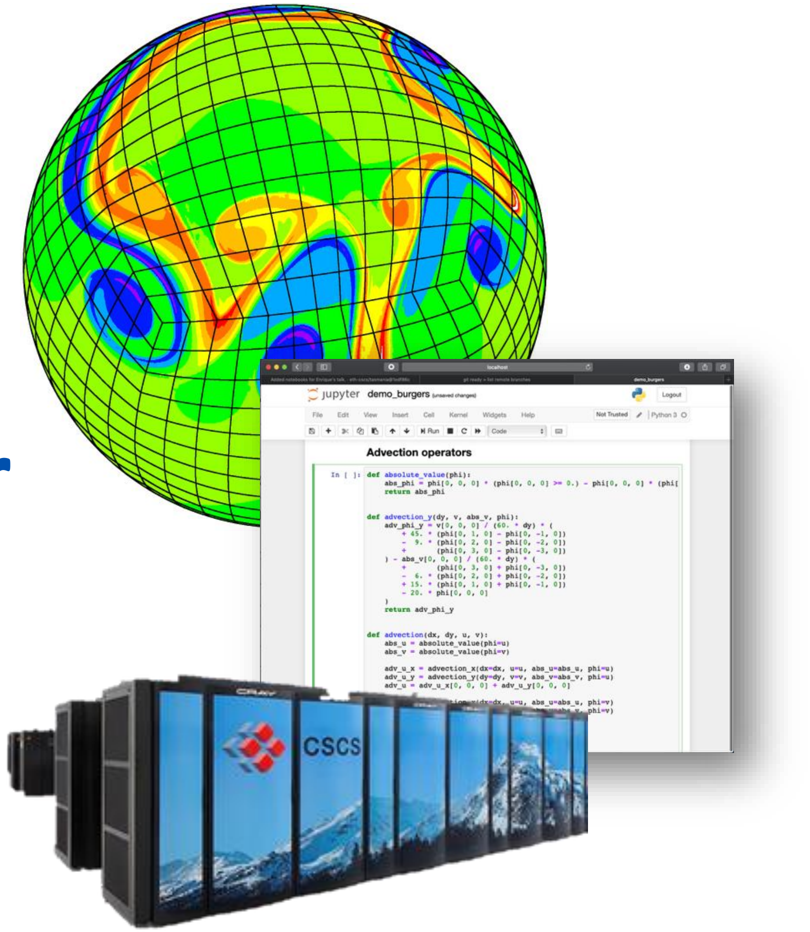
High Performance Computing for Weather and Climate (HPC4WC)

Content: Introduction

Lecturer: Oliver Fuhrer

Block course 701-1270-00L

Summer 2025

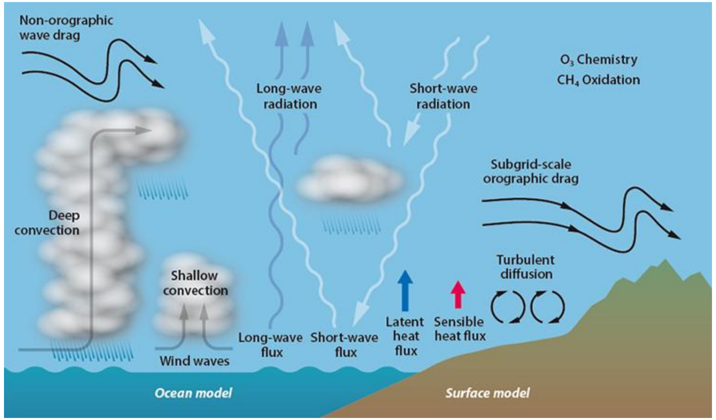


Learning goals

- Understand why weather and climate simulation requires high-performance computing (HPC)
- Understand why stencil computations are a key algorithmic motif of weather and climate models
- Understand a simple stencil program: higher-order monotonic diffusion (Xue 2000, MWR)

Modeling the Earth system

Physical understanding



Governing equations (e.g. atmosphere)

$$\frac{d}{dt}\mathbf{v} = -2\boldsymbol{\Omega} \times \mathbf{v} - \frac{1}{\rho}\nabla_3 p + \mathbf{g} + \mathbf{F}$$ Conservation of momentum (Navier-Stokes)

$$C_v \frac{d}{dt}(\rho q) + p\frac{d}{dt}\left(\frac{1}{\rho}\right) = J$$ Conservation of energy (1st Law of Thermodynamics)

$$\frac{\partial}{\partial t}(\rho) = -\nabla_3 \cdot (\rho \mathbf{v})$$ Conservation of air mass

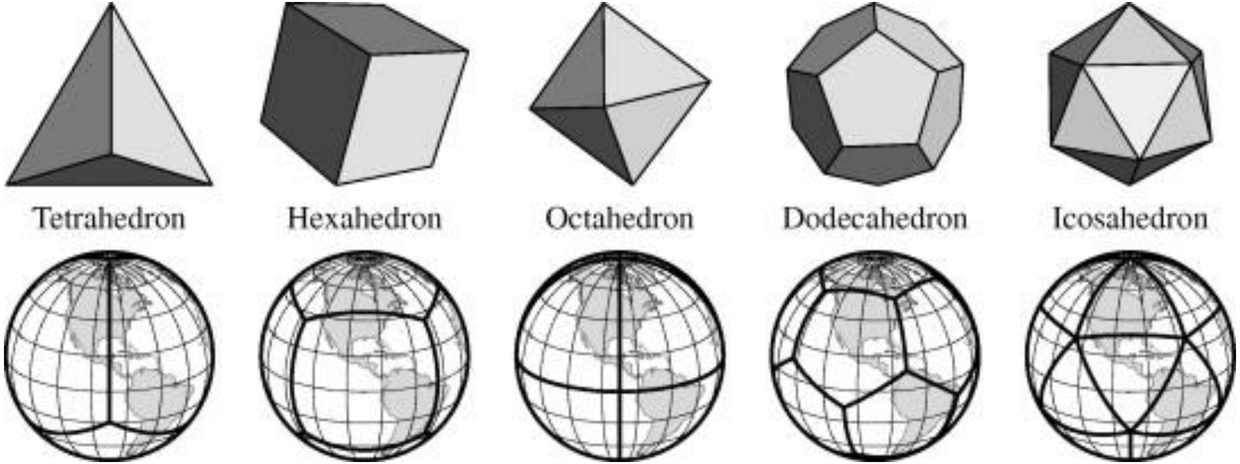
$$\frac{\partial}{\partial t} = -\nabla_3 \cdot (\rho \mathbf{v} q) + \rho\,(E - C)$$ Continuity of water vapor mass

$$p = \rho R T$$ Equation of state (Ideal gas law)

