LMDZ

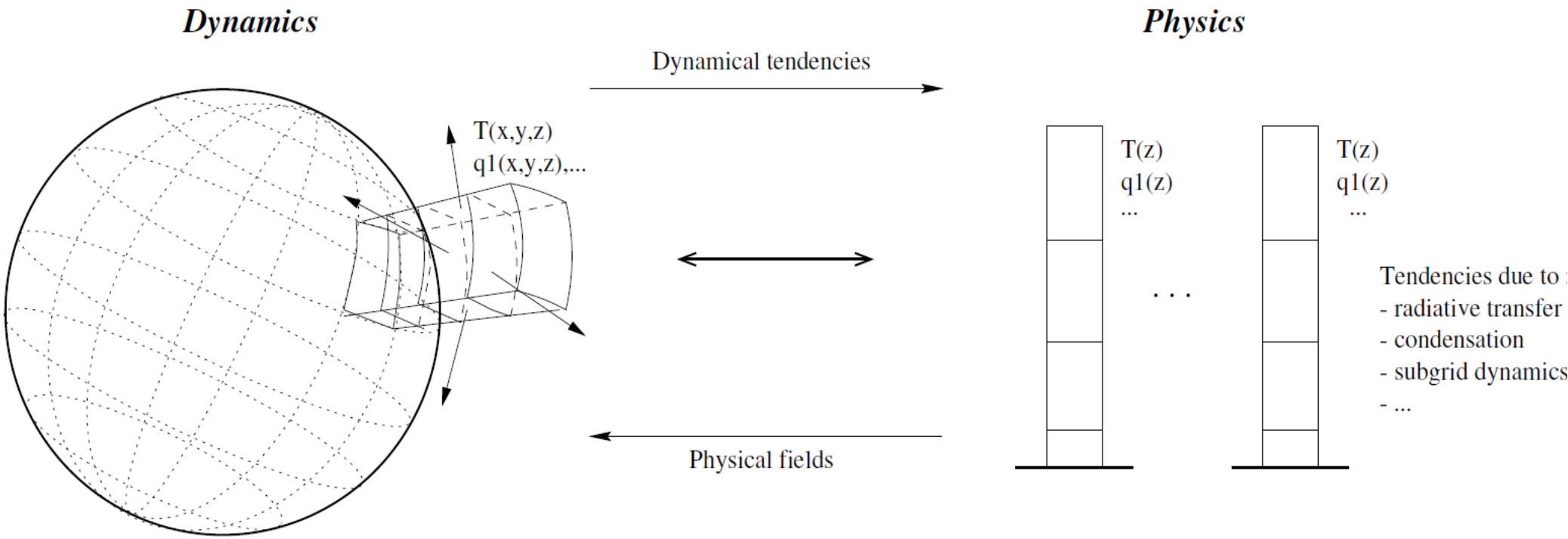
Dynamics/physics organization,
Grids,
Time stepping,
Dissipation...

LMDZ courses, December 17 2014

Overview of course topics

- Grids:
  - Horizontal grids in the physics & dynamics
  - Vertical discretization
- Time marching:
  - Generalities about time marching schemes
  - What is used in LMDZ
  - Longitudinal polar filter
- Lateral diffusion and sponge layer:
  - Energy cascade
  - Illustrative example of diffusion
  - Sponge layer near model top

Grids in LMDZ



Separation between physics and dynamics:

- "dynamics": solving the GFD equations on the sphere; usually with the assumption of a hydrostatic balance and thin layer approximation. Valid for all terrestrial planets.
- "physics": (planet-specific) local processes, local to individual atmospheric columns.

Horizontal grids in LMDZ

Grid dimensions specified when compiling LMDZ:

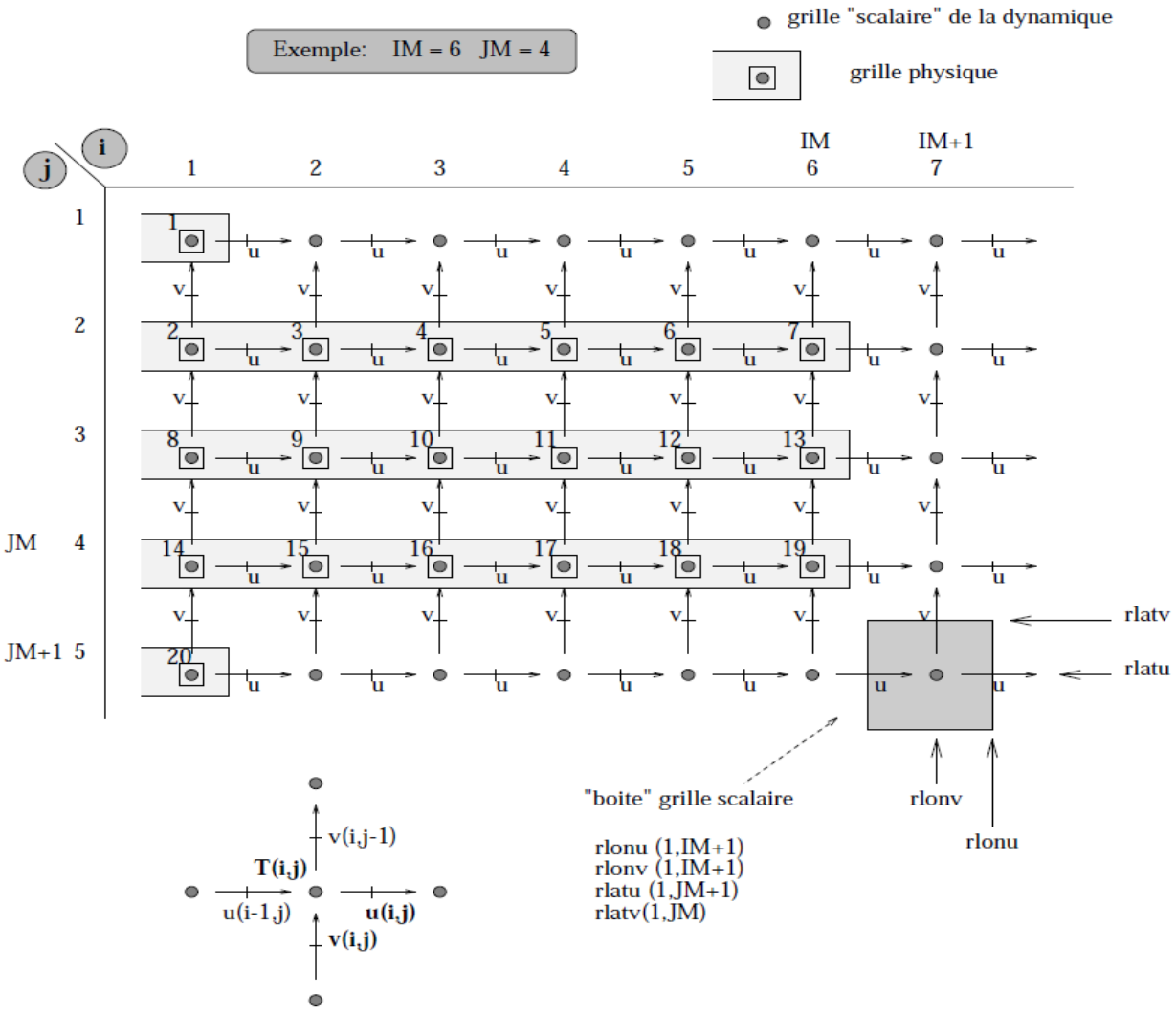
makelmdz[_fcm] -d iimxjjmxllm ...

In the dynamics:

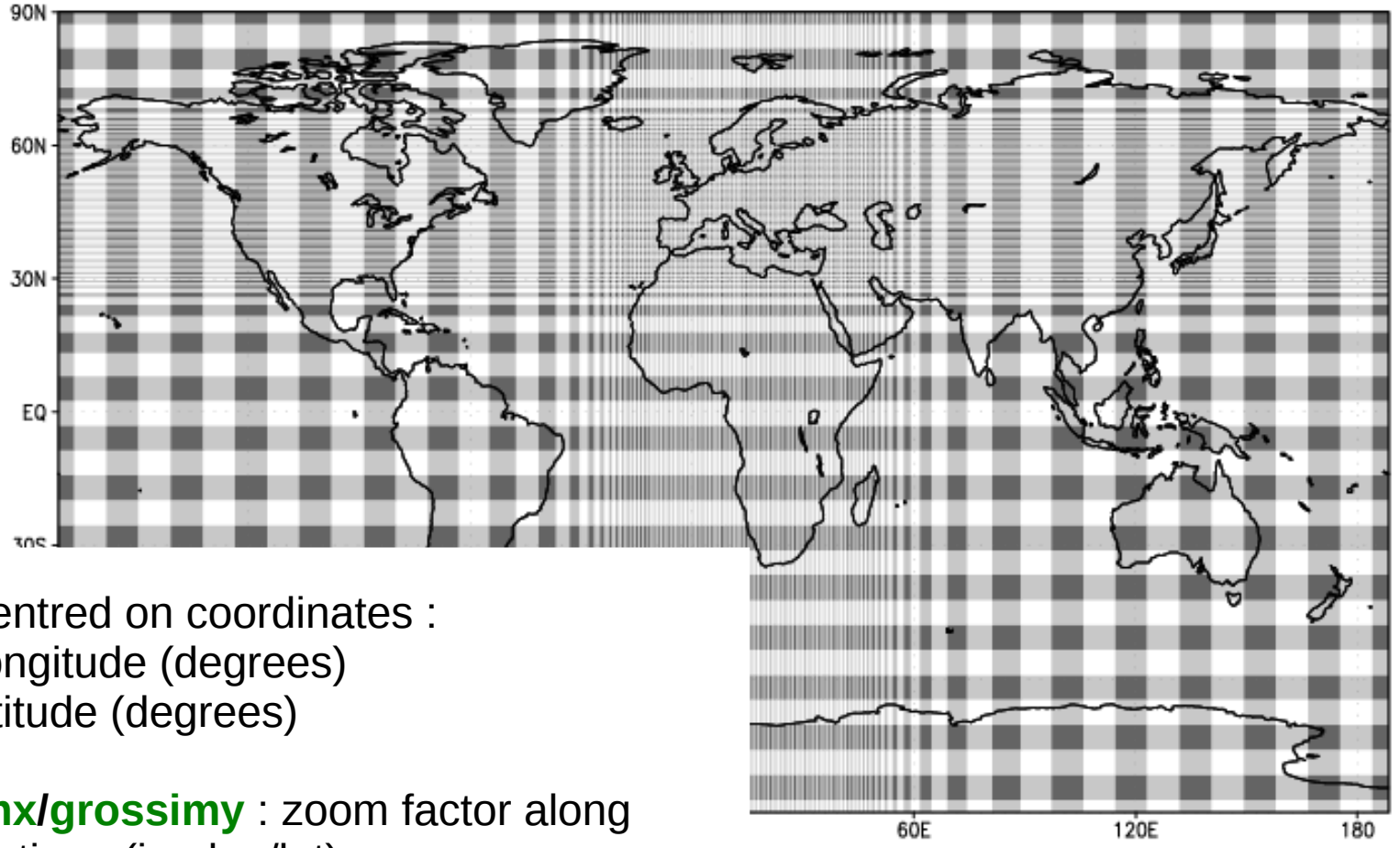
- Staggered grids, u, v and scalars (temperature, tracers) are on different meshes
- Global lonxlat grids with redundant grid points
  - at the poles
  - in longitude

