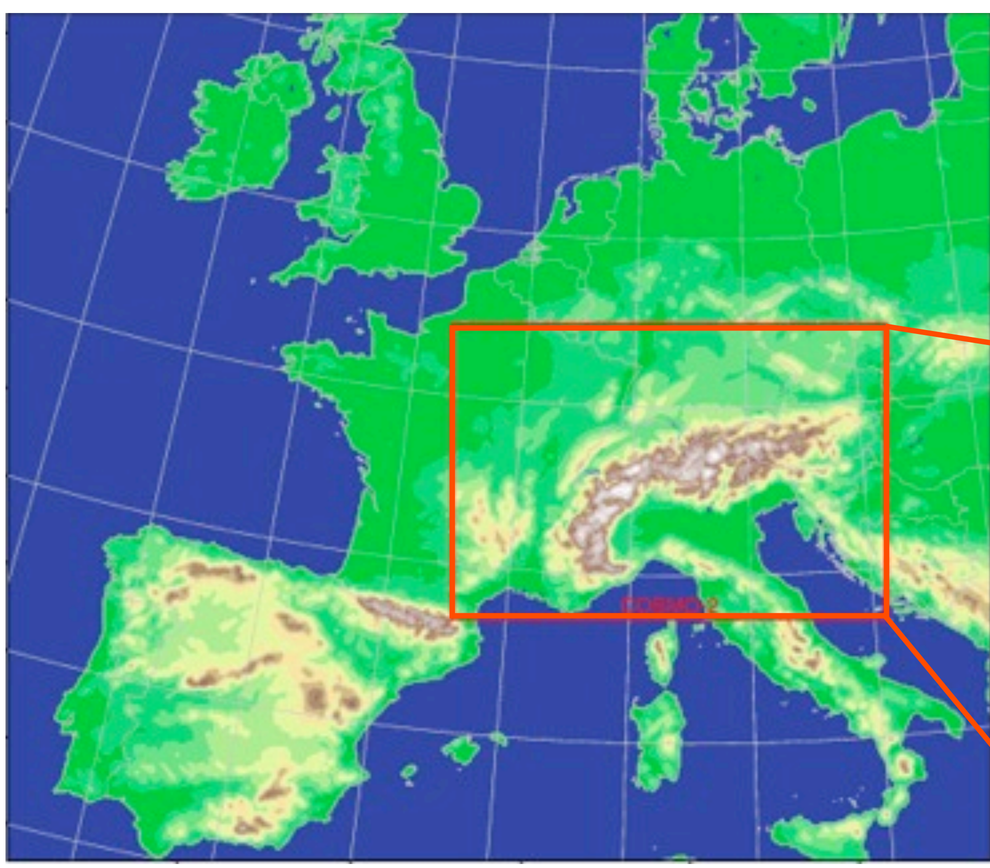
**Co-Design**

# Numerical weather predictions for Switzerland (MeteoSwiss)

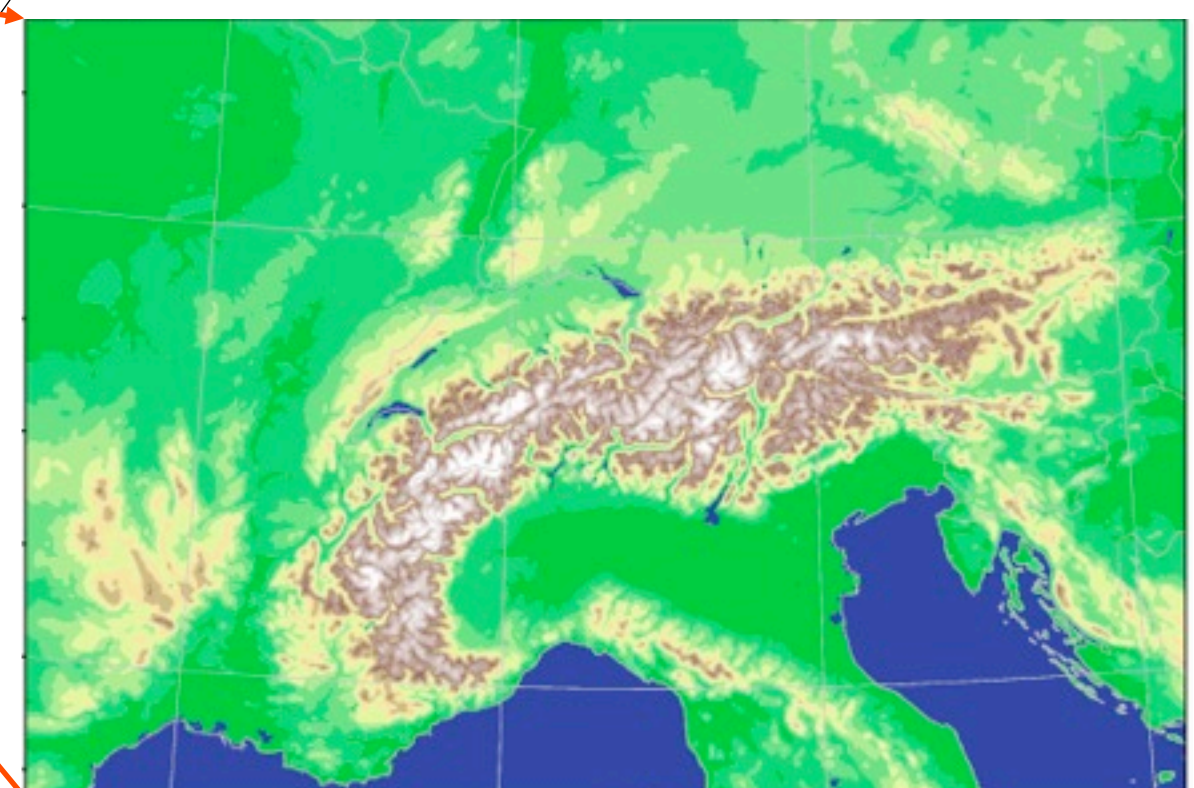


**ECMWF: boundary conditions**
**16km, 91 layers**
**2x per day**

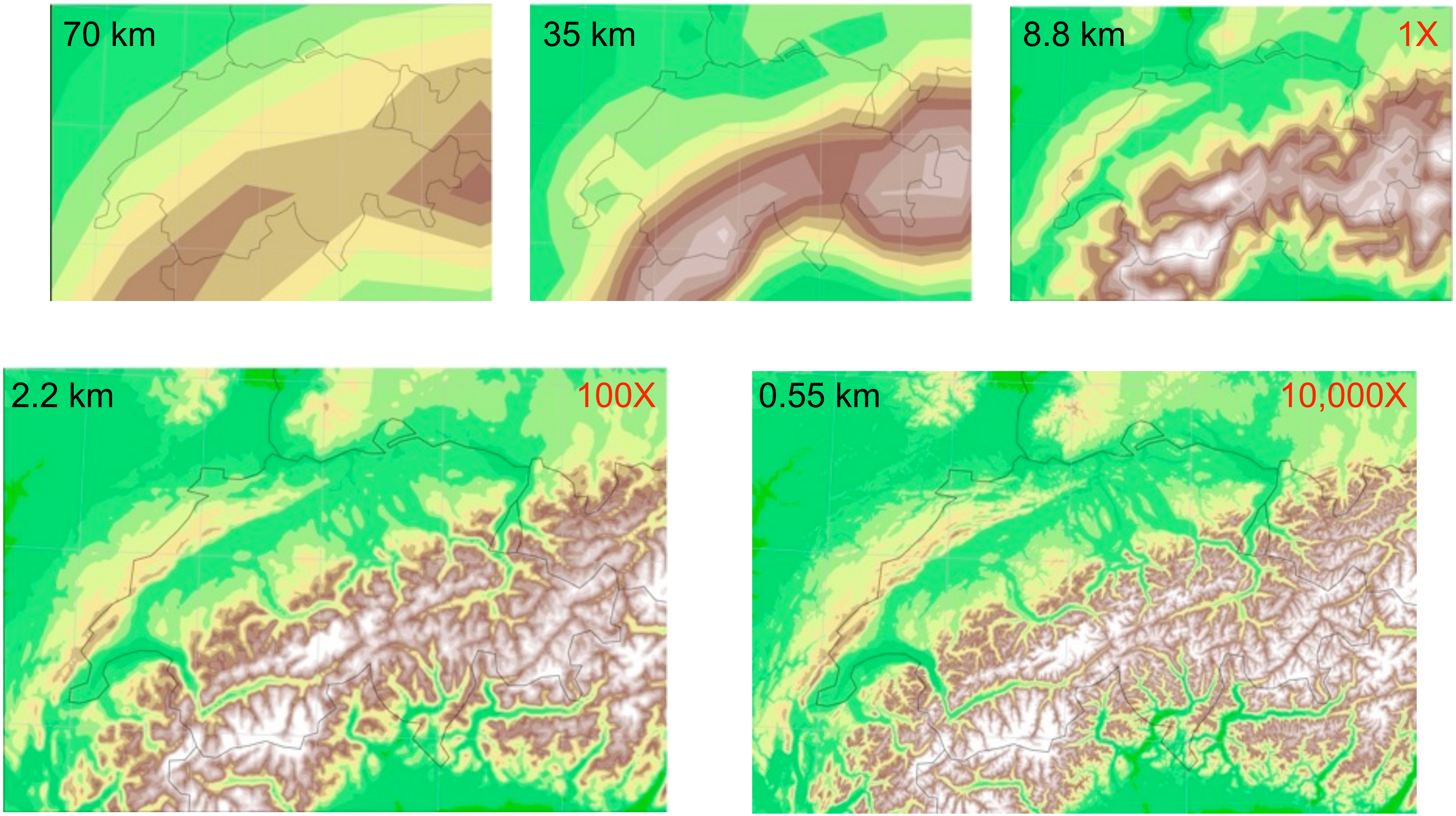
**COSMO-7 : 3x per day 72h forecast**
**6.6 km lateral grid, 60 layers**



**COSMO-2: 8x per day 24h forecast**
**2.2 km lateral grid , 60 layer**



# Why resolution is such an issue for Switzerland



# COSMO: Consortium for Small Scale Modeling a regional climate and weather model

- Website see: http://www.cosmo-model.org
- Currently used by the following weather services:
  - DWD (Germany); MCH (Switzerland); USAM (Italy); HNMS (Greece); IMGW (Poland); NMA (Romania); RHM (Russia);
- Last number of Scientists – climate and meteorology
- Several codes are part of this model:
  - COSMO-CCLM (weather will be discussed here)
  - COSMO-ARTH (advection and chemistry for aerosols etc.)
  - COSMO-RAPS (benchmark for vendors)
  - ...
- Algorithmic approach is very similar to WRF
  - Structured gird; finite difference / stencil
- The consortium is well managed
  - It is not hard to find the code you have to change!

