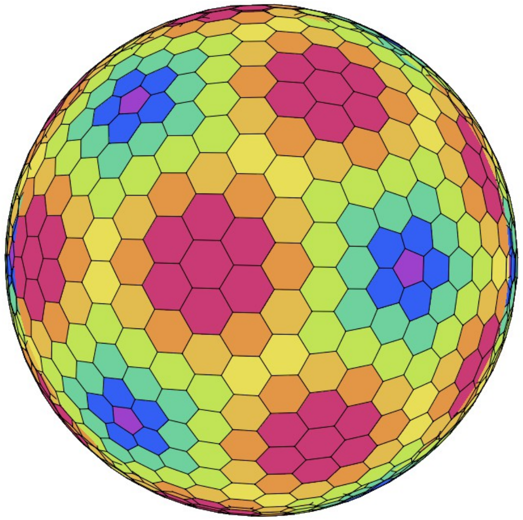
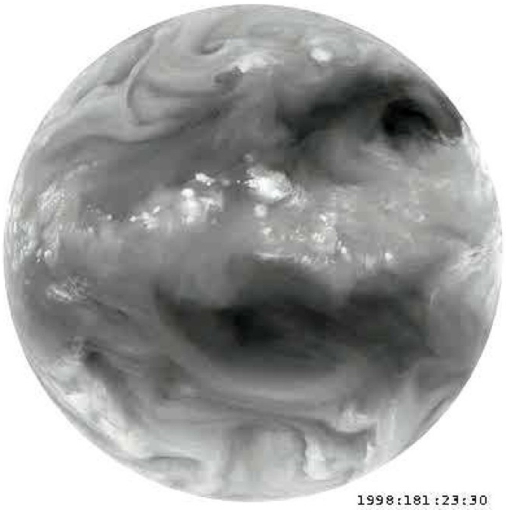
The DYNAMICO dynamical core : theory, implementation and outlook

Thomas Dubos
École Polytechnique, LMD/IPSL, Palaiseau, France

Yann Meurdesoif
CEA, LSCE/IPSL, Orsay, France

$$\delta \int \mathcal{L} \mathrm{d}t = 0$$





DYNAMICO

Equations of motion		*shallow-water*
		shalllow-atmosphere, hydrostatic
	ongoing :	*deep-atmosphere, fully compressible*

Conservation properties

Mass (air and species)
Energy

Formulation

Mass : flux-form
Momentum : ***Hamiltonian vector-invariant form***

Vertical coordinate

Terrain-following mass-based

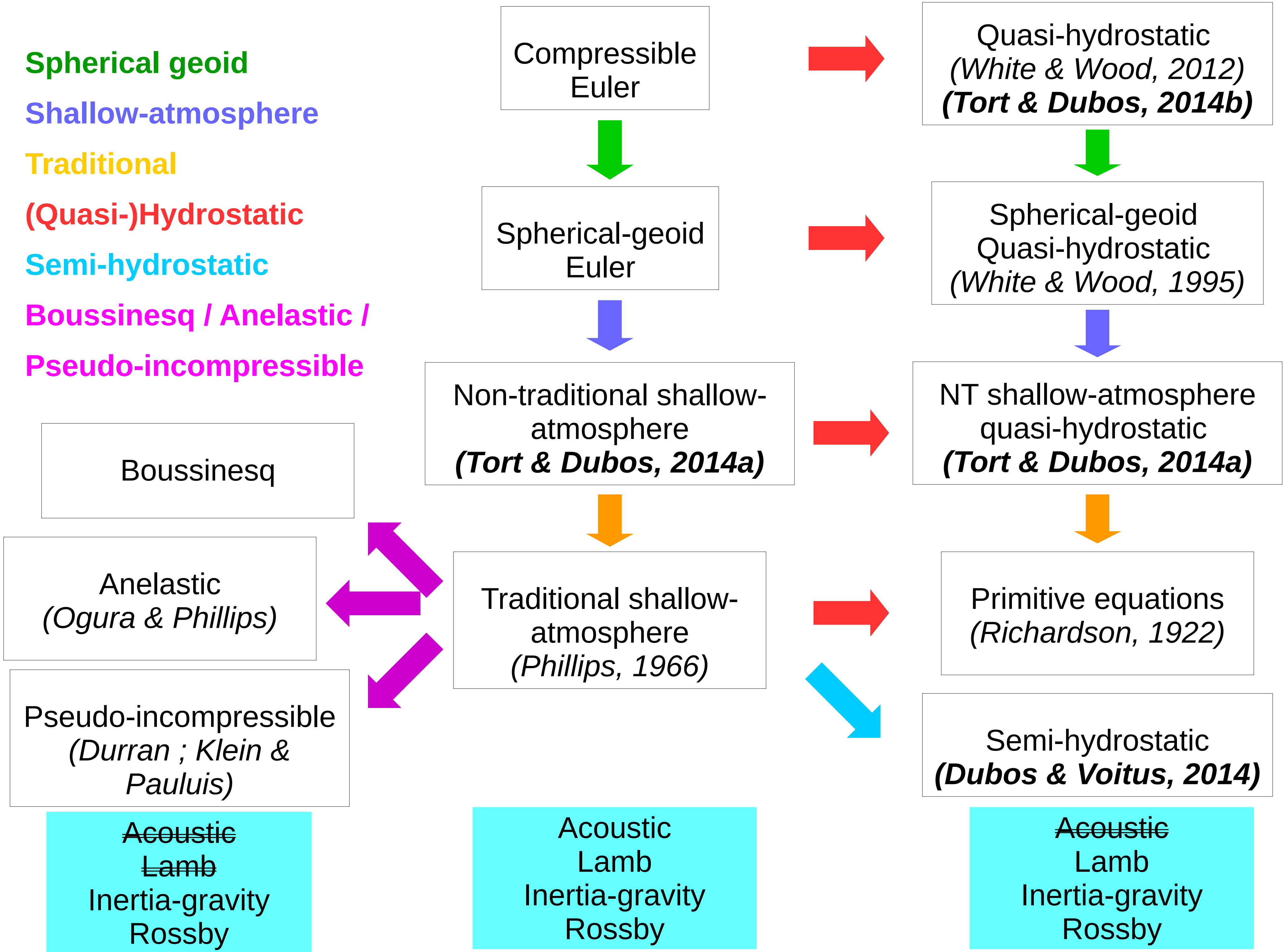
Numerics

Mass : finite volume
Momentum : low-order ***mimetic finite difference***
Mesh : ***icosahedral-hexagonal C-grid***
Time : (additive) Runge-Kutta (HEVI)

Computing

MPI / OpenMP
Scales at least to O(10^4), including ***I/O***

- Hamiltonian formulations
- DYNAMICO status


Spherical geoid
Shallow-atmosphere
Traditional
(Quasi-)Hydrostatic
Semi-hydrostatic
Boussinesq / Anelastic /
Pseudo-incompressible
Compressible Euler
Quasi-hydrostatic (White & Wood, 2012) (Tort & Dubos, 2014b)
Spherical-geoid Euler
Spherical-geoid Quasi-hydrostatic (White & Wood, 1995)
Non-traditional shallow-atmosphere (Tort & Dubos, 2014a)
NT shallow-atmosphere quasi-hydrostatic (Tort & Dubos, 2014a)
Boussinesq
Anelastic (Ogura & Phillips)
Traditional shallow-atmosphere (Phillips, 1966)
Primitive equations (Richardson, 1922)
Pseudo-incompressible (Durran ; Klein & Pauluis)
Semi-hydrostatic (Dubos & Voitus, 2014)
~~Acoustic~~
~~Lamb~~
Inertia-gravity
Rossby
Acoustic
Lamb
Inertia-gravity
Rossby
~~Acoustic~~
Lamb
Inertia-gravity
Rossby

Least action principle

density, *entropy*, *velocity*

$$\delta \int L(\mathbf{x}, \rho, s, \mathbf{u})\mathrm{d}m\mathrm{d}t = 0$$

L=kinetic+Coriolis -internal-potential

$$(x, y, z) \to (\xi^1, \xi^2, \xi^3) \qquad \mathbf{u}$$

$$u^i = D\xi^i / Dt$$

contravariant relative velocity

$$D/Dt = \partial_t + u^i \partial_i$$

$$v_i = \partial L / \partial u^i$$

covariant absolute velocity

$$v_i = G_{ij} u^j + R_j$$

Euler-Lagrange equations

Tort & Dubos, JAS 2014

$$\frac{Dv_i}{Dt} - \frac{\partial L}{\partial \xi^i} = \frac{1}{\mu} \partial_i \left(\mu^2 \frac{\partial L}{\partial \mu} \right)$$

Hamiltonian formulation

Dubos & Tort, MWR 2014

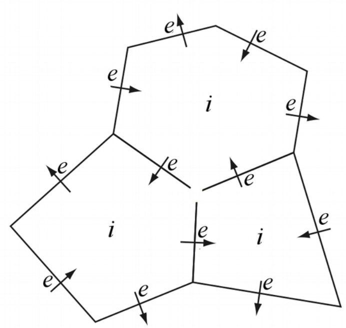
Curl form

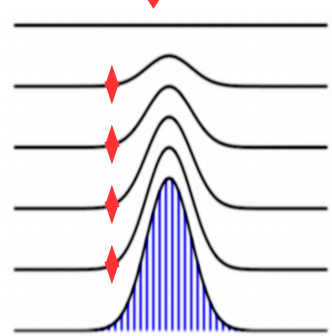
$$\frac{\partial v_i}{\partial t} = \dots$$

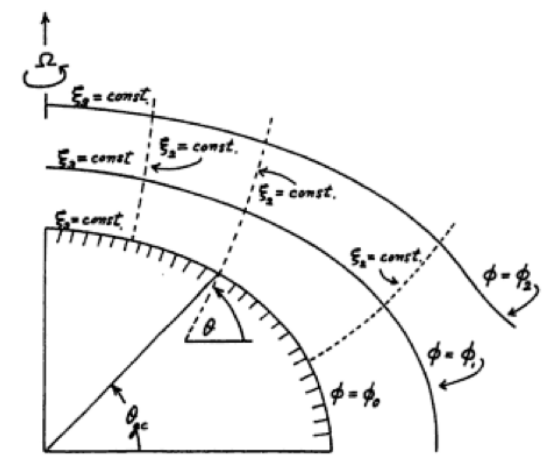
Flux form

$$\frac{\partial}{\partial t}(\mu v_i) = \dots$$

Discretize !







Surfaces of constant ξ_2 and constant ξ_3

Eulerian coordinates in planetary frame

Generalized vertical coordinate

Kinematics / dynamics separation

Mass budget in shallow-atmosphere vs deep-atmosphere geometry :

$$\frac{\partial\rho}{\partial t} + \frac{1}{a\cos\phi}\frac{\partial}{\partial\lambda}\rho u + \frac{1}{a\cos\phi}\frac{\partial}{\partial\phi}\rho\cos\phi v + \frac{\partial}{\partial r}\rho w = 0$$

$$u = a\cos\phi\dot\lambda,\ v = a\dot\phi,\ w = \dot r \qquad \mu = a^2\cos\phi\rho$$

$$\frac{\partial\rho}{\partial t} + \frac{1}{r\cos\phi}\frac{\partial}{\partial\lambda}\rho u + \frac{1}{r\cos\phi}\frac{\partial}{\partial\phi}\rho\cos\phi v + r^{-2}\frac{\partial}{\partial r}\rho r^2 w = 0$$

$$u = r\cos\phi\dot\lambda,\ v = r\dot\phi,\ w = \dot r \qquad \mu = r^2\cos\phi\rho$$

$$\frac{\partial\mu}{\partial t} + \frac{\partial}{\partial\lambda}\left(\mu\dot\lambda\right) + \frac{\partial}{\partial\phi}\left(\mu\dot\phi\right) + \frac{\partial}{\partial r}\left(\mu\dot r\right) = 0$$

$$(\lambda,\phi)\to\mathbf{x}\in S^2 \qquad r\to\eta$$

Kinematics / dynamics separation

- Transport works in a metric-independent computational space $(\mathbf{x},\ \eta) \in S^2 \times [0,1]$
- Metric is needed only when converting between covariant and contravariant
- Diagnosing contravariant (flux) from covariant (momentum) is the business of the dynamics

  => *separation of concerns*

$$\frac{\partial \mu}{\partial t} + \frac{\partial}{\partial \mathbf{x}}\left(\mu \dot{\mathbf{x}}\right) + \frac{\partial}{\partial \eta}\left(\mu \dot{\eta}\right) = 0$$
$$\frac{\partial}{\partial t}\mu\theta + \frac{\partial}{\partial \mathbf{x}}\left(\theta\mu \dot{\mathbf{x}}\right) + \frac{\partial}{\partial \eta}\left(\theta\mu \dot{\eta}\right) = 0$$

Energy / vorticity conserving schemes and the curl (vector-invariant) form

$$\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \times \mathbf{u} + \mathbf{f} \times \mathbf{u} + \nabla \Phi + \frac{1}{\rho} \nabla p = 0$$

$$\mathbf{u} \cdot \nabla \mathbf{u} = (\nabla \times \mathbf{u}) \times \mathbf{u} + \nabla \frac{\mathbf{u} \cdot \mathbf{u}}{2}$$

$$\partial_t \mathbf{u} + (\mathbf{f} + \nabla \times \mathbf{u}) \times \mathbf{u} + \nabla \left(\frac{\mathbf{u} \cdot \mathbf{u}}{2} + \Phi \right) + \frac{1}{\rho} \nabla p = 0$$

- Curl-form : standard starting point to devise energy/vorticity-conserving schemes
- intimately connected to Hamiltonian formulation (Salmon, 1983 ; Gassmann, 2012)
- for **compressible flow,** Hamiltonian formulation well-established if coordinate system is **Eulerian**

A change of perspective

- Usual perspective : choose *prognostic fields* and write down *evolution equations*
- Now suppose evolution equations are known for all prognostic variables. Then how an arbitrary functional of those prognostic variables evolves is entirely known :

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathcal{F}[\mathbf{v},\, \mu,\, \Theta] = \int \left(\frac{\delta \mathcal{F}}{\delta v_i}\partial_t v_i + \frac{\delta \mathcal{F}}{\delta \mu}\partial_t \mu + \frac{\delta \mathcal{F}}{\delta \Theta}\partial_t \Theta \right) \mathrm{d}x^1 \mathrm{d}x^2 \mathrm{d}x^3$$

- Conversely, *such an evolution equation contains all the equations of motion*

New perspective : write a single evolution equation for an arbitrary functional of prognostic fields

- Remark : by design $\frac{\mathrm{d}}{\mathrm{d}t}\mathcal{F}$ is linear in the functional derivatives of $\mathcal{F}$

Hamiltonian formulation

- A Hamiltonian formulation is such an evolution equation of the special form :

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathcal{F} = \{\mathcal{F}, \mathcal{H}\}$$

where H is total energy and the Poisson bracket is bilinear, antisymmetric and satisfies the Jacobi identity

- nice, but where is my curl form ?