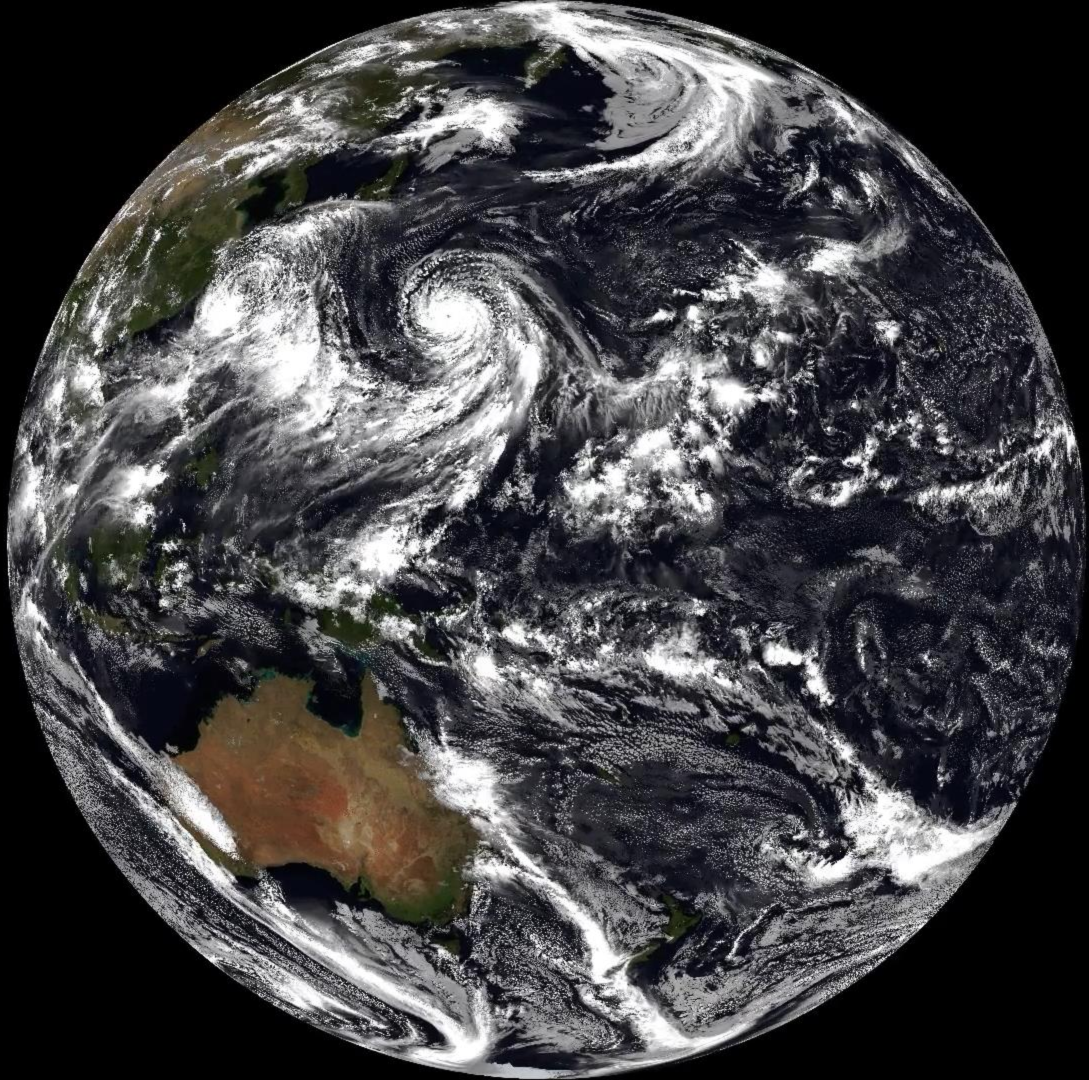
Variables: $\{\mathbf{v}, p, T, \rho, q\}$

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

Grids on a sphere

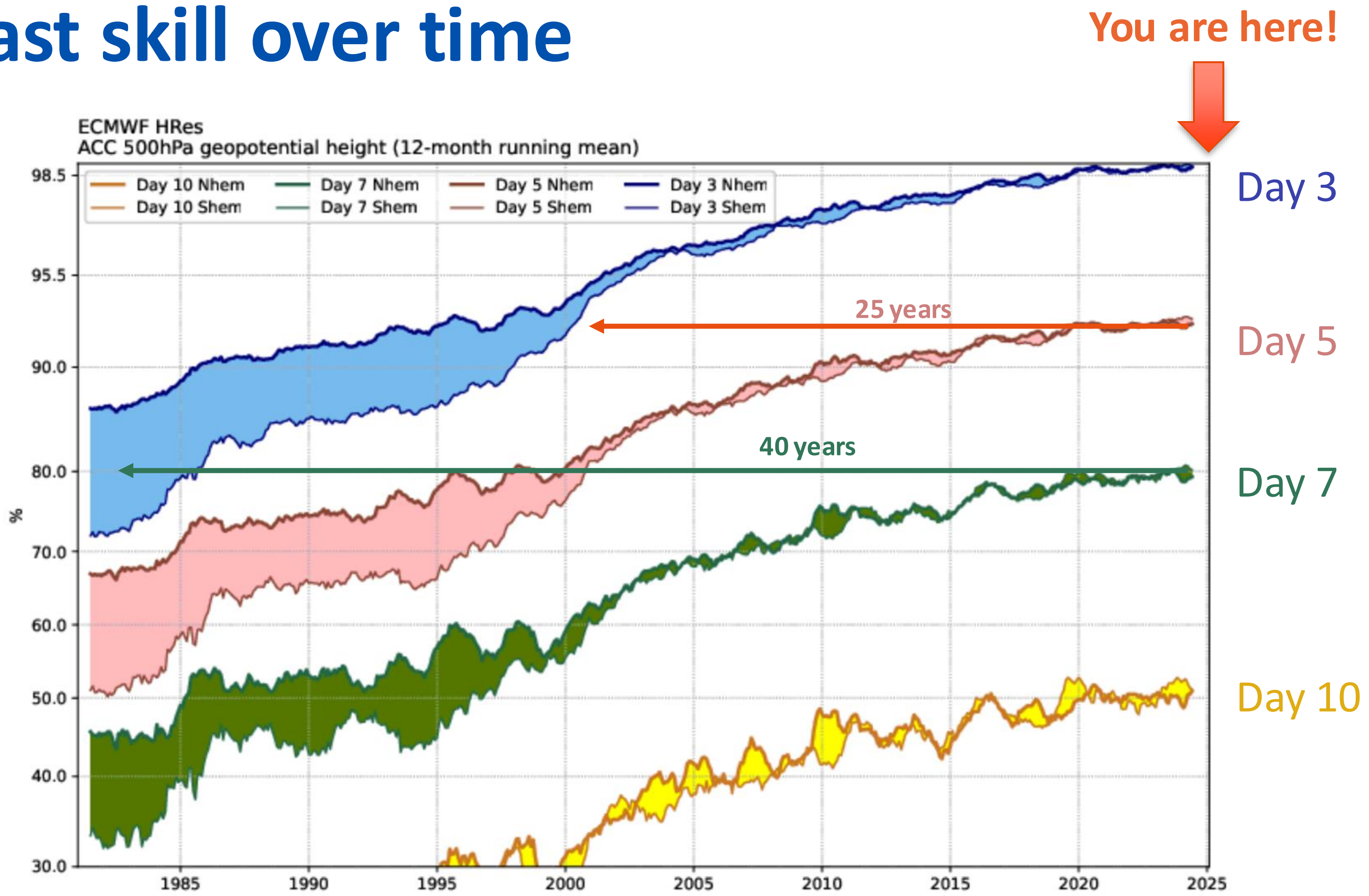


(Image source https://www.cosmic-core.org/free/article-42-geometry-platonic-solids-part-3-spherical-stereographic-solids/