# Dynamics in COSMO-CCLM

**velocities**

$$\frac{\partial u}{\partial t} = -\left\{\frac{1}{a\cos\varphi}\frac{\partial E_h}{\partial\lambda} - vV_a\right\} - \dot{\zeta}\frac{\partial u}{\partial\zeta} - \frac{1}{\rho a\cos\varphi}\left(\frac{\partial p'}{\partial\lambda} - \frac{1}{\sqrt{\gamma}}\frac{\partial p_0}{\partial\lambda}\frac{\partial p'}{\partial\zeta}\right) + M_u$$

$$\frac{\partial v}{\partial t} = -\left\{\frac{1}{a}\frac{\partial E_h}{\partial\varphi} + uV_a\right\} - \dot{\zeta}\frac{\partial v}{\partial\zeta} - \frac{1}{\rho a}\left(\frac{\partial p'}{\partial\varphi} - \frac{1}{\sqrt{\gamma}}\frac{\partial p_0}{\partial\varphi}\frac{\partial p'}{\partial\zeta}\right) + M_v$$

$$\frac{\partial w}{\partial t} = -\left\{\frac{1}{a\cos\varphi}\left(u\frac{\partial w}{\partial\lambda} + v\cos\varphi\frac{\partial w}{\partial\varphi}\right)\right\} - \dot{\zeta}\frac{\partial w}{\partial\zeta} + \frac{g}{\sqrt{\gamma}}\frac{\rho_0}{\rho}\frac{\partial p'}{\partial\zeta} + M_w + g\frac{\rho_0}{\rho}\left\{\frac{(T-T_0)}{T} - \frac{T_0 p'}{T p_0} + \left(\frac{R_v}{R_d} - 1\right)q^v - q^l - q^f\right\}$$

**pressure**

$$\frac{\partial p'}{\partial t} = -\left\{\frac{1}{a\cos\varphi}\left(u\frac{\partial p'}{\partial\lambda} + v\cos\varphi\frac{\partial p'}{\partial\varphi}\right)\right\} - \dot{\zeta}\frac{\partial p'}{\partial\zeta} + g\rho_0 w - \frac{c_{pd}}{c_{vd}}pD$$

**temperature**

$$\frac{\partial T}{\partial t} = -\left\{\frac{1}{a\cos\varphi}\left(u\frac{\partial T}{\partial\lambda} + v\cos\varphi\frac{\partial T}{\partial\varphi}\right)\right\} - \dot{\zeta}\frac{\partial T}{\partial\zeta} - \frac{1}{\rho c_{vd}}pD + Q_T$$

**water**

$$\frac{\partial q^v}{\partial t} = -\left\{\frac{1}{a\cos\varphi}\left(u\frac{\partial q^v}{\partial\lambda} + v\cos\varphi\frac{\partial q^v}{\partial\varphi}\right)\right\} - \dot{\zeta}\frac{\partial q^v}{\partial\zeta} - (S^l + S^f) + M_{q^v}$$

$$\frac{\partial q^{l,f}}{\partial t} = -\left\{\frac{1}{a\cos\varphi}\left(u\frac{\partial q^{l,f}}{\partial\lambda} + v\cos\varphi\frac{\partial q^{l,f}}{\partial\varphi}\right)\right\} - \dot{\zeta}\frac{\partial q^{l,f}}{\partial\zeta} - \frac{g}{\sqrt{\gamma}}\frac{\rho_0}{\rho}\frac{\partial P_{l,f}}{\partial\zeta} + S^{l,f} + M_{q^{l,f}}$$

**turbulence**

$$\frac{\partial e_t}{\partial t} = -\left\{\frac{1}{a\cos\varphi}\left(u\frac{\partial e_t}{\partial\lambda} + v\cos\varphi\frac{\partial e_t}{\partial\varphi}\right)\right\} - \dot{\zeta}\frac{\partial e_t}{\partial\zeta} + K_m^v\frac{g\rho_0}{\sqrt{\gamma}}\left\{\left(\frac{\partial u}{\partial\zeta}\right)^2 + \left(\frac{\partial v}{\partial\zeta}\right)^2\right\} + \frac{g}{\rho\theta_v}F^{\theta_v} - \frac{\sqrt{2}e_t^{3/2}}{\alpha_M l} + M_{e_t}$$