The curl (vector-invariant) form and Hamiltonian formulation

$$\partial_t \mathbf{u} + (\mathbf{f} + \nabla \times \mathbf{u}) \times \mathbf{u} + \nabla\left(\frac{\mathbf{u}\cdot\mathbf{u}}{2} + \Phi\right) + \frac{1}{\rho}\nabla p = 0$$

$$\partial_t v_i + (\partial_j v_i - \partial_i v_j)\, u^j + \partial_i\left(k + \Phi\right) + \theta \partial_i \pi = 0$$

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathcal{F} = \{\mathcal{F}, \mathcal{H}\}$$

- means that all tendencies can be written as expressions that are *linear in the functional derivatives of total energy*

$$\mathcal{H}\left[v_i, \mu, \Theta\right] = \int \left(\frac{1}{2} g^{ij}(v_i - R_i)(v_j - R_j) + \Phi + e\left(\alpha = \frac{J}{\mu}, \theta\frac{\Theta}{\mu}\right)\right)\mu \mathrm{d}x^1\mathrm{d}x^2\mathrm{d}x^3$$

- To do that let us first find those functional derivatives : $v_i = g_{ij}u^j + R_i, k = \frac{1}{2}g_{ij}u^i u^j$

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathcal{H} = \int \left(\mu u^i \partial_t v_i + (k + \Phi)\,\partial_t \mu + \pi \partial_t \Theta\right)\mathrm{d}x^1\mathrm{d}x^2\mathrm{d}x^3$$

The curl (vector-invariant) form and Hamiltonian formulation

$$\boxed{\begin{aligned} \partial_t \mu + \partial_i \frac{\delta \mathcal{H}}{\delta v_i} &= 0 \\ \partial_t \Theta + \partial_i \left(\theta \frac{\delta \mathcal{H}}{\delta v_i} \right) &= 0 \\ \partial_t v_i + \frac{\partial_j v_i - \partial_i v_j}{\mu} \frac{\delta \mathcal{H}}{\delta v_j} + \partial_i \frac{\delta \mathcal{H}}{\delta \mu} + \theta \partial_i \frac{\delta \mathcal{H}}{\delta \Theta} &= 0 \end{aligned}}$$

$$\frac{\mathrm{d}}{\mathrm{d}t} \mathcal{F} = \{\mathcal{F}, \mathcal{H}\}$$

- means that all tendencies can be written as expressions that are *linear in the functional derivatives of total energy*

$$\mathcal{H}\left[v_i, \mu, \Theta\right] = \int \left(\frac{1}{2} g^{ij} (v_i - R_i)(v_j - R_j) + \Phi + e\left(\alpha = \frac{J}{\mu}, \theta \frac{\Theta}{\mu} \right) \right) \mu \mathrm{d}x^1 \mathrm{d}x^2 \mathrm{d}x^3$$

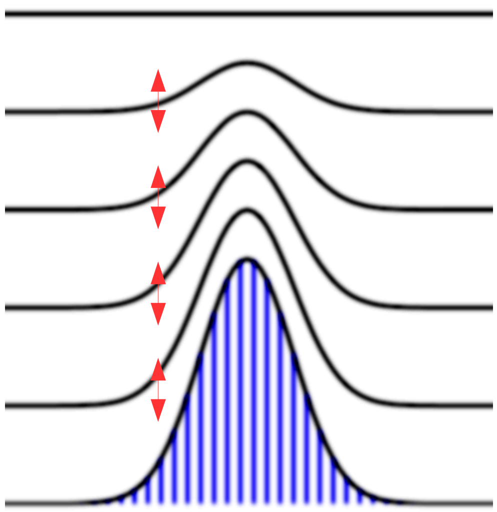
- To do that let us first find those functional derivatives : $v_i = g_{ij} u^j + R_i, k = \frac{1}{2} g_{ij} u^i u^j$

$$\frac{\mathrm{d}}{\mathrm{d}t} \mathcal{H} = \int \left(\mu u^i \partial_t v_i + (k + \Phi)\, \partial_t \mu + \pi \partial_t \Theta \right) \mathrm{d}x^1 \mathrm{d}x^2 \mathrm{d}x^3$$

Energy / vorticity conserving schemes and the curl (vector-invariant) form

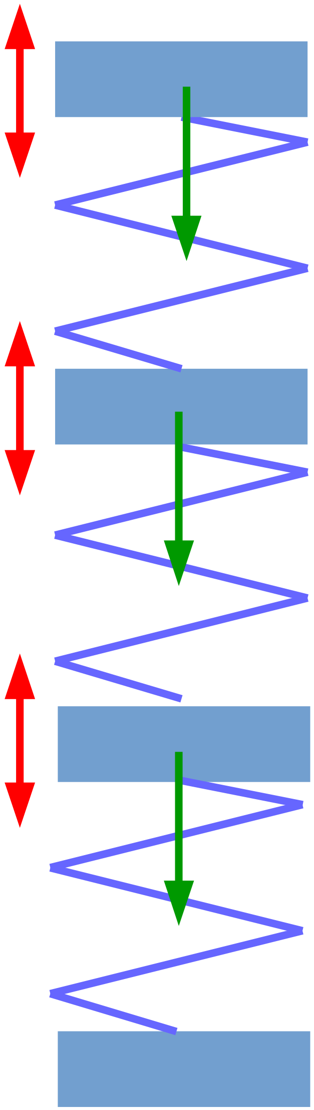
$$\partial_t \mathbf{u} + (\mathbf{f} + \nabla \times \mathbf{u}) + \nabla \left(\frac{\mathbf{u} \cdot \mathbf{u}}{2} + \Phi \right) + \frac{1}{\rho} \nabla p = 0$$

- curl-form standard starting point to devise energy/vorticity-conserving schemes
- intimately connected to Hamiltonian formulation (Salmon, 1983 ; Gassmann, 2012)
- for **compressible flow,** Hamiltonian formulation well-established if coordinate system is **Eulerian**



We want to handle **non-Eulerian** vertical coordinates :

- Isentropic/isopycnal
- based on mass / hydrostatic pressure
- natural representation of **hydrostatic adjustment**



Hamiltonian formulation for generalized vertical coordinates :
Dubos & Tort (MWR 2014)

Hamiltonian formulation in generalized vertical coordinates

(Dubos & Tort, MWR 2014)

$$\begin{aligned}
\partial_t \mu + \partial_\eta \left(\mu \dot\eta\right) + \partial_i \frac{\delta \mathcal{H}}{\delta v_i} &= 0, \\
\partial_t \Theta + \partial_\eta \left(\Theta \dot\eta\right) + \partial_i \left(\theta \frac{\delta \mathcal{H}}{\delta v_i}\right) &= 0, \\
\partial_t v_i + \dot\eta \partial_\eta v_i + v_3 \partial_i \dot\eta + \frac{\partial_j v_i - \partial_i v_j}{\mu} \frac{\delta \mathcal{H}}{\delta v_j} + \partial_i \frac{\delta \mathcal{H}}{\delta \mu} + \theta \partial_i \left(\frac{\delta \mathcal{H}}{\delta \Theta}\right) &= 0, \\
\partial_t V_3 + \partial_\eta \left(V_3 \dot\eta\right) + \frac{\delta \mathcal{H}}{\delta \Phi} &= 0, \\
\partial_t \Phi + \dot\eta \partial_\eta \Phi - \frac{\delta \mathcal{H}}{\delta V_3} &= 0.
\end{aligned}$$

Integration by parts
+ invariance w.r.t. vertical coordinate
=> conservation of energy

Isentropic / Isopycnal	*Mass-based*	*Eulerian*
$\dot\eta = 0$	Diagnosed from horizontal mass flux	$\partial_t \Phi = 0$

Hamiltonian formulation in generalized vertical coordinates

(Dubos & Tort, MWR 2014)

$$
\begin{aligned}
\partial_t \mu + \partial_\eta \left(\mu \dot\eta\right) + \partial_i \frac{\delta \mathcal{H}}{\delta v_i} &= 0, \\
\partial_t \Theta + \partial_\eta \left(\Theta \dot\eta\right) + \partial_i \left(\theta \frac{\delta \mathcal{H}}{\delta v_i}\right) &= 0, \\
\partial_t v_i + \dot\eta \partial_\eta v_i + v_3 \cancel{\partial_i \eta} + \frac{\partial_j v_i - \partial_i v_j}{\mu} \frac{\delta \mathcal{H}}{\delta v_j} + \partial_i \frac{\delta \mathcal{H}}{\delta \mu} + \theta \partial_i \left(\frac{\delta \mathcal{H}}{\delta \Theta}\right) &= 0, \\
\cancel{\partial_t V_3} + \partial_\eta \left(\cancel{V_3 \dot\eta}\right) + \frac{\delta \mathcal{H}}{\delta \Phi} &= 0, \\
\cancel{\partial_t \Phi} + \cancel{\dot\eta \partial_\eta \Phi} - \cancel{\frac{\delta \mathcal{H}}{\delta V_3}} &= 0.
\end{aligned}
$$

Hydrostatic $\quad v_3 \equiv \frac{\partial L}{\partial u^3} = 0$

Isentropic / Isopycnal	*Mass-based*	*Eulerian*
$\dot\eta = 0$	Diagnosed from horizontal mass flux	$\partial_t \Phi = 0$

Physical space

$$\lambda, \varphi, \Phi, g$$

$$u, v, w$$

$$\alpha = 1/\rho,\ s,\ r$$

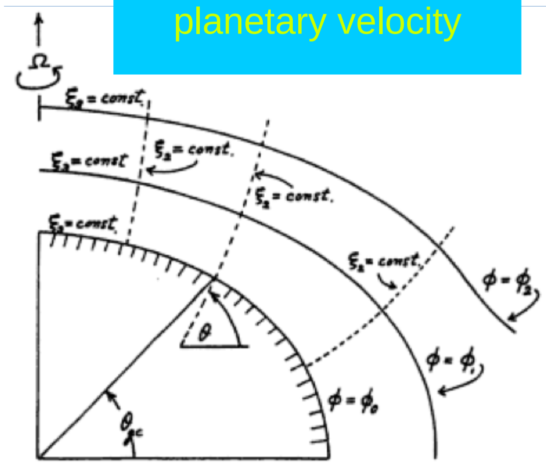
Thermodynamics

$$p, T, \chi$$

Geopotential Coordinates

Metric, gravity,

planetary velocity



Surfaces of constant ξ_2 and constant ξ_3

Computational space $S^2 \times [0,1]$

$$\mathbf{v} = \mathbf{G}\,(\mathbf{x}, \Phi) \cdot \dot{\mathbf{x}} + \mathbf{R}(\mathbf{x}, \Phi)$$

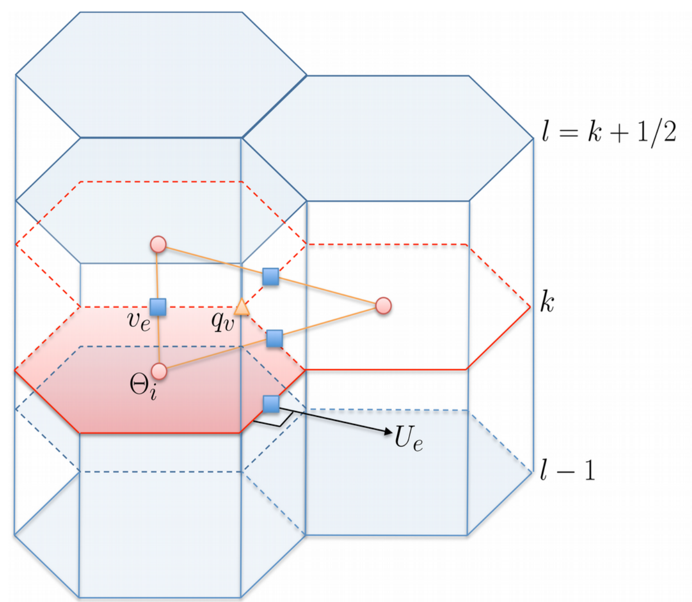
$\mathbf{x}$ $\qquad$ η

Horizontal mesh

Icosahedral C-grid

Vertical mesh

Lorenz



Discrete space

$$m_{ik} = \int\int\int \mu \mathrm{d}\mathbf{x}\mathrm{d}\eta$$

$$F_{il} = \int\int \mu\dot{\eta}\mathrm{d}\mathbf{x}$$

$$v_{ek} = \int \mathbf{v} \cdot \mathrm{d}\mathbf{x}$$

$$\alpha_{ik} = \alpha(p_{ik}, s_{ik}),$$

- Discrete integration by parts (Bonaventura & Ringler, 2005 ; Taylor, 2010)
- Energy- and vorticity- conserving Coriolis discretization (TRiSK : Thuburn et al., 2009 ; Ringler et al., 2010)