In the physics:

- Colocated variables
- No global lonxlat horizontal grid, columns are labelled **using a single index** (from North Pole to South Pole)



LMDZ, Z for Zoom

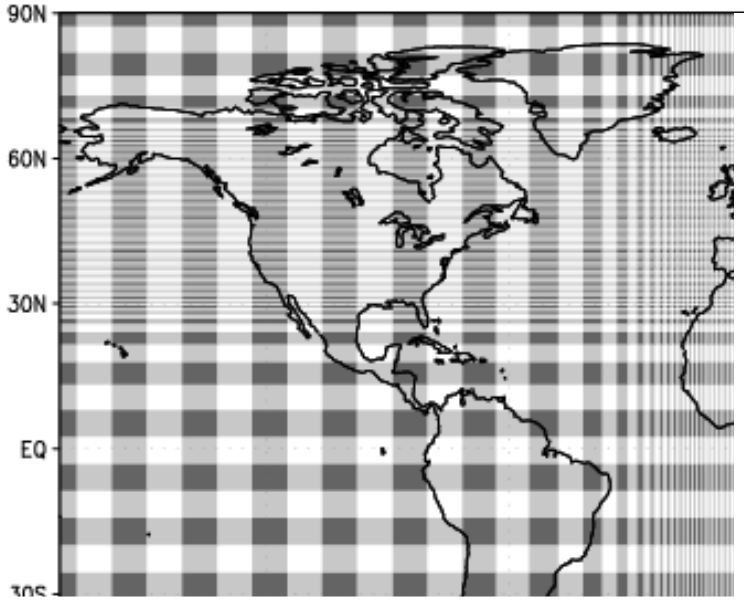


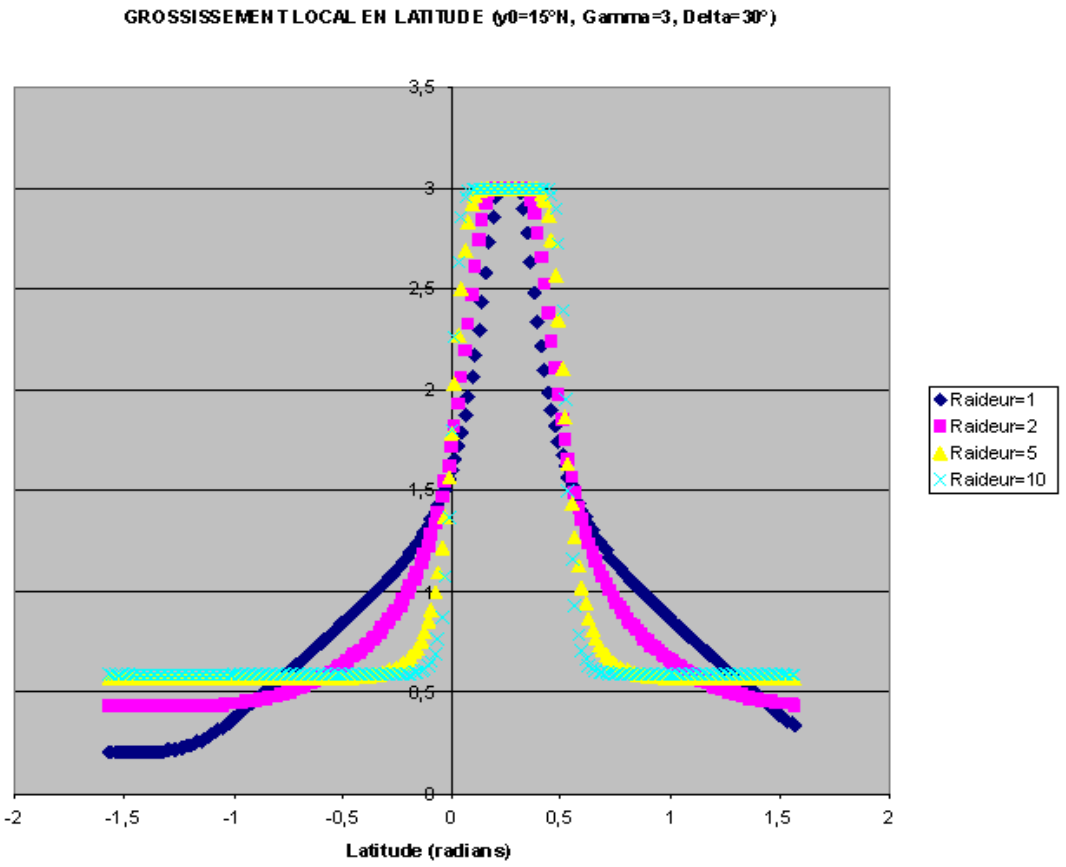
Zoom centred on coordinates :
clon : longitude (degrees)
clat : latitude (degrees)

grossimx/grossimy : zoom factor along x/y directions (i.e. lon/lat)
Computed as the ratio of the smallest mesh (I.e. in the zoom), compared to the mesh size for a global regular grid with the same total number of points.

dzoomx/dzoomy : fraction of the grid containing the zoomed area: dzoom*360° by dzoomy*180°

LMDZ, Z for Zoom





Zoom centred on coordinates :
clon : longitude (degrees)
clat : latitude (degrees)

grossimx/grossimy : zoom factor along x/y directions (i.e. lon/lat)
Computed as the ratio of the smallest mesh (I.e. in the zoom), compared to the mesh size for a global regular grid with the same total number of points.

taux/tauy : steepness of the transition between inner zoom and outer zoom meshes (typically one tries to avoid sharp transitions; tau ~ 3 is a reasonable value)

Nudging in LMDZ

$$\frac{\partial u}{\partial t} = \frac{\partial u}{\partial t}_{GCM} + \frac{u_{analyse} - u}{\tau}$$

$$\frac{\partial v}{\partial t} = \frac{\partial v}{\partial t}_{GCM} + \frac{v_{analyse} - v}{\tau}$$

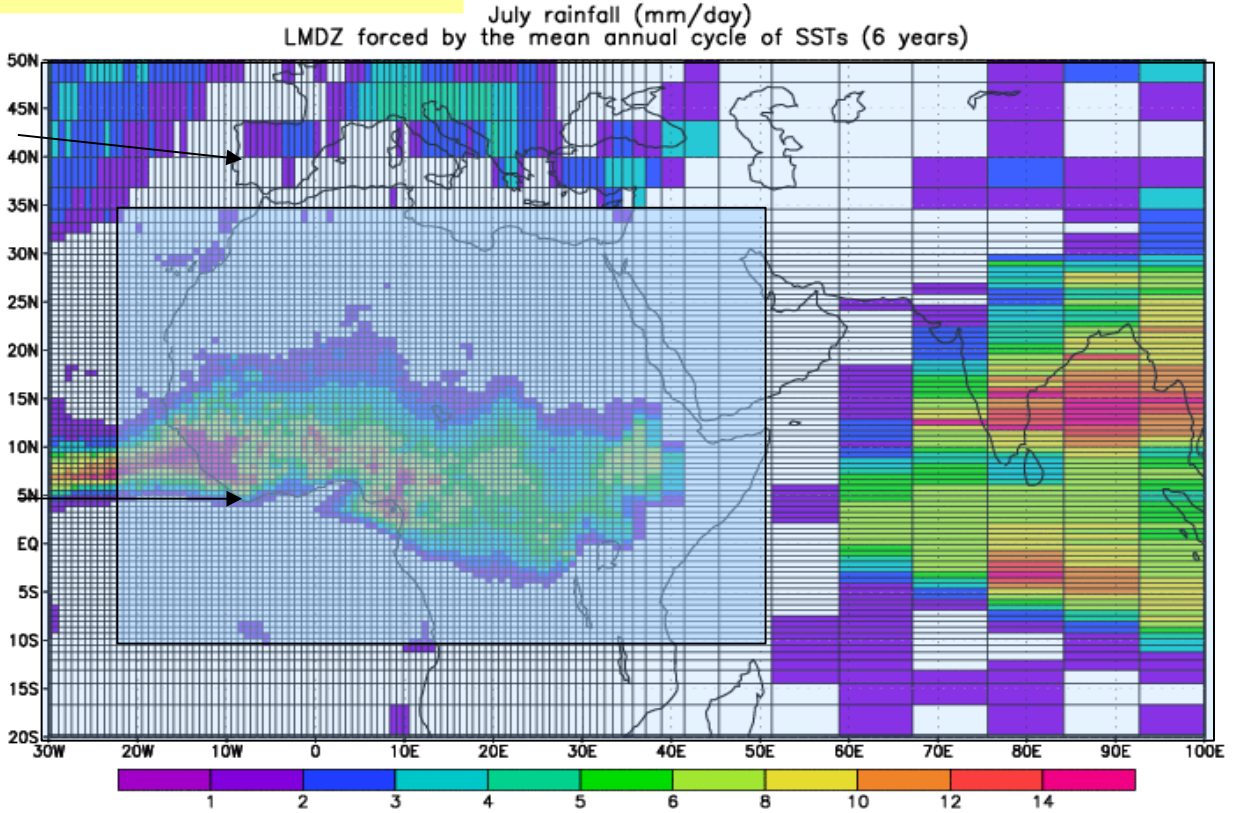
Nudging towards analyses or reanalyses with given time constants $u_{analyse}$ $v_{analyse}$

