Schweizerische Eidgenossenschaft
Confédération suisse
Confederazione Svizzera
Confederaziun svizra

Federal Department of Home Affairs FDHA

ETH zürich

C2SM Center for Climate Systems Modeling

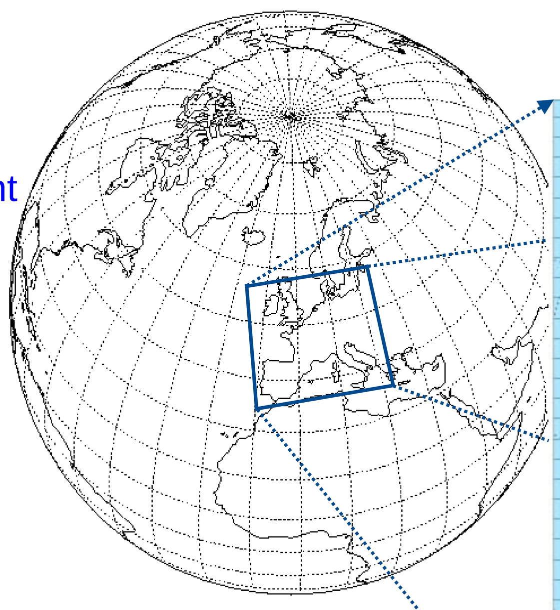
CSCS Centro Svizzero di Calcolo Scientifico Swiss National Supercomputing Centre

Numerical Weather Prediction at MeteoSwiss

Philippe Steiner
Federal Office of Meteorology and Climatology MeteoSwiss, Zurich, Switzerland

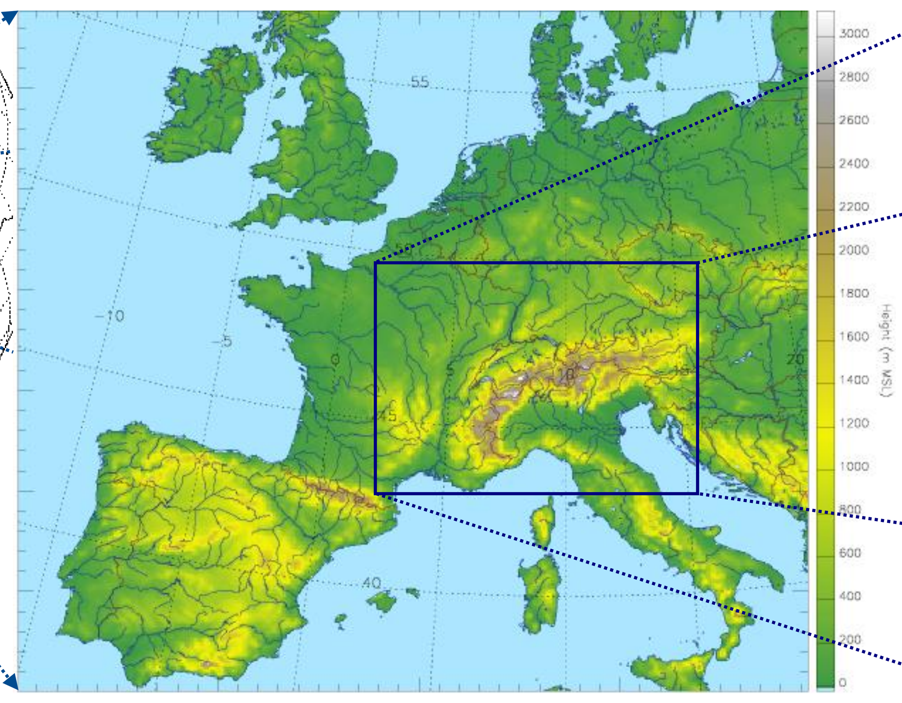
Swiss implementation of the COSMO-Model

- **Prognostic variables**
  pressure, 3 wind components, temperature, specific humidity, cloud water, cloud ice, rain, snow, turbulent kinetic energy (TKE), 4 different pollen species. COSMO-2: also graupel
- **Coordinates** general terrain-following height-based vertical levels, Lorenz staggering; Arakawa-C, rotated Lat/Lon horizontal grid
- **Dynamics** 2-timelevel 3rd order Runge-Kutta
- **Physics**
  bulk microphysics for atmospheric water content, multilayer soil module, radiation, turbulence, sso, COSMO-7: Tiedtke mass flux convection scheme COSMO-2: explicit deep convection
- **Computers**
  2 Cray XE6 (production / backup & development) at Swiss National Supercomputing Centre, CSCS 144 / 336 AMD 2.1 GHz MagnyCours processors with 1728 / 4032 computational processing cores Together, the systems can reach a peak performance of 50 TFlops.
- **Time to solution**
  27 minutes for 33h COSMO-2
  Effective performance 450 Gflops (5% of peak)



