INSTITUT PIERRE SIMON LAPLACE
DES SCIENCES DE L'ENVIRONNEMENT

# Modélisation numérique du climat sur architectures haute performance : quelques succès et défis à l'IPSL


Thomas Dubos
*LMD/IPSL, École Polytechnique*

*Marie-Alice Foujols*
*IPSL, CNRS*

*Yann Meurdesoif*
*LSCE/IPSL, CEA*



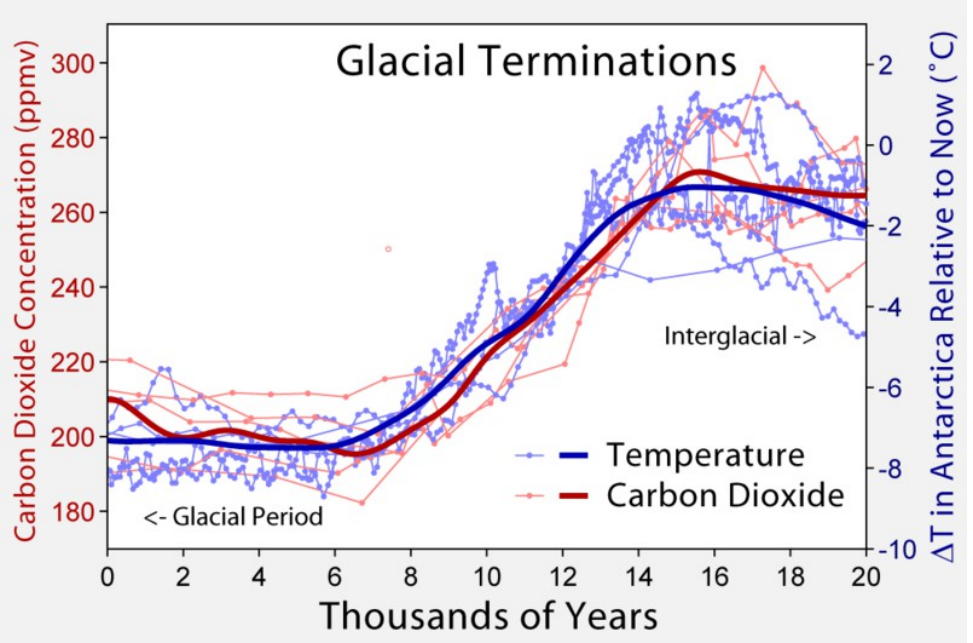
*Séminaire Aristote, 22 mai 2019, École Polytechnique*

# Modélisation numérique du climat sur architectures haute performance : quelques succès et défis à l'IPSL

- Contexte et quelques chiffres
- Du vectoriel au massivement parallèle
- Vers l'exascale ? des défis techniques, mais pas seulement

Le temps : une réalisation
- Simuler
- Prévoir


Le climat : une statistique et son évolution
- Modéliser
- Comprendre/expliquer
- Anticiper

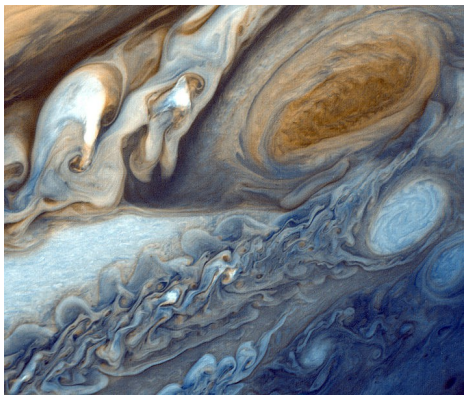
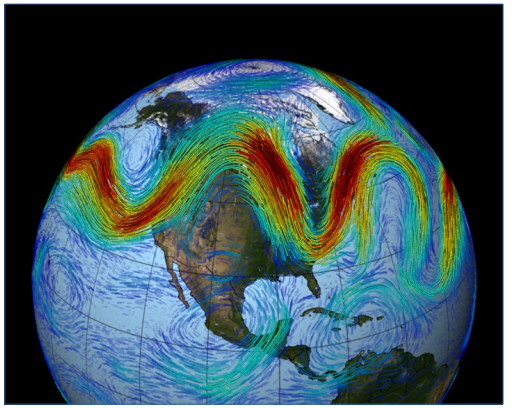


# *The atmosphere : a gravity-dominated, compressible flow*

## Characteristic scales

- *Velocity* : Sound c ~ 340m/s — Wind U ~ 30m/s
- *Time* : Buoyancy oscillations N ~ g/c ~$10^{-2}$ $s^{-1}$ — Coriolis f ~ $10^{-4}$ $s^{-1}$
- *Length* : **Scale height $H=c^2/g$=10km** — **Rossby radius : R=c/f ~ 1000 km**

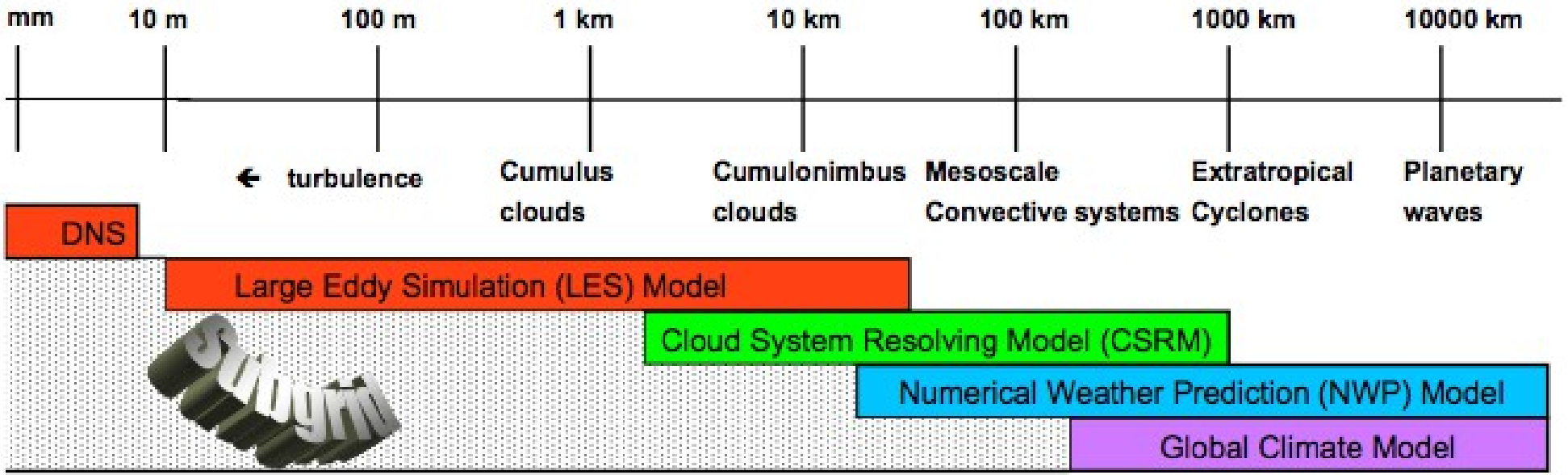


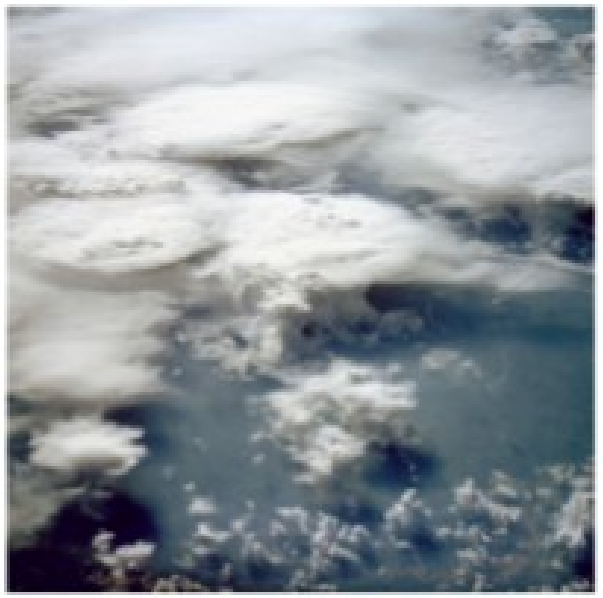
small-scale | scale height | mesoscale | synoptic | planetary