Energy-conserving 3D core

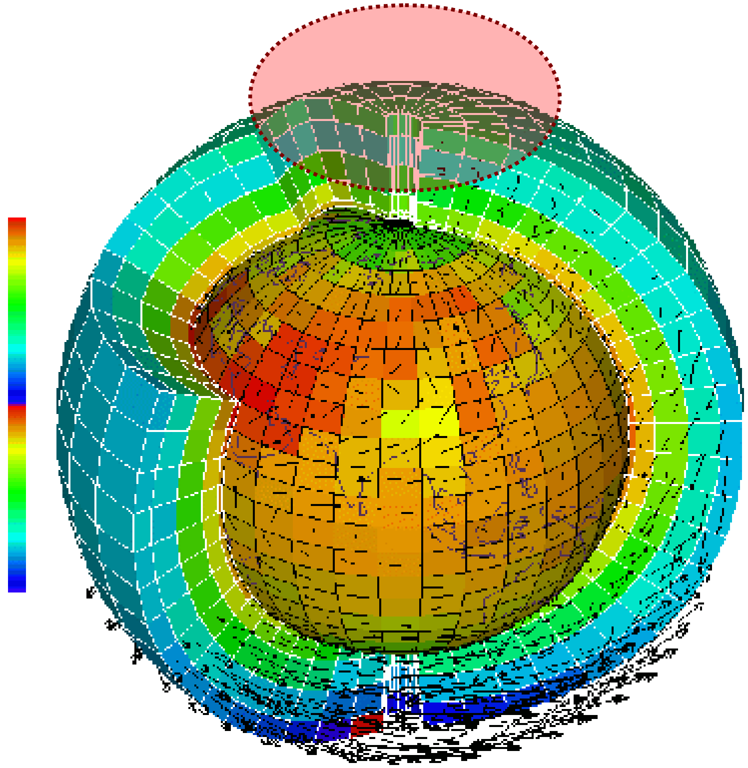
Hamiltonian structure of the equations of motion can :

- Guide the choice of prognostic variables
- Help with discrete conservation of energy
  (up to time truncation errors)

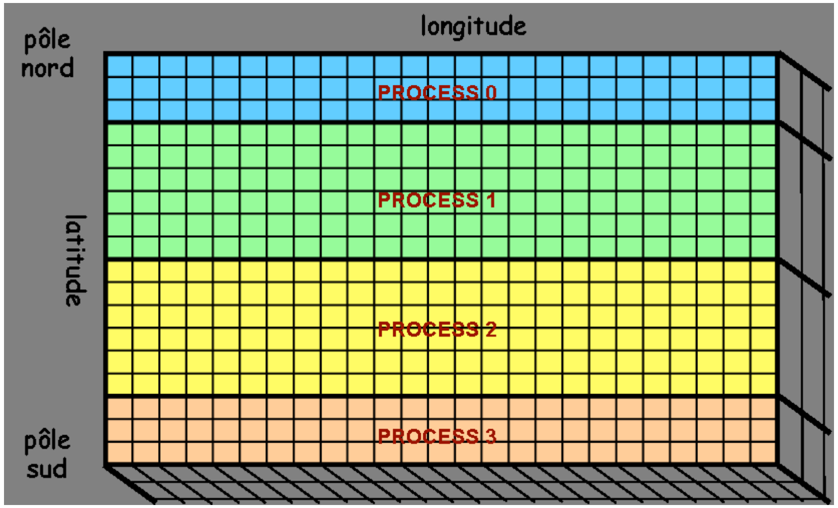
- Create opportunities for *modularity / separation of concerns* :
    - Bracket (general structure) independent from specific equation set
    - Hamiltonian specific to certain equation set :
      equation of state, shallow-atmosphere vs deep-atmosphere

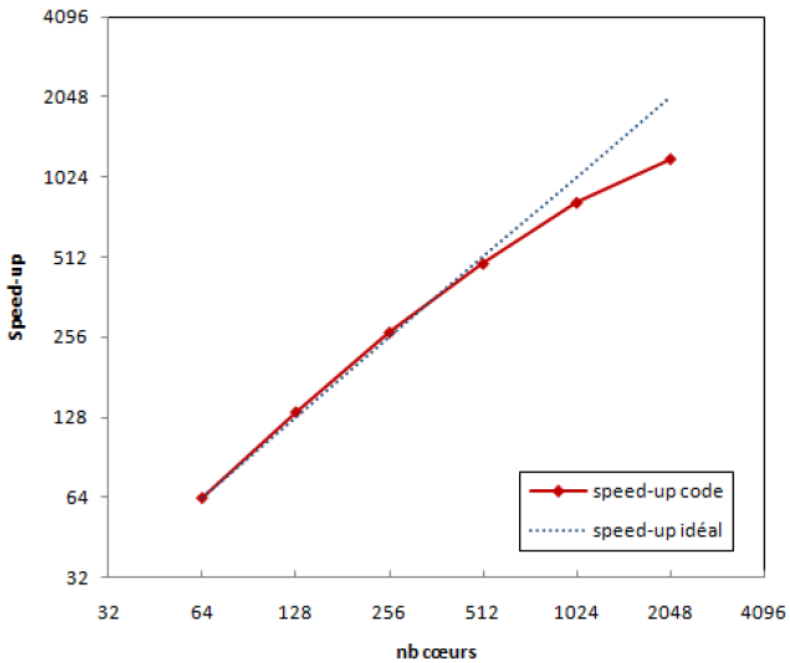
- Implemented in DYNAMICO with FD but other building blocks possible :
  - spectral elements (horizontal)
  - compatible finite elements (GungHo)

- Hamiltonian formulations
- DYNAMICO status



LMD-Z lon-lat core





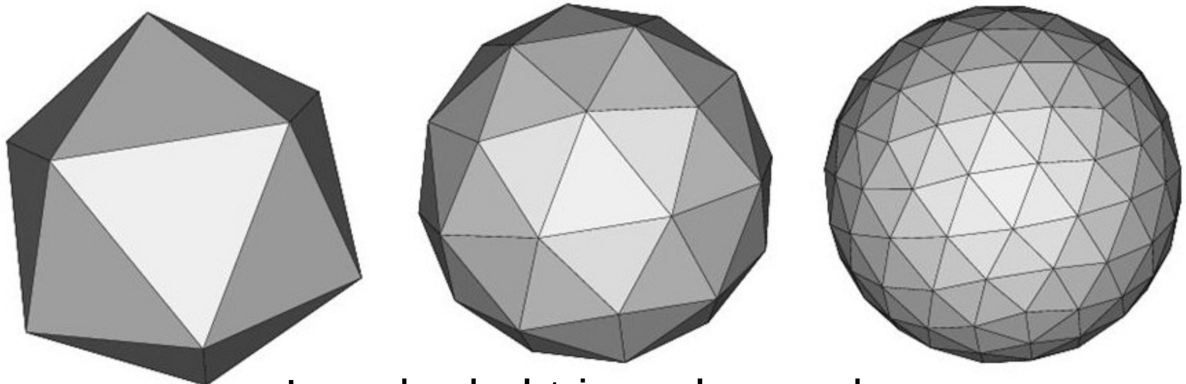
Y. Meurdesoif, 2010

1/4 degree (25km) => max. 360 MPI processes

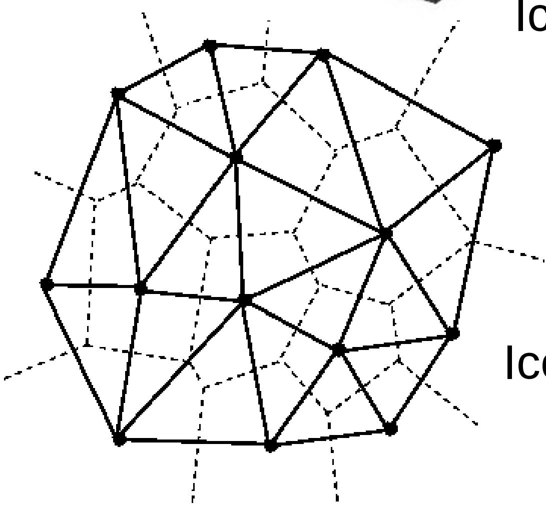
=> a couple thousand cores

Mesh partitioning for parallel computing

nbp=1 *nbp=2* *nbp=4*

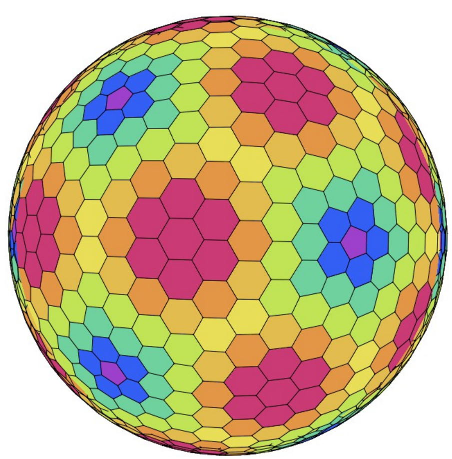


Icosahedral-triangular mesh
10*nbp^2+2 *vertices*

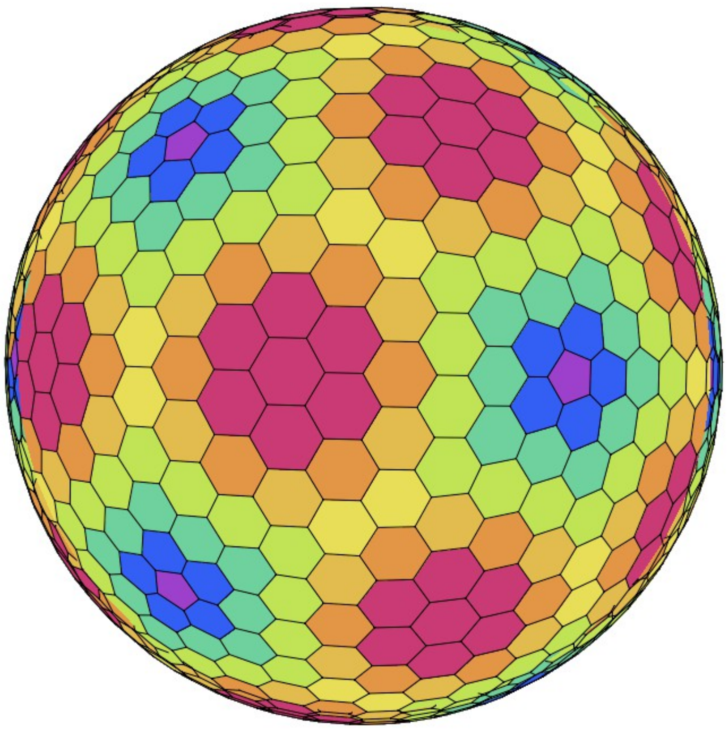


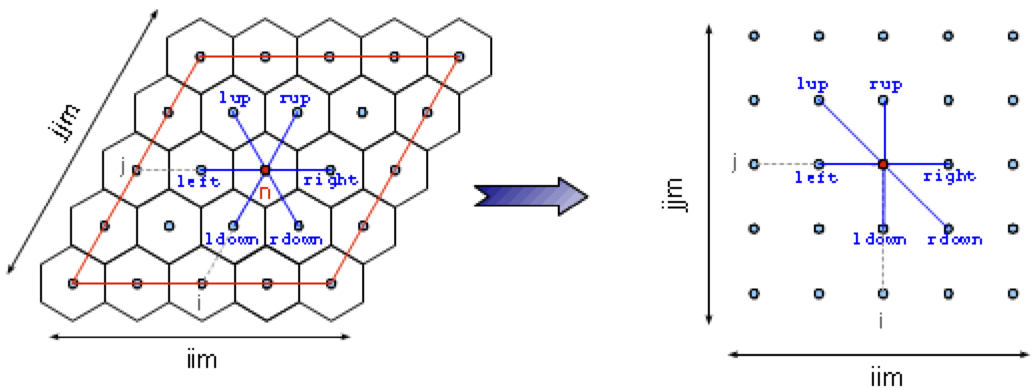
Voronoi dual

Icosahedral-hexagonal mesh
10*nbp^2+2 *cells*

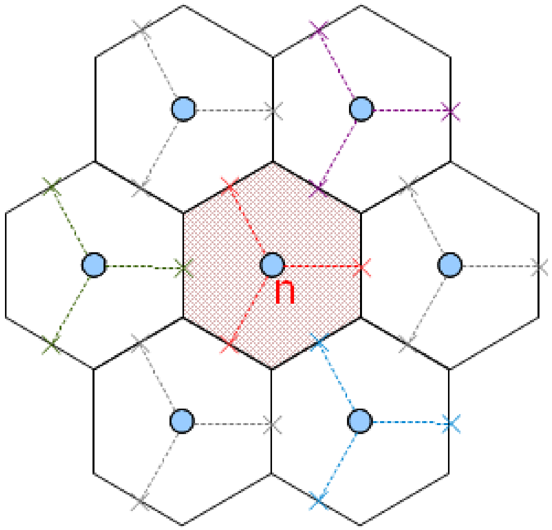


- Easy to partition into 10 x nsplit^2 domains
- About (nbp/nsplit)^2 cells per domain = MPI process
- Nbp/nsplit>10 for performance
- Ex : 25km ~10^6 cells ~10 000 MPI processes