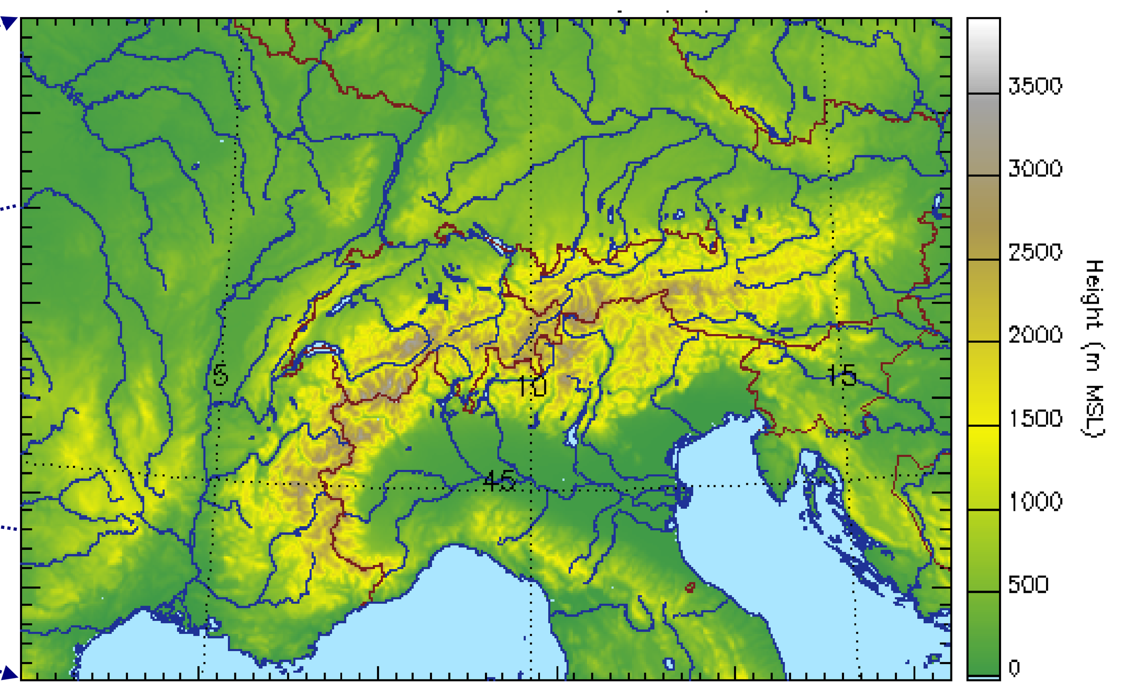
Global Integrated Forecast System **IFS** (ECMWF, ~16km resolution)

COSMO-7



COSMO-7 domain (maximum height at 3140m).

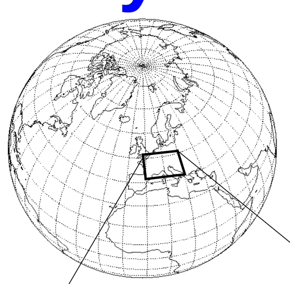
COSMO-2



COSMO-2 domain (maximum height of 3944m).

Mesh size	3/50°, **~6.6km**	1/50°, **~2.2km**
Domain	393 x 338 x 60 = 7'970'040 grid points	520 x 350 x 60 = 10'920'000 grid points
Forecasts	+72h at 00, 06 and 12 UTC	+33h at 00, 06, 09, 12, 15, 18, 21 UTC, +45h at 03 UTC
Boundary conditions	Hourly update from IFS	Hourly update from COSMO-7
Initial conditions	Newtonian relaxation (nudging) to surface and upper air observations, intermittent cycle of 3h assimilation	Same as COSMO-7, but with use of radar data over Switzerland (latent heat nudging)

Valley winds with COSMO-1 Jürg Schmidli (ETH)