1 km | **10 km** | 100 km | **1000 km** | 10000 km

# Les nuages : différentes échelles, différents processus



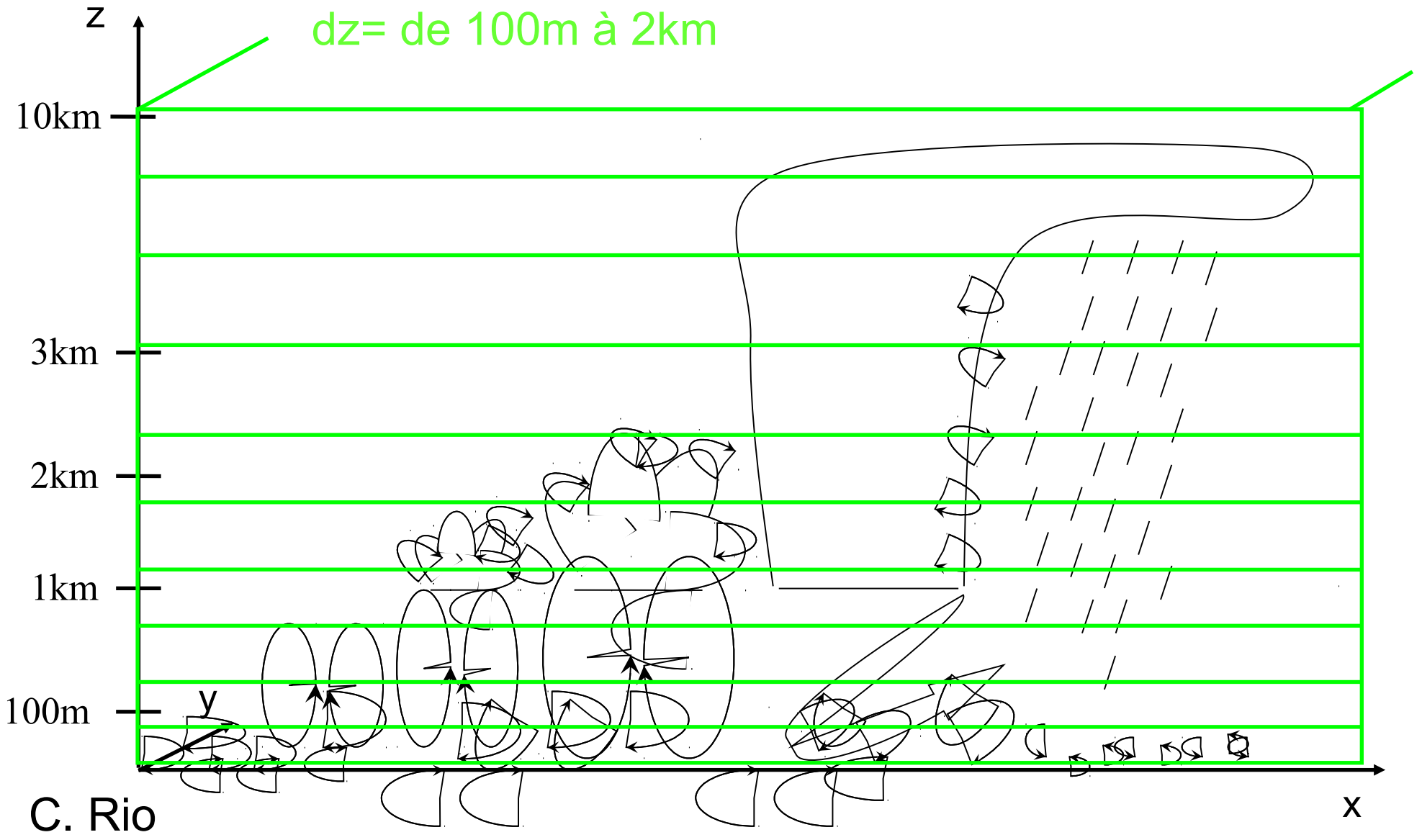

9/2/2008



P. Siebesma

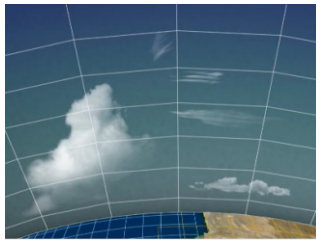
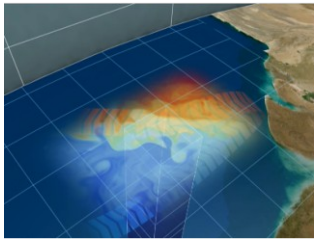
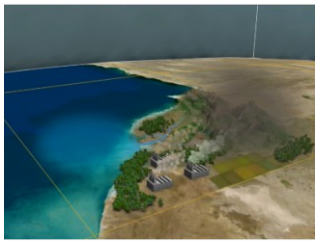
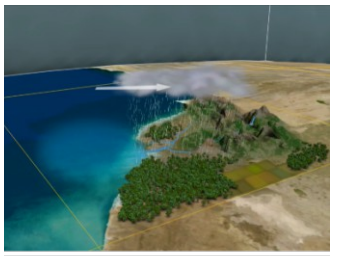
# La modélisation des nuages convectifs

Modèle de circulation générale (GCM):
dx=dy=de 50 à 400km
dz= de 100m à 2km




C. Rio

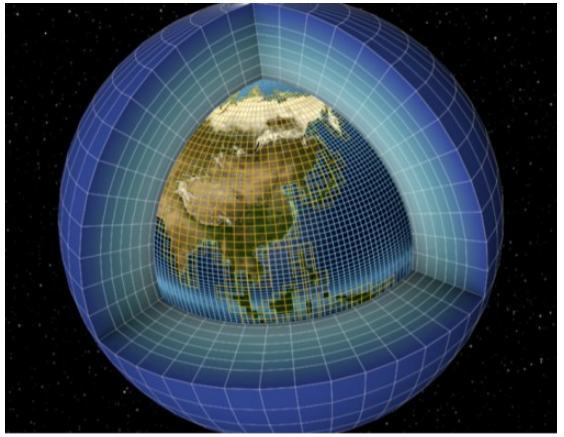
# Earth System Modelling at IPSL

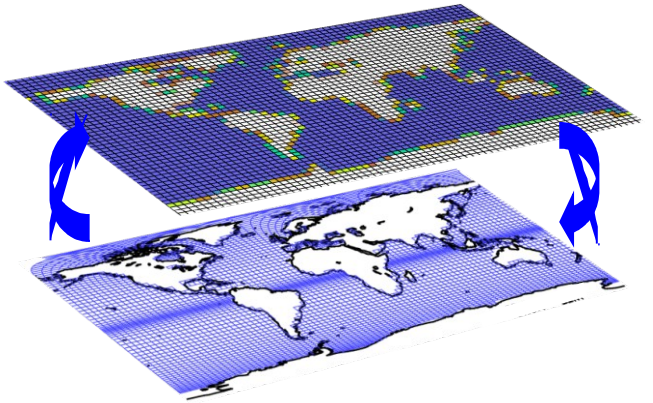
**INCA / REPROBUS**
(chimie atmosphérique)
(aérosol)

**LMDZ**
(atmosphère)

**ORCHIDEE**
(surfaces continentales)
(végétation)



**OASIS**
(coupleur)

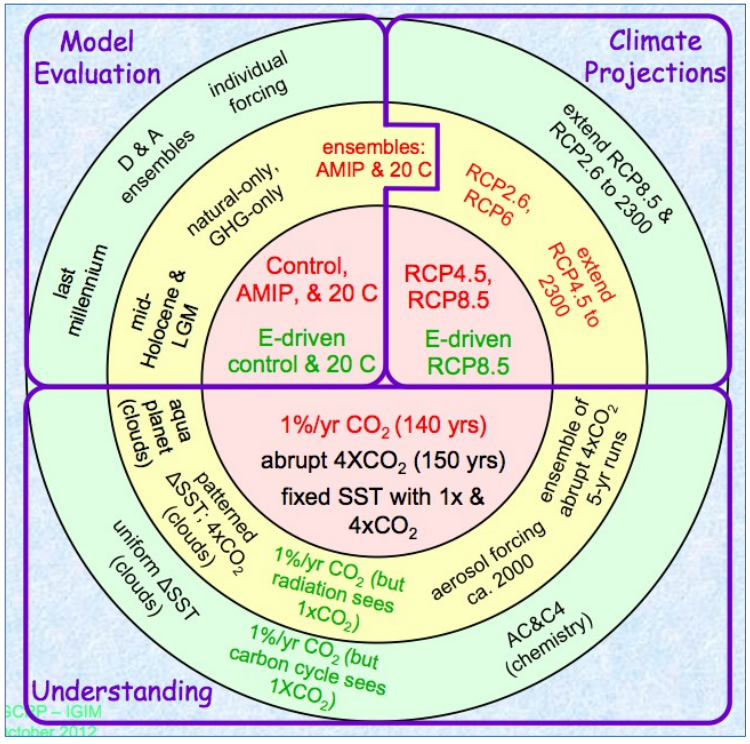


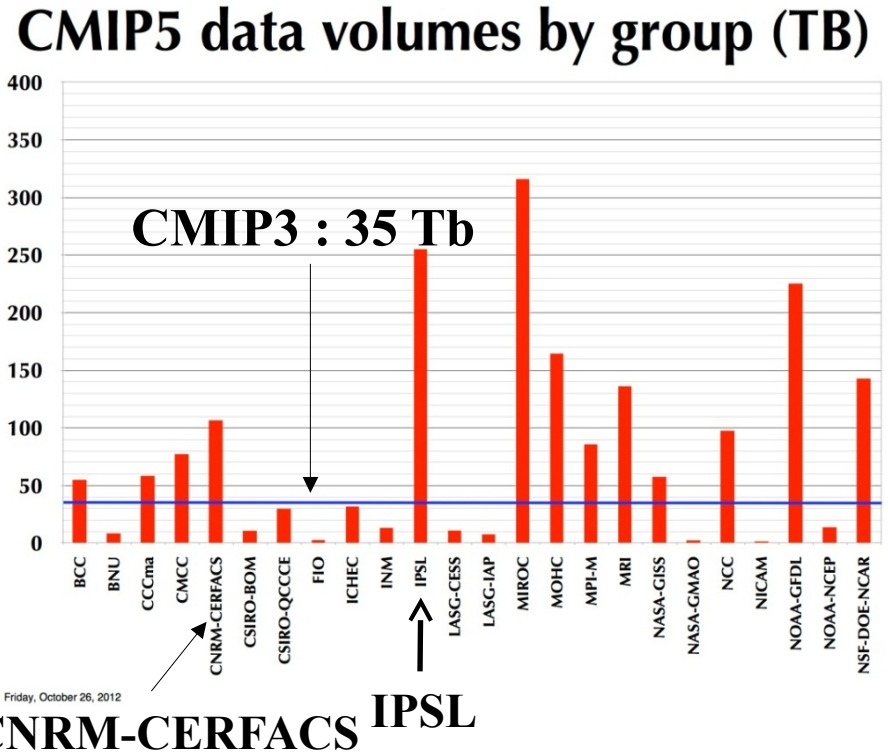
**OPA**
(océan)