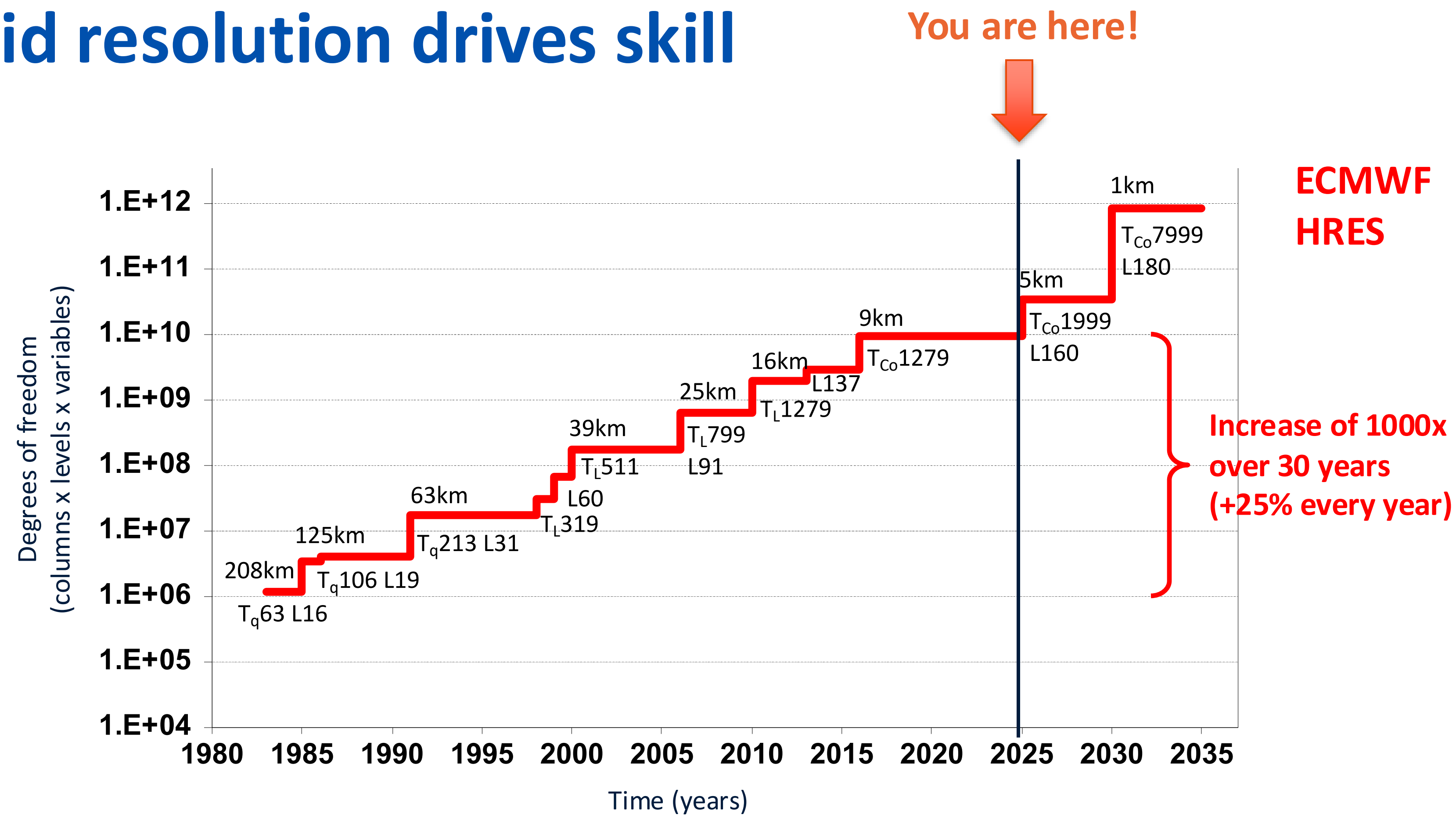
$6 \times 3072^2 \times 80$

Forecast skill over time



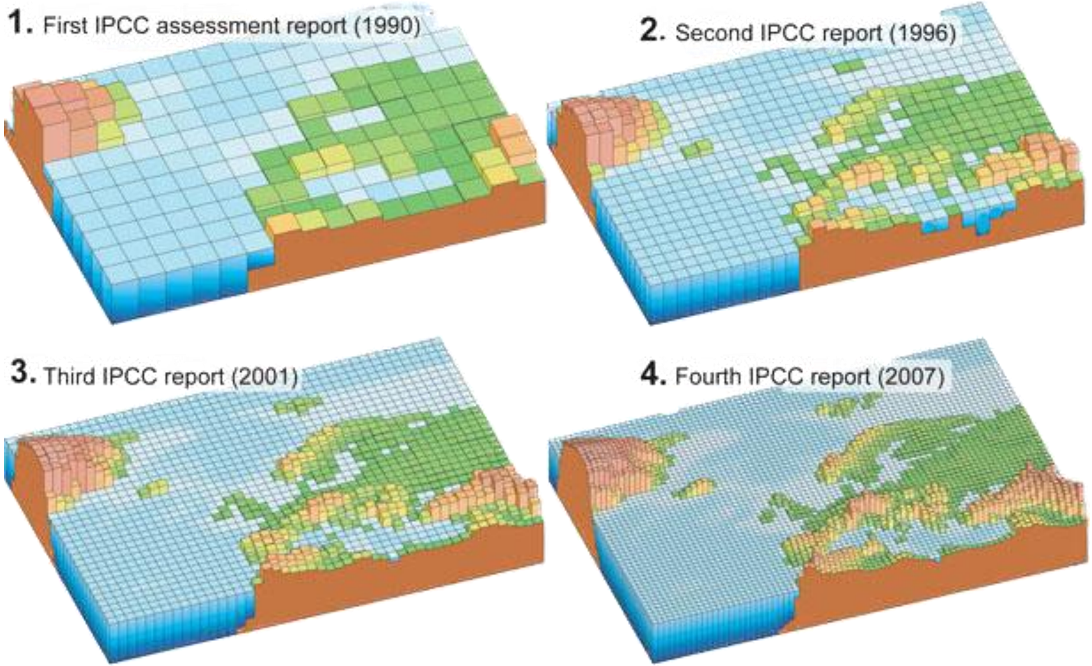
(Image source https://www.ecmwf.int/en/forecasts/charts/catalogue/plwww_m_hr_ccaf_adrian_ts?facets=undefined&time=2020051100)