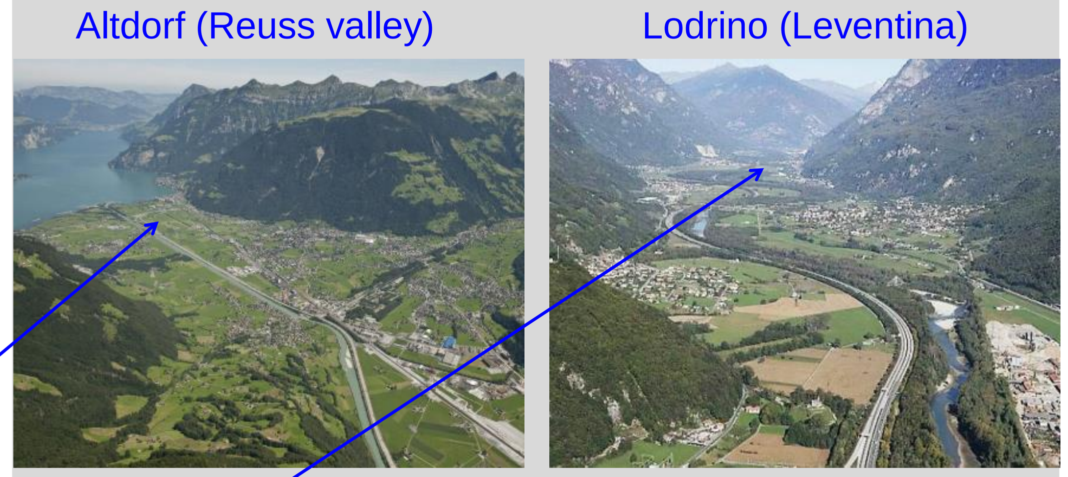
IFS in 2016: ~9km resolution

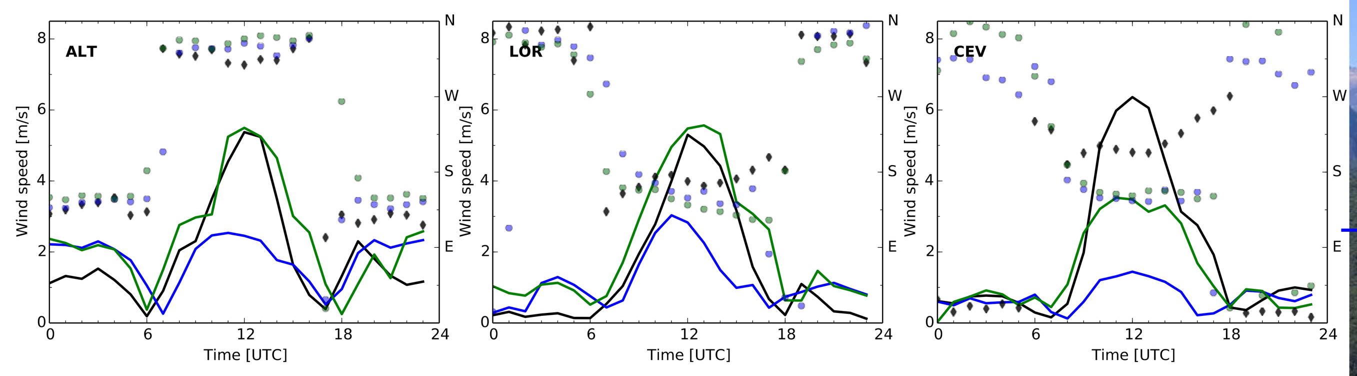
Next deterministic model: COSMO-1
Δx = 1.1 km, Δt = 10 s
1158 x 774 x 80 grid points
33 – 45h day forecasts, 8 x per day

Figure 1) COSMO-1 domain (maximum height at 4270m)

RESULTS

- Benefit of high resolution for wind speed. Narrow valley stations like Cevio are more difficult (Fig. 2)
- Large improvement of valley winds for most sites (Fig. 3 and 4)
- Critical factors are resolution and external parameters. Need high-resolution surface data for 1km simulations





Observations, COSMO-2, COSMO-1

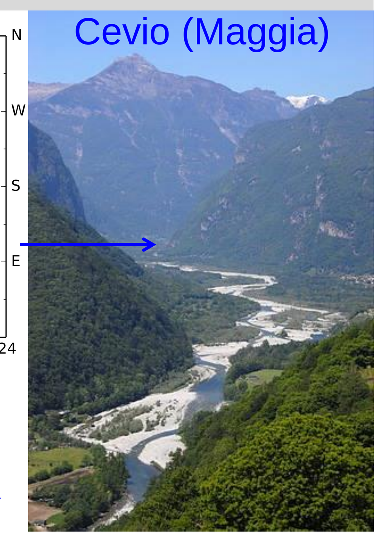


Figure 2) Valley wind speed (−) and direction (◊) in Altdorf (Reuss valley), Lodrino (Leventina) and Cevio (Maggia) 8-days average for July 9 - 27, 2006