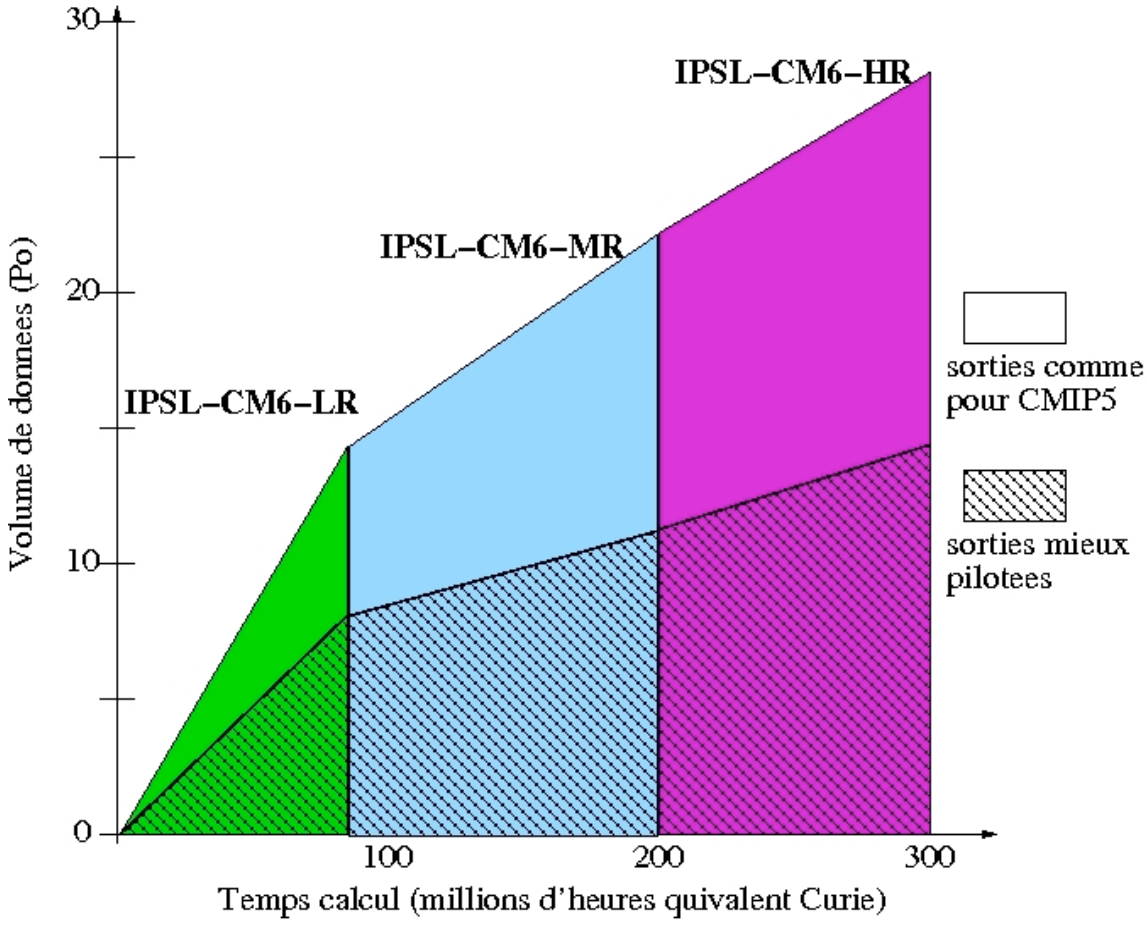
**LIM**
(glace de mer)

**PISCES**
(biogéochimie marine)

**NEMO**


Model Evaluation
Climate Projections
Understanding
individual forcing
D & A ensembles
ensembles: AMIP & 20 C
natural-only, GHG-only
RCP2.6, RCP6
extend RCP8.5 & RCP2.6 to 2300
last millennium
mid-Holocene & LGM
Control, AMIP, & 20 C
E-driven control & 20 C
RCP4.5, RCP8.5
E-driven RCP8.5
extend RCP4.5 to 2300
aqua planet (clouds)
1%/yr CO2 (140 yrs)
abrupt 4XCO2 (150 yrs)
fixed SST with 1x & 4xCO2
ensemble of abrupt 4xCO2 5-yr runs
patterned ΔSST, 4xCO2 (clouds)
uniform ΔSST (clouds)
1%/yr CO2 (but radiation sees 1xCO2)
aerosol forcing ca. 2000
1%/yr CO2 (but carbon cycle sees 1XCO2)
AC&C4 (chemistry)


CMIP5 data volumes by group (TB)
CMIP3 : 35 Tb
400
350
300
250
200
150
100
50
0
BCC
BNU
CCCma
CMCC
CNRM-CERFACS
CSIRO-BOM
CSIRO-QCCCE
FIO
ICHEC
INM
IPSL
LASG-CESS
LASG-IAP
MIROC
MOHC
MPI-M
MRI
NASA-GISS
NASA-GMAO
NCC
NICAM
NOAA-GFDL
NOAA-NCEP
NSF-DOE-NCAR
Friday, October 26, 2012
CNRM-CERFACS
IPSL

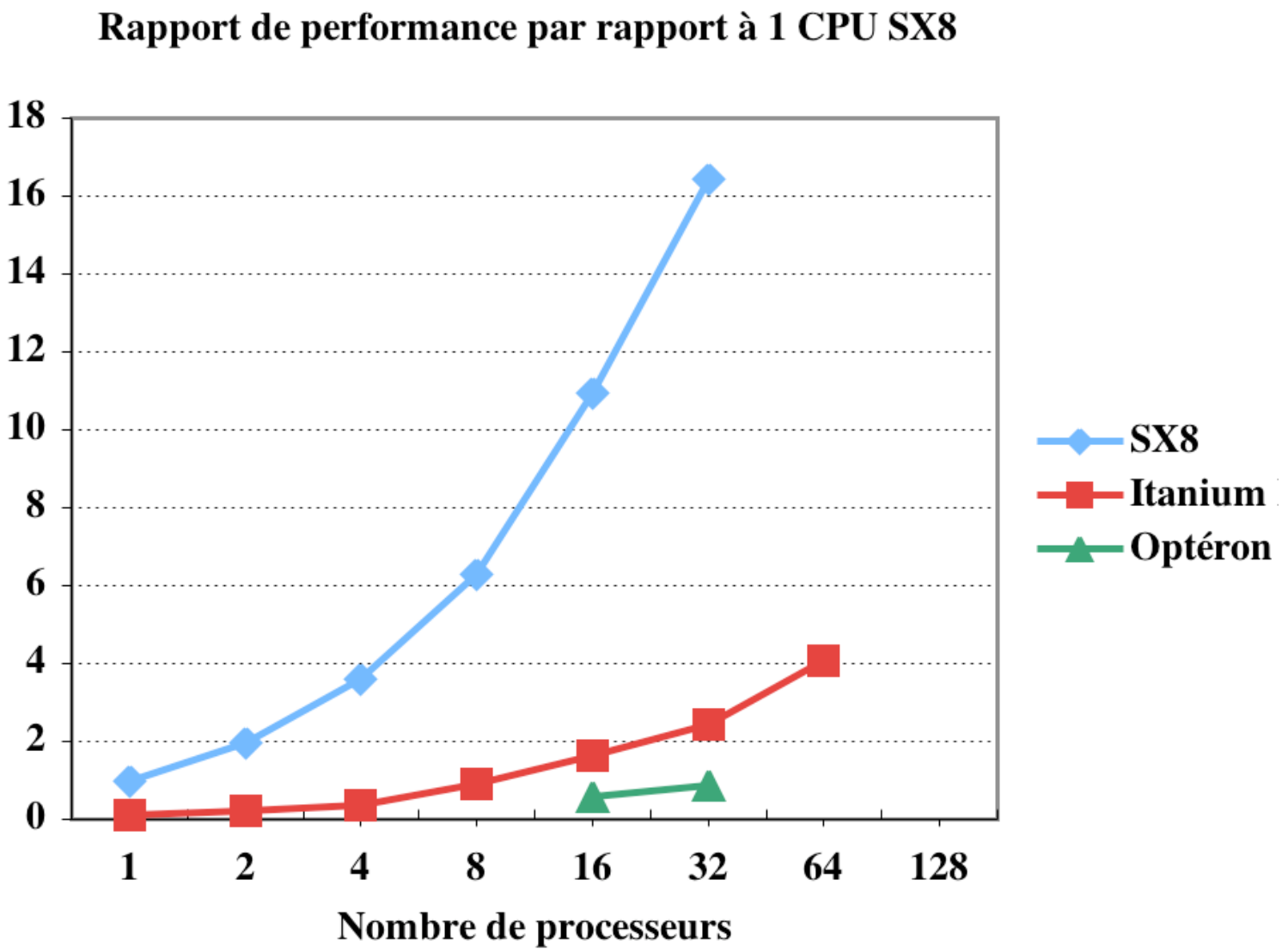

30
20
10
0
Volume de donnees (Po)
IPSL–CM6–HR
IPSL–CM6–MR
IPSL–CM6–LR
100
200
300
Temps calcul (millions d'heures quivalent Curie)
sorties comme pour CMIP5
sorties mieux pilotees