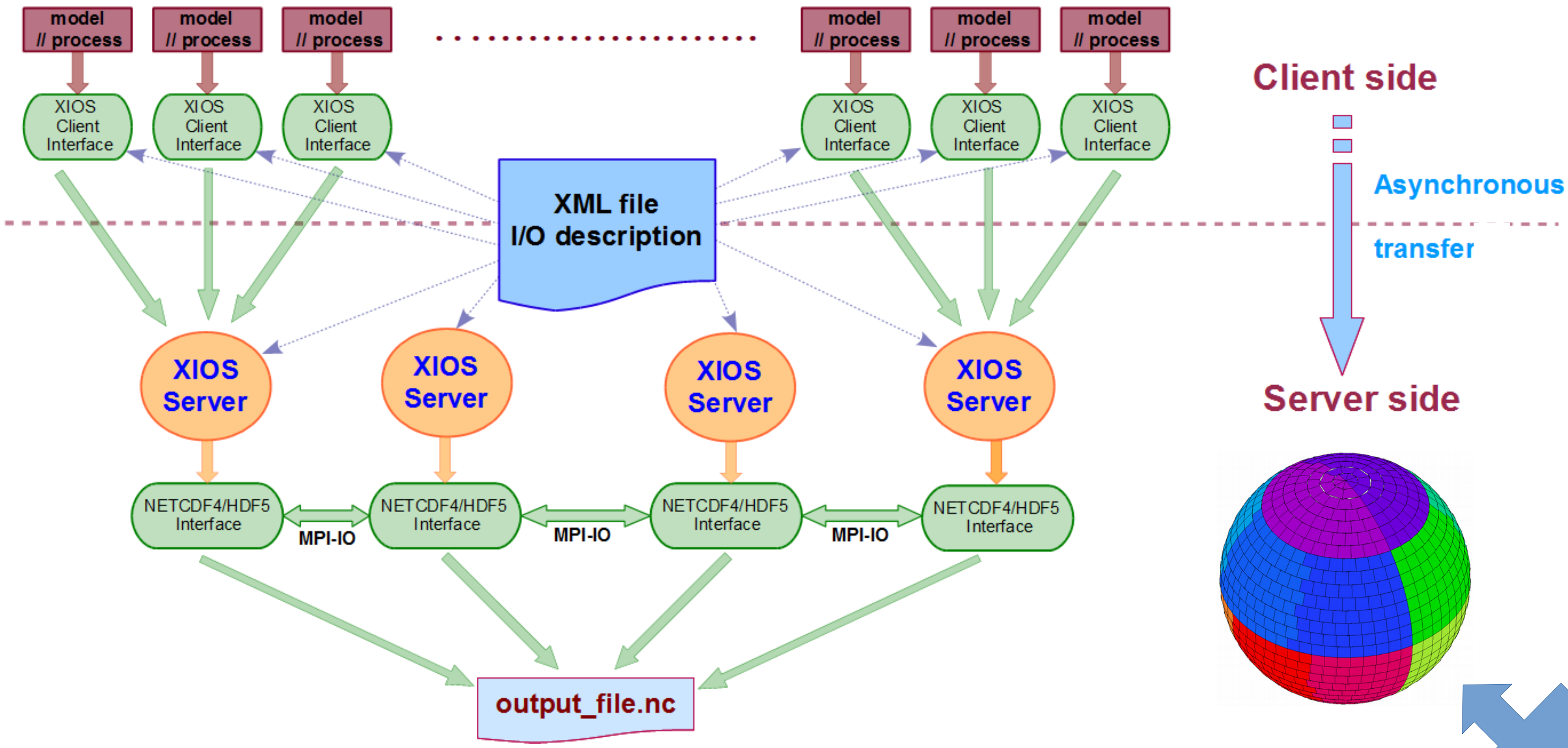
- Vertical direction in outer loops
- **Direct access to neighbours via constant offsets**
- No special case for pentagons (handled by metrics)

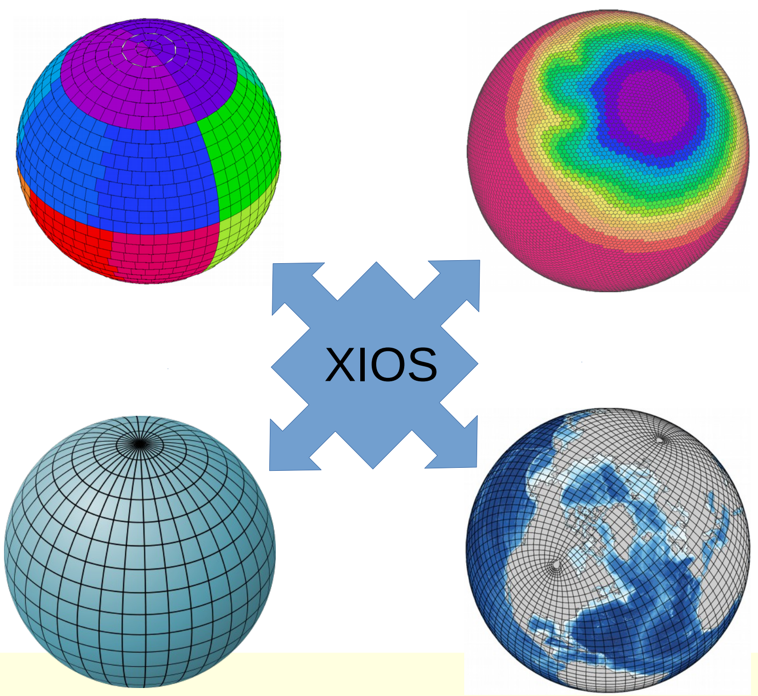


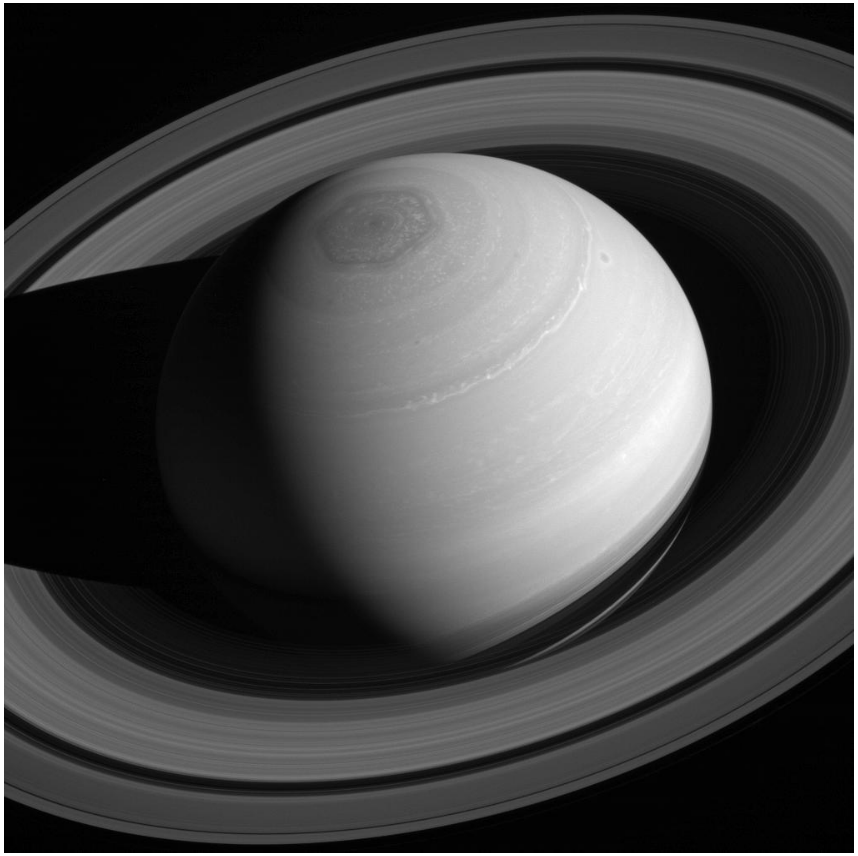

```
DO ij=ij_begin,ij_end
   ! convm = -div(mass flux), sign convention as in Ringler et a
   convm(ij,l)= -1./Ai(ij)*(ne_right*hflux(ij+u_right,l)   +  &
         ne_rup*hflux(ij+u_rup,l)         +  &
         ne_lup*hflux(ij+u_lup,l)         +  &
         ne_left*hflux(ij+u_left,l)       +  &
         ne_ldown*hflux(ij+u_ldown,l)     +  &
         ne_rdown*hflux(ij+u_rdown,l))
ENDDO
```

XIOS (Y. MEURDESOIF) : XML I/O SERVER

PARALLEL ASYNCHRONOUS I/O - ONLINE POST-PROCESSING LIBRARY AND SERVER







Saturn GCM simulations

Grid

- Horizontal resolution: $1/2°$ (+ tests $1/4°$ & $1/8°$)
- Vertical levels: 32 levels from 3 bars to 1 mbar

Boundary conditions

- Initial: steady-state temperature from 1D run, no winds
- Dissipation (SGS): very strong ($\mathcal{D}^+$), strong ($\mathcal{D}^=$), regular ($\mathcal{D}^-$)
- Type 1: 64 levels + sponge layer on uppermost 4 levels
- Type 2: Bottom drag $|\lambda| > 33°$: $\tau = 100$d ($\mathcal{F}^+$), 1000d ($\mathcal{F}^-$) [Liu and Schneider JAS 2010]

Machinery

- MPI+openMP code run on Occigen cluster in CINES
- cores: 1200 ($1/2°$), 9000 ($1/4°$), 11520-30000 ($1/8°$)
- Results are shown after 20000 Saturn days integrations

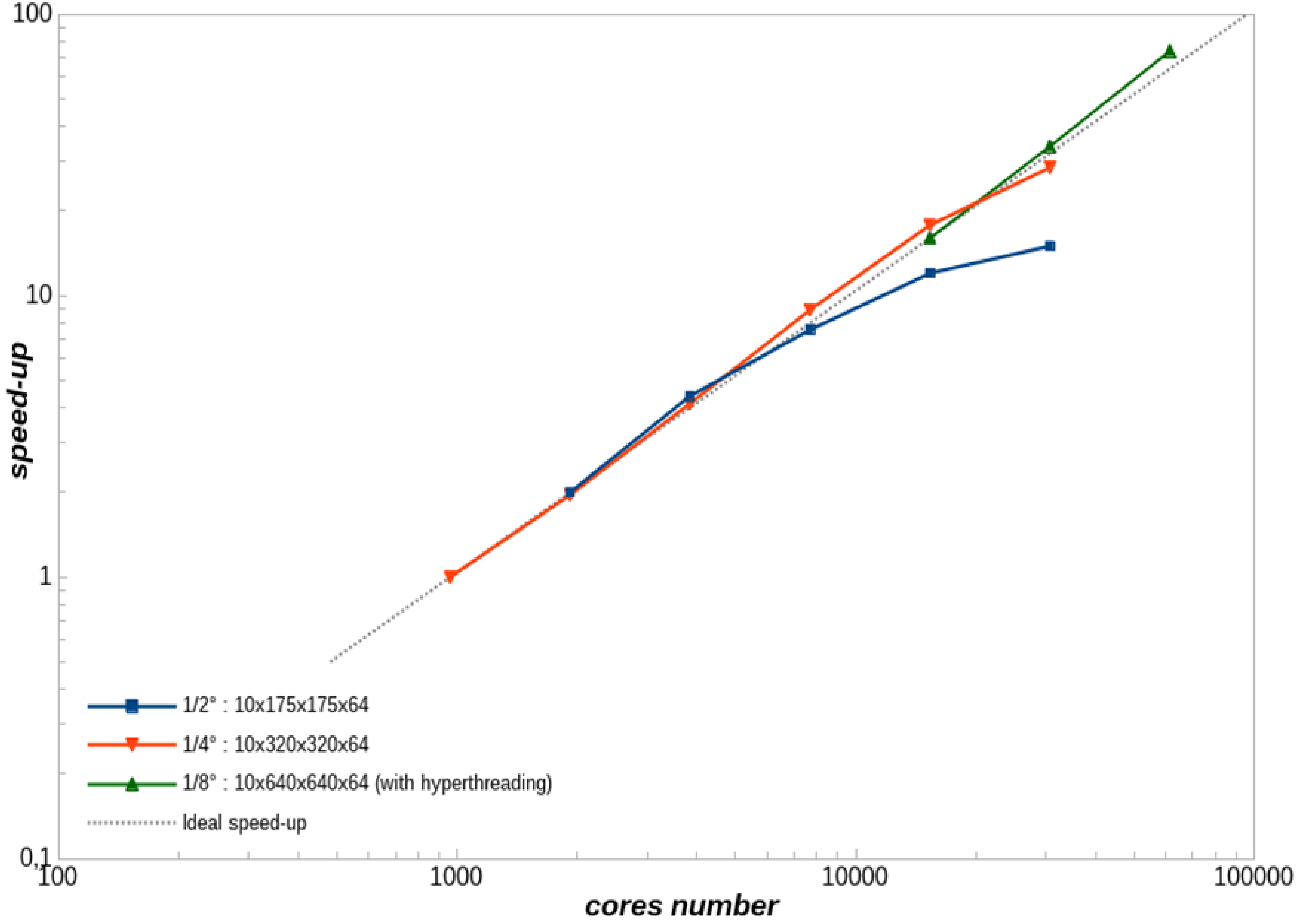
1 Sat day = 10 h ; 1 Sat year = 30 Earth yr