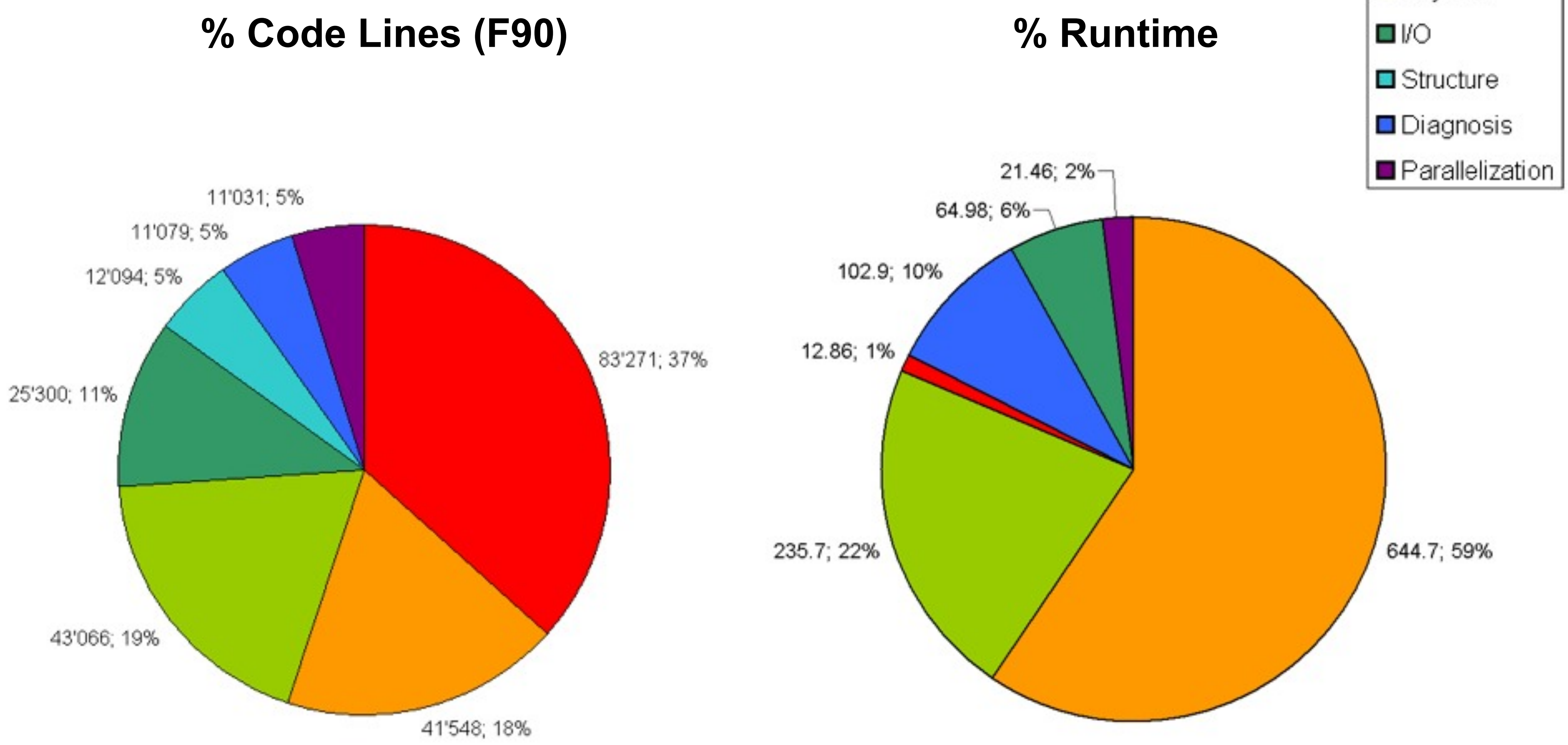
## Timestep

implicit (sparse) — explicit (RK3) — implicit (sparse solver) — explicit (leapfrog)

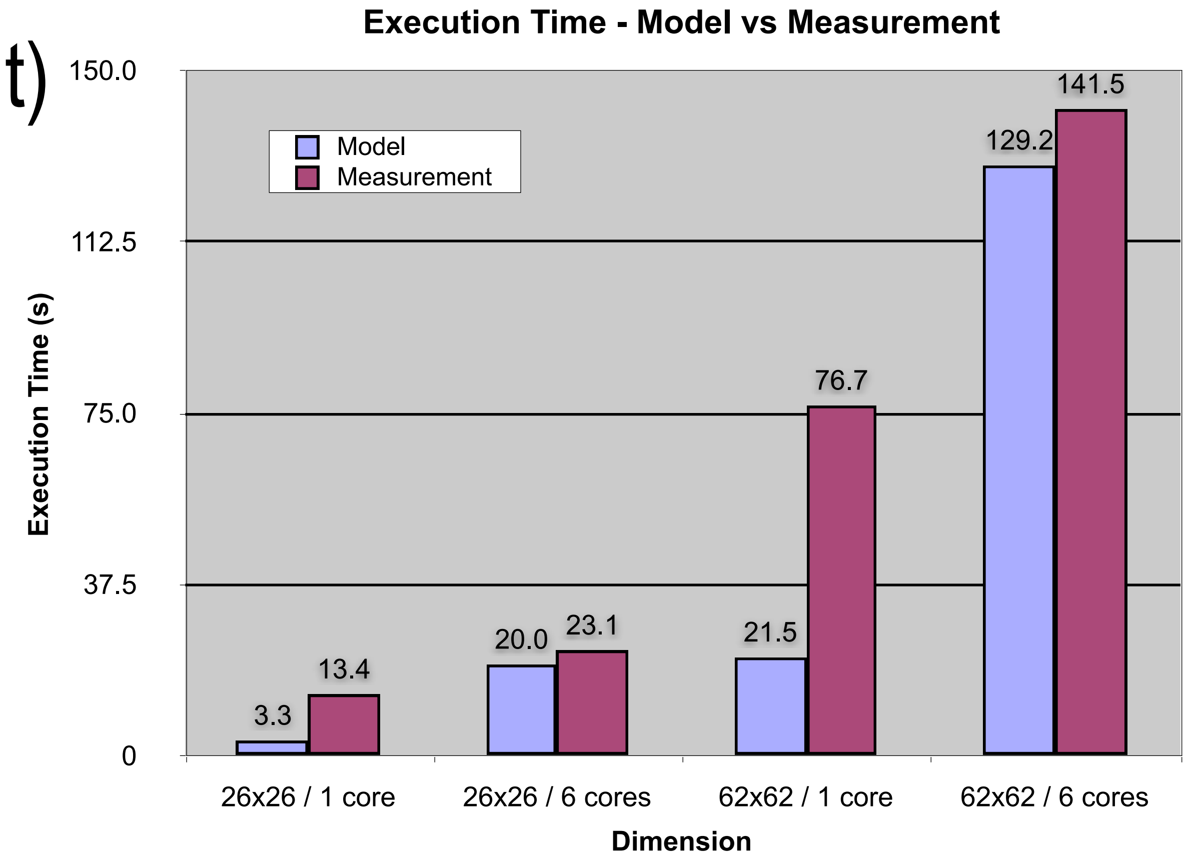
- 1x: physics et al. tendencies
- 3x: horizontal adv., vertical adv.
  - ~10x: fast wave solver
- water adv.