Grid resolution drives skill

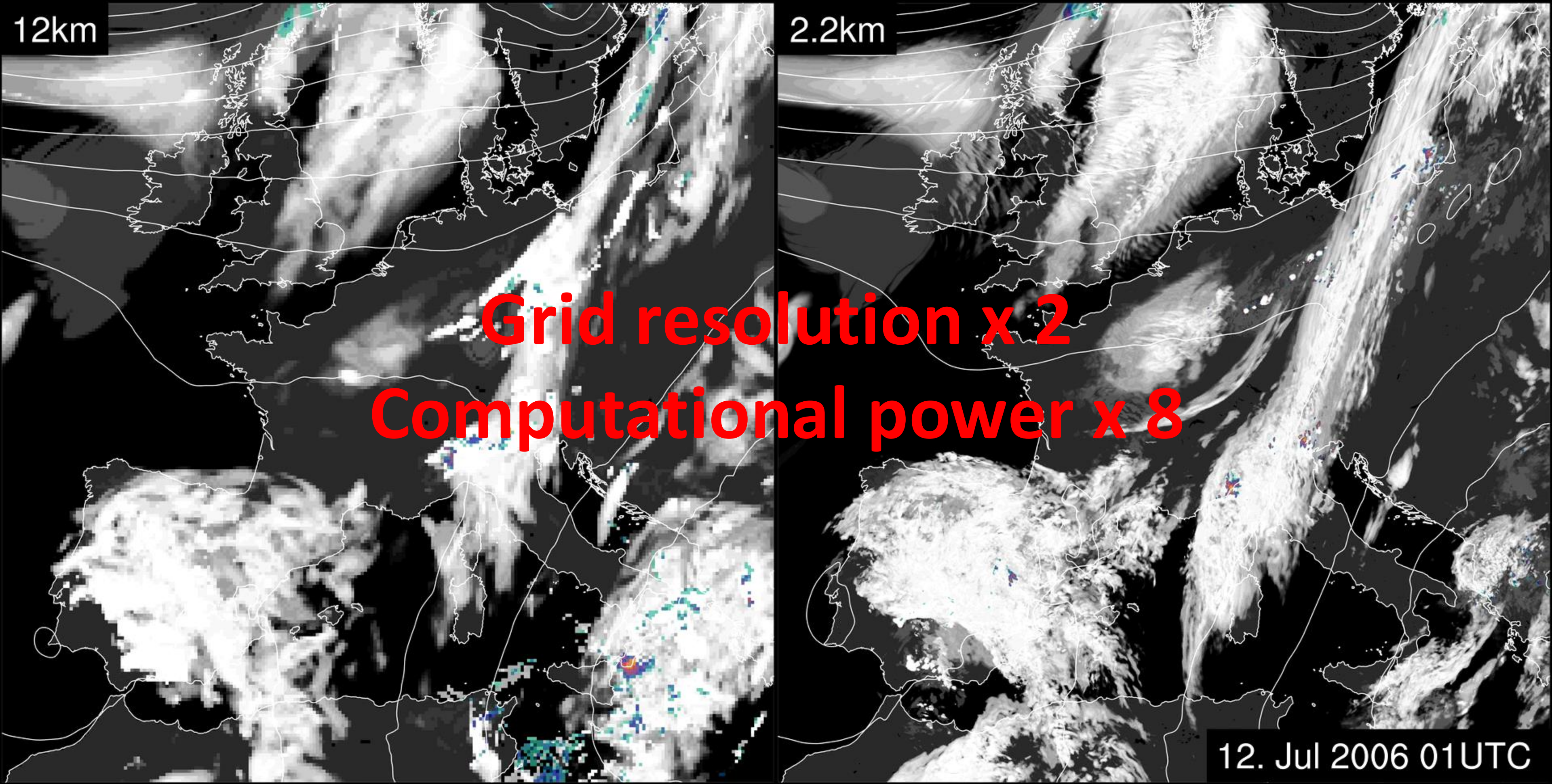


(Image source Schulthess et al. 2019, IEEE Race to Exascale Computing)

Grid resolution in the IPCC



(Image source https://nas-sites.org/climate-change/climatemodeling/common/images/image_page_3_2_resolution.png

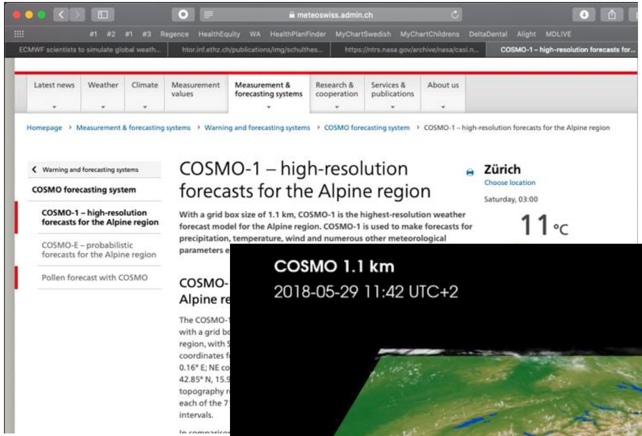


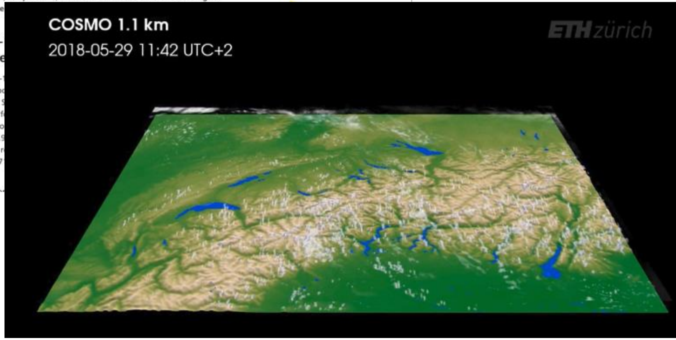
Grid resolution x 2
Computational power x 8

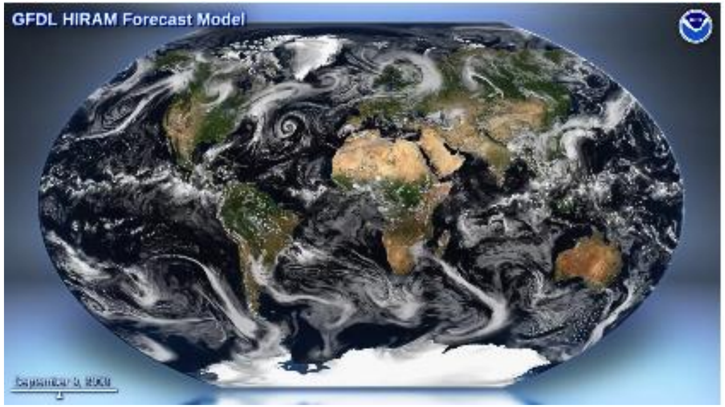
Example applications

- Swiss national weather forecast ([ICON-CH1-EPS]())
  $1.15 \times 10^8 \times 80$ gridpoints
  $\Delta x = 1.05$ km
  $\Delta t = 10$ s
  11 members
  80 x faster than real time (0.2 SYPD)

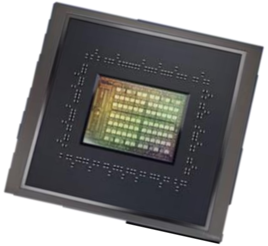
- Global coupled climate model with 1.25 km resolution ([ICON Sapphire]())
  $3.26 \times 10^8 \times 80$ gridpoints
  $\Delta x = 1.25$ km
  $\Delta t = 10$ s
  1000 x faster than real time (3 SYPD)







Why HPC?