# 10-year perspective for CMIP

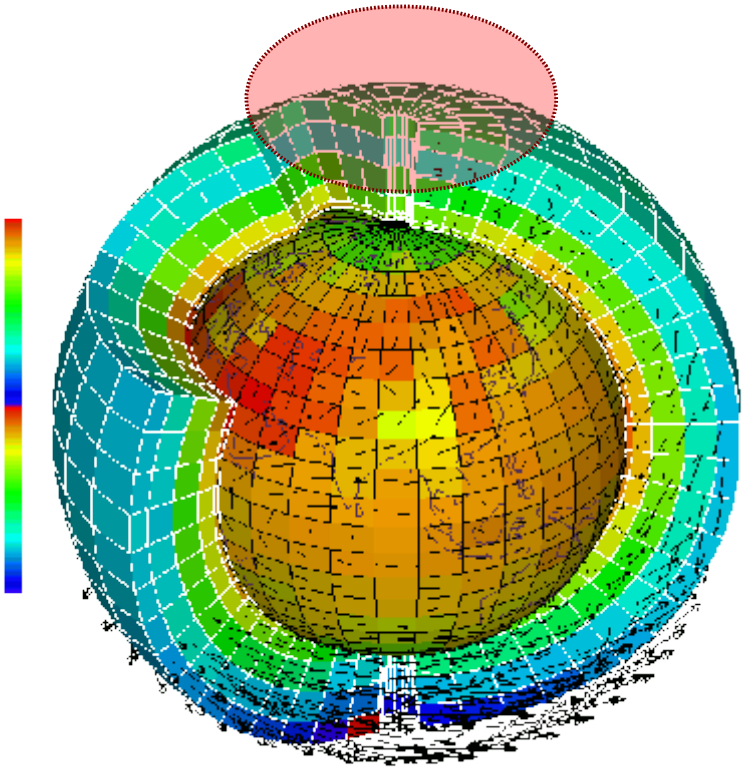
| | CMIP5 | CMIP6 | CMIP7 |
|---|---|---|---|
| Year | 2012 | 2017 | 2022 |
| Power factor | 1 | 30 | 1000 |
| Npp | 200 | 357 | 647 |
| Resolution [km] | 100 | 56 | 31 |
| Number of mesh points [millions] | 3,2 | 18,1 | 108,4 |
| Ensemble size | 120 | 214 | 388 |
| Number of variables | 800 | 1068 | 1439 |
| Interval of 3-dimensional output (hours) | 6 | 4 | 3 |
| Years simulated | 90000 | 120170 | 161898 |
| Storage density | 0,00002 | 0,00002 | 0,00002 |
| **Distributed Archive Size (Pb)** | **3,19** | **86,05** | **2260,20** |

- Contexte et quelques chiffres
- Du vectoriel au massivement parallèle
- Vers l’exascale ? des défis techniques, mais pas seulement

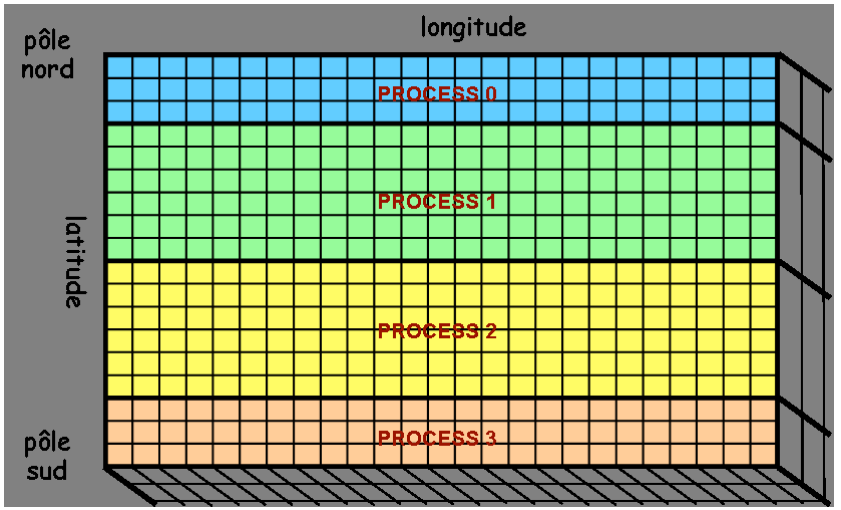
2007 : les codes du modèle IPSL-CM sont vectorisés et parallèles (MPI)

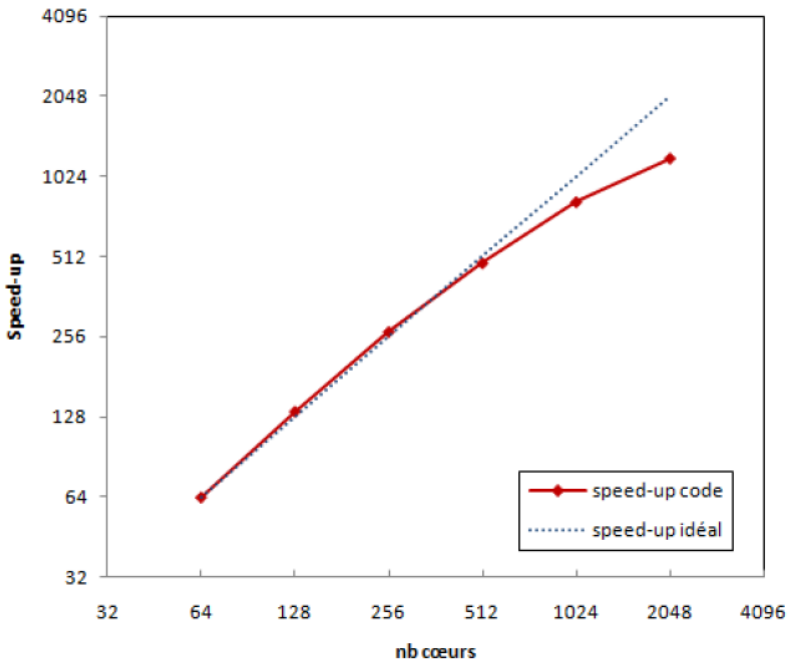


*The pole problem : FFT filters around the pole for stability => global dependency (Williamson, 2007)*



- LMD-Z lon-lat core



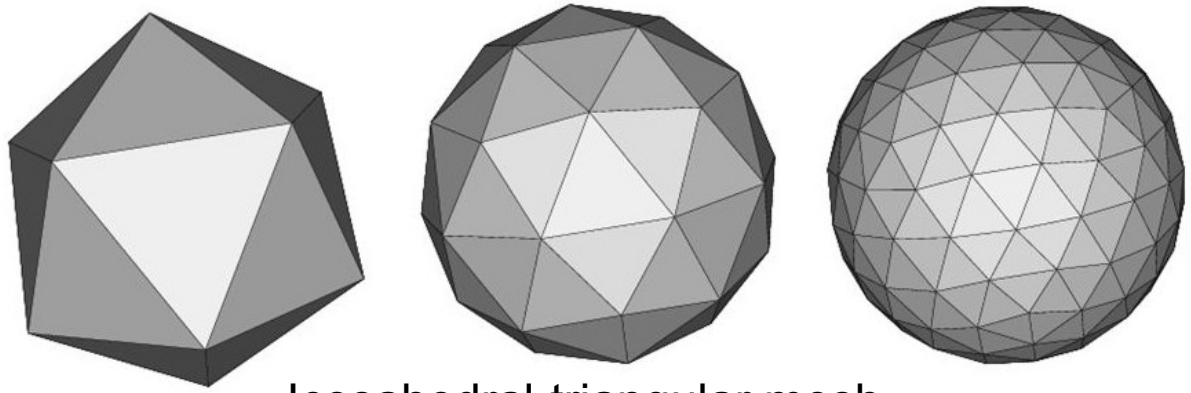




Y. Meurdesoif, 2010

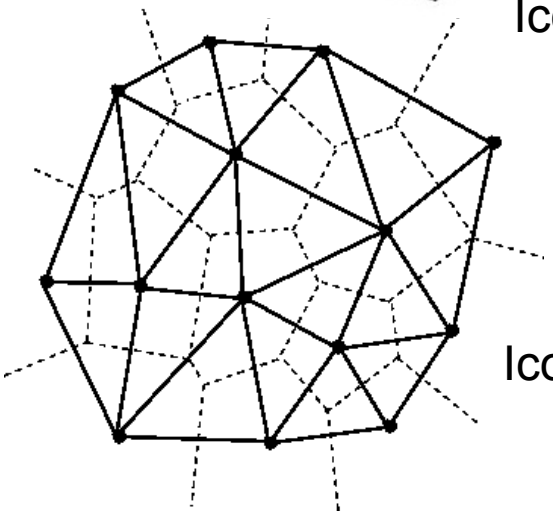
1/4 degré (25km) => max. 240 processus MPI