Physical parameterizations ⇒ 1D computations of forcings on each grid point

☞ Radiative transfer ⇒ Guerlet et al. Icarus 2014

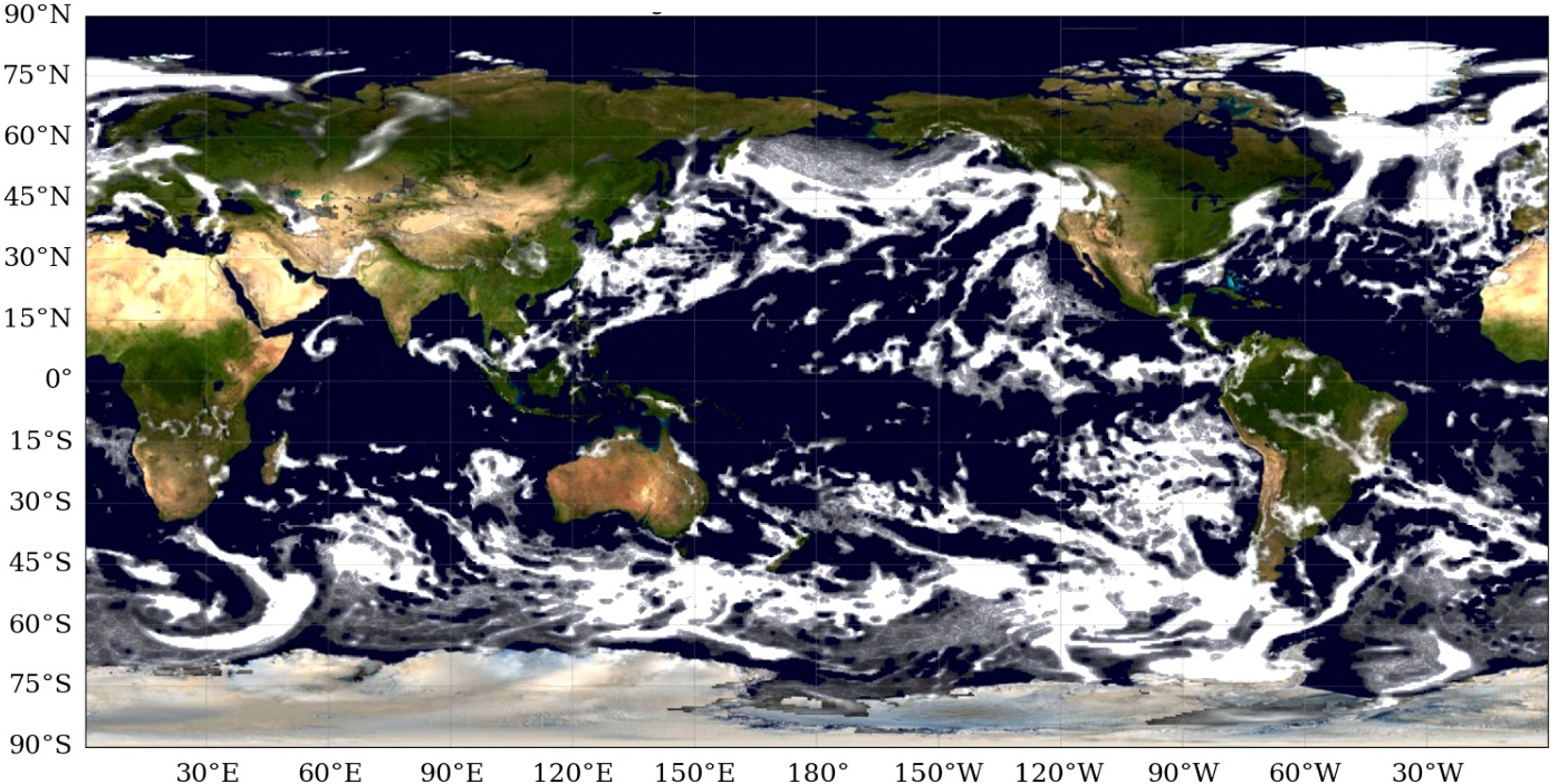
- correlated-k scheme for IR and VIS heating rates [Wordsworth et al. 2010]
- gases CH_4, C_2H_6, C_2H_2 with optimized spectral discretization
- HITRAN 2012 database + Karkoschka and Tomasko 2010 for CH_4 around 1μm
- collision-induced absorption H_2-H_2 and H_2-He [Wordsworth et al. 2012]
- Rayleigh scattering H_2, He
- simple two-layer aerosol model [constrained by Roman et al. 2013]
  - tropospheric haze layer $180 - 660$ mbar / $\tau \sim 8$ / $r = 2\mu$m
  - stratospheric haze layer $1 - 30$ mbar / $\tau \sim 0.1$ / $r = 0.1\mu$m
- free bottom surface with internal heat flux
- incoming flux: ring shadowing, oblateness

☞ Turbulent diffusion + dry convective adjustment [Hourdin et al. 1993]

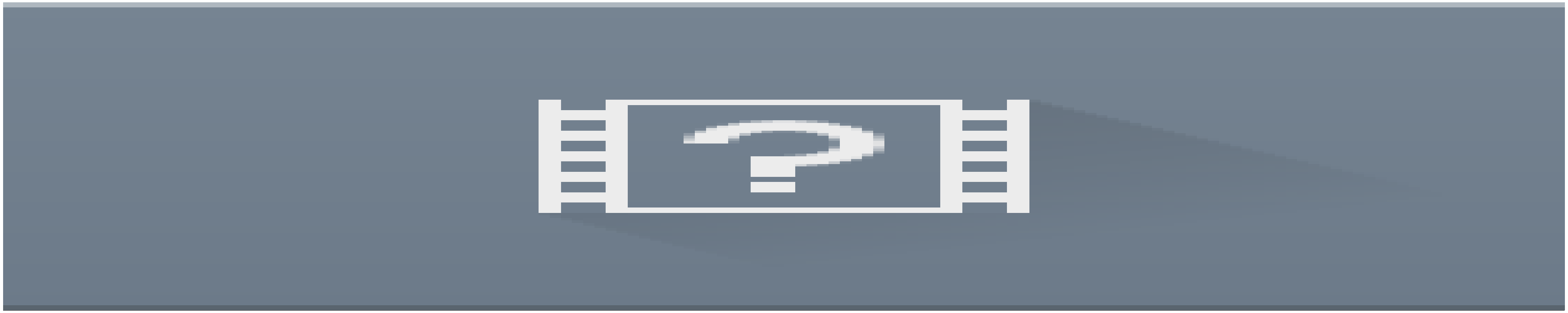

100
10
1
0,1
speed-up
100
1000
10000
100000
cores number
1/2° : 10x175x175x64
1/4° : 10x320x320x64
1/8° : 10x640x640x64 (with hyperthreading)
Ideal speed-up

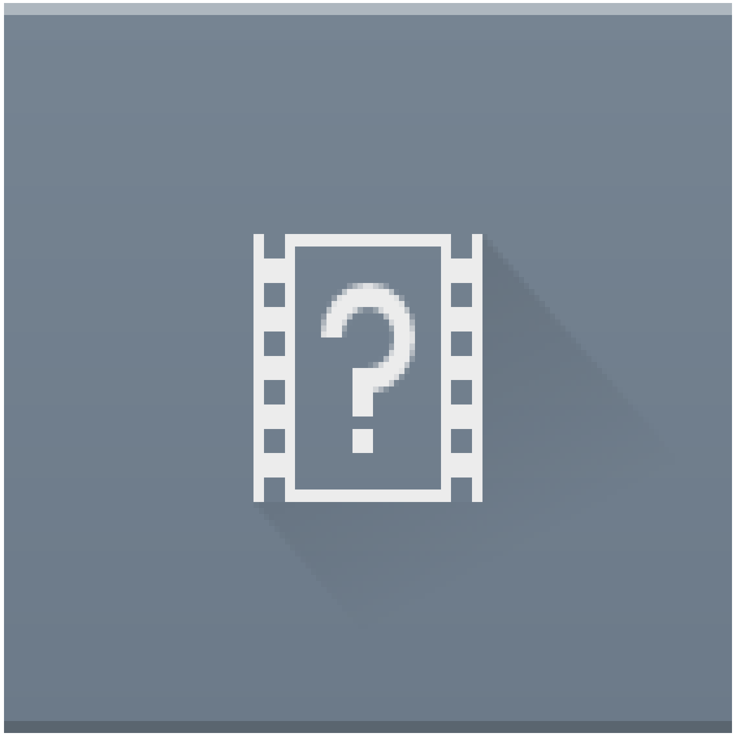
IPSL-CM7A-HR :

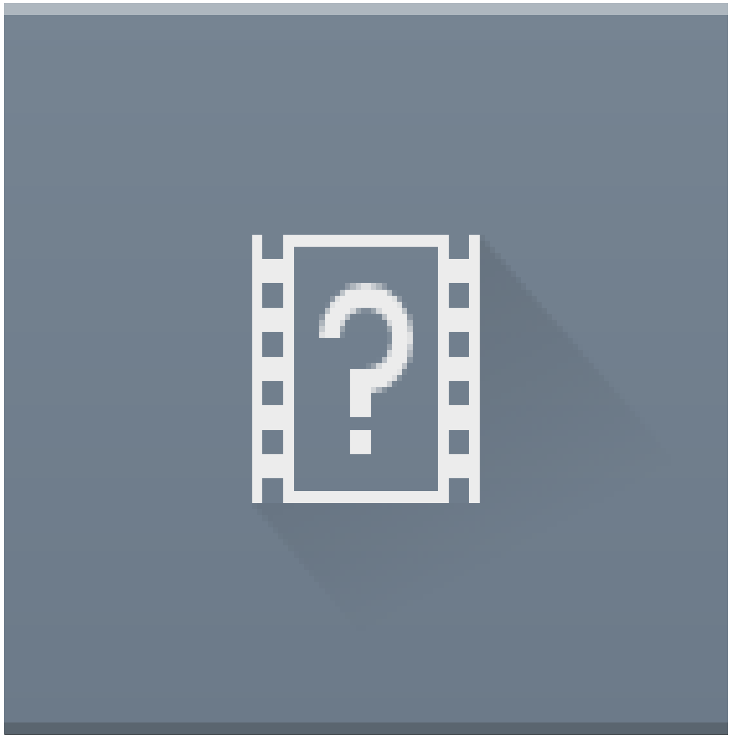
DYNAMICO-LMDZ-ORCHIDEE, 50km
5 SYPD @ 2848 cores – full physics, I/O
256 000 atmospheric columns x 80 vertical levels = 20 million cells