NVIDIA Grace CPU
@3.44 Ghz, 250 W

- Swiss national weather forecast (ICON-CH1-EPS)
  $1.15 \times 10^8 \times 80$ gridpoints
  $\Delta x = 1.05$ km
  $\Delta t = 10$ s
  11 members
  80 x faster than real time

~ 1’000 CPUs

~ 250 kW (= 320 households)

- Global coupled climate model with 1.25 km resolution (ICON Sapphire)
  $3.26 \times 10^8 \times 80$ gridpoints
  $\Delta x = 1.25$ km
  $\Delta t = 10$ s
  ~~1000 x faster than real time~~
  Currently only 100 x feasible!

~ 500’000 CPUs

~ 125 MW (approx. 10% of Gösgen)

~130 CPUs

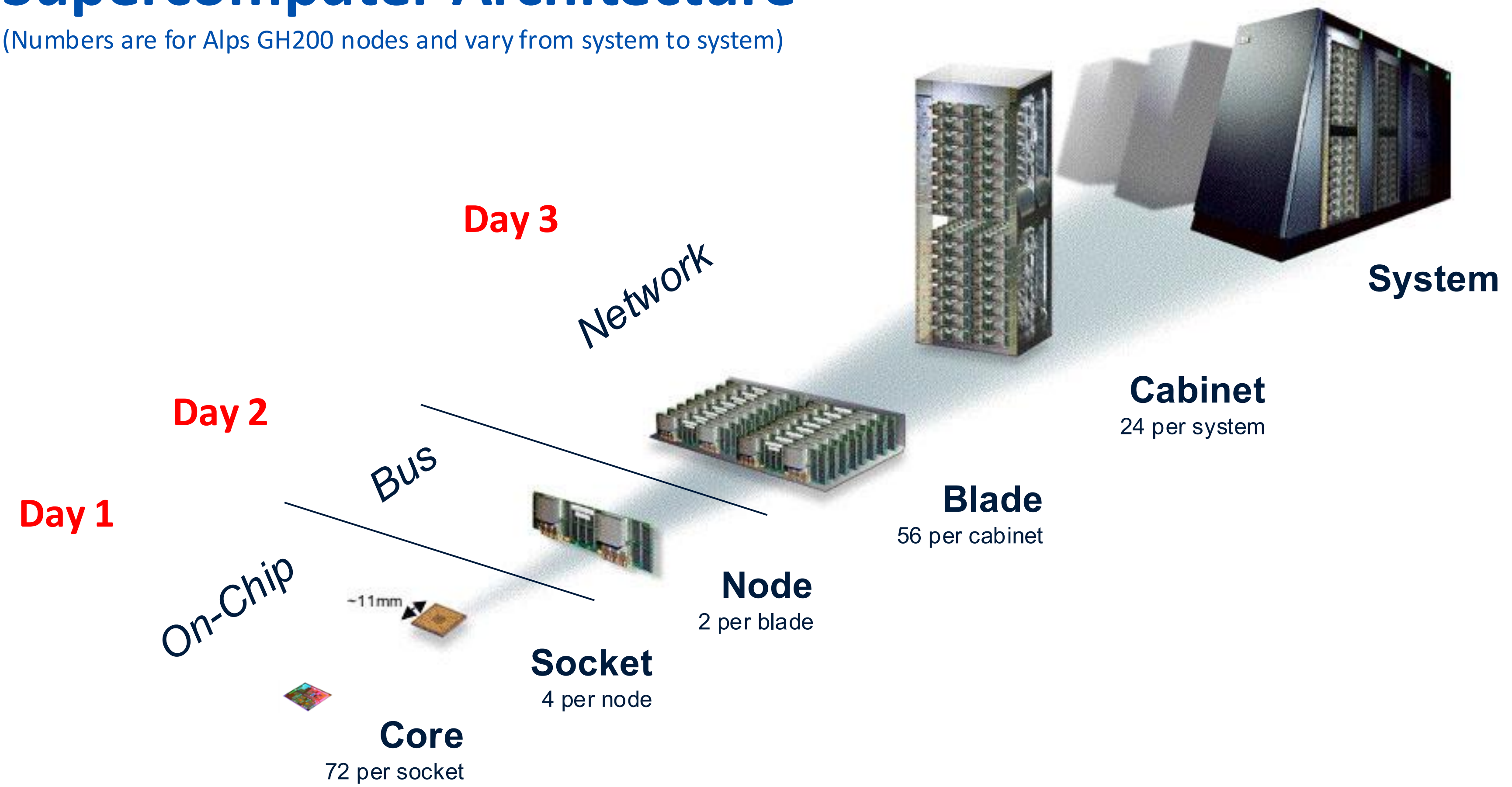
(only sufficent for one ICON-CH1-EPS member)

14’000 CPUs

(~ 15 x more dense!)

Supercomputer Architecture

(Numbers are for Alps GH200 nodes and vary from system to system)



“*High Performance Computing* is the practice of aggregating computing power in a way that delivers much higher performance than one could get out of a typical desktop computer or workstation in order to solve large problems in science, engineering, or business.”

Lab Exercises

01-roofline-model.ipynb

- Learn about performance metrics and how to compute theoretical peak values.
- Learn about arithmetic intensity and performance limiters.

02-stencil-program.ipynb

- Determine arithmetic intensity of a stencil program.
- Apply a performance profiling tool to gain insight into performance.
- Show limitations of the von Neumann model for understanding performance.

03-caches-data-locality.ipynb

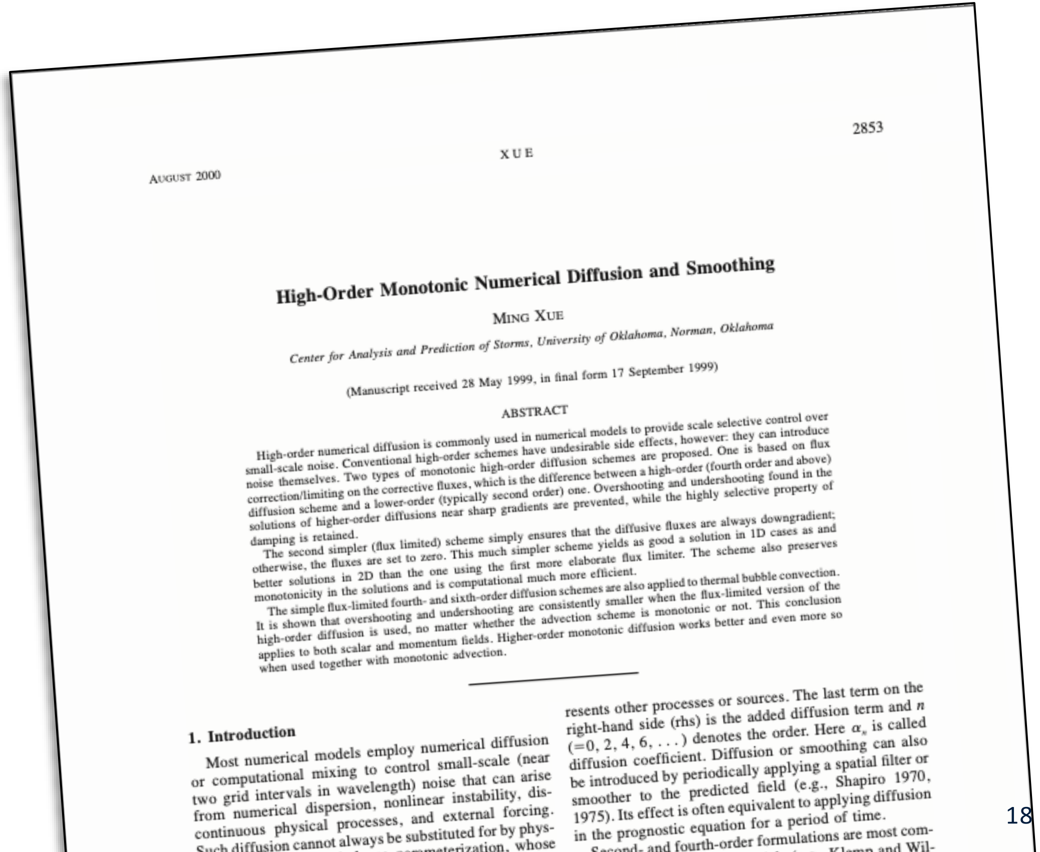
- Determine arithmetic intensity of a stencil program.
- Apply a performance profiling tool to gain insight into performance.
- Show limitations of the von Neumann model for understanding performance.