# Mesh partitioning for parallel computing

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

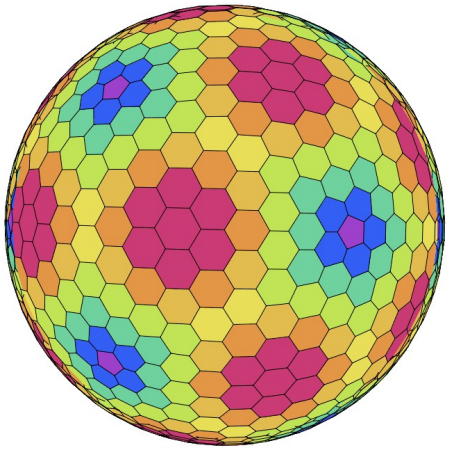


Icosahedral-triangular mesh
$10*nbp^2+2$ *vertices*

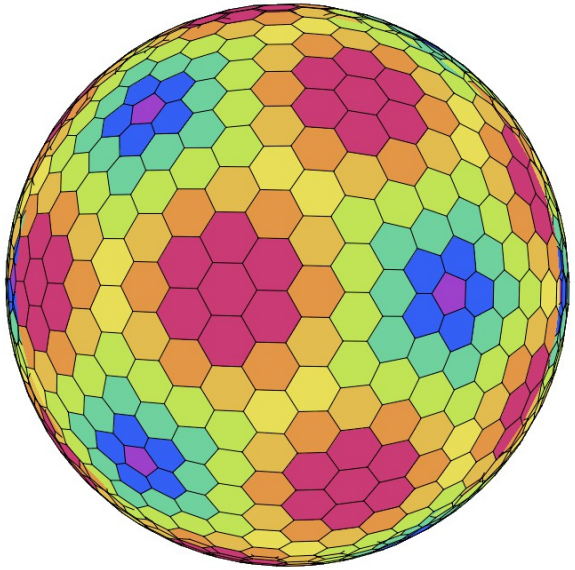


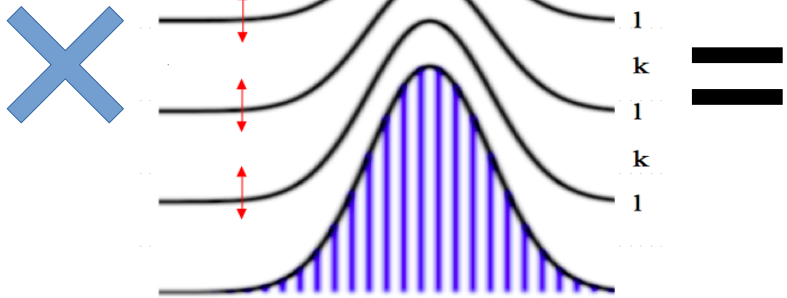
*Voronoi dual*

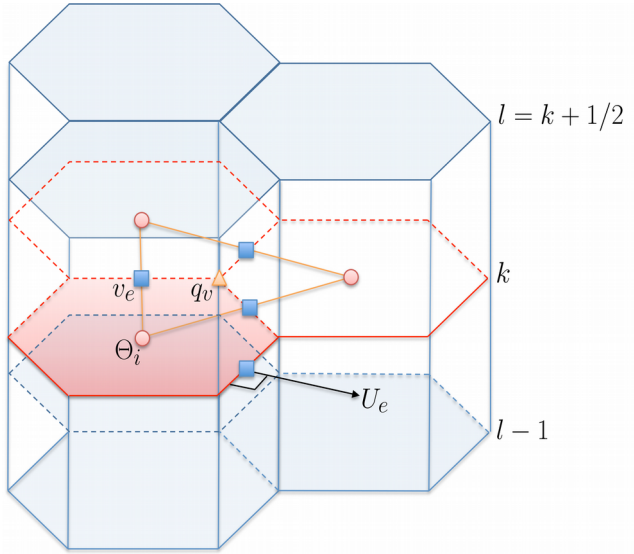
Icosahedral-hexagonal mesh
$10*nbp^2+2$ *cells*



- 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



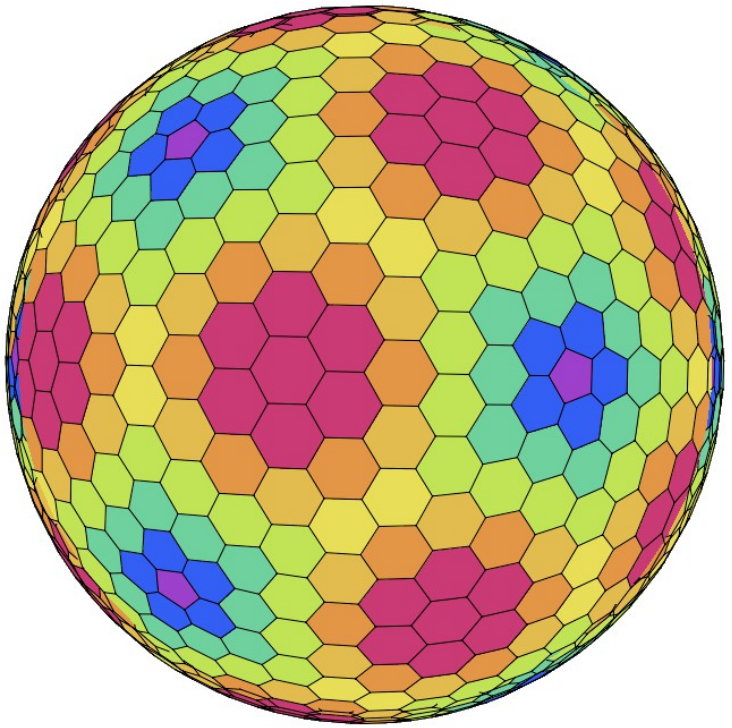


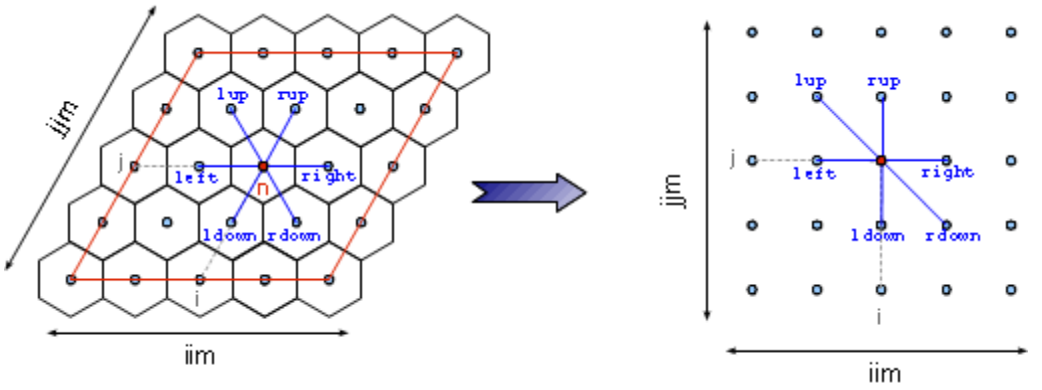




- Discrete integration by parts (Bonaventura & Ringler, 2005 ; Taylor, 2010)
- Energy- and vorticity- conserving Coriolis discretization
- (TRiSK : Thuburn et al., 2009 ; Ringler et al., 2010)

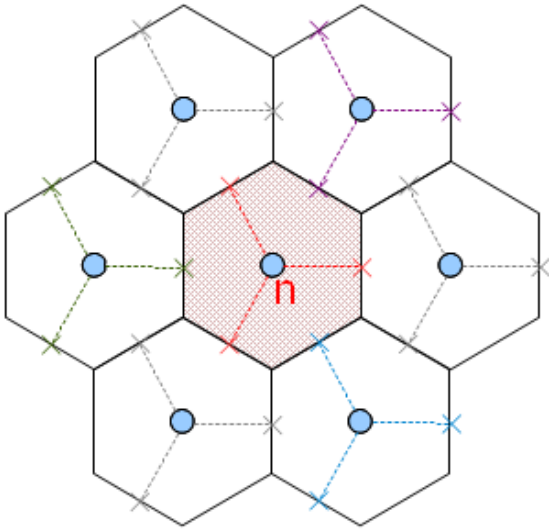
**Energy-conserving 3D core**

- (Tort & Dubos, 2015 ; Dubos et al., 2015)



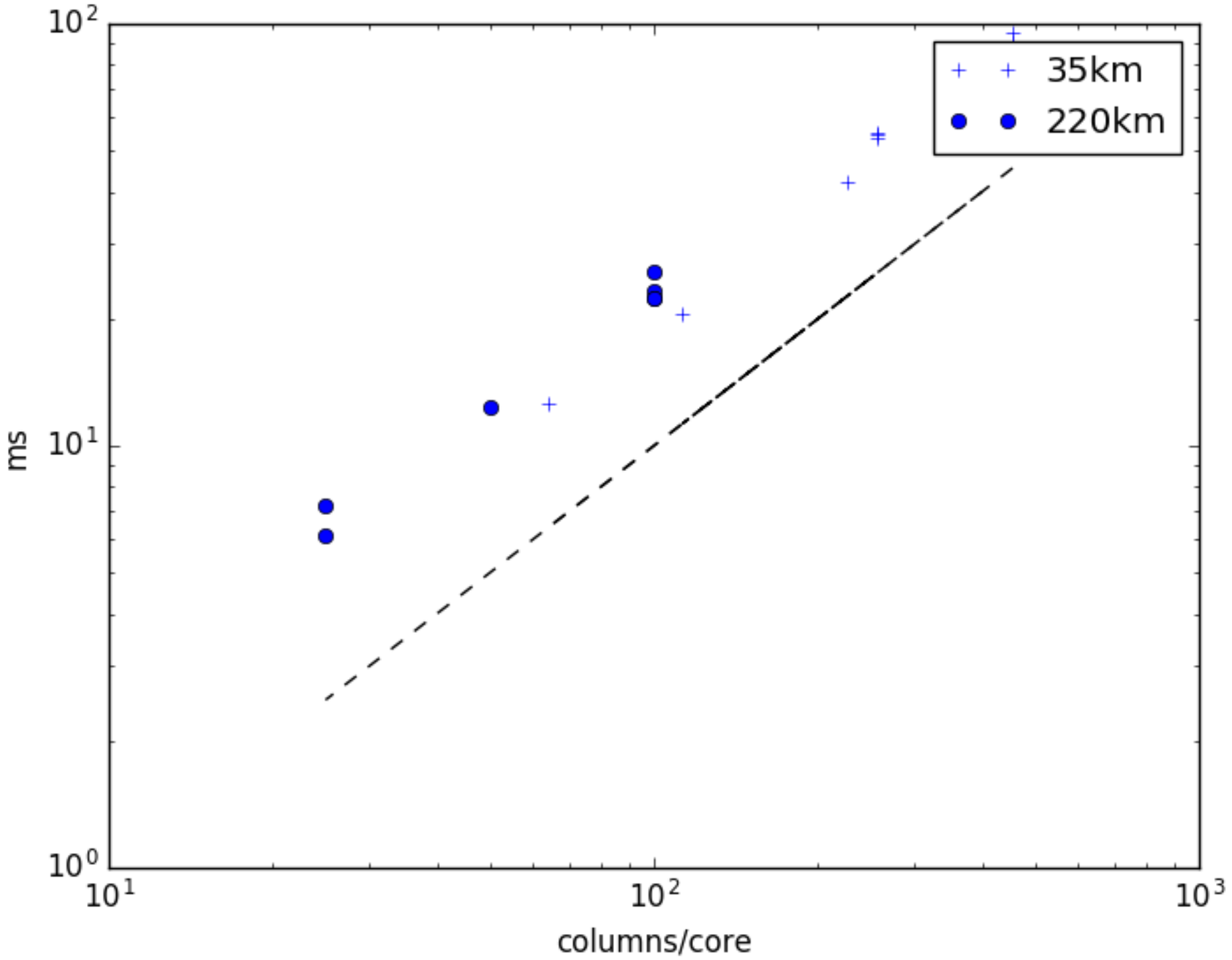



- Vertical direction in outer loops
- **Direct access to neighbours via constant offsets**
- No special case for pentagons (handled by metrics)




```
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))
END DO
```