Low-level cloudiness – HighResMIP experiment

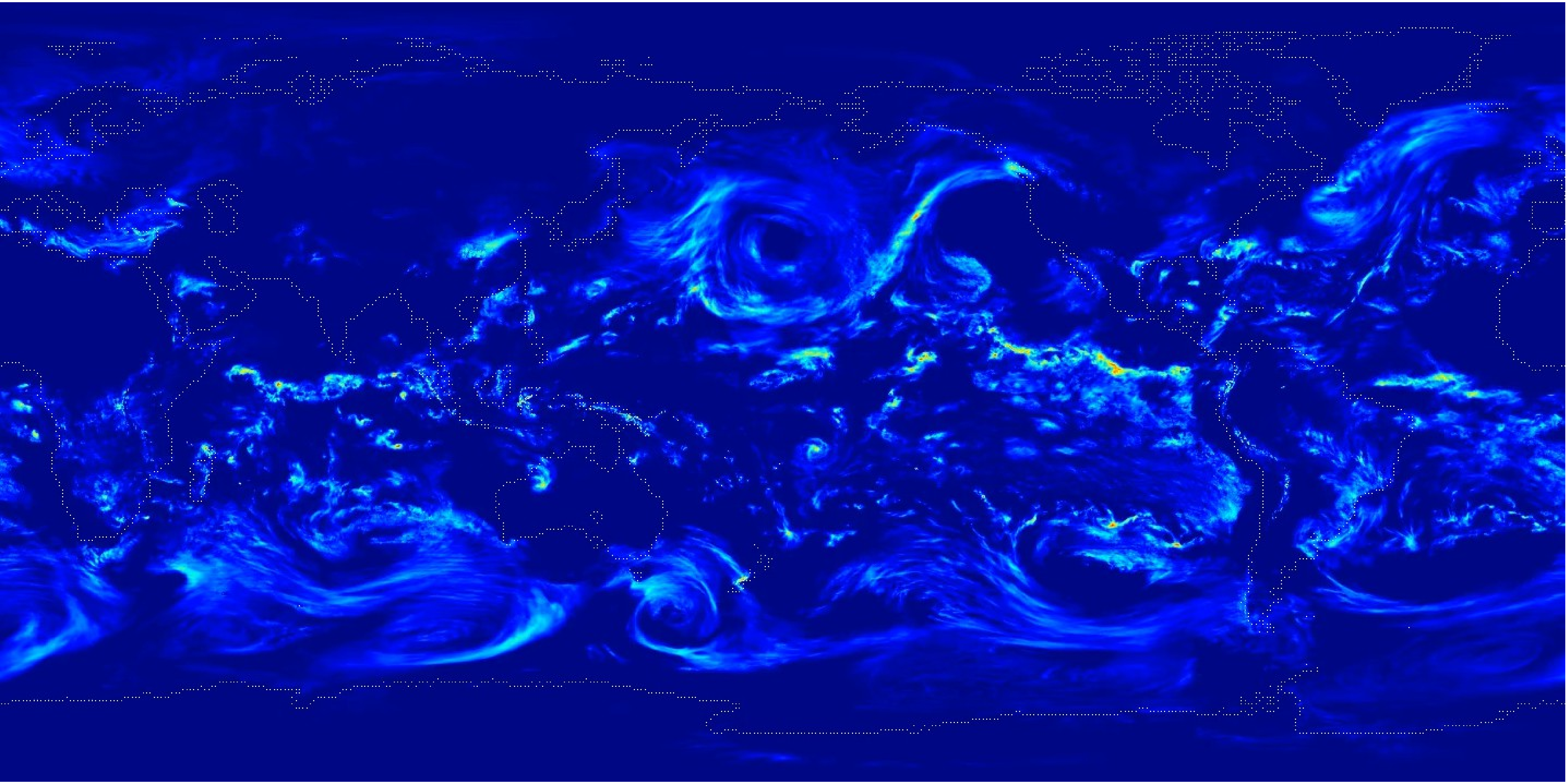






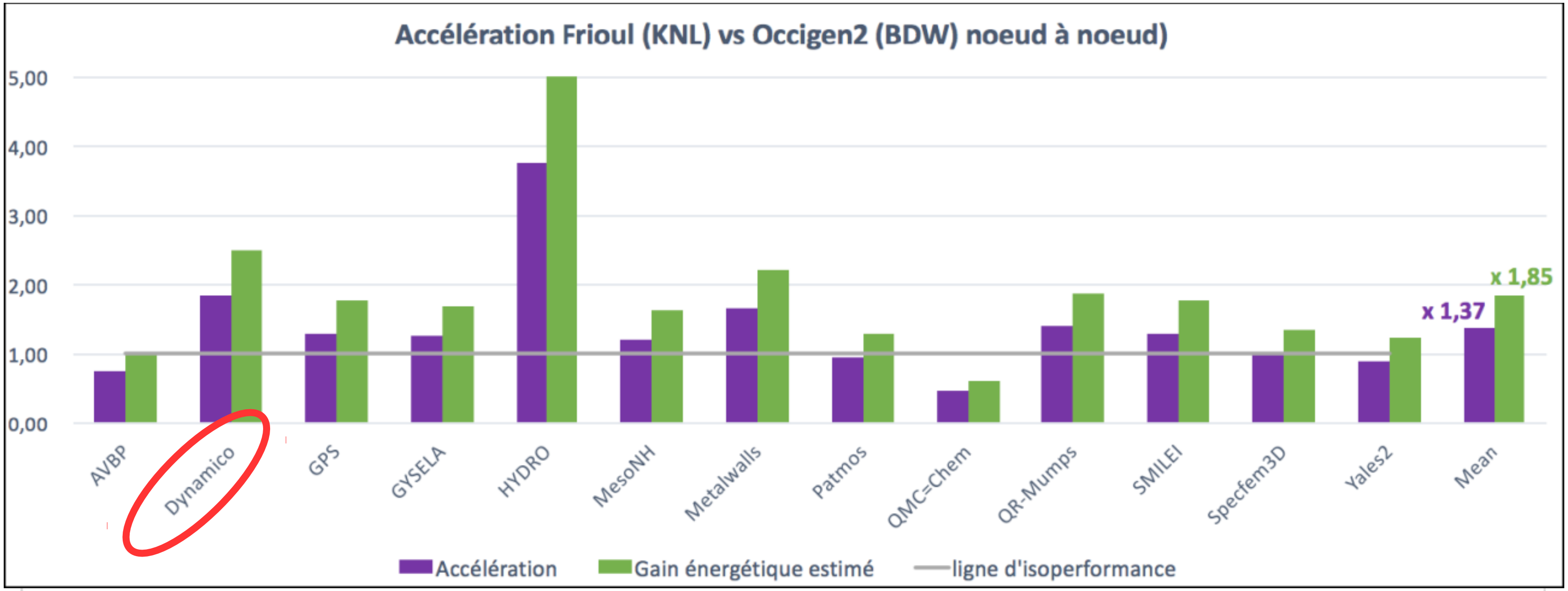
DYNAMICO-LMDZ, 25km

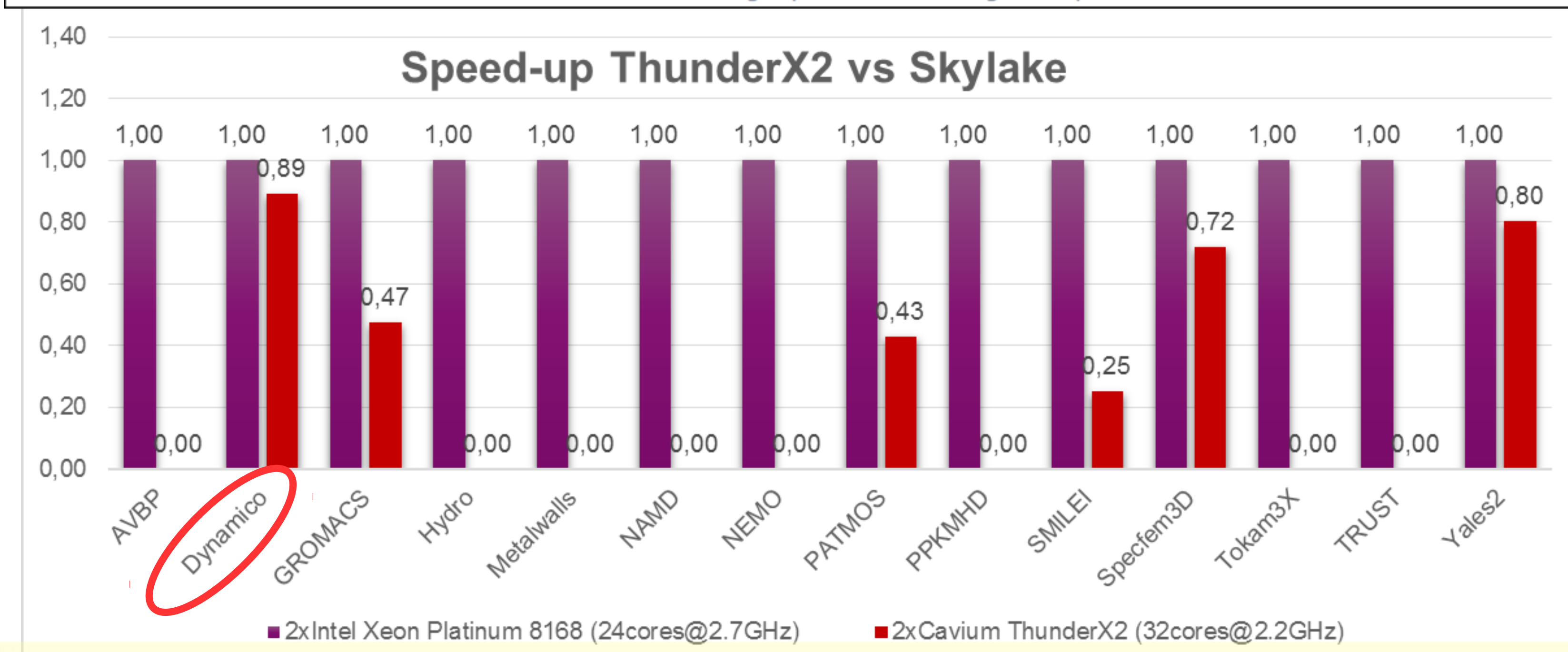
(1 024 000 atmospheric columns x 80 vertical levels)



Liquid water

DYNAMICO on fancy CPUs : KNL and ARM

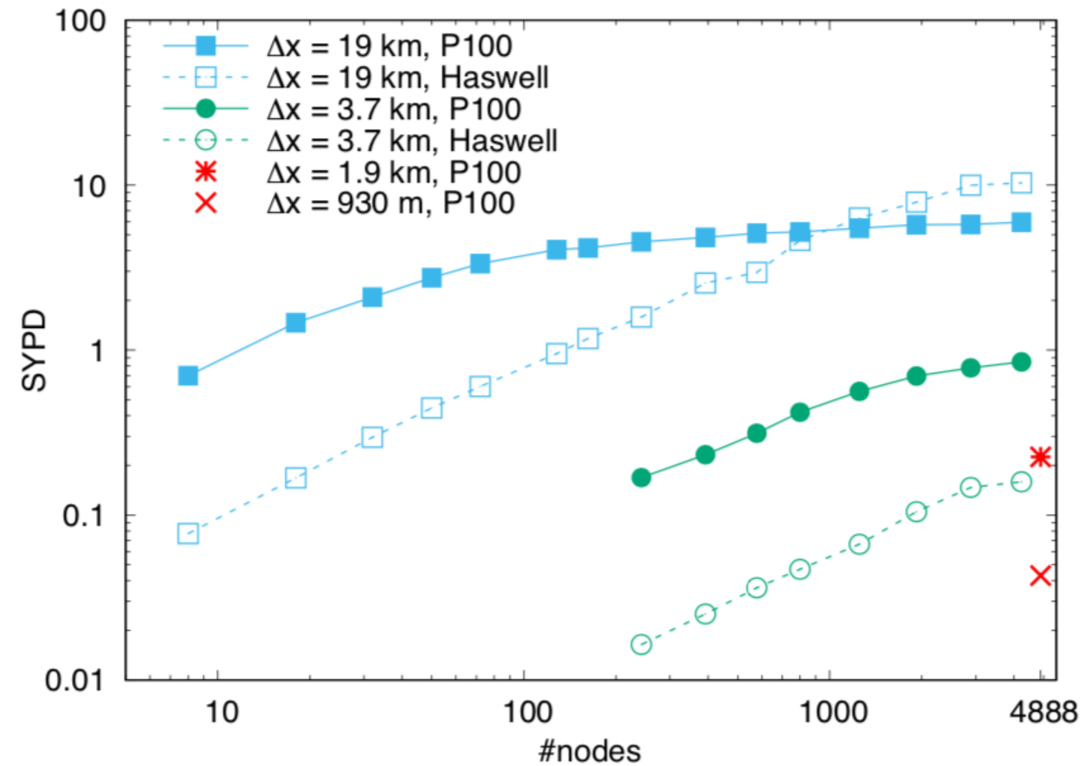




DYNAMICO on GPUs : Nvidia V100

# columns	16 000	64 000	256 000	1 024 000
40 CPUs	68 ms	282 ms	1193 ms	5102 ms
1 GPU	26,6 ms (x 2.5)	77 ms (x3.6)	299 ms (x 4.0)	Not enough memory
2 GPUs	14,7 ms (x 4.6)	40 ms (x 7)	152 ms (x 7.9)	649 ms (x 7.9)
4 GPUs	11,5 ms (x 5.9)	24 ms (x 11.8)	75 ms (x 15.9)	304 ms (x 16.8)

```
!$acc parallel loop collapse(2)
DO l = ll_begin, ll_end
  DO ij = ij_begin_ext,ij_end_ext
    uu_right = 0.5*(rhodz(ij,l)+rhodz(ij+t_right,l))*u(ij+u_right,l)
    uu_right = uu_right*le_de(ij+u_right)
    hflux(ij+u_right,l) = uu_right
    uu_lup = 0.5*(rhodz(ij,l)+rhodz(ij+t_lup,l))*u(ij+u_lup,l)
    uu_lup = uu_lup *le_de(ij+u_lup)
    hflux(ij+u_lup,l) = uu_lup
    uu_ldown = 0.5*(rhodz(ij,l)+rhodz(ij+t_ldown,l))*u(ij+u_ldown,l)
    uu_ldown = uu_ldown*le_de(ij+u_ldown)
    hflux(ij+u_ldown,l) = uu_ldown
  END DO
END DO
```



Fuhrer et al. 2017 https://doi.org/10.5194/gmd-2017-230

Prospects for higher-resolution *climate* modelling

- CMIP requires a throughput of x10000 (30SYPD)
- Some climate modelling still doable with x1000 (3SYPD)
- Ability to attain x1000 depends on maximum stable time step (numerics) and walltime needed to perform one time step (implementation)
- Assuming a large enough machine, reducing walltime is a **strong scaling problem**
- For DYNAMICO, dt (in sec) is about 2.5*dx (in km)

  => ms/time step required to reach 3SYPD :

  - 25km 60 ms per full time step
  - 8km 20 ms per full time step
  - 1km 2.5 ms per full time step

CMIP6 physics (79 vertical levels) cost 2-3 ms per column per call
(24 SYPD with 96 calls per day, 36 columns per core)

If physics are called every 5 dynamics time steps :

100 columns/core	=> 20+(100*2/5)=60 ms	=>	3 SYPD at 25 km with 10 000 cores
30 columns/core	=> 10+(30*2/5)=22 ms	=>	3 SYPD at 8 km with 300 000 cores ?
10 columns/core	=> 3+(10*2/5)=7 ms	=>	1 SYPD at 1 km with 50 million cores ??