Learning goals

- Understand why weather and climate simulation requires high-performance computing (HPC)
- Understand why stencil computations are a key algorithmic motif of weather and climate models
- Understand a simple stencil program: higher-order monotonic diffusion (Xue 2000, MWR)

Higher-order diffusion (Xue 2000, MWR)

- Atmospheric and ocean models often need some form of numerical filtering to control small-scale noise.
- [Xue 2000 (Monthly Weather Review)]() published a class of higher-order monotonic filters that are frequently used.
- We will focus on the 4^{th}-order non-monotonic diffusion for this course.



High-Order Monotonic Numerical Diffusion and Smoothing

Ming Xue

Center for Analysis and Prediction of Storms, University of Oklahoma, Norman, Oklahoma

(Manuscript received 28 May 1999, in final form 17 September 1999)

ABSTRACT

High-order numerical diffusion is commonly used in numerical models to provide scale selective control over small-scale noise. Conventional high-order schemes have undesirable side effects, however: they can introduce noise themselves. Two types of monotonic high-order diffusion schemes are proposed. One is based on flux correction/limiting on the corrective fluxes, which is the difference between a high-order (fourth order and above) diffusion scheme and a lower-order (typically second order) one. Overshooting and undershooting found in the solutions of higher-order diffusions near sharp gradients are prevented, while the highly selective property of damping is retained.

The second simpler (flux limited) scheme simply ensures that the diffusive fluxes are always downgradient; otherwise, the fluxes are set to zero. This much simpler scheme yields as good a solution in 1D cases as and better solutions in 2D than the one using the first more elaborate flux limiter. The scheme also preserves monotonicity in the solutions and is computational much more efficient.

The simple flux-limited fourth- and sixth-order diffusion schemes are also applied to thermal bubble convection. It is shown that overshooting and undershooting are consistently smaller when the flux-limited version of the high-order diffusion is used, no matter whether the advection scheme is monotonic or not. This conclusion applies to both scalar and momentum fields. Higher-order monotonic diffusion works better and even more so when used together with monotonic advection.

1. Introduction

Most numerical models employ numerical diffusion or computational mixing to control small-scale (near two grid intervals in wavelength) noise that can arise from numerical dispersion, nonlinear instability, discontinuous physical processes, and external forcing. Such diffusion cannot always be substituted for by phys-

resents other processes or sources. The last term on the right-hand side (rhs) is the added diffusion term and n (=0, 2, 4, 6, . . .) denotes the order. Here α_n is called diffusion coefficient. Diffusion or smoothing can also be introduced by periodically applying a spatial filter or smoother to the predicted field (e.g., Shapiro 1970, 1975). Its effect is often equivalent to applying diffusion in the prognostic equation for a period of time.

Governing equations

High-order monotonic diffusion
(see paper)

$$\frac{\partial \phi}{\partial t} = S + (-1)^{n/2+1} \alpha_n \nabla^n \phi, \qquad (1)$$

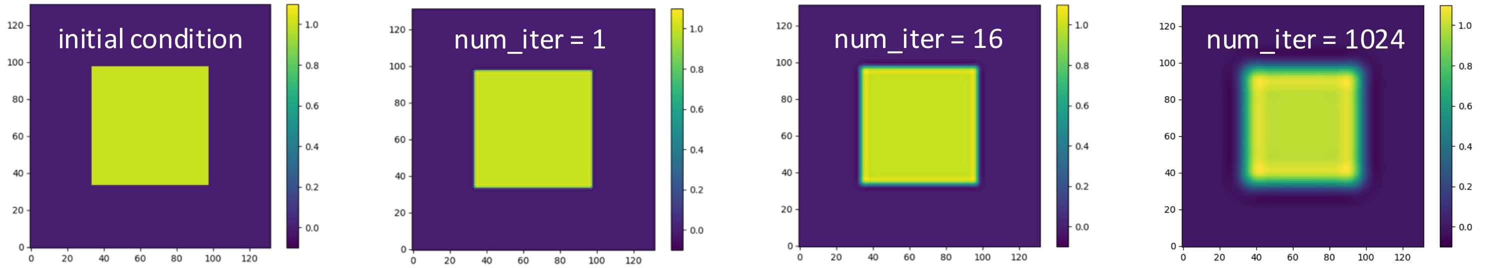
Simplifications (see paper)

- ignore other processes (S=0)
- 4th-order (n=4)
- without limiter

$$S = 0 \quad n = 4 \quad \text{horizontal}$$

$$\frac{\partial \phi}{\partial t} = -\alpha_4 \nabla_h^4 \phi$$
$$= -\alpha_4 \Delta_h^2 \phi \qquad \Delta_h = \nabla_h^2$$

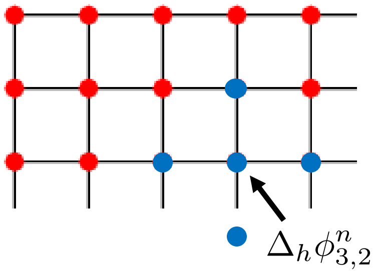
$$\boxed{\frac{\partial \phi}{\partial t} = -\alpha_4 \Delta_h \left(\Delta_h \phi\right)}$$



Discretization: Finite difference method (FDM)

Computational grid

Store values at gridpoints (i,j)



$$x_{i,j} = i\,\Delta x \quad y_{i,j} = j\,\Delta y$$

$$t^n = n\,\Delta t$$

$$\phi^n_{i,j} = \phi(x_{i,j}, y_{i,j}, t^n)$$

Spatial disretization

2nd-order centered

$$\Delta_h \phi^n_{i,j} \approx \left(-4\,\phi^n_{i,j} + \phi^n_{i-1,j} + \phi^n_{i+1,j} + \phi^n_{i,j-1} + \phi^n_{i,j+1}\right)/\Delta x^2$$