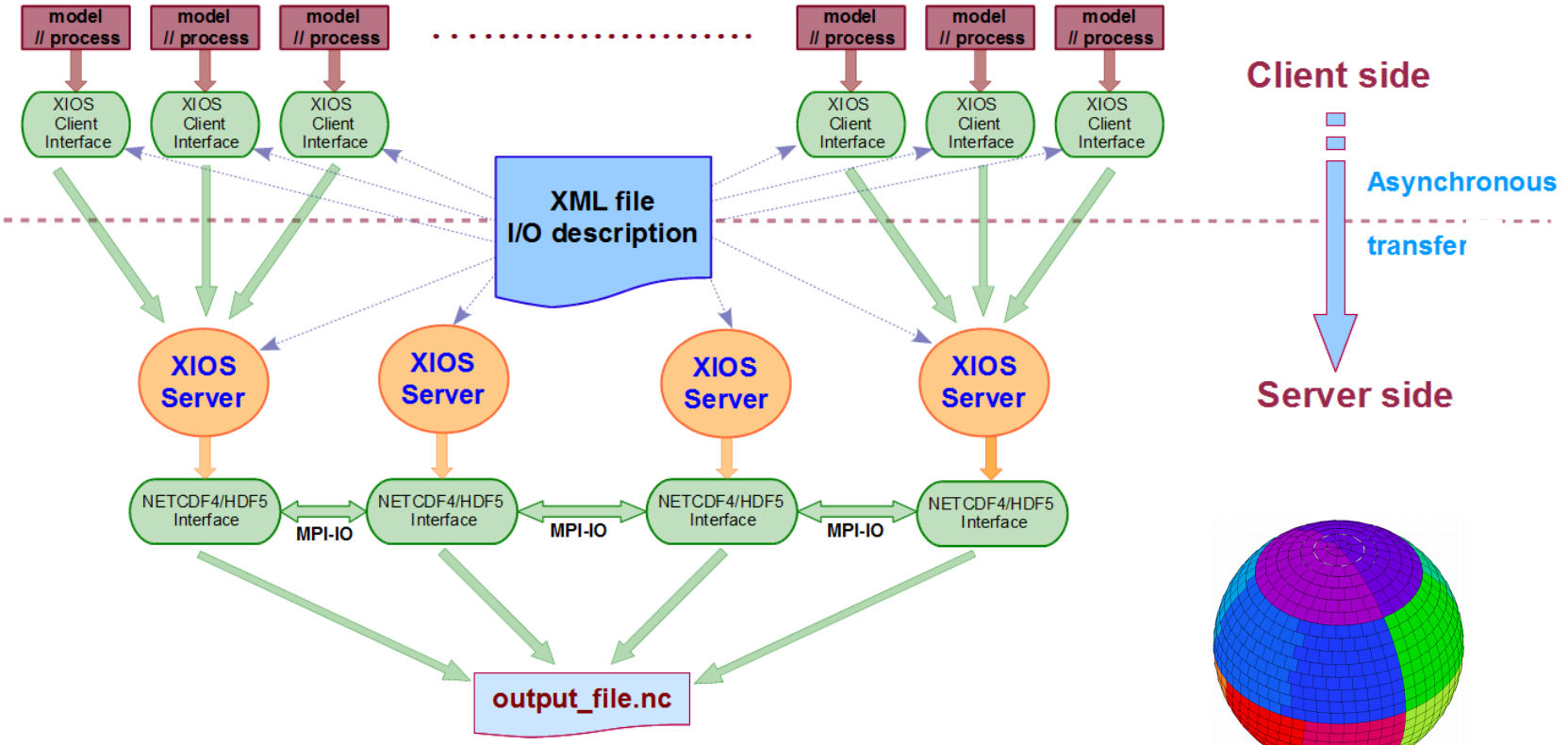
Curie (Intel Sandy Bridge) MPI/OpenMP Dynamics + 4 tracers no phys, no I/O

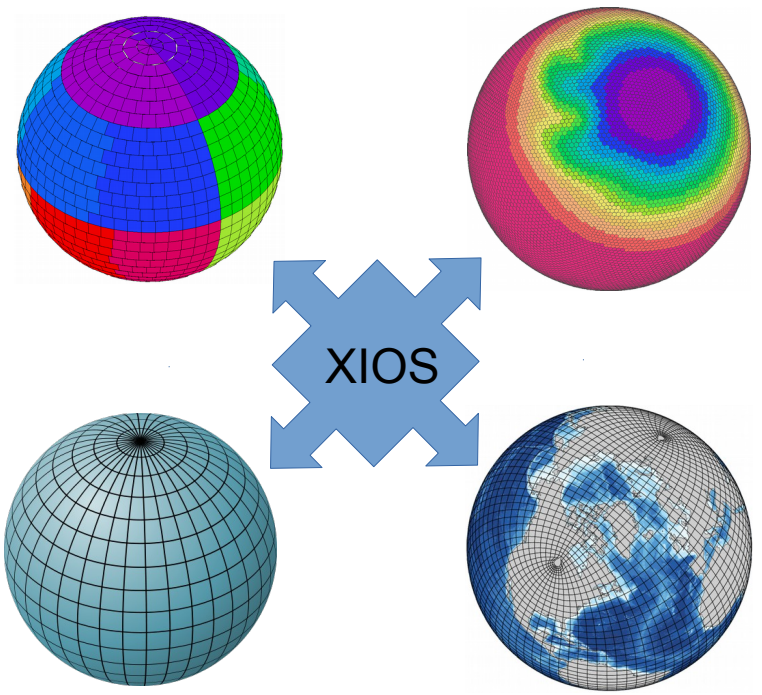


- CMIP requires a throughput of x10000 (30SYPD)
- Some climate modelling still doable with x1000 (3SYPD)
- Ability to attain x1000 depends on maximum stable time step (numerics) and walltime needed to perform one time step (implementation)
- Assuming a large enough machine, reducing walltime is a **strong scaling problem**
- For DYNAMICO, dt (in sec) is about 2.5*dx (in km)
- => 3SYPD
  - At 25km resolution requires about 60 ms per full time step
  - At 8km resolution requires about 20 ms per full time step
  - At 1km resolution requires about 2.5 ms per full time step