# COSMO-CCLM: Lines vs. Runtime

Runtime based 2 km production model of MeteoSwiss



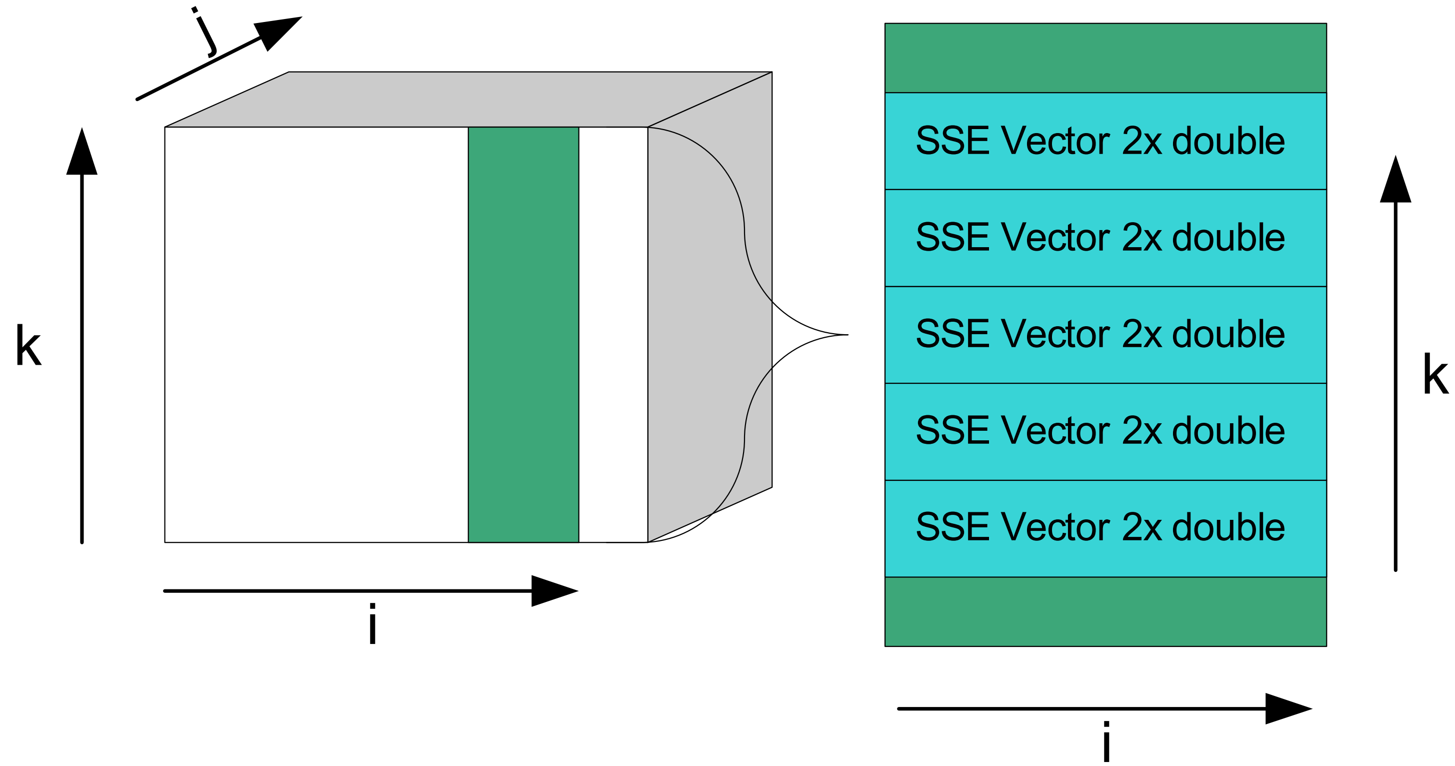
# Algorithmic motifs and performance model