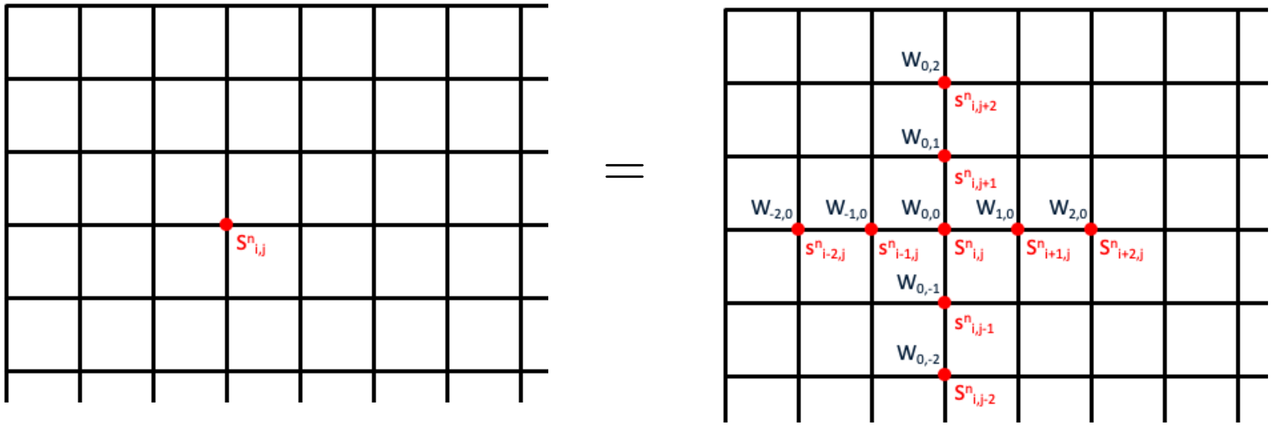
Time discretization

1st-order forward (Euler)

$$\partial_t \phi^n_{i,j} \approx \left(\phi^{n+1}_{i,j} - \phi^n_{i,j}\right)/\Delta t$$

Stencil computation (5x5, sparse)

$$s^{n+1}_{i,j} = \sum_{i_{\rm rel},j_{\rm rel}} w_{i_{\rm rel},j_{\rm rel}}\, s^{n}_{i+i_{\rm rel},j+j_{\rm rel}}$$



weighted sum of grid values
compact neighborhood
same pattern ∀ gridpoints

“A *stencil computation* is an algorithmic motif where the value at a certain grid point is computed from a compact neighborhood of gridpoints in it’s vicinity on the computational grid using the same pattern for every gridpoint.”

Stencil program

$$\frac{\partial \phi}{\partial t} = -\alpha_4 \, \Delta_h \left(\Delta_h \phi \right)$$

Discretization

$$\Delta_h \phi^n_{i,j} \approx \left(-4\, \phi^n_{i,j} + \phi^n_{i-1,j} + \phi^n_{i+1,j} + \phi^n_{i,j-1} + \phi^n_{i,j+1} \right) / \Delta x^2$$

$$\partial_t \phi^n_{i,j} \approx \left(\phi^{n+1}_{i,j} - \phi^n_{i,j} \right) / \Delta t$$

$$\alpha = \alpha_4 \frac{\Delta t}{\Delta x^2}$$

Implementation

```
do num_iter:
  tmp = lap(in)
  out = lap(tmp)
  out = in - α out
  swap(in, out)
```

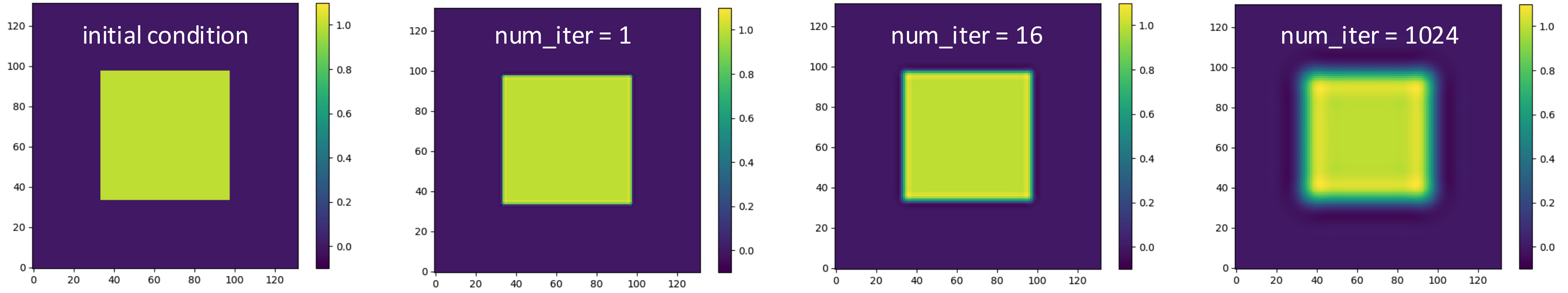
Structure of stencil_2d.F90 / stencil_2d.py

$$\mathtt{tmp}_{\mathrm{i,j,k}} = lap(\mathtt{in}_{i,j,k}) = -4\,\mathtt{in}_{i,j,k} + \mathtt{in}_{i-1,j,k} + \mathtt{in}_{i+1,j,k} + \mathtt{in}_{i,j-1,k} + \mathtt{in}_{i,j+1,k}$$

$$\mathtt{out}_{\mathrm{i,j,k}} = lap(\mathtt{tmp}_{i,j,k}) = -4\,\mathtt{tmp}_{i,j,k} + \mathtt{tmp}_{i-1,j,k} + \mathtt{tmp}_{i+1,j,k} + \mathtt{tmp}_{i,j-1,k} + \mathtt{tmp}_{i,j+1,k}$$

$$\mathtt{out}_{\mathrm{i,j,k}} = \mathtt{in}_{\mathrm{i,j,k}} - \alpha\,\mathtt{out}_{\mathrm{i,j,k}}$$

$$\mathtt{if}\quad \mathtt{iter} < \mathtt{num\_iter} \quad : \quad \mathrm{swap}\,(\mathtt{in}, \mathtt{out})$$