Example of nudging parameters:

ok_guide = y
guide_T = n , guide_p = n , guide_q = n
guide_u = y , guide_v = y
tau_min_u = 0.0208333 (days)
tau_max_u = 10 (days)
tau_min_v = 0.0208333 (days)
tau_max_v = 10 (days)

Strong nudging (τ=30min)

Weak to moderate nudging (τ=10 days)



Vertical discretization in LMDZ

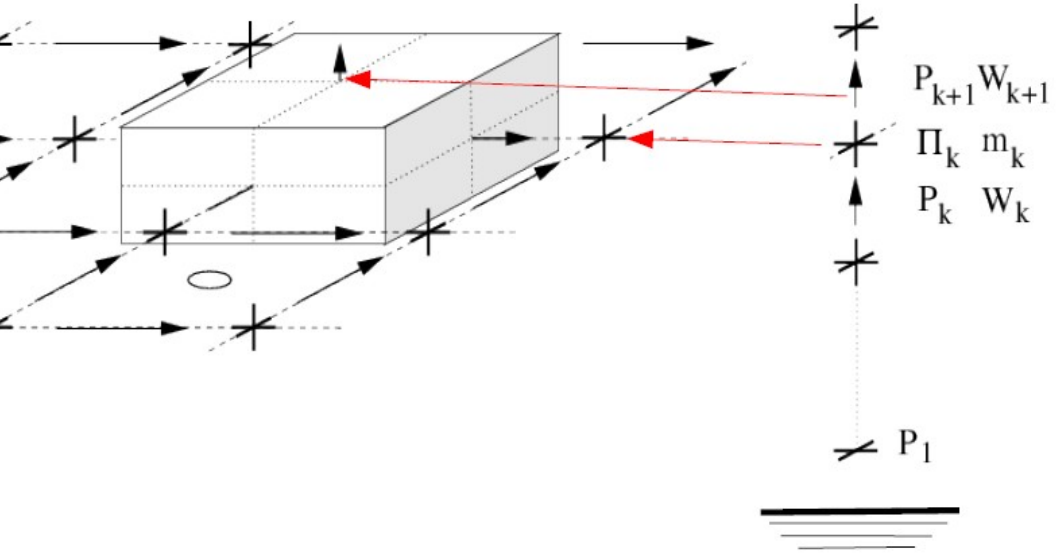
- Model levels are hybrid sigma-pressure levels:

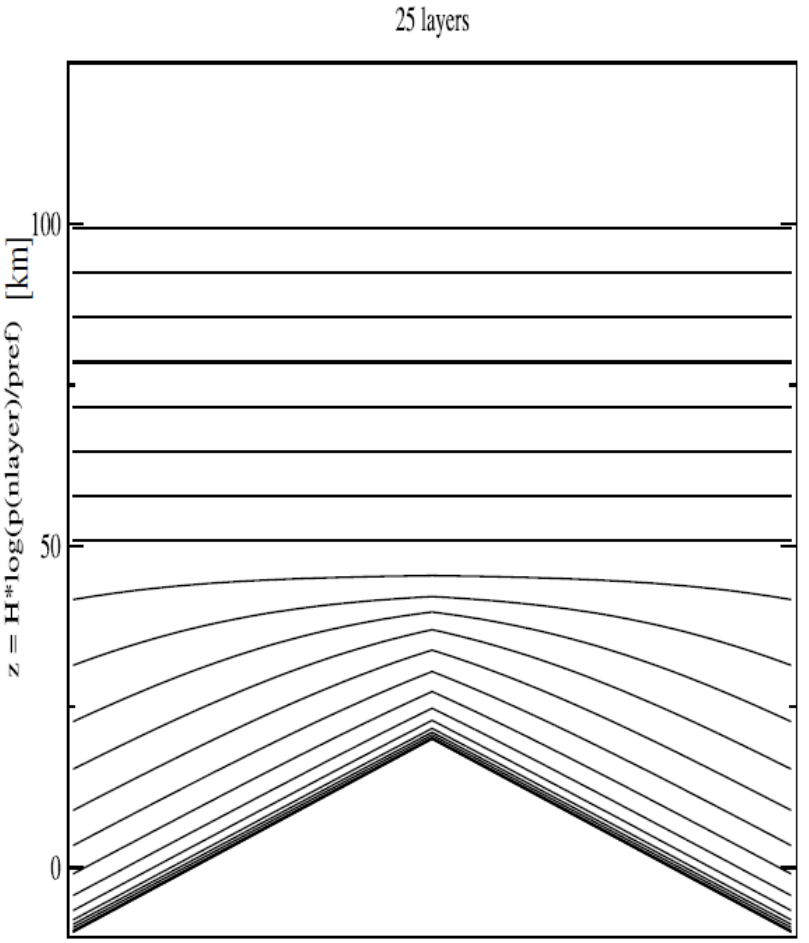
  P(level,time) = **ap**(level) + **bp**(level) . **Ps** (time)

  hybrid coordinates **ap**(k) and **bp**(k) are fixed for a given model run

  Surface pressure **Ps**(t) varies during the run

- Near the surface ap ~ 0 => bp(k) ~ P/Ps
- At high altitudes , bp ~ 0 => ap(k) ~ P





Vertical discretization in LMDZ

- Setting model levels via the def files (also a function of number of vertical levels) :

  vert_sampling = strato : a default for stratospheric extensions

  vert_sampling = sigma : automated generation of purely sigma levels

  vert_sampling = param : load values from a “sigma.def” file

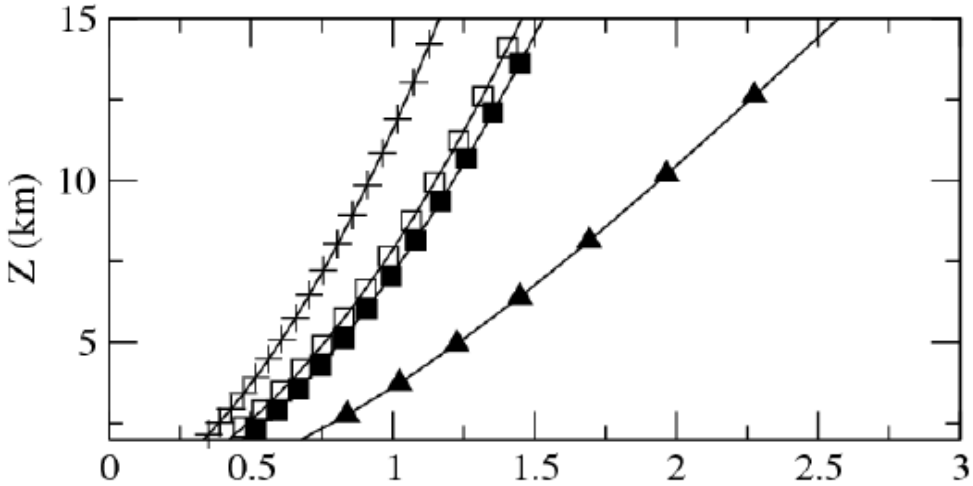
  vert_sampling = tropo : a default for tropospheric simulations

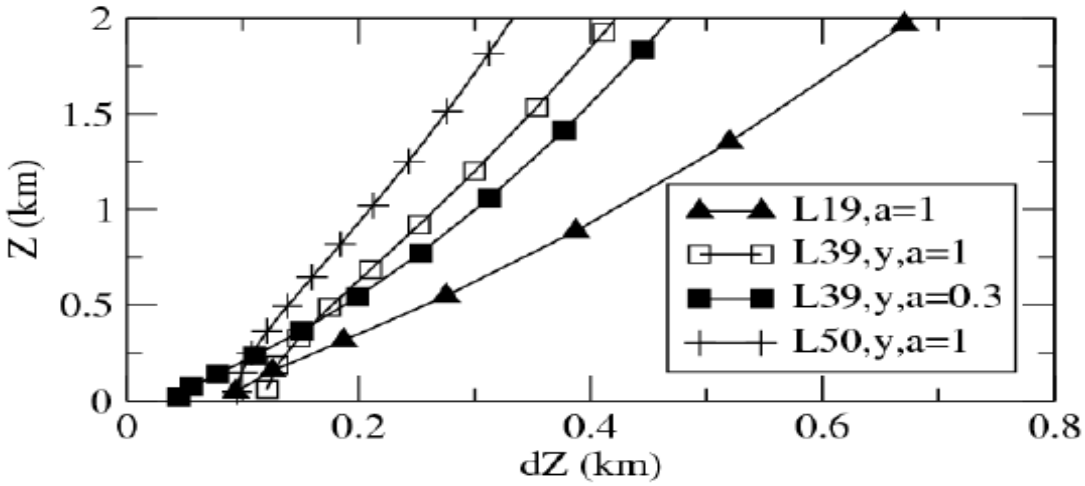
  vert_sampling = read : read ap() and bp() from file “hybrid.txt”

  => Typically you don't need to mess with the vertical discretization,

  the default behaviour most likely matches your needs.