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

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

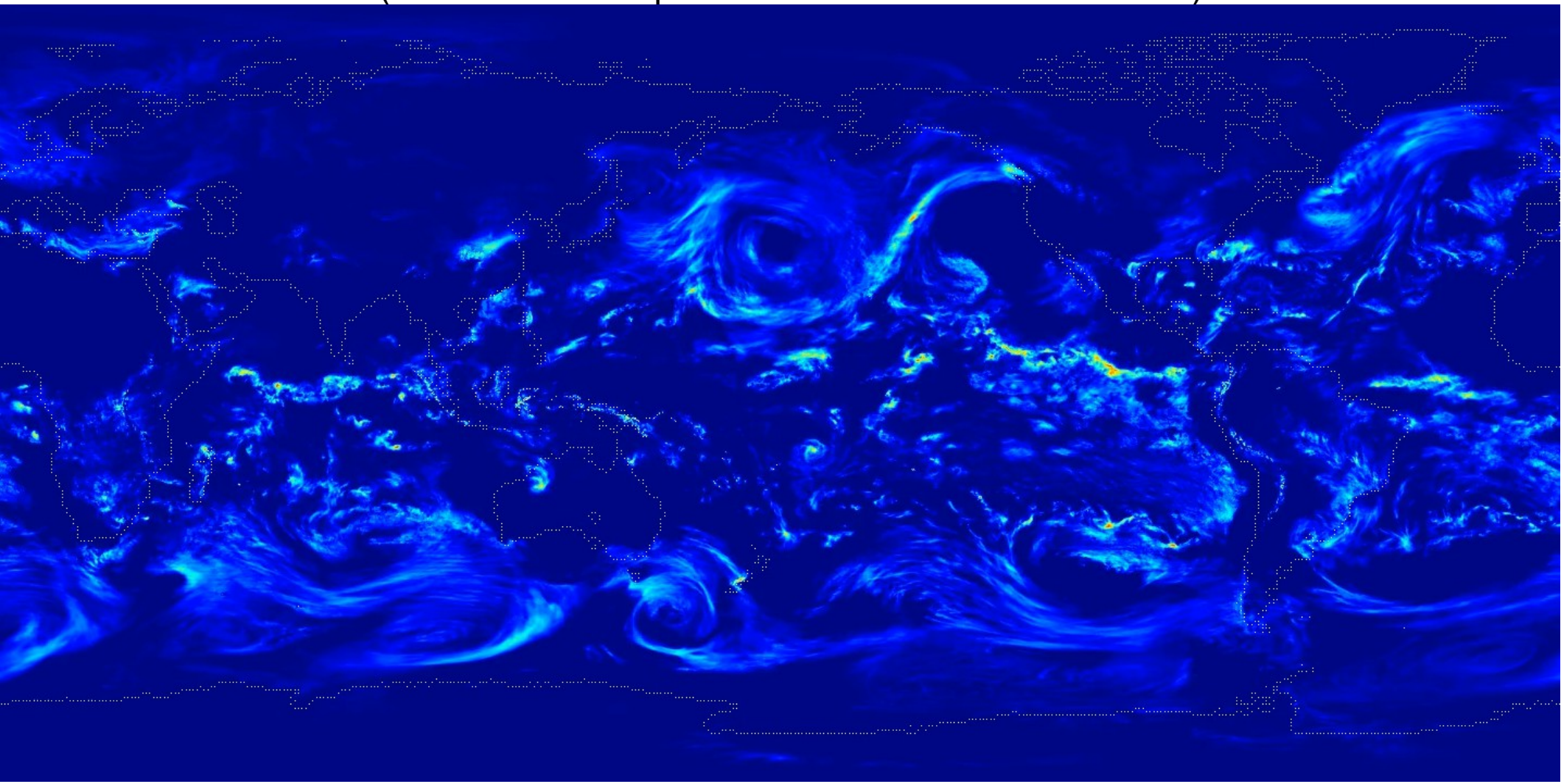


# IPSL-CM7A-HR :

DYNAMICO-LMDZ, 25km

(1 024 000 atmospheric columns x 80 vertical levels)

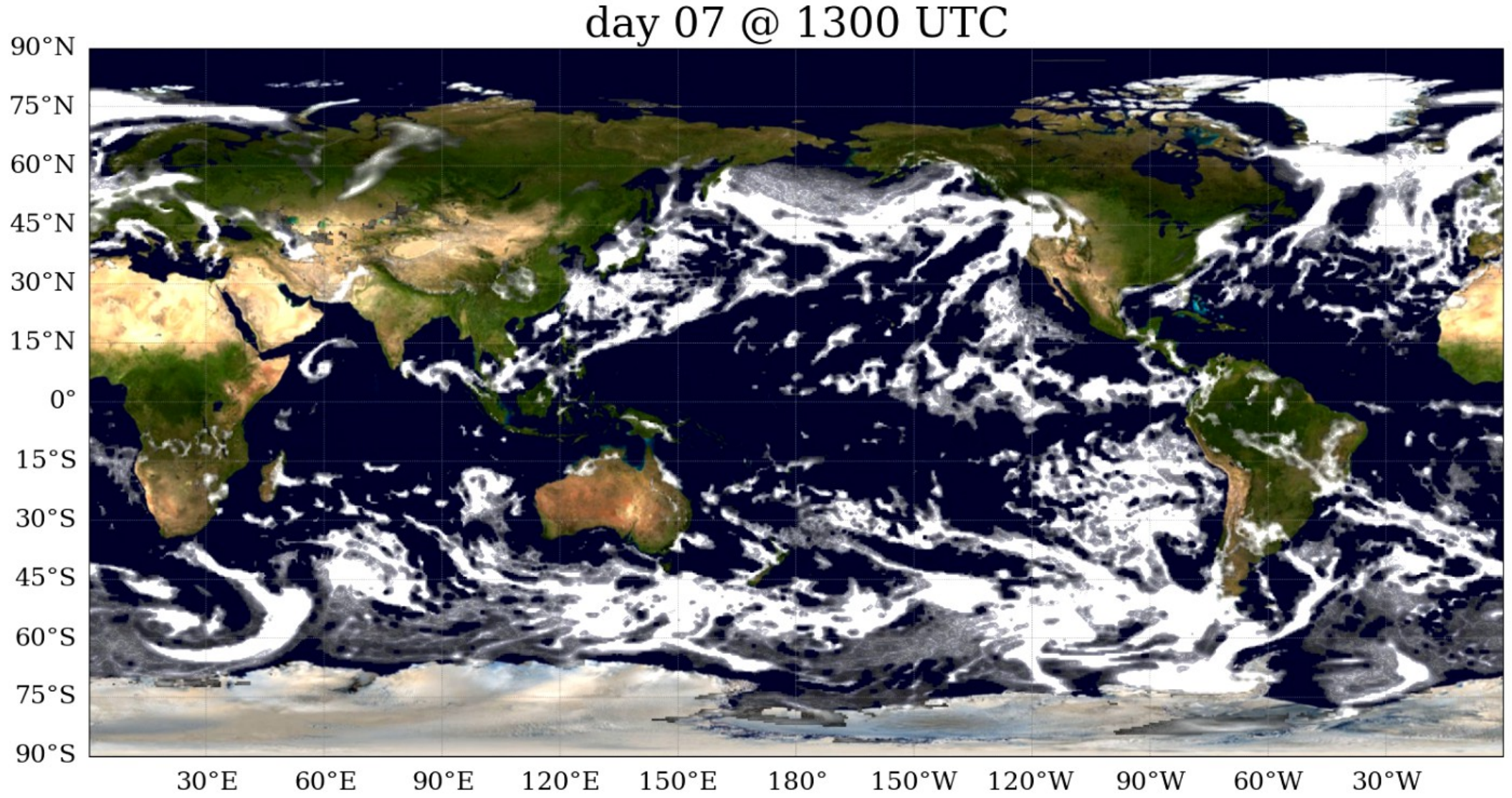


Liquid water

# IPSL-CM7A-HR :

DYNAMICO-LMDZ, 50km

(256 000 atmospheric columns x 80 vertical levels = 20 million cells)




Low-level cloudiness

- Contexte et quelques chiffres
- Du vectoriel au massivement parallèle
- Vers l’exascale ? des défis techniques, mais pas seulement

# Défi technique : adaptation des codes à de nouvelles architectures

Ex : portage de DYNAMICO sur GPU Nvidia V100
(Y. Meurdesoif, A. Durocher (LSCE/IPSL) / HPE / contrat de progrès IDRIS)

*« Le contrat de progrès a pour objectif d'accompagner les communautés utilisatrices de l'IDRIS ... le but final étant, en cas de succès, de pouvoir migrer ces communautés d'utilisateurs identifiées sur la partition convergée... »*
*« Pour être validée, la version GPU d'un code devra … être en moyenne sur l'ensemble des cas tests au moins 4 fois plus performante que les versions non-accélérées en comparaison nœud à nœud. »*

DYNAMICO : ~10 000 LOC
dont ~2000 LOC critiques

=> portage manuel
- par insertion de directives OpenACC
- adaptation de la gestion mémoire, halos, ...

| Nb colonnes atmosphère | 16000 | 64000 | 256000 |
|---|---|---|---|
| 40 procs MPI (CPU) | 68 ms | 282 ms | 1193 ms |
| 1 GPU | 26,6 ms | 77 ms | 299 ms |
| Accélération | 2,5 | 3,6 | 4,0 |

```
!$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
```

# Défi technique : adaptation des codes à de nouvelles architectures

LMDZ (paramétrisations sous-maille) : ~100 000 LOC

- Profil très plat, pas de ‘hot spot’
- Rétrocompatibilité : on supprime rarement du code existant => parties très anciennes (>30 ans)
- Code très touffu
- Parties écrites « maison » : évoluent en permanence
- Parties importées (code radiatif RRTM) : expertise manquante

Si des moyens étaient obtenus pour le faire :

- Est-ce un investissement (architecture pérenne) ou une dépense (architecture sans lendemain) ?
- Comment être performant sur plusieurs architectures avec un code unique ?
- Comment faire évoluer après portage un code communautaire écrit par des physiciens ?