- Note: current data layout of fields is *u*(x,y,z,t)
- Stencil computations
  - Used for explicit integration
  - Fast-Wave solver (integrating acoustic waves) uses only small stencils – 3x3x3
  - Advection uses 4x4x4, 7x1x1 or 1x7x1 stencils
- Tri-diagonal solver
  - Used for implicit integration in the vertical dimension (z)
  - This may be extended to 5-diagonal solver (increase coupling in z-dimension)
- With current data layout – performance is totally bandwidth limited!
  - Good starting point for the performance model is to count memory references and use memory bandwidth of processor

# Main steps to improve efficiency / time to solution

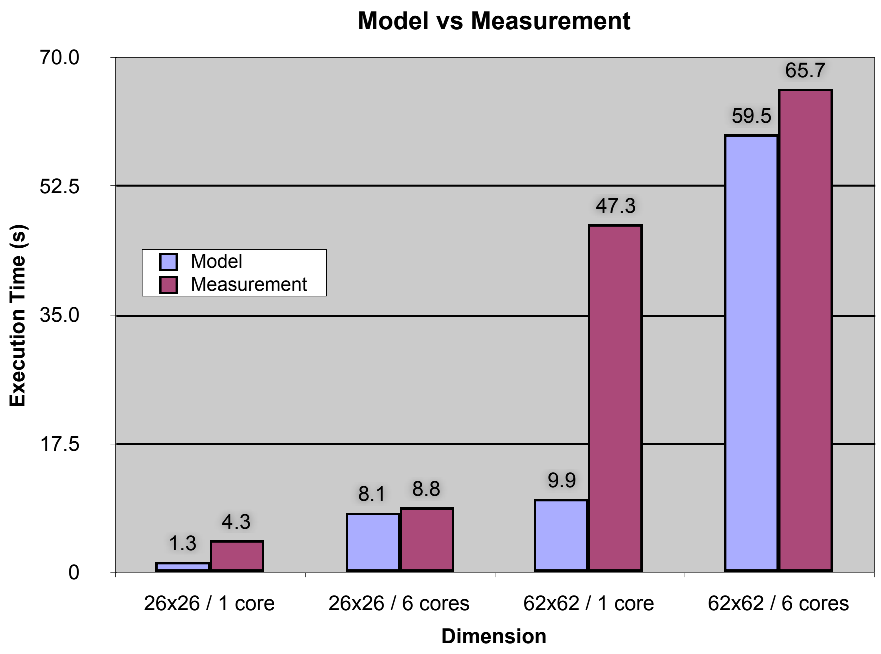
- Change data structures:
  - From: *u*(x,y,z) to *u*(z,y,x) or something in between
- Other obvious ideas
  - Reduce pre-calculations
  - Merge loops
  - Access memory through and relative locations – exploits locality of the stencil
- Since we are memory bandwidth bound – consider GPUs



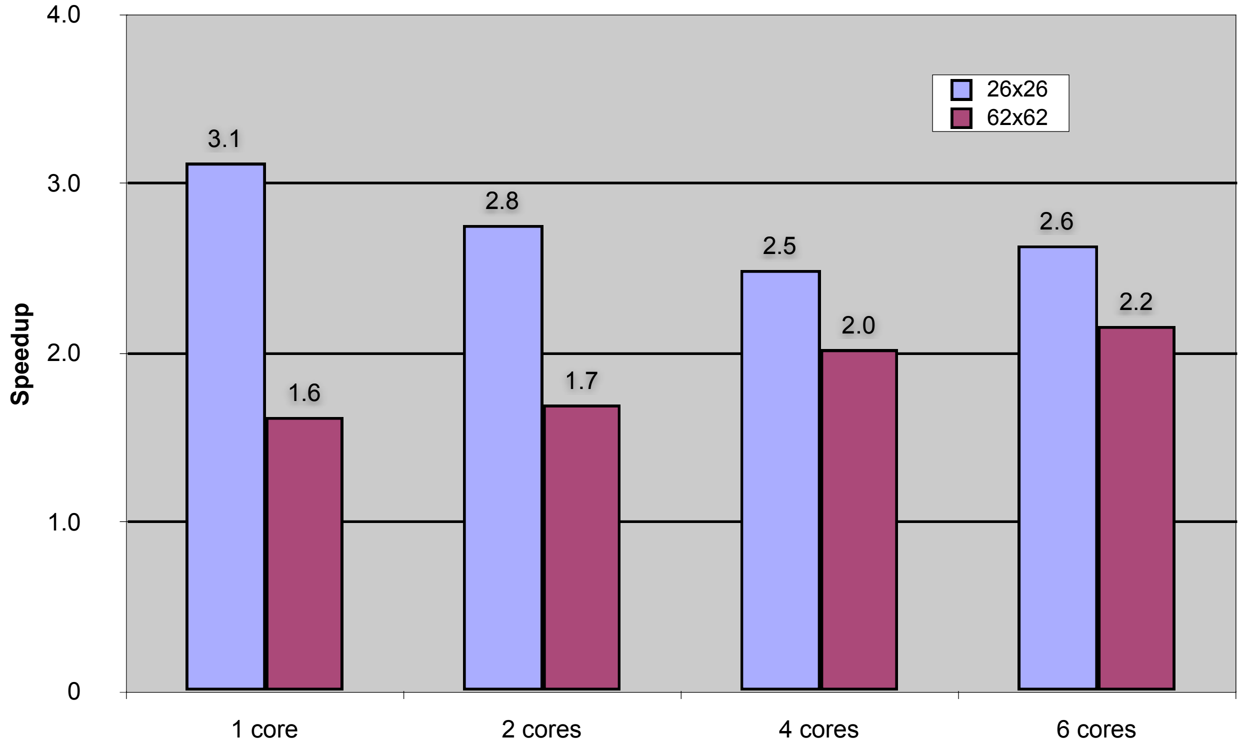

This will affect about 80k lines of code – do the performance gains justify the effort?