  => Check out routine dyn3d_common/disvert.F90





Time marching schemes

- The big picture: you want to solve

$$\begin{aligned} \frac{dx(t)}{dt} &= F(x,t) \\ x(t=0) &= x_0 \end{aligned}$$

- And it is all about using a time marching scheme, built on Taylor expansion for evaluation of the time derivative, and choosing at which time level t=n.dt F(x,t) is to be evaluated

Time marching schemes

- Explicit Euler scheme (1st order in time):

$$\begin{aligned} \frac{dx(t)}{dt} &\simeq \frac{x_{i+1} - x_i}{\delta t} \\ F(x,t) &\simeq F(x_i, t_i) \end{aligned}$$

- Implicit Euler scheme (1st order in time):

$$\begin{aligned} \frac{dx(t)}{dt} &\simeq \frac{x_{i+1} - x_i}{\delta t} \\ F(x,t) &\simeq F(x_{i+1}, t_{i+1}) \end{aligned}$$

- Crank-Nicholson scheme (2nd order in time):

$$\begin{aligned} \frac{dx(t)}{dt} &\simeq \frac{x_{i+1} - x_i}{\delta t} \\ F(x,t) &\simeq \frac{F(x_{i+1}, t_{i+1}) + F(x_i, t_i)}{2} \end{aligned}$$

Time marching schemes

- Matsuno scheme: a predictor-corrector (Euler explicit-Euler Implicit) scheme (1st order):

$$\begin{aligned} \frac{dx(t)}{dt} &\simeq \frac{x_{i+1} - x_i}{\delta t} \\ x^p(t_{i+1}) &= x(t_i) + \delta t . F(x(t_i), t_i) \\ F(x,t) &\simeq F(x^p(t_{i+1}), t_{i+1}) \end{aligned}$$

- Leapfrog scheme: use encompassing time steps to evaluate the derivative (2nd order):

$$\begin{aligned} \frac{dx(t)}{dt} &\simeq \frac{x_{i+1} - x_{i-1}}{2\delta t} \\ F(x,t) &\simeq F(x(t_i), t_i) \end{aligned}$$

Time marching schemes

- Illustrative example, on a decay equation

$$\frac{dq(t)}{dt} = -\frac{1}{\tau} q(t) \longrightarrow q(t) = q_0 e^{-\frac{t}{\tau}}$$

- Resulting integration schemes:

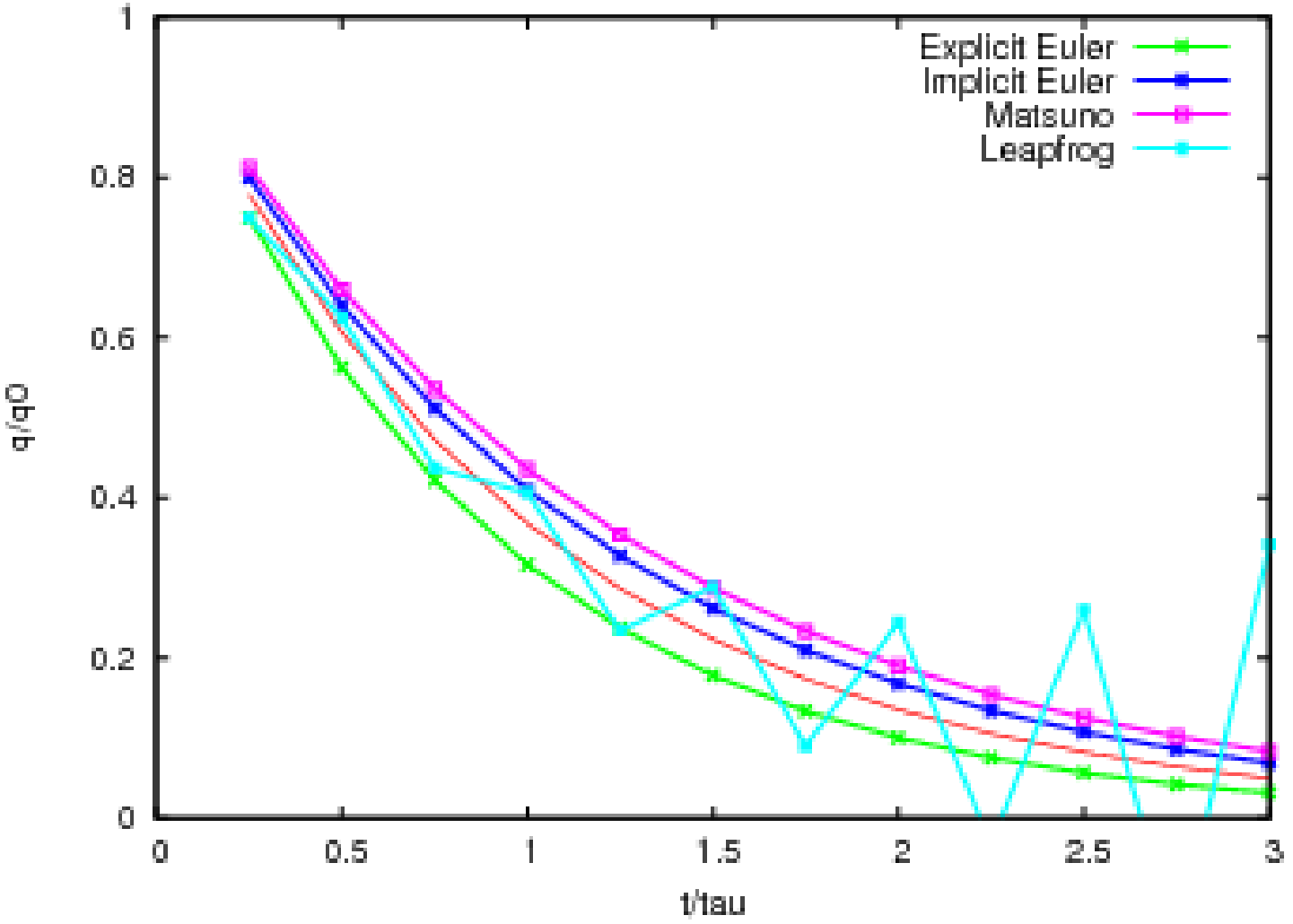
$$\text{EE}: \quad q^{n+1} = \left[1 - \frac{\delta t}{\tau}\right] q^n$$

$$\text{EI}: \quad q^{n+1} = \left[\frac{1}{1 + \delta t/\tau}\right] q^n$$

$$\text{CN}: \quad q^{n+1} = \left[\frac{1 - \delta t/(2\tau)}{1 + \delta t/(2\tau)}\right] q^n$$

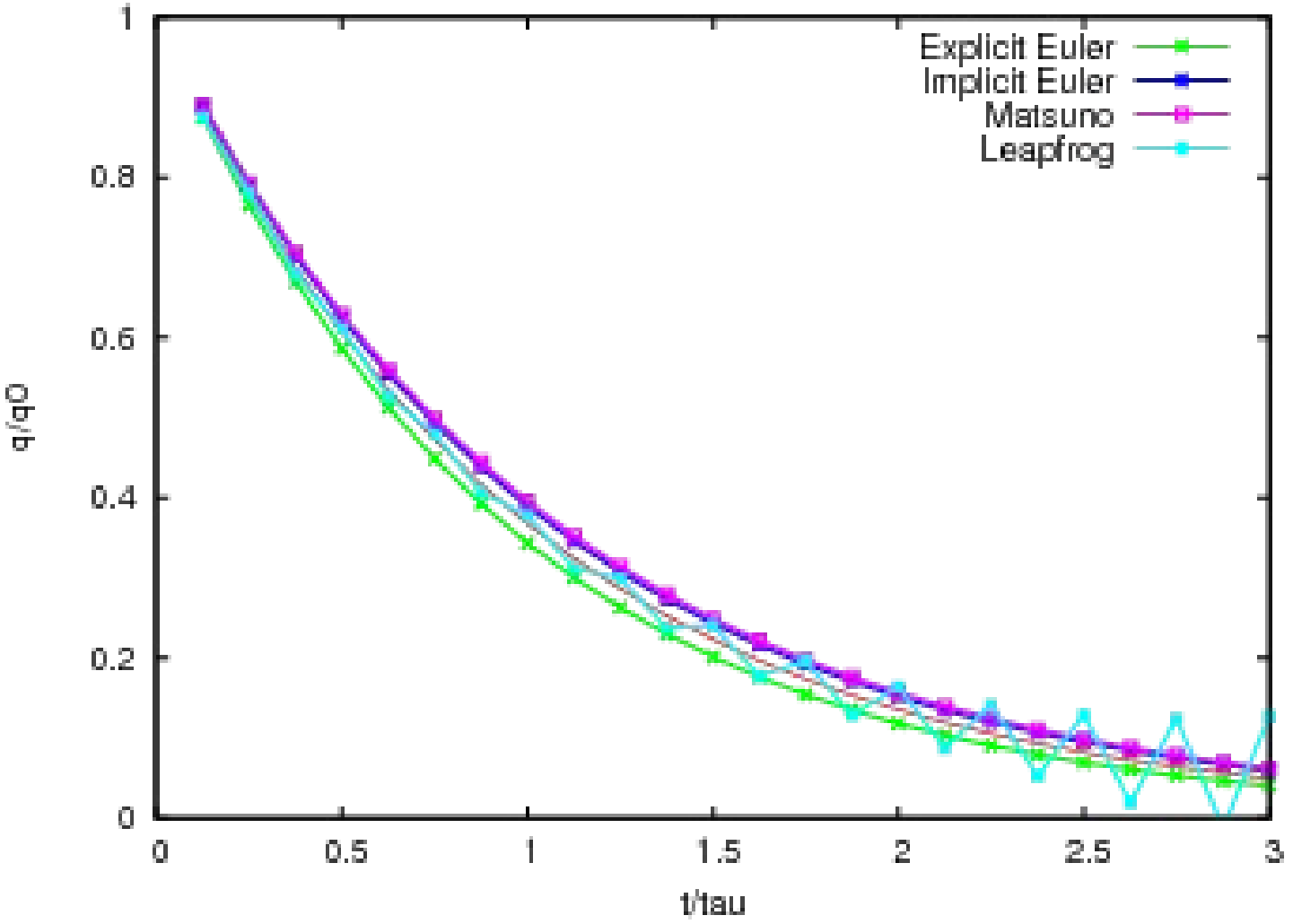
- Stability requirement (CFL) for EE : dt/tau < 2