=> *questions liées au développement communautaire des codes*

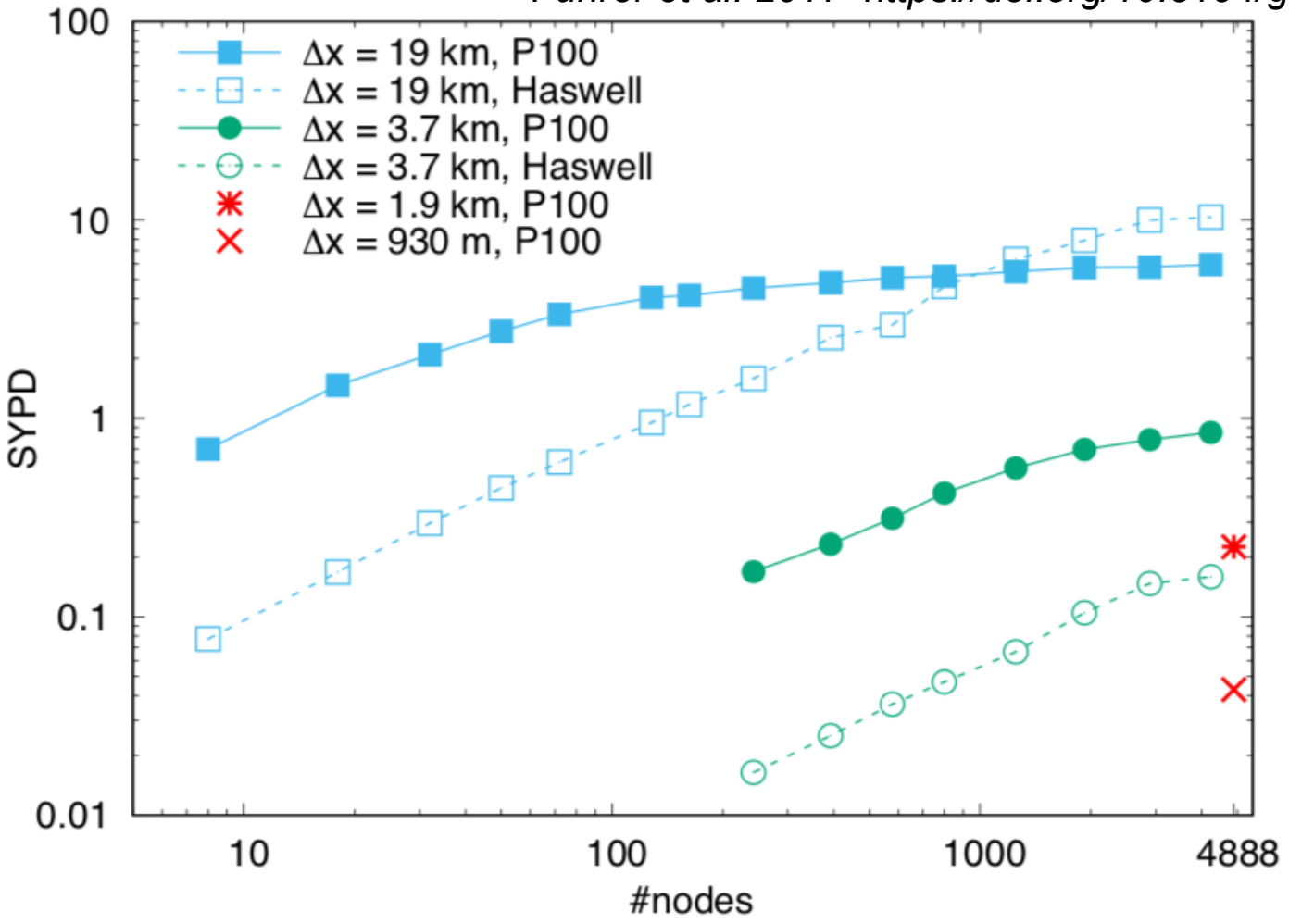
Dans l’air du temps :

- ‘separation of concerns’, ‘domain-specific languages’
- machine learning

# Dans l'air du temps : 'separation of concerns' et domain-specific languages

Modèle COSMO + DSL Stella/GridTools (C++ / templates)

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

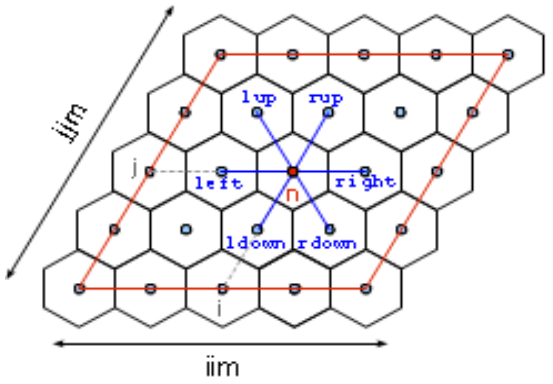



Écueils potentiels : pérennité du DSL, « learning curve », usine à gaz ?

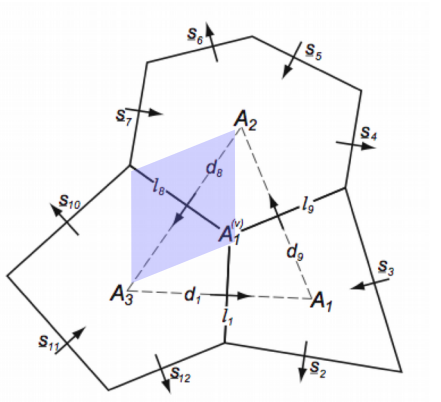
# Dans l'air du temps : 'separation of concerns' et domain-specific languages

## DySL : DYNAMICO-specific « language »

*An experiment in code maintainability and separation of concerns*




```
49 KERNEL('div')
50   FORALL_CELLS_EXT()
51     ON_PRIMAL
52       div_ij=0.
53       FORALL_EDGES
54         div_ij = div_ij + SIGN*LE_DE*u(EDGE)
55       END_BLOCK
56       divu(CELL) = div_ij / AI
57     END_BLOCK
58   END_BLOCK
59 END_BLOCK
```



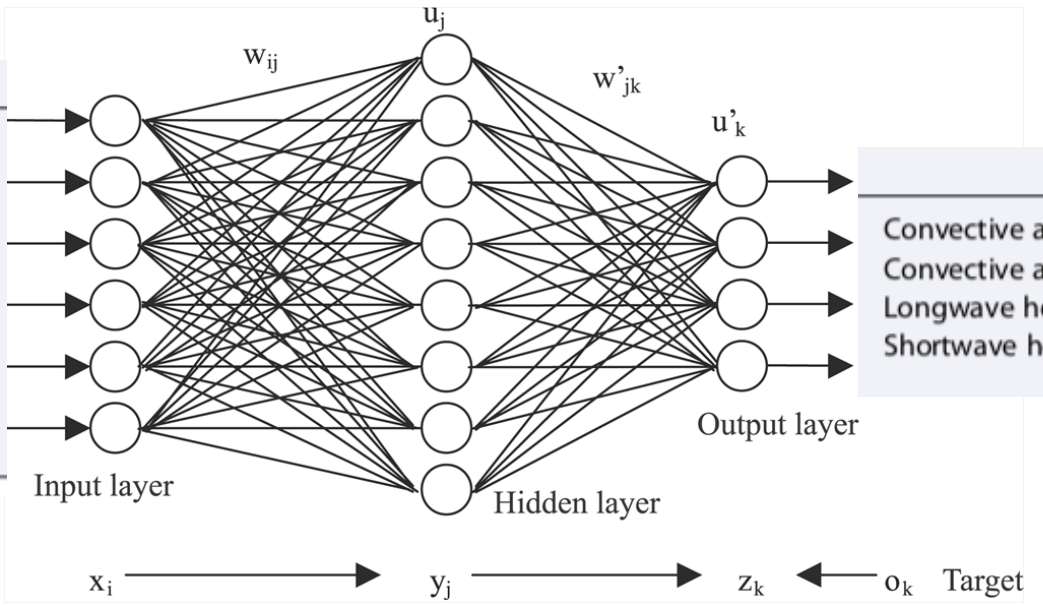

```
1    DO l = ll_begin, ll_end
2  !DIR$ SIMD
3        DO ij=ij_begin_ext, ij_end_ext
4           div_ij=0.
5           div_ij = div_ij + ne_rup*le_de(ij+u_rup)*u(ij+u_rup,l)
6           div_ij = div_ij + ne_lup*le_de(ij+u_lup)*u(ij+u_lup,l)
7           div_ij = div_ij + ne_left*le_de(ij+u_left)*u(ij+u_left,l)
8           div_ij = div_ij + ne_ldown*le_de(ij+u_ldown)*u(ij+u_ldown,l)
9           div_ij = div_ij + ne_rdown*le_de(ij+u_rdown)*u(ij+u_rdown,l)
10          div_ij = div_ij + ne_right*le_de(ij+u_right)*u(ij+u_right,l)
11          divu(ij,l) = div_ij / Ai(ij)
12       END DO
13    END DO
14
```

```
2     DO ij = 1, primal_num
3        DO l = 1,llm
4           div_ij=0.
5           DO iedge = 1, primal_deg(ij)
6              edge = primal_edge(iedge,ij)
7              div_ij = div_ij + primal_ne(iedge,ij)*le_de(edge)*u(l,edge)
8           END DO
9           divu(l,ij) = div_ij / Ai(ij)
10       END DO
11    END DO
```

- Simplicité plutôt que généralité => faible coût d'entrée
- Génération de blocs de code Fortran par manipulation du pseudo-code « DySL »
- => faible coût de sortie
- Approches plus sophistiquées (parser/backend) envisagées avec J. Bigot (MdLS)

# Dans l’air du temps : machine learning

| Input variables |
| --- |
| Temperature at beginning of time step |
| Humidity at beginning of time step |
| Surface pressure |
| Sensible heat flux |
| Latent heat flux |
| Temperature tendency from dynamics |
| Humidity tendency from dynamics |
| Incoming solar radiation |
| Size of stacked array |




| Output variables |
| --- |
| Convective and turbulent temperature tendency |
| Convective and turbulent humidity tendency |
| Longwave heating tendency |
| Shortwave heating tendency |

**Stephan Rasp[a,b,1], Michael S. Pritchard[b], and Pierre Gentine[c,d]**
www.pnas.org/cgi/doi/10.1073/pnas.1810286115

- Appel à routine(s) complexes/obscures remplacé par réseau de neurone
- => générique, potentiellement efficace sur GPU/architecture convergée
- Encore beaucoup de questions : contraintes physiques, utilisation hors domaine d’apprentissage ...

- IPSL : une transition vers le parallélisme massif opérée et opérationnelle (CMIP6)
- Vers l'exascale ? des défis techniques, mais pas seulement
  - Exploiter le parallélisme massif peut nécessiter une reformulation profonde du contenu du modèle (ex : DYNAMICO)
  - Les architectures exascale auront-elle la scalabilité forte nécessaire pour la modélisation du climat ?
  - Le but n'est pas de porter nos codes, mais d'arriver à un modèle de développement qui assure une certaine portabilité de la performance
  - Les solutions pour parvenir à ce but doivent tenir compte du caractère communautaire du développement, et du rôle de laboratoire scientifique joué par le modèle
    - solutions « maison » : sur mesure mais ressources, expertise ?
    - solutions externes : besoins spécifiques, pérennité, coût de sortie, learning curve ?
    - contourner le problème : machine learning ??

*Merci de votre attention*