# Study performance gains & development cost with a prototype

- Extract 2-3k lines code and validate agains performance model

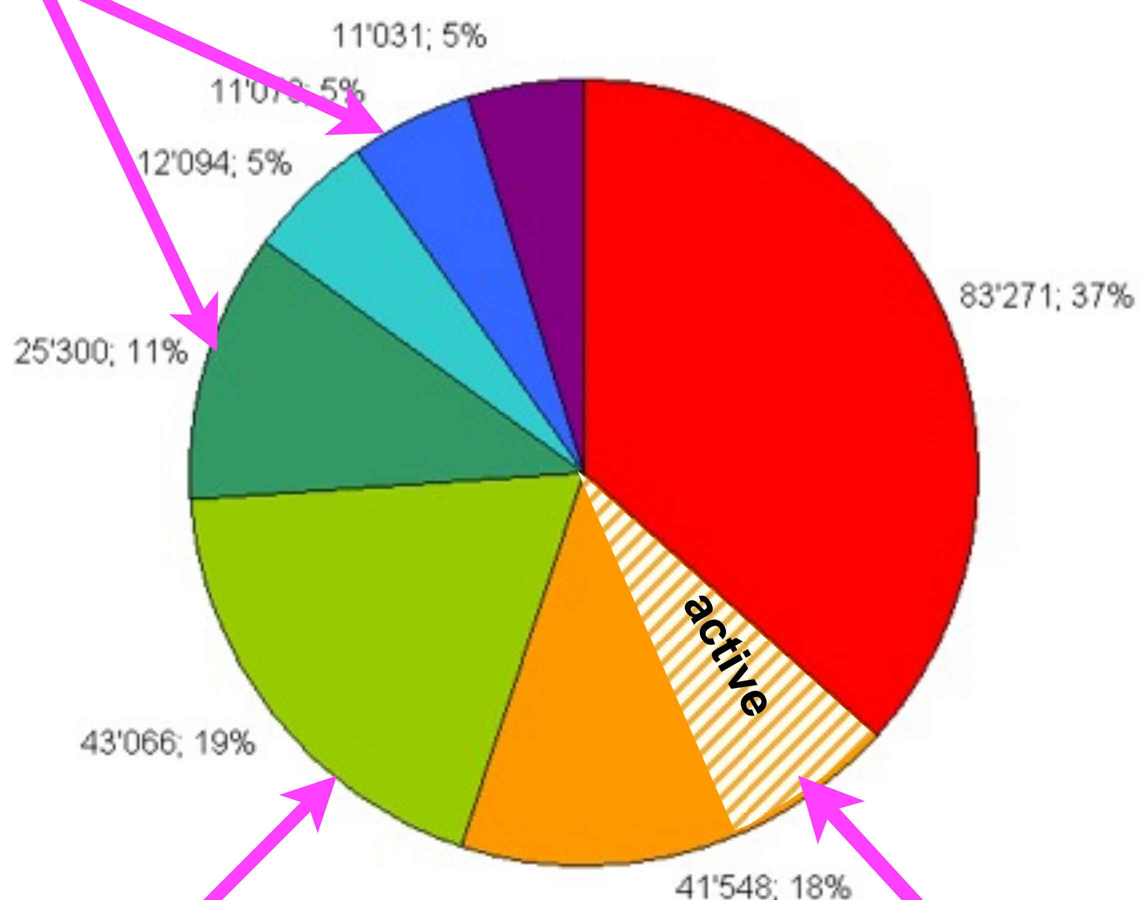

- Rewrite this prototype (several times) and measure performance gains

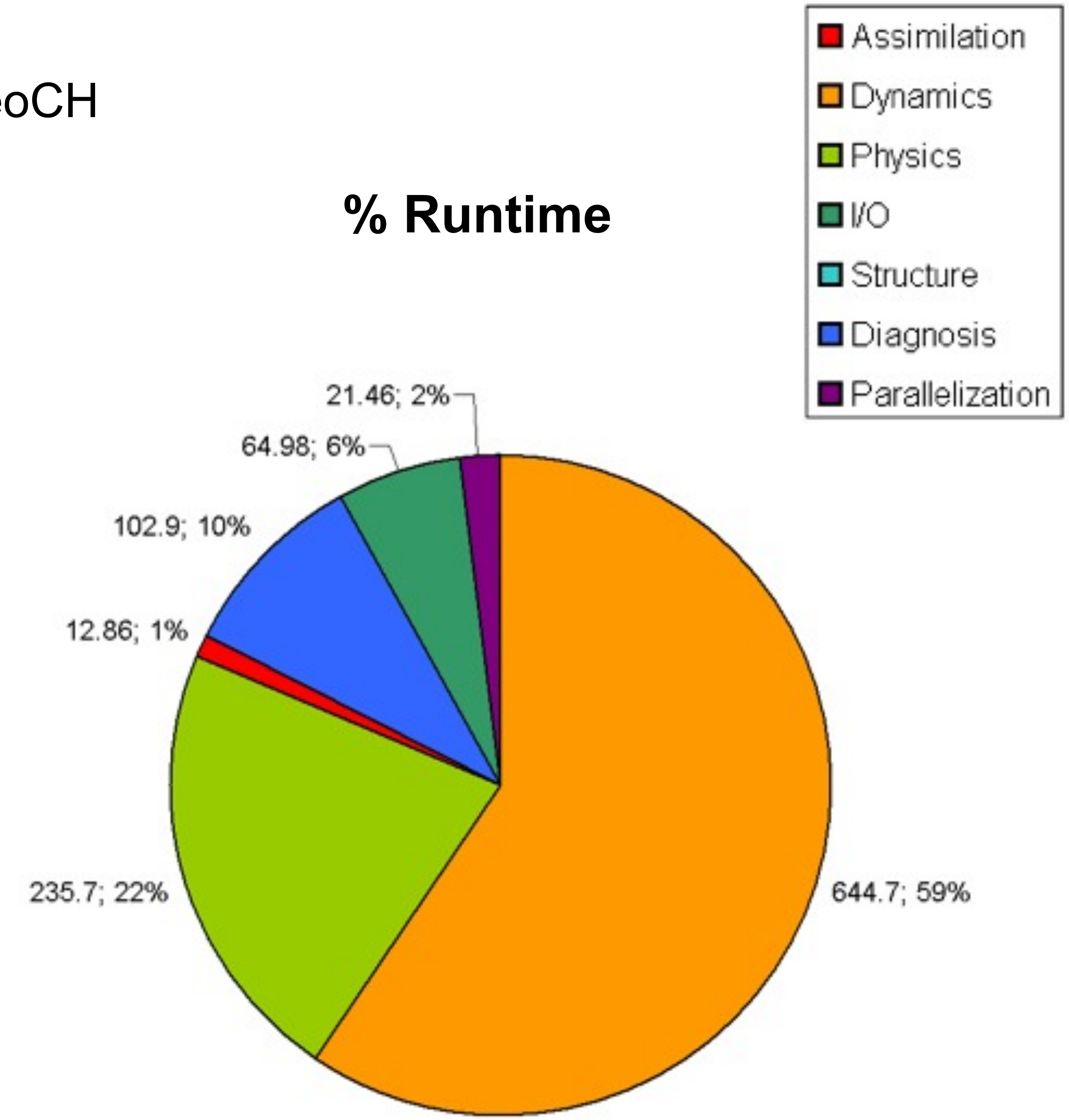


# COSMO-CCLM: Lines vs. Runtime

Runtime based 2 km production problem of MeteoCH

**Look at I/O subsystems & in-situ visualization (workflow!)**




**Adapt to new data structures & port to GPUs with directives**

**Rewrite in C++ with Cuda kernels, explore DSEL (requires change in data structures)**

# Structure of the co-design approach to COSMO

- Study / model / method / algorithm
  - in order to quickly impact the operational model we decided not to change the model, method of solution or any of the algorithms (for now)
- Develop performance model of current code
  - validate against full production runs
- Extract prototype-kernels & study potential performance gains
  - demonstrate and quantify advantages of refactoring the code to scientists (C++ rewrite!)
- Rewrite substantial part of the code
  - focus on heavily multithreaded version first; add domain decomposition in second step
  - based on custom built stencil library
  - conservative implementation for X86 & multi-core
  - aggressive approach with CUDA for GPU (high memory bandwidth)
  - consider DSL for implementation & DSEL for higher level abstraction + auto-tuning
- Longer-term considerations: change workflow to reduce I/O & storage
  - consider in-situ visualization