Time marching schemes



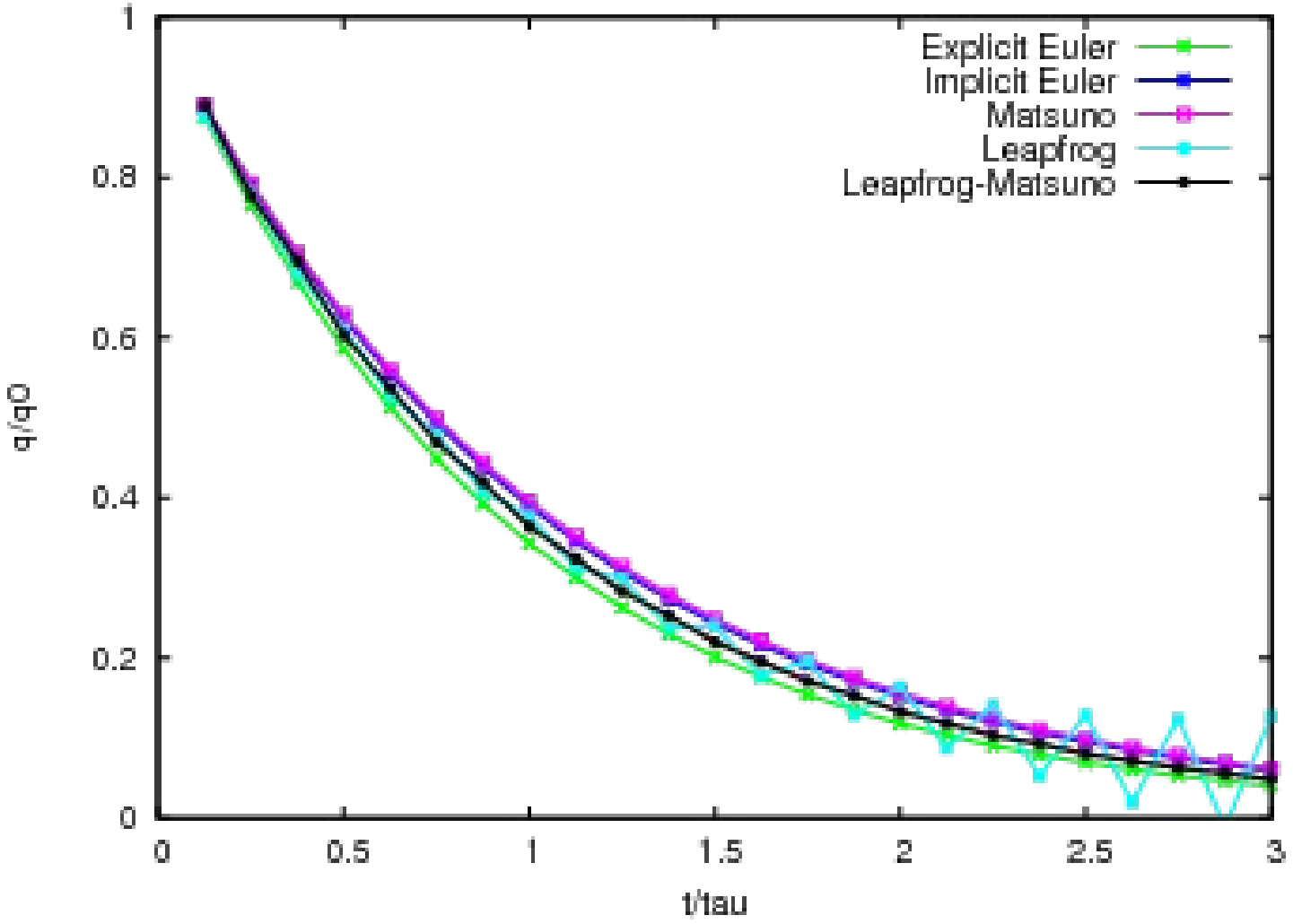
- 4 integration steps per unit of t/tau

Time marching schemes



- 8 integration steps per unit of t/tau

Time marching schemes



- 8 integration steps per unit of t/tau

Time marching in LMDZ

Time splitting between physics/dynamics/dissipation:

$$\frac{\partial \psi}{\partial t} = Dyn(\psi) + Phy(\psi) + Dissip(\psi)$$

- Dynamics : Leapfrog-Matsuno scheme

  Leapfrog steps with a Mastuno step every iperiod step
- Physics : Explicit Euler

  Every iphysiq dynamical steps (multiple of iperiod)
- Dissipation: Explicit Euler

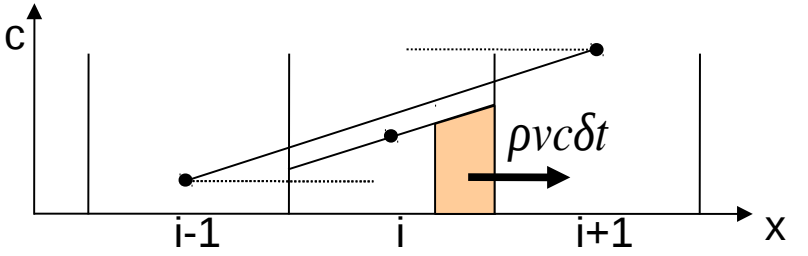
  Every dissip_period dynamical steps (multiple of iperiod)

Tracer advection in LMDZ

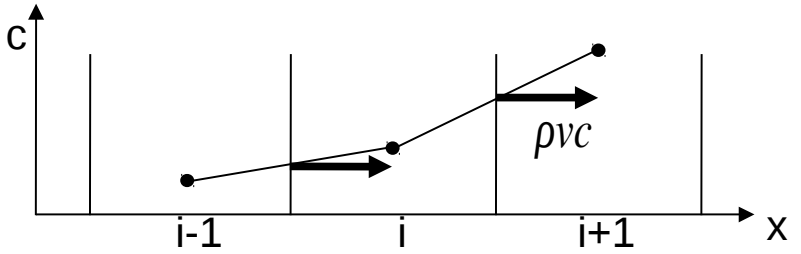
Use of the Van Leer I scheme (1977), a second order finite volume scheme with slope limiters (e.g. MUSCL, MINMOD) (Hourdin et Armengaud, 1999).

Guaranties of fundamental physical properties of transport : conservation of the total quantity, postitivity, monotony, non amplification of extremas, weak numerical diffusion

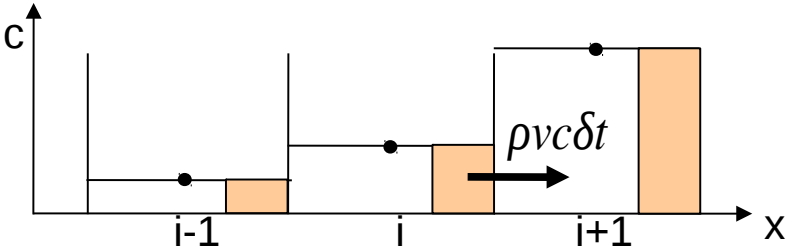
Scheme I by Van Leer (1977)



Centered finite differences (second order)

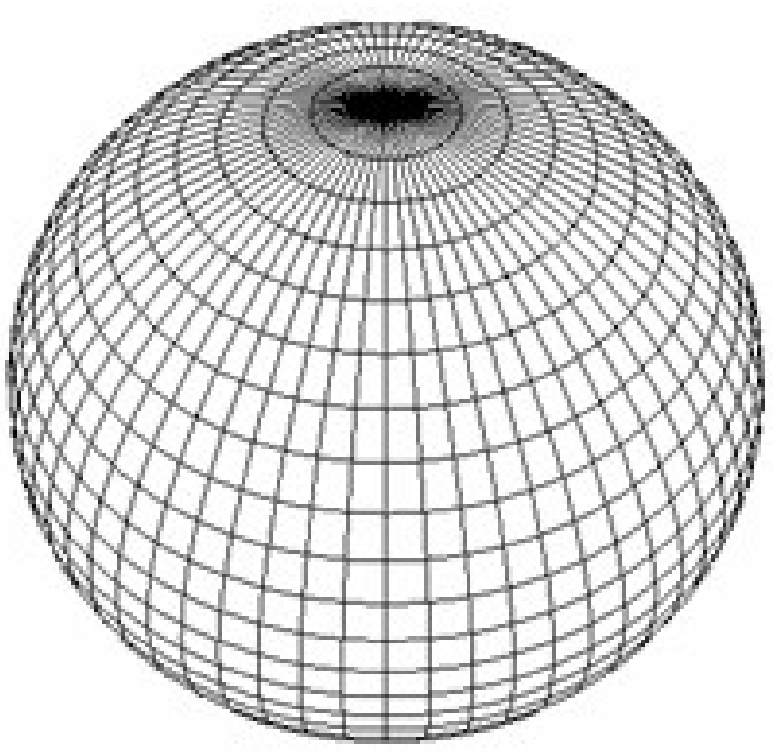


Upwind first order scheme (Godunov, 1952)



- CFL requirement, for an advection velocity Umax : Umax.(dt/dx) = cte , with cte ~ 0(1)

Longitudinal polar filter



- A lon-lat grid implies that the meshes tighten dramatically as the pole is approached.
- CFL conditions there would dictate using an extremely small time step for the time marching scheme.