Ongoing (or rather unfinished ?) work not covered in this talk :

- Coupling with ocean (NEMO) => IPSL-CM
- Fully unstructured re-implementation
  => variable resolution (à la MPAS)
- Limited-area
- Non-hydrostatic dynamics
- Deep-atmosphere / non-traditional dynamics
- Fancy equation of state (Venus)

Summary

Hamiltonian and variational structures

- help consider different equation sets in a unified way and clarify basic concepts : kinematics vs dynamics, velocity vs momentum
- separate the problem of conserving energy from other numerical properties : accuracy, dispersion, …

DYNAMICO : now mature as part of IPSL-CM and for planets

Separation of concerns :

- currently addressed as a software design problem
- claim / feeling : also and maybe primarily a higher-level (physics, numerics) design goal
- work needed by physicists, mathematicians to split models into smaller pieces with minimal interaction (e.g. transport, thermodynamics, geometry…)

EXTRA

Hydrostatic primitive equations : continuous

shallow atmosphere

traditional approximation

$$\mathbf{v} = a^2\dot{\mathbf{x}} + \mathbf{R} \qquad \mathbf{R} = a^2 \cos\phi\,\Omega\,\mathbf{e}_\lambda$$

potential energy

mass-weighted potential temperature

$$H[\mu,\Theta,\Phi,\mathbf{v}] = \int \left[\left(\Phi + e\left(\frac{a^2\partial_\eta\Phi}{g\mu},\frac{\Theta}{\mu}\right)\right) + \frac{1}{2}a^{-2}\left(\mathbf{v}-\mathbf{R}\right)^2\right]\mu\,\mathrm{d}^2\mathbf{x}\,\mathrm{d}\eta$$

Internal energy

specific volume

horizontal kinetic energy K_H

Bernoulli

Exner

Hydrostatic balance

mass flux

$$\frac{\mathrm{d}H}{\mathrm{d}t} = \int \left[\left(\Phi + K_H\right)\partial_t\mu + \pi\partial_t\Theta + \left(\mu + \frac{a^2}{g}\partial_\eta p\right)\partial_t\Phi + \mu\dot{\mathbf{x}}\cdot\partial_t\mathbf{v}\right]\mathrm{d}^2\mathbf{x}\,\mathrm{d}\eta$$

$$\frac{\delta H}{\delta\mu} \qquad \frac{\delta H}{\delta\Theta} \qquad \frac{\delta H}{\delta\Phi} \qquad \frac{\delta H}{\delta\mathbf{v}}$$

All terms in the equations of motion can be expressed as functional derivatives of total energy

Hydrostatic primitive equations : discrete

(Lagrangian vertical coordinate – extra terms for vertical transport when mass-based)

mass-weighted potential temperature

TRiSK

shallow-atmosphere metric

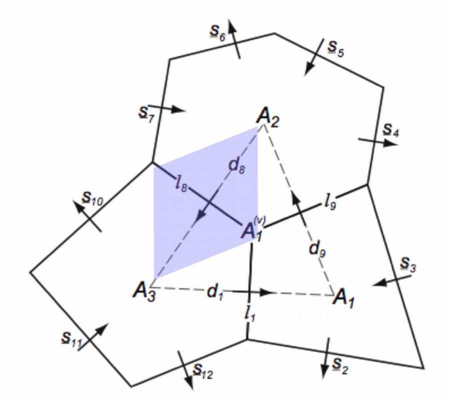
planetary velocity

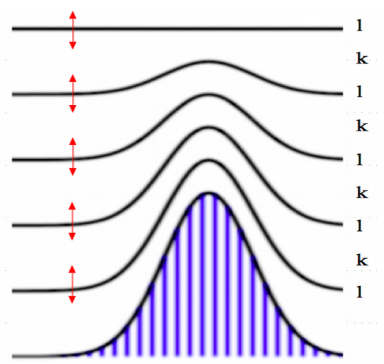
$$H = K(m_{ik},\, v_{ek}) + P(m_{ik},\, \Theta_{ik},\, \Phi_{il})$$

$$K = \sum_{ike} m_{ik} \frac{A_{ie}}{A_i} \left(\frac{v_{ek} - R_e}{a d_e} \right)^2$$

normal velocity

$$P_{HPE} = \sum_{ik} m_{ik} \left[\overline{\Phi_i}^k + e \left(\alpha_{ik} = \frac{A_i \delta_k (a^2 \Phi_i / g)}{m_{ik}}, \theta_{ik} = \frac{\Theta_{ik}}{m_{ik}} \right) \right]$$





mass flux

centered/upwind flux

$$\partial_t m_{ik} + \delta_i \frac{\partial H}{\partial v_{ek}} = 0 \qquad \partial_t \Theta_{ik} + \delta_i \left(\theta^*_{ek} \frac{\partial H}{\partial v_{ek}} \right) = 0$$

$$\partial_t v_{ek} + \left(\frac{\delta_v v_k}{\overline{m_k}^v} \frac{\partial H}{\partial v_{ek}} \right)^\perp + \delta_e \frac{\partial H}{\partial m_{ik}} + \theta^*_{ek} \delta_e \frac{\partial H}{\partial \Theta_{ik}} = 0$$

SW pot. Vort. Bernoulli function Exner function

Discrete energy budget : Lagrangian vertical coordinate

mass flux

centered/upwind

$$\partial_t m_{ik} + \delta_i \frac{\partial H}{\partial v_{ek}} = 0 \qquad \partial_t \Theta_{ik} + \delta_i \left(\theta^*_{ek} \frac{\partial H}{\partial v_{ek}} \right) = 0$$

$$\partial_t v_{ek} + \left(\frac{\delta_v v_k}{\overline{m_k}^v} \frac{\partial H}{\partial v_{ek}} \right)^\perp + \delta_e \frac{\partial H}{\partial m_{ik}} + \theta^*_{ek} \delta_e \frac{\partial H}{\partial \Theta_{ik}} = 0$$

SW pot. Vort.

Bernoulli function

Exner function

discrete div

discrete integration by parts

discrete grad

$$\frac{\mathrm{d}H}{\mathrm{d}t} = \sum \frac{\partial H}{\partial m_{ik}} \partial_t m_{ik} + \frac{\partial H}{\partial \Theta_{ik}} \partial_t \Theta_{ik} + \frac{\partial H}{\partial v_{ek}} \partial_t v_{ek} + \frac{\partial H}{\partial \Phi_{il}} \partial_t \Phi_{il} = 0$$

Hydrostatic balance

$$\Rightarrow p_{ik} \Rightarrow \alpha_{ik} \Rightarrow \Phi_{il}$$

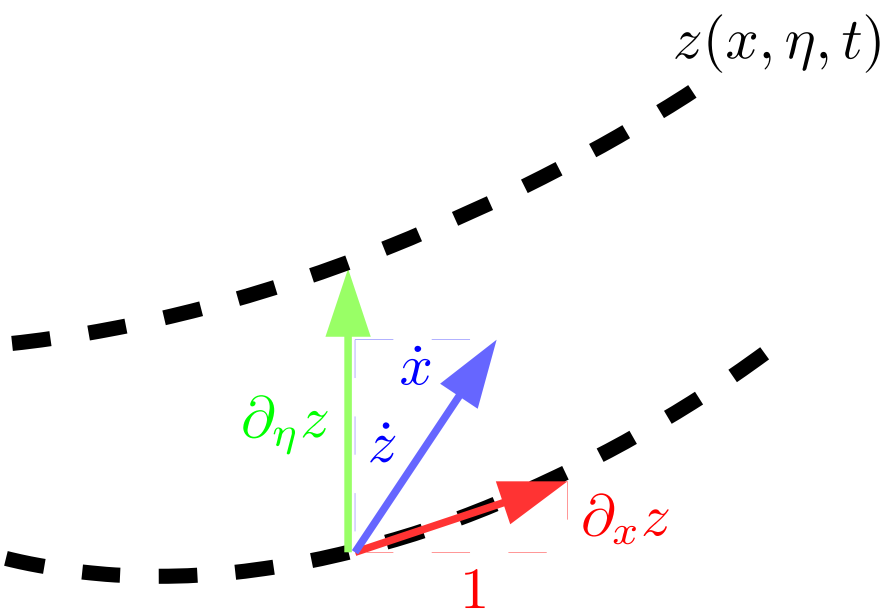
Non-Lagrangian vertical coordinate : also possible to cancel additional contributions from vertical transport (Tort et al., QJRMS 2015)

What is new in NH (fully compressible) compared to hydrostatic ?

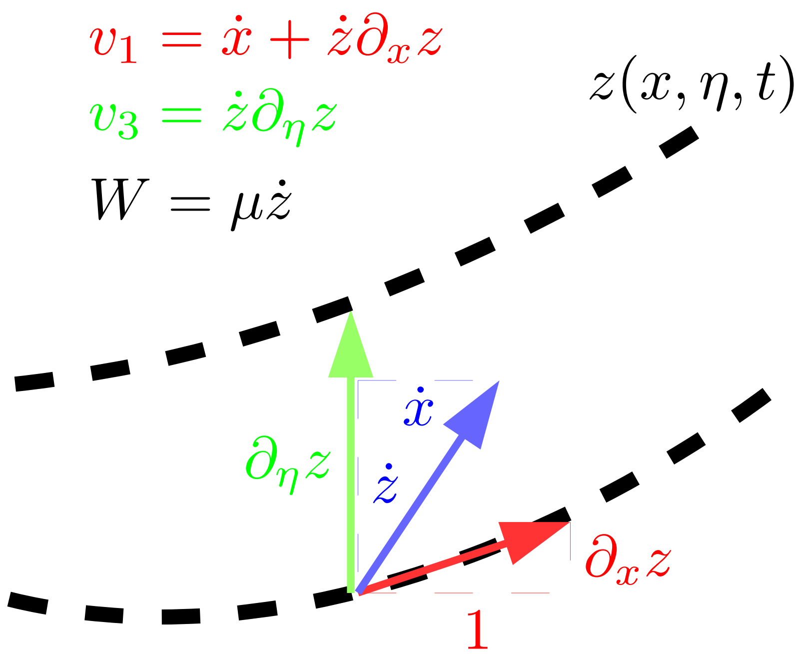
- More *energy* : vertical kinetic energy K_V not neglected any more
- More *prognostic* variables : geopotential, vertical momentum

Example : Cartesian (x,z) slice, non-rotating

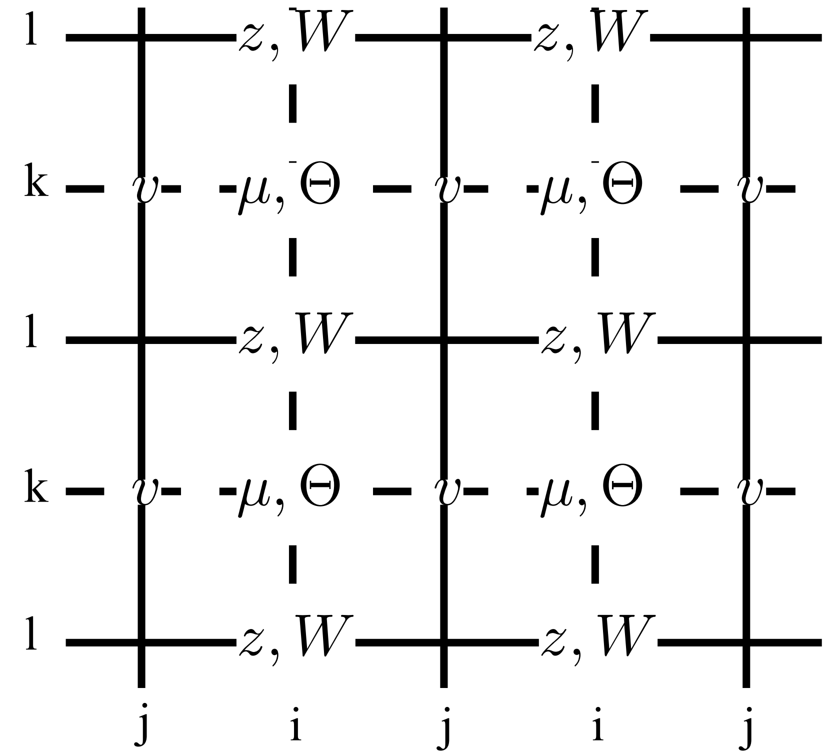
$$v_1 = \dot{x} + \dot{z}\partial_x z$$
$$v_3 = \dot{z}\partial_\eta z$$
$$W = \mu\dot{z}$$



$$\mathcal{H} = \int \left[\frac{\left(v_1 - \frac{W}{\mu}\partial_x z\right)^2 + \left(\frac{W}{\mu}\right)^2}{2} + e\left(\frac{\partial_\eta z}{\mu}, \frac{\Theta}{\mu}\right) + gz \right] \mu \mathrm{d}x\mathrm{d}\eta$$



C-grid Lorentz staggering



$$K_H = \frac{1}{2}\sum\left(v^2_{jk}\overline{\mu_i}^j - 2\overline{W_l}^j\overline{v_j}^l D_j z_l + \frac{\overline{(D_j z)^2}^i W^2_{il}}{\overline{\mu_i}^l}\right) \simeq \frac{1}{2}\int\left(v - \partial_x z\frac{W}{\mu}\right)^2 \mu \mathrm{d}x\mathrm{d}\eta$$

$$\frac{\partial K_H}{\partial v_{jk}} = \overline{\mu_i}^j v_{jk} - \overline{D_j z_l \overline{W_l}^j}$$

« horizontal » mass flux

$$\frac{\partial K_H}{\partial z_{il}} = D_i\left(\overline{W_l}^j\overline{v_j}^l - D_j z_l \frac{\overline{W^2_{il}}^j}{\overline{\mu_i}^l}\right)$$