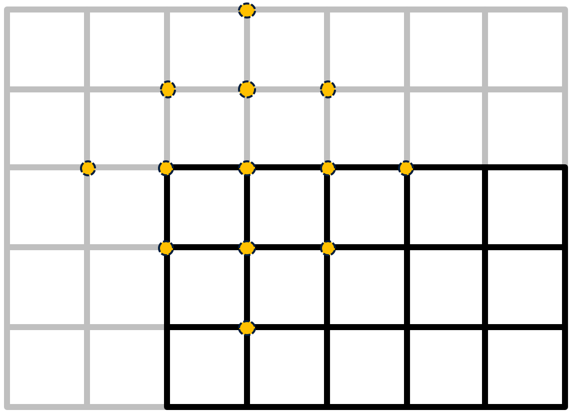
Halo points

`halo_update( in )`

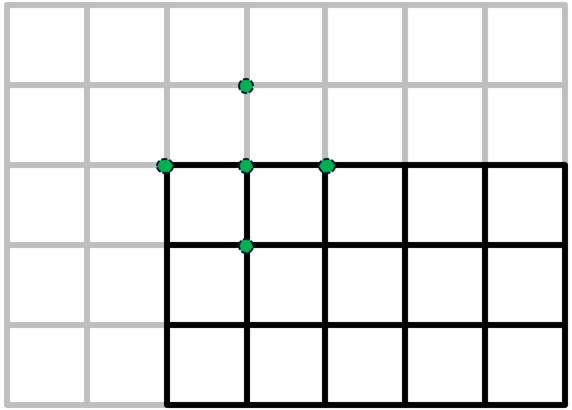
$$\mathtt{tmp}_{\mathrm{i,j,k}} = lap(\mathtt{in}_{i,j,k}) \quad = \quad -4\,\mathtt{in}_{i,j,k} + \mathtt{in}_{i-1,j,k} + \mathtt{in}_{i+1,j,k} + \mathtt{in}_{i,j-1,k} + \mathtt{in}_{i,j+1,k}$$

$$\mathtt{out}_{\mathrm{i,j,k}} = lap(\mathtt{tmp}_{i,j,k}) \quad = \quad -4\,\mathtt{tmp}_{i,j,k} + \mathtt{tmp}_{i-1,j,k} + \mathtt{tmp}_{i+1,j,k} + \mathtt{tmp}_{i,j-1,k} + \mathtt{tmp}_{i,j+1,k}$$

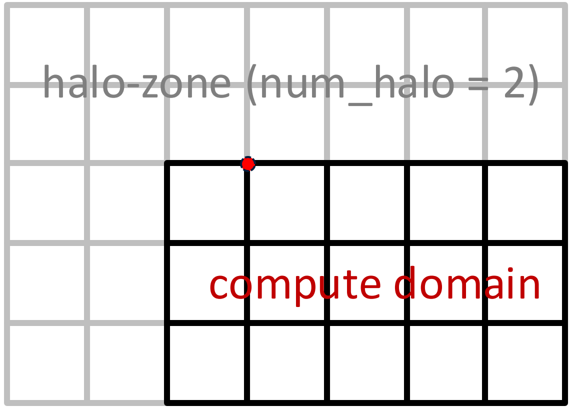
$$\mathtt{out}_{\mathrm{i,j,k}} \quad = \quad \mathtt{in}_{\mathrm{i,j,k}} - \alpha\ \mathtt{out}_{\mathrm{i,j,k}}$$



`in(i,j,1)`

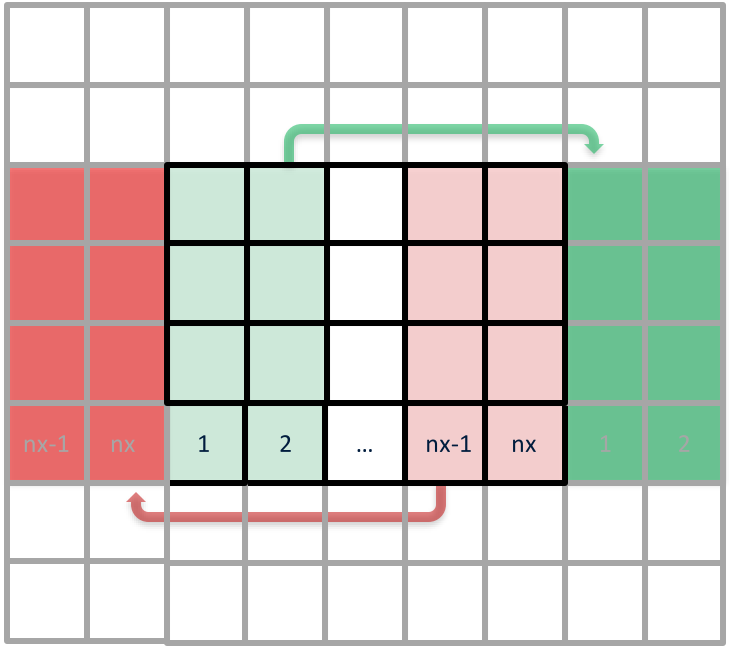


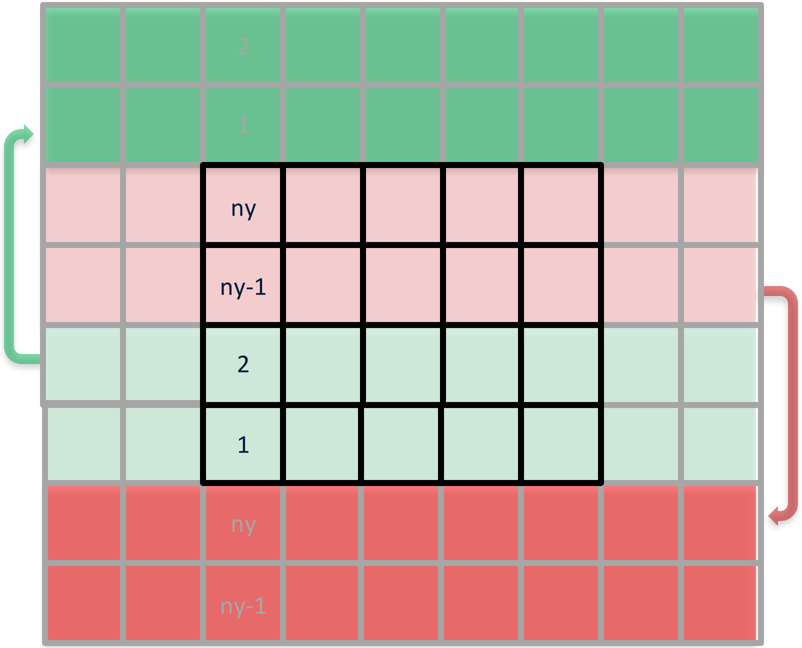
`tmp(i,j,1)`



`out(i,j,1)`

Halo update (periodic)





field(nx + 2 * num_halo, ny + 2 * num_halo, nz)

Not stencils

- (Some) implicit methods (e.g. vertical advection)
- Spectral methods (e.g. IFS dynamical core)
- Other algorithmic motifs
  - Reductions
  - Searches
  - …

Summary

- Spatial resolution drives required computational power $1/\Delta x^3$, this is why we need supercomputers for modeling weather and climate.
- While models use a myriad of different numerical techniques to solve the governing PDEs, the main algorithmic motif resulting from these methods are stencil computations.

Our course will **focus exclusively on stencil computations**, but the concepts are more general and transferable.

Lab Exercises

01-roofline-model.ipynb

- Learn about performance metrics and how to compute theoretical peak values.
- Learn about arithmetic intensity and performance limiters.

02-stencil-program.ipynb

- Determine arithmetic intensity of a stencil program.
- Apply a performance profiling tool to gain insight into performance.
- Show limitations of the von Neumann model for understanding performance.

03-caches-data-locality.ipynb

- Determine arithmetic intensity of a stencil program.
- Apply a performance profiling tool to gain insight into performance.
- Show limitations of the von Neumann model for understanding performance.