"Top-six" stations

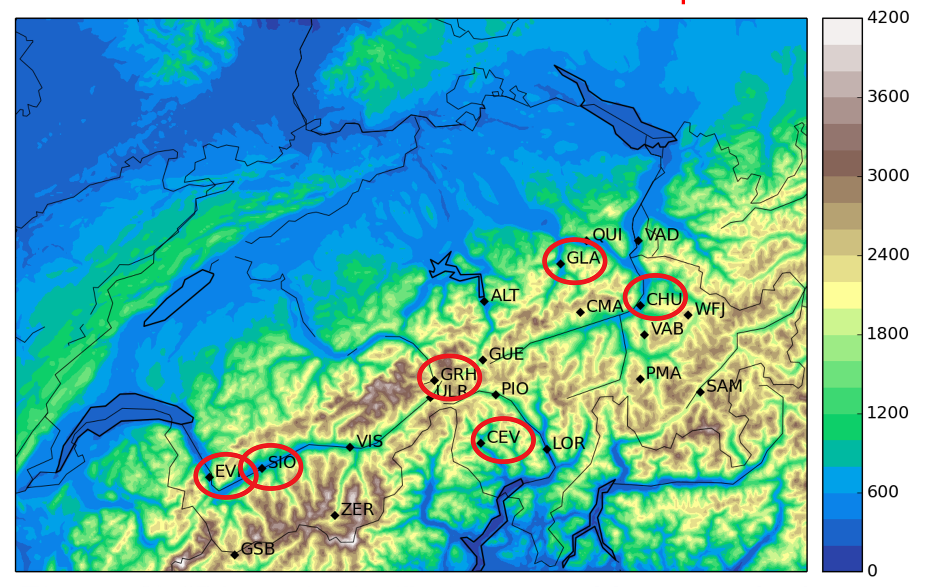


Figure 3) "Diurnal wind" stations with mean (July 9-27, 2006) maximum wind > 4 m/s → 21 stations

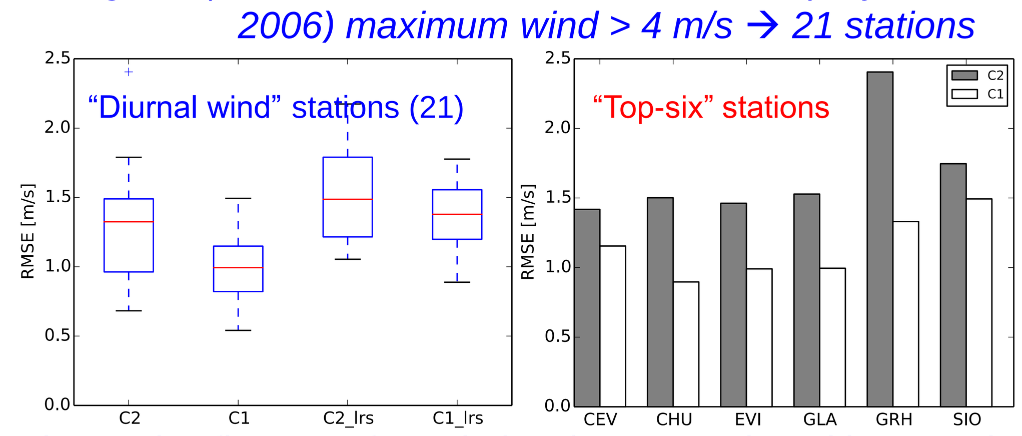
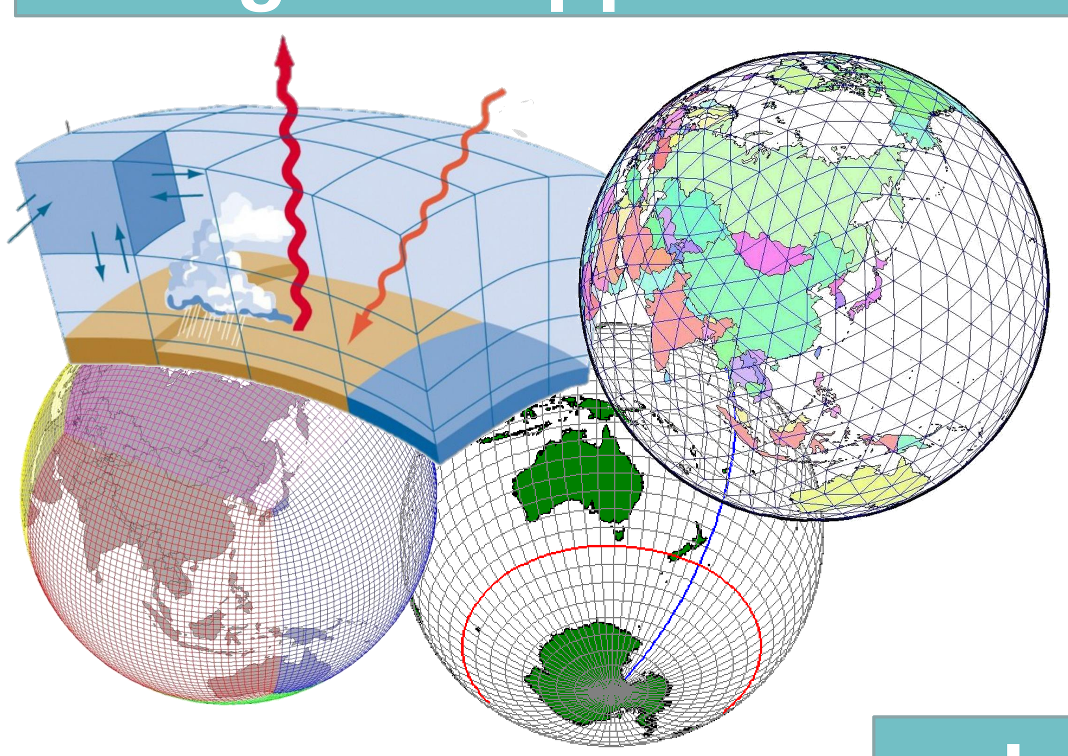