« horizontal » advection of vertical momentum

slow fast

$$H = K_H[\mu, \Phi, \mathbf{v}, W] + K_V[\mu, \Phi, W] + P[\mu, \Theta, \Phi]$$

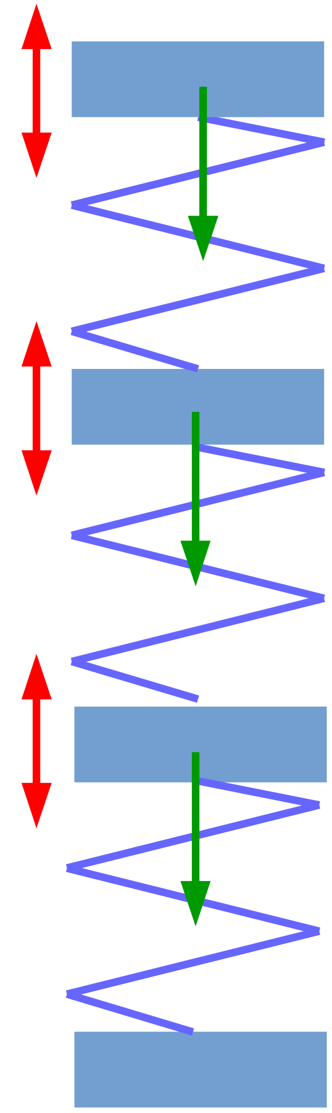
$$\partial_t \mu + \partial_\eta \left(\mu \dot{\eta} \right) + \partial_{\mathbf{x}} \cdot \frac{\delta H}{\delta \mathbf{v}} = 0$$

$$\partial_t \Theta + \partial_\eta \left(\theta \mu \dot{\eta} \right) + \partial_{\mathbf{x}} \cdot \theta \frac{\delta H}{\delta \mathbf{v}} = 0$$

$$\partial_t \mathbf{v} + \dot{\eta} \partial_\eta \mathbf{v} + v_3 \partial_{\mathbf{x}} \dot{\eta} + \frac{\partial_{\mathbf{x}} \times \mathbf{v}}{\mu} \times \frac{\delta H}{\delta \mathbf{v}} + \partial_{\mathbf{x}} \frac{\delta K_H}{\delta \mu}$$
$$+ \partial_{\mathbf{x}} \frac{\delta (K_V + P)}{\delta \mu} + \theta \partial_{\mathbf{x}} \frac{\delta P}{\delta \Theta} = 0$$

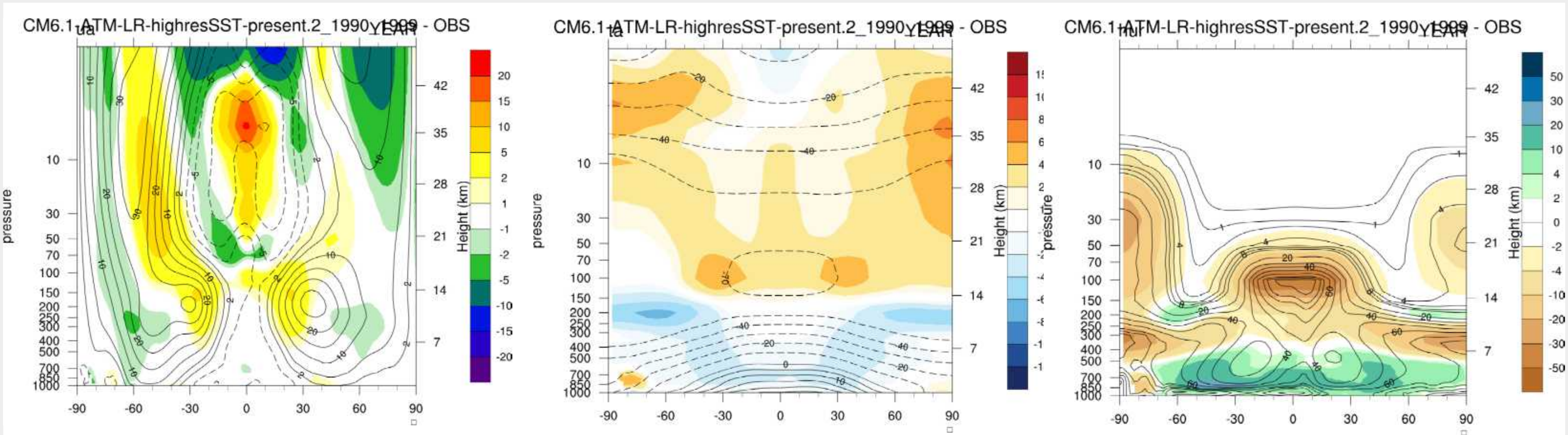
$$\partial_t \Phi + \dot{\eta} \partial_\eta \Phi - \frac{\delta K_H}{\delta W} - \frac{\delta K_V}{\delta W} = 0$$

$$\partial_t W + \partial_\eta \left(\dot{\eta} W \right) + \frac{\delta K_H}{\delta \Phi} + \frac{\delta (K_V + P)}{\delta \Phi} = 0$$



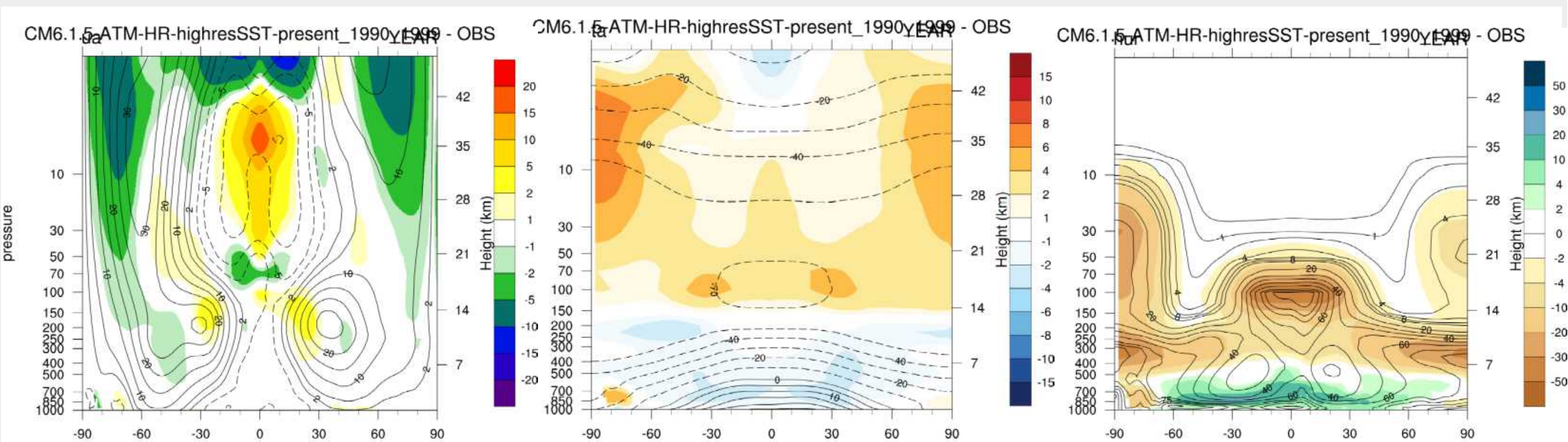
- The implicit problem only couples vertical position and vertical momentum
- eliminate W and obtain a scalar tridiagonal implicit problem for Φ

Zonal wind *Temperature* *Relative humidity*



IPSL-CM6.1-LR

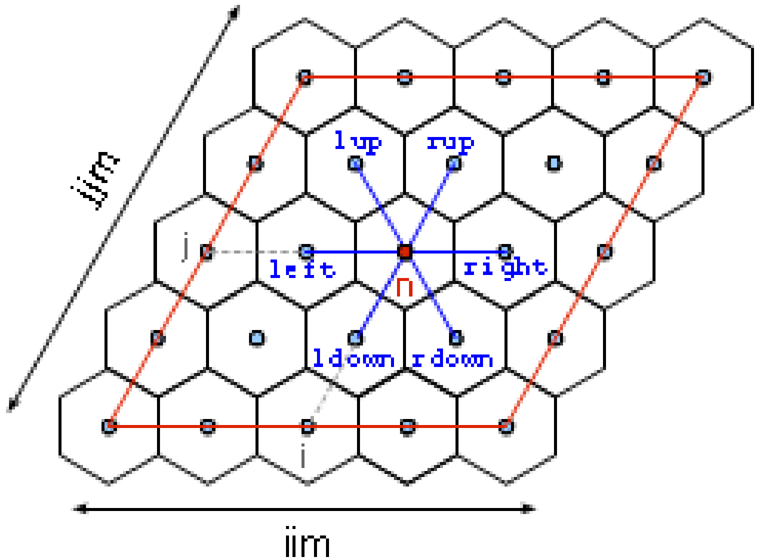
Contours : OBS (ERA) *Colors : bias = model-OBS*



IPSL-CM6.1-HR

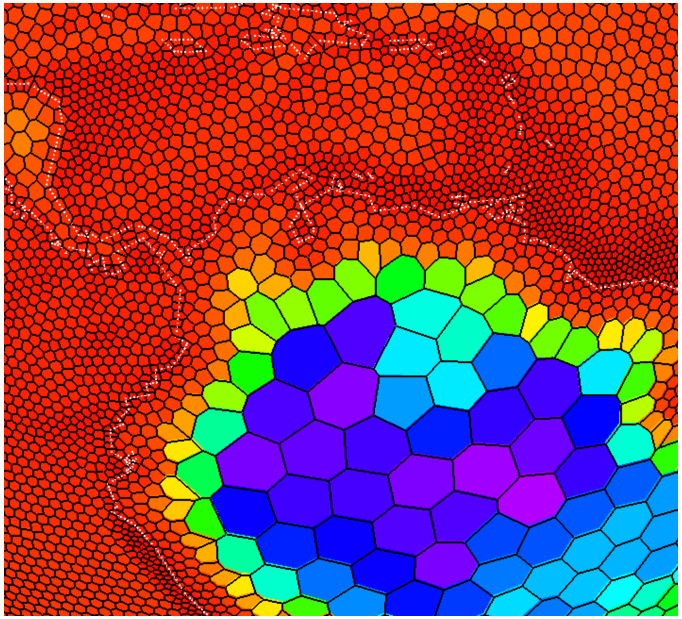
=> some reduction of mean biases at higher resolution

Unstructured-mesh capability



Structured mesh

- Quasi-uniform resolution
- Zoom possible but limited
- Regular data access / compute pattern



Unstructured mesh

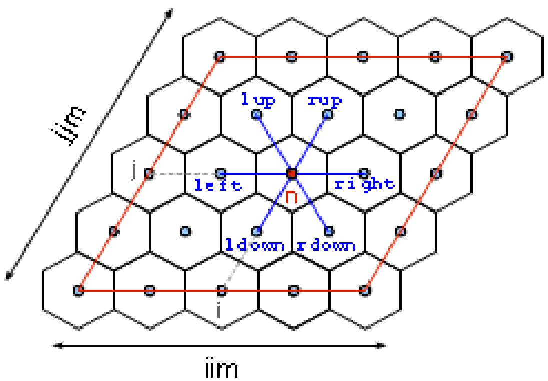
- Variable resolution
- Very flexible zoom capability
- Irregular data access / compute pattern
- Needs scale-aware physics

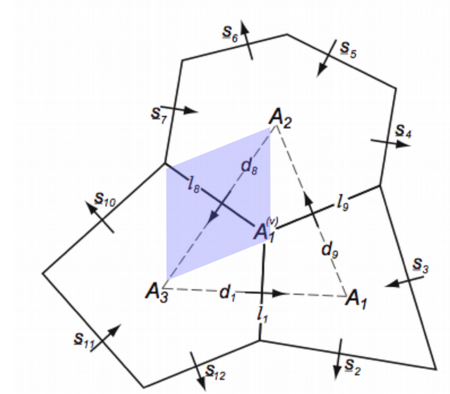
Status :

- *Prototype available in the « devel» branch*
- *MPI/OpenMP parallelism*
- *More work ahead (performance, transport scheme, ...)*

DySL : Dynamico-specific « language »

```
KERNEL('div')
  FORALL_CELLS_EXT()
    ON_PRIMAL
      div_ij=0.
      FORALL_EDGES
        div_ij = div_ij + SIGN*LE_DE*u(EDGE)
      END_BLOCK
      divu(CELL) = div_ij / AI
    END_BLOCK
  END_BLOCK
END_BLOCK
```






```
   DO l = ll_begin, ll_end
!DIR$ SIMD
      DO ij=ij_begin_ext, ij_end_ext
         div_ij=0.
         div_ij = div_ij + ne_rup*le_de(ij+u_rup)*u(ij+u_rup,l)
         div_ij = div_ij + ne_lup*le_de(ij+u_lup)*u(ij+u_lup,l)
         div_ij = div_ij + ne_left*le_de(ij+u_left)*u(ij+u_left,l)
         div_ij = div_ij + ne_ldown*le_de(ij+u_ldown)*u(ij+u_ldown,l)
         div_ij = div_ij + ne_rdown*le_de(ij+u_rdown)*u(ij+u_rdown,l)
         div_ij = div_ij + ne_right*le_de(ij+u_right)*u(ij+u_right,l)
         divu(ij,l) = div_ij / Ai(ij)
      END DO
   END DO
```

```
   DO ij = 1, primal_num
      DO l = 1,llm
         div_ij=0.
         DO iedge = 1, primal_deg(ij)
            edge = primal_edge(iedge,ij)
            div_ij = div_ij + primal_ne(iedge,ij)*le_de(edge)*u(l,edge)
         END DO
         divu(l,ij) = div_ij / Ai(ij)
      END DO
   END DO
```

- Code generator purely text-based (no grammar / parser / syntax tree)
- Macro substitution (cpp) and inlining (jinja2)
- Generates *human-readable* code chunks to be #included in hand-written Fortran code
- Lightweight, low exit cost