# Leaving the comfort zone of separated concerns

- Decompose application into algorithmic motifs
- Understand the hierarchy of motifs – the "dwarfs" are not all dwarfs
- Design systems or components therefore against these dwarfs

## Algorithmic motifs and their arithmetic intensity

Arithmetic intensity: number of operations per word of memory transferred

Rank-1 update in HF-QMC → Rank-N update in DCA++

Finite difference / stencil

QMR in WL-LSMS

Linpack

Sparse linear algebra

BLAS1

Dense matrix-matrix

BLAS2

FFT

BLAS3

O(1) → O(log N) → O(N)

- Proper (theoretical) description of heterogenous systems
  - Eckert and Mauchly's invention is named after von Neumann!
- ...

# Metric for “super” in supercomputing?

- Peak performance vs. sustained floating point performance – which application?
  - Linpack benchmark (Top500)
  - Particular code under production conditions (Gordon Bell Prize)
- Is sustained floating point performance really the right metric?
  - Much better than peak performance but doesn’t do justice for large classes of applications
- There is probably no simple answer!
- Optimize simulation systems for things that matter for the scientists / engineers
- **Time to solution** (minimize of good enough)
- **Energy to solution** / total cost of ownership

# Collaborators (co-design COSMO)

- Oliver Fuhrrer (MeteoCH)
- Tobias Gysi & David Müller (SCS)
- Will Sawyer, Ugo Vareto, Mauro Bianco, Sadaf Alam & others (CSCS)
- Uli Schättler, Michael Baldauf (DWD)

# QUESTIONS / COMMENTS?