- Longitudinal (Fourier) filtering, removing high spatial frequencies, is used to enforce that resolved features are at the level of those at ~60°

Energy spectra and lateral dissipation

- Observations (Nastrom & Gage 1985, Lindborg 1999) collected over length scales from a few to thousands of km display a characteristic energy cascade (from Skamarock, 2004).

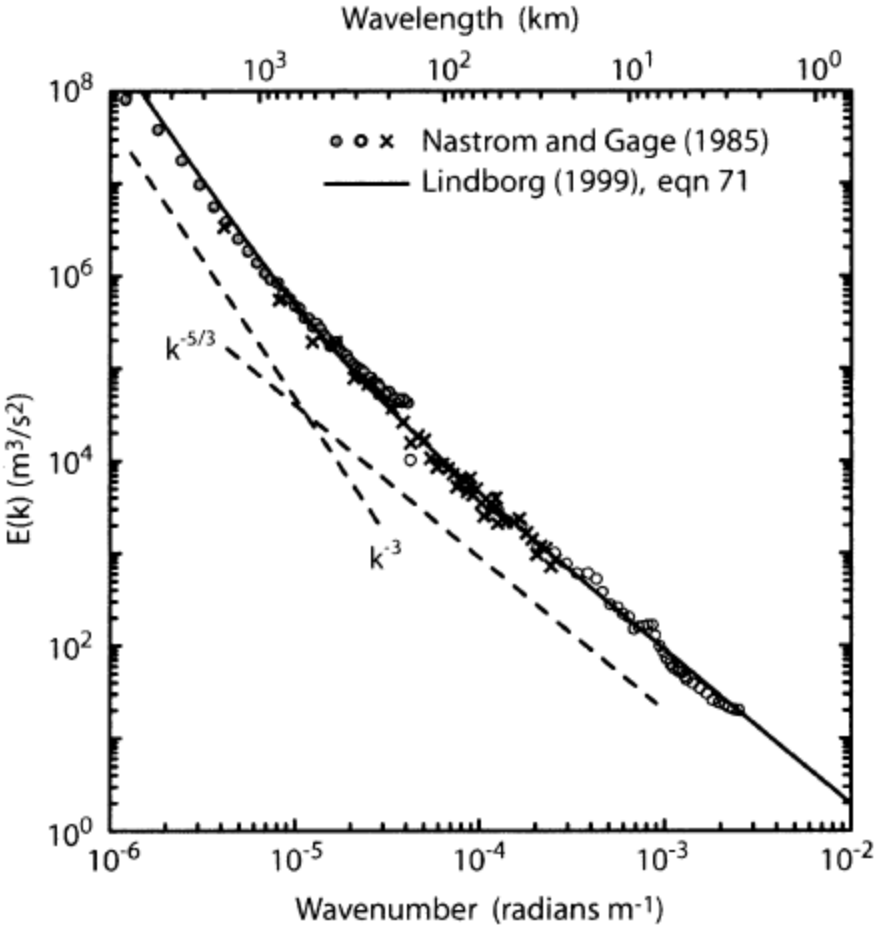
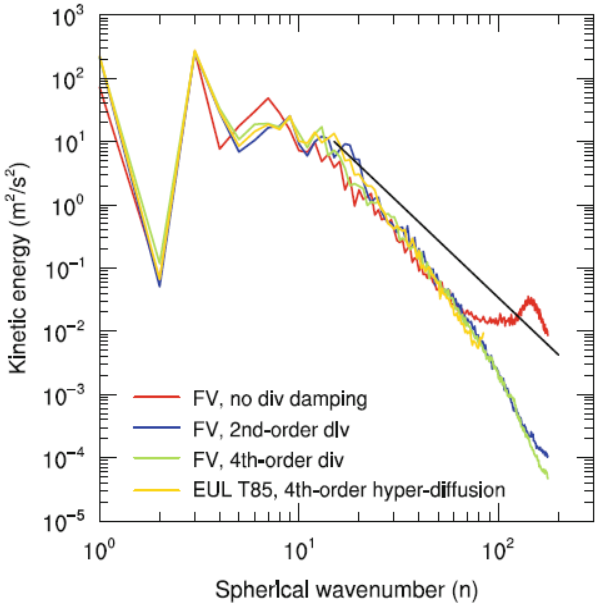


FIG. 1. Nastrom and Gage (1985) spectrum derived from the GASP aircraft observations (symbols) and the Lindborg (1999) functional fit to the MOZAIC aircraft observations.

- In order to fulfil the observed energy cascade from resolved scales to unresolved scales in GCMs, a dissipation term is added:

$$Dissip(\psi) = \frac{(-1)^{q+1}}{\tau} \nabla^{2q} \psi$$

Lateral dissipation in GCMs as a tool to pin the energy cascade



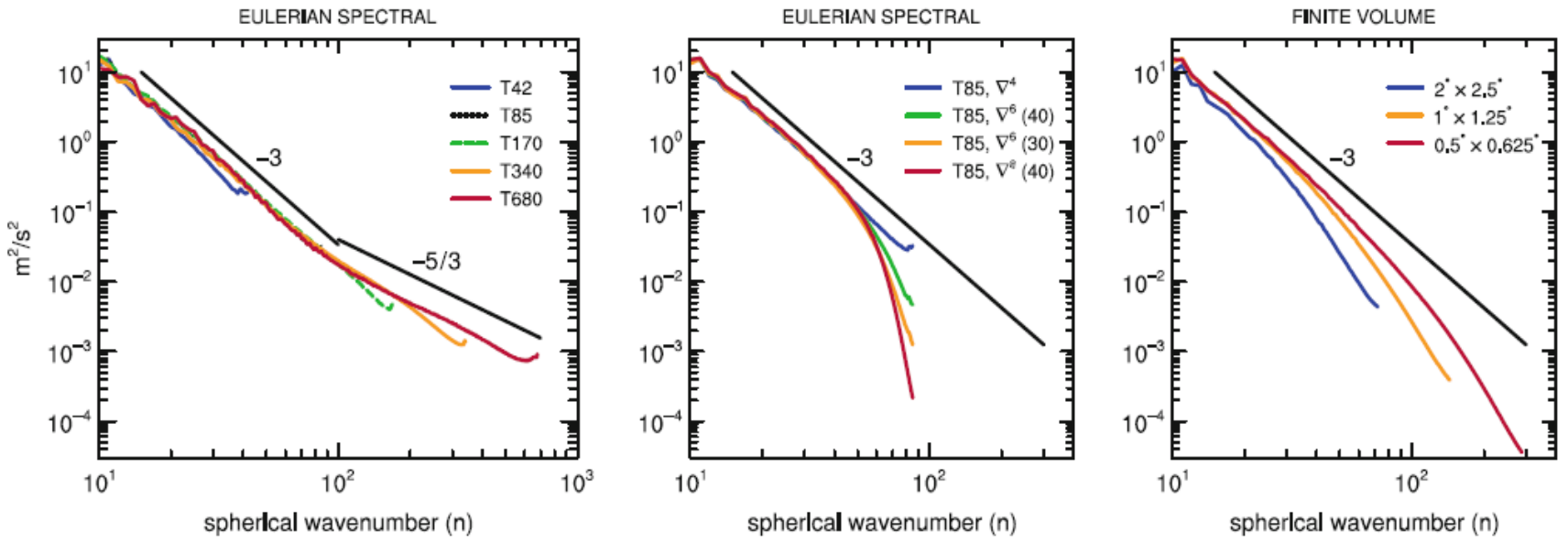


Fig. 13.4 250 hPa kinetic energy spectra as a function of the spherical wavenumber (n) in aqua-planet simulations from (*left*) CAM Eulerian spectral dynamical core with ∇^4 diffusion for different resolutions, (*center*) T85L26 Eulerian spectral dynamical with ∇^4, ∇^6 and ∇^8 diffusion, and (*right*) CAM Finite Volume (FV) dynamical core for different *lat* × *lon* resolutions in degrees and 26 levels

Illustrative example of dissipation

- Simple 1D diffusion equation toy model:

$$\frac{dA}{dt} = \nu \frac{\partial^2 A}{\partial x^2}$$

- Von Neumann (Fourier mode) analysis

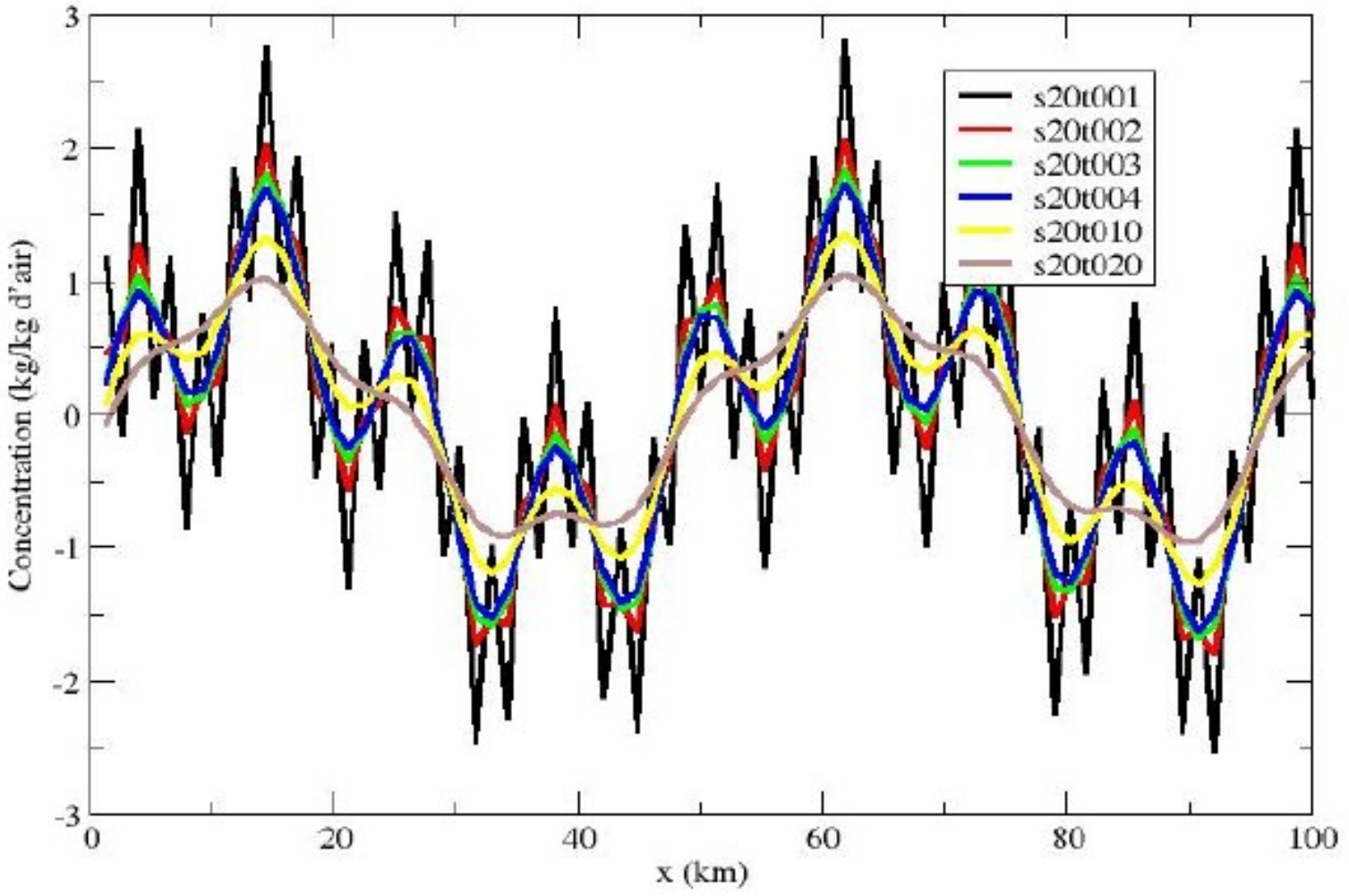
$$A_k(t) = a_k(t).sin(kx)$$

- Explicit Euler time marching (with stability condition!):

$$a_k^{n+1} = \left(1 - \nu k^2 \Delta t\right) a_k^n$$

Note that mode damping is stronger for large k

Illustrative example of dissipation



- Temporal evolution, from an initial condition consisting of 2 sine modes and an extreme (2 grid points wavelength) “numerical mode”

Controlling dissipation in LMDZ

- Parameters in file gcm.def:

dissip_period: Apply dissipation every dissip_period dynamical steps (or specify 0 to let model pick an appropriate value)

nitergdiv: number of iterations on velocity dissipation operator grad.div

nitergrot: number of iterations on velocity dissipation operator grad.rot

niterh: number of iterations on temperature dissipation operator div.grad

Usual values: nitergdiv=1, nitergrot=2, niterh=2

tetagdiv: dissipation time scale (s) for smallest wavelength for u,v (grad.div component)

tetagrot: dissipation time scale (s) for smallest wavelength for u,v (grad.rot component)

tetatemp: dissipation time scale (s) for smallest wavelength for potential temperature (div.grad)

values depend on horizontal resolution

Controlling dissipation in LMDZ

- Parameters in file gcm.def:

  tetagdiv: dissipation time scale (s) for smallest wavelength for u,v (grad.div component)

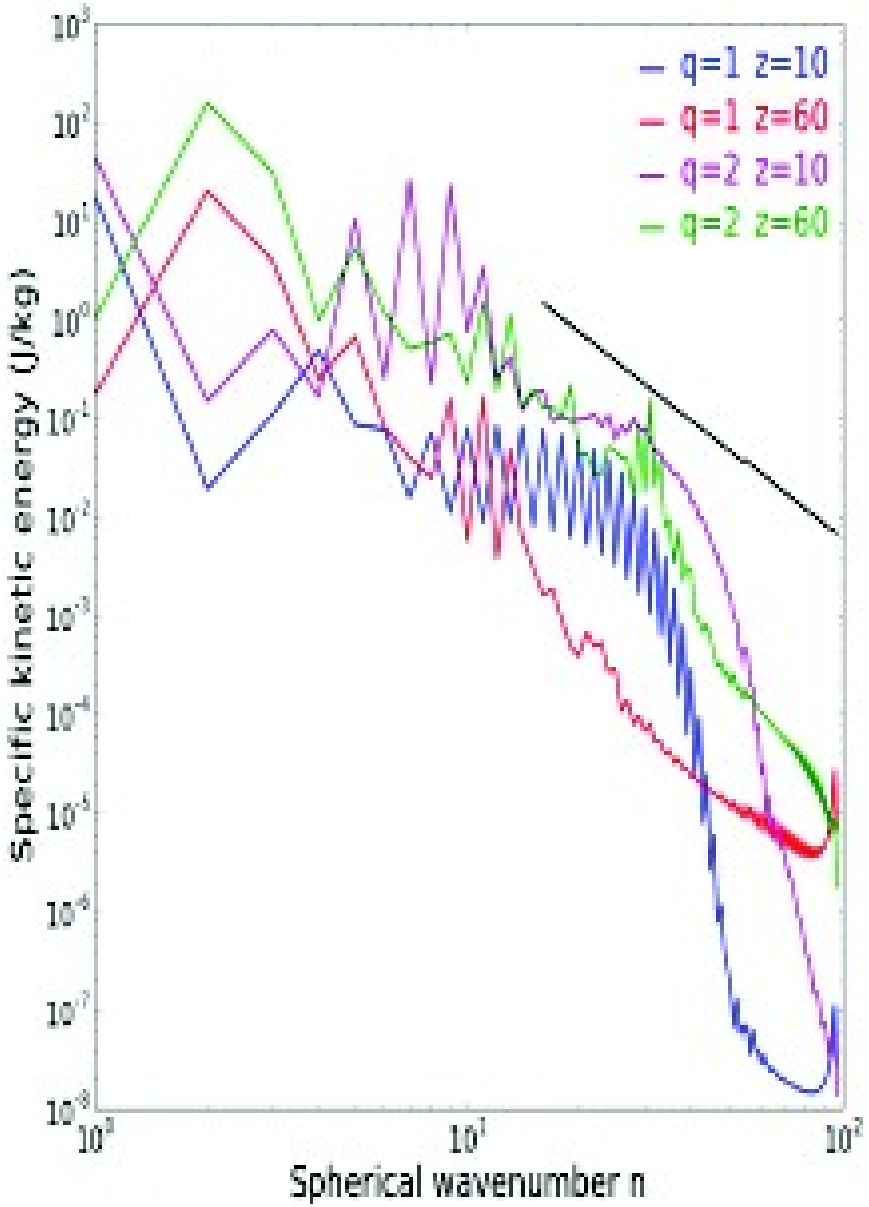
  tetagrot: dissipation time scale (s) for smallest wavelength for u,v (grad.rot component)

  tetatemp: dissipation time scale (s) for smallest wavelength for potential temperature (div.grad)

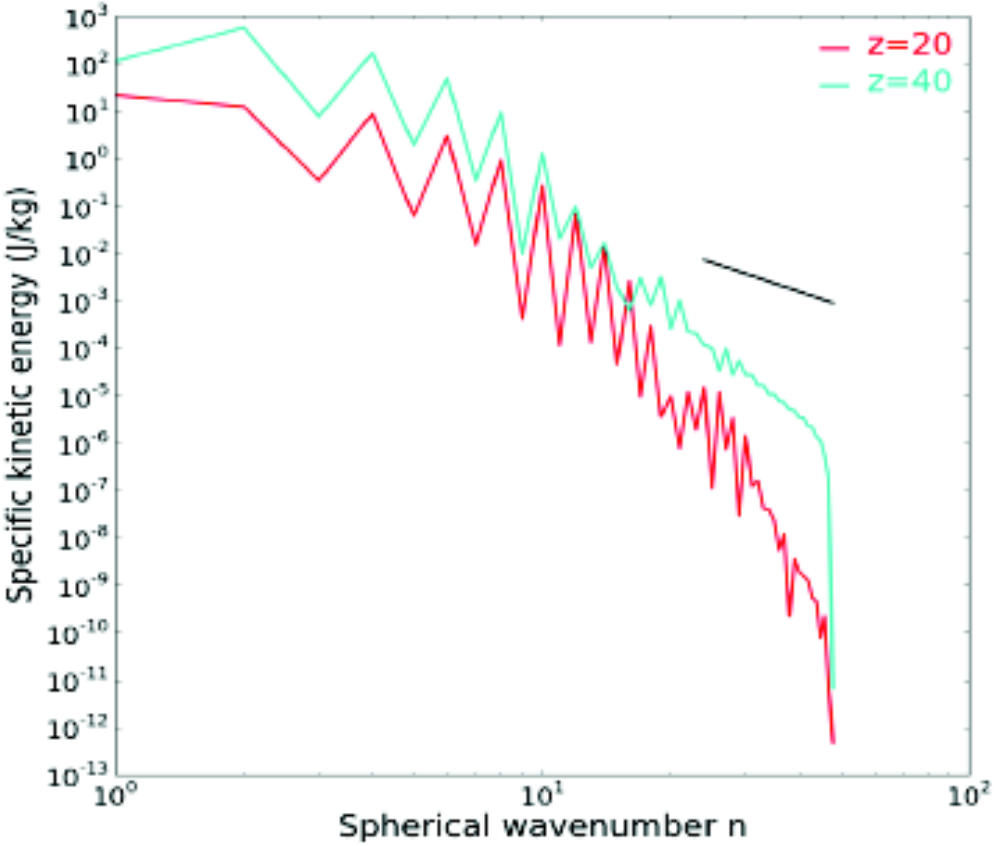
  optimal teta values depend on horizontal resolution