Figure 4) Influence of resolution (COSMO-1/2) and low resolution surface data (lrs) on diurnal cycle RMSE of wind speed.

GridTools

Lucas Benedicic[1], Mauro Bianco[1], Paolo Crosetto[1], Oliver Fuhrer[2], Carlos Osuna[3], Thomas C. Schulthess[1]

Targeted Applications



- ✓ Regular Grids
  - ✓ Multidimensional arrays
- ✓ Structured grids
  - ✓ Regular tessellations
- ✓ Icosahedral grid
- ✓ Cube on sphere
  - ✓ Fixed refinements on regular sub-regions

Scope of GridTools

- ✓ **Set of libraries** for stencil methods
  - ❖ **Describing** stencils
    - Iteration constraints - Data Structures
    - Computation units - Interfaces
  - ❖ **Composing** stencils
  - ❖ **Halo update**
  - ❖ **Boundary conditions**
- ✓ **Block structured** grids (hierarchical structure)
- ✓ Interoperable by **sharing concepts**
- ✓ Suited for **other** application fields
  - ❖ No application field specific constructs
- ✓ One does not have to adhere to all of them!

General Properties

- ✓ Industry **quality tools** for development
- ✓ Works on multiple backends (**GPUs or CPUs**)
- ✓ Single language for the **whole** application
- ✓ Library **composition**
- ✓ Library **abstraction**
- ✓ **Separation** of concerns
- ✓ **Incremental** optimization
- ✓ **Extensibility**

Challenges

- ✓ **Debug ability** / Verifiability
- ✓ Steep programmer **learning** curve
- ✓ Maintainability: **modular design** with multiple backends

Laplace Operator in C++

```
struct lap_function {
  typedef output_accessor<0> out;
  typedef input_accessor<1, range<-1,1,-1,1> > in;
  typedef arg_list<out, in> arg_list_type;

  template <typename Accessors>
  static void Do(Accessors const & eval, x_lap) {
    eval(out()) = eval(4*in() -
                  in( 1, 0, 0) + in( 0, 1, 0) +
                  in(-1, 0, 0) + in( 0,-1, 0));  }
};
```

Laplace Operator in Python

```
class LaplaceStencil (MultiStageStencil):
  def kernel (self, out, in):
    for p in self.interior_points (out,
                                   halo=(-1,1,-1,1)):
      out [p] = 4 * in [p] - (
        in_data[p+(1,0,0)] + in_data[p+(0,1,0)] +
        in_data[p+(-1,0,0)] + in_data[p+(0,-1,0)])
```

[1]: Swiss National Supercomputing Centre, CSCS
[2]: MeteoSwiss
[3]: Center for Climate Systems Modeling, C2SM