- Moreover there is a multiplicative factor for the dissipation coefficient, which increases with model levels (see dyn3d_common/inidissip.F90), which can be controlled by flag “vert_prof_dissip”

Some spectra (from LMDZ/Saturn)

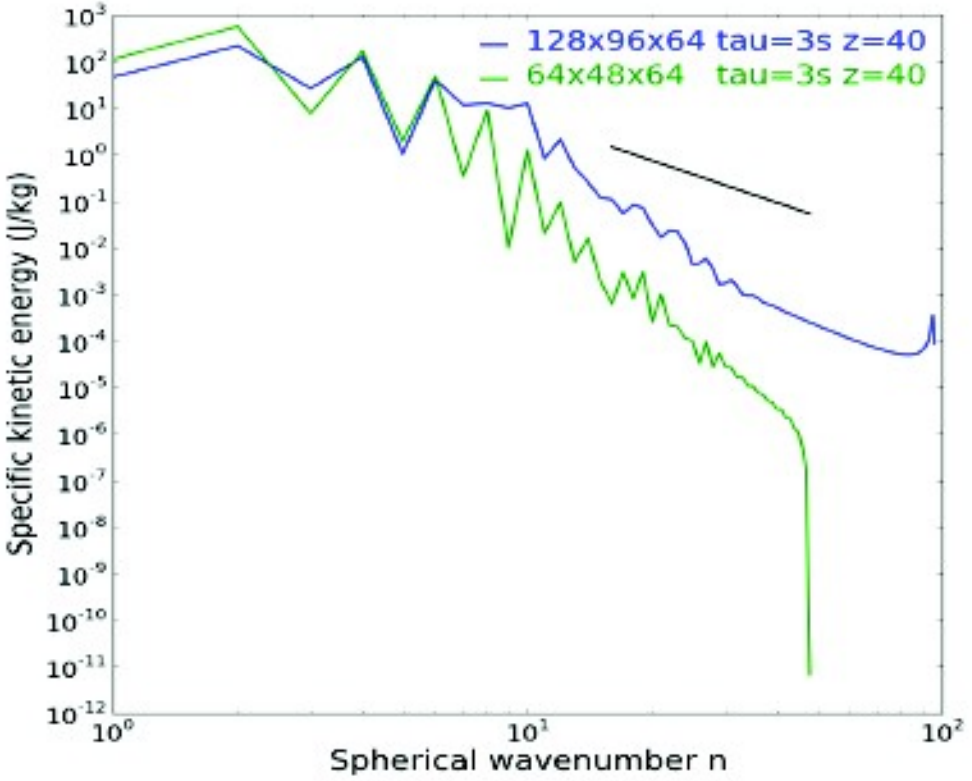


Energy spectra for simulations with 1 and 2 diffusion order.

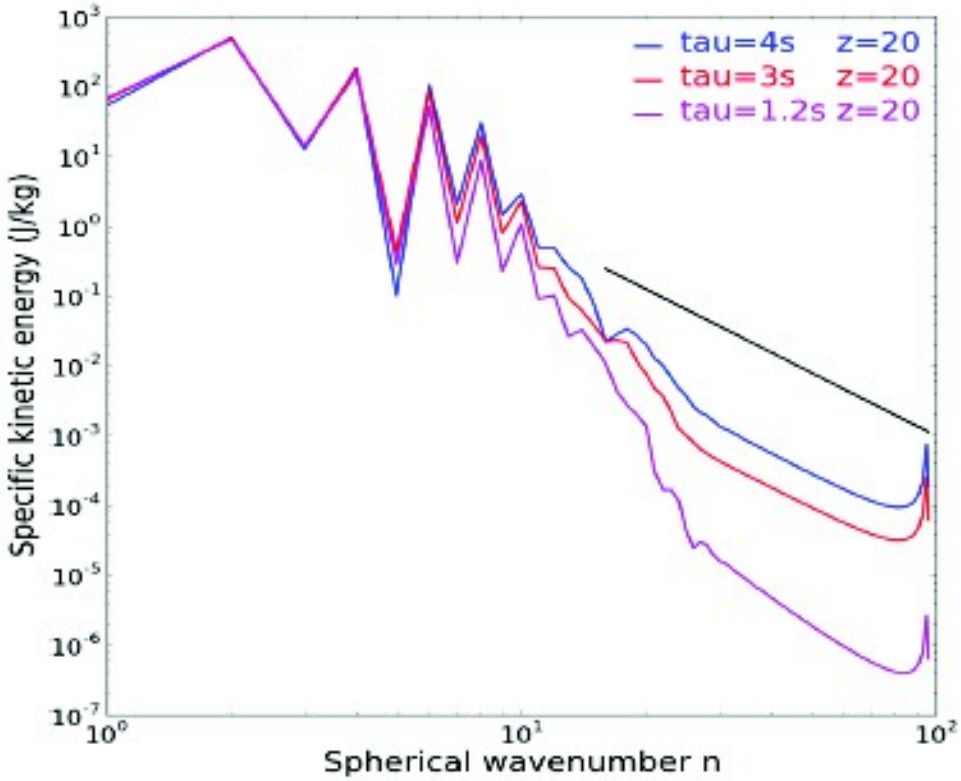


Energy spectra at two different altitude levels z in the northern summer on a 64x48x64 grid. τ value is 3s. Spectra ranges up to $n = 48$.

Some spectra (from LMDZ/Saturn)



Energy spectra at the same altitude and time with $\tau = 3s$ on a 64X48X64 and 128x96x64 grids.



Spectra of simulations on a 128x96x64 grid with different τ (1s, 3s, 4s).

The sponge layer

- In addition to lateral dissipation, it is necessary to damp vertically propagating waves (non-physically reflected downward from model top).
- The sponge layer is limited to topmost layers (usually 4) and added during the dissipation step.
- Sponge modes and parameters (gcm.def):

  iflag_top_bound: 0 for no sponge, 1 for sponge over 4 topmost layers, 2 for sponge from top to 100 times topmost layer pressure

  mode_top_bound: 0 for no relaxation, 1 to relax u,v to zero, 2 to relax u,v to their zonal mean, 3 to relax u,v and potential temperature to their zonal mean.

  tau_top_bound: inverse of characteristic time scale at the topmost layer (halved at each successive descending layer)

Some examples of typical values for various resolutions

From test simulations using various horizontal resolutions (Foujols et al.)
http://forge.ipsl.jussieu.fr/igcmg/wiki/ResolutionIPSLCM4_v2

Tableau récapitulatif des différences entre résolutions et avec le couplé standard

	R44	**R97**	**R99**	**R149**	**R1414**	**R1914**
Résolutions						
atmosphère	44x43x19	96x71x19	96x95x19 (*)	144x96x19	144x142x19	192x142x19 (*)
océan	ORCA2	ORCA2	ORCA2	ORCA2	ORCA2	ORCA2
Etat initial						
océan	2L20 - 20 ans 30 12 1859	2L20 -30 12 1859	2L20 -30 12 1859	2L20 - 30 12 1859	2L20 - 30 12 1859	2L20 - 30 12 1859
atmosphère	10 ans forcé AMIP 1980-1989 F44A	10 ans forcé AMIP 1980-1989 F97A	10 ans forcé AMIP 1980-1989 F99A	10 ans forcé AMIP 1980-1989 F149A	10 ans forcé AMIP 1980-1989 F1414A	10 ans forcé AMIP 1980-1989 F1914A
coupleur	défaut	défaut	défaut	défaut	défaut	défaut
atmosphère (idem avec dynzon)	10 ans forcé AMIP 1980-1989 F44AZ	10 ans forcé AMIP 1980-1989 F97AZ	10 ans forcé AMIP 1980-1989 F99AZ	10 ans forcé AMIP 1980-1989 F149AZ	10 ans forcé AMIP 1980-1989 F1414AZ	10 ans forcé AMIP 1980-1989 F1914AZ
coupleur	défaut	défaut	défaut	défaut	défaut	défaut
Paramètres						
day_step	240	480	480 (=96x71)	720	720	960 (**)
nitergdiv	1					
iphysiq	5	10	10 (=)	15	15	20 (**)
tetagdiv	36000.	5400. (**)	5400. (=)	5400. (**)	3600.	3600. (=144x142)
tetagrot	18000.	7200. (**)	7200. (=)	7200. (**)	5400.	5400. (=)
tetatemp	18000.	7200. (**)	7200. (=)	7200. (**)	5400.	5400. (=)
cvl_corr	1.0	1.0	1.0 (=)	1.0	1.0 (**)	1.0 (=)

Rules of thumb for run.def parameters

- Time steps in LMDZ:

  dynamical time steps: dtvr = daysec / day_step

  physics time step: dtphys = iphysiq * dtvr

  dissipation time step: dtdiss = dissip_period * dtvr

  tracer advection time step: dtvrtrac = iapp_trac * dtvr

- Constraints to be aware of:

  dtvr limited by CFL for waves: Cmax.dt < min(dx,dy)

  dtrtrac limited by advection CFL: Umax.dt < min(dx,dy)

  iphysiq, dtvrtrac, dissip_period should be multiples of iperiod

Rules of thumb for run.def parameters

- Constraints to be aware of (continued):

  dissipation time step should be much smaller than dissipation timescales:

  dtdiss << tetatgdiv, tetagrot, tetatemp

- Changing time step with resolution on a regular grid:

  day_step(max(iim,jjm)=N) ~ day_step(max(iim,jjm)=M) * M/N

- Time step for a zoomed simulation, compared to regular grid:

  day_step(zoom) ~ day_step(regular)*max(grossismx,grossismy)