MPAS Basics

Prof. Dr. Pedro S. Peixoto
&
Dr. Felipe A. V. B. Alves

Applied Mathematics
Instituto de Matemática e Estatística
Universidade de São Paulo

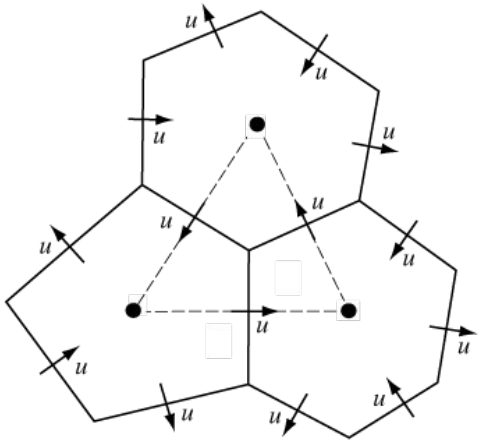
Based on MPAS tutorials by NCAR et al

Oct 2023

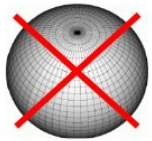
Key references

MPAS
Model for Prediction Across Scales

- **Horizontal dynamics** (discussed in previous classes): Ringler, T., J. Thuburn, J. Klemp and W. Skamarock, 2010: A unified approach to energy conservation and potential vorticity dynamics on arbitrarily structured C-grids, Journal of Computational Physics.
- **Time integration** (discussed in previous classes): Wicker, L.J. and Skamarock, W.C., 2002. Time-splitting methods for elastic models using forward time schemes. Monthly weather review, 130(8), pp.2088-2097.
- **Advection** (would require more time in classes to be discussed): Skamarock, W.C. and Gassmann, A., 2011. Conservative transport schemes for spherical geodesic grids: High-order flux operators for ODE-based time integration. Monthly Weather Review, 139(9), pp.2962-2975.
- **Vertical** (will not be discussed in this course): Klemp, J.B., Skamarock, W.C. and Dudhia, J., 2007. Conservative split-explicit time integration methods for the compressible nonhydrostatic equations. Monthly Weather Review, 135(8), pp.2897-2913.
- **3D Model** (This class): Skamarock, W.C., Klemp, J.B., Duda, M.G., Fowler, L.D., Park, S.H. and Ringler, T.D., 2012. A multiscale nonhydrostatic atmospheric model using centroidal Voronoi tesselations and C-grid staggering. Monthly Weather Review, 140(9), pp.3090-3105.



MPAS Development



2005 Global lat-lon (WRF) problematic.

2006 Triangles - problems with divergence.

2007 Yin-Yang: local conservation past 1st-order accuracy?
Cubed-sphere: Corner point problems?

2008 Hex grid: C-grid problem solved for perfect hex mesh.
C-grid problem solved for general Voronoi mesh.

2009 Unstructured-mesh MPAS SW eqns. solver.
MPAS hydrostatic eqns. solver.

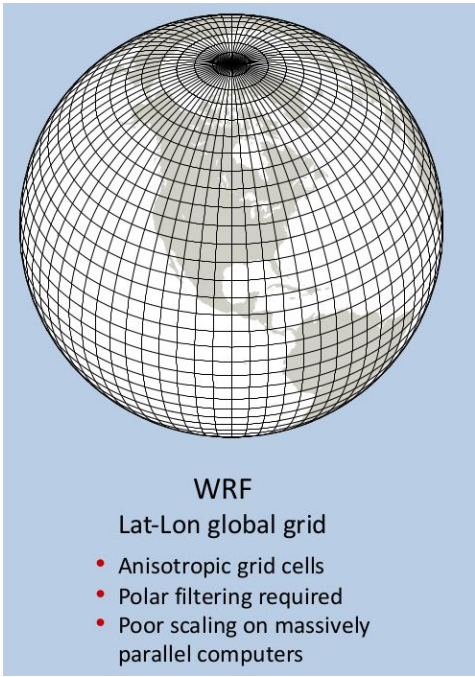
2010 MPAS nonhydrostatic eqns. solver.
Hydrostatic MPAS in CAM/CESM.

2011 WRF-NRCM physics in MPAS.

2012 DART data assimilation.

2013 3km global mesh tests on Yellowstone.
MPAS V1.0 release (atmosphere, ocean)
MPAS-Atmosphere real-time TC forecast testing.

2014 Scale-aware physics testing begins.

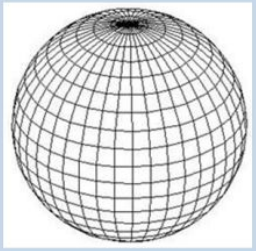


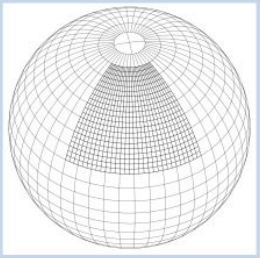
WRF
Lat-Lon global grid

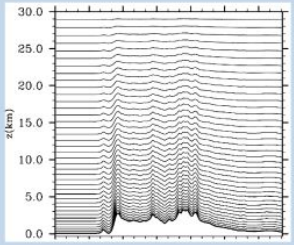
- Anisotropic grid cells
- Polar filtering required
- Poor scaling on massively parallel computers

From: https://github.com/pedrospeixoto/MPAS-BR/blob/master/docs/mpas_br/mpas_refs/01.MPAS_overview.pdf

MPAS vs WRF





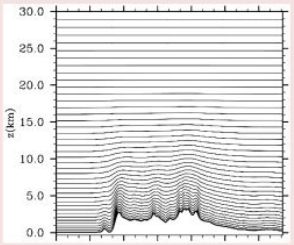


WRF Characteristics

- Lat-Lon global grid
  - Anisotropic grid cells
  - Polar filtering required
  - Poor scaling on massively parallel computers
- Grid refinement through domain nesting
  - Flow distortions at nest boundaries
- Pressure-based terrain-following sigma vertical coordinate

MPAS Characteristics

- Unstructured Voronoi (hexagonal) grid
  - Good scaling on massively parallel computers
  - No pole problems
- Smooth grid refinement on a conformal mesh
  - Increased accuracy and flexibility in varying resolution
- Height-based hybrid smoothed terrain-following vertical coordinate
  - Improved numerical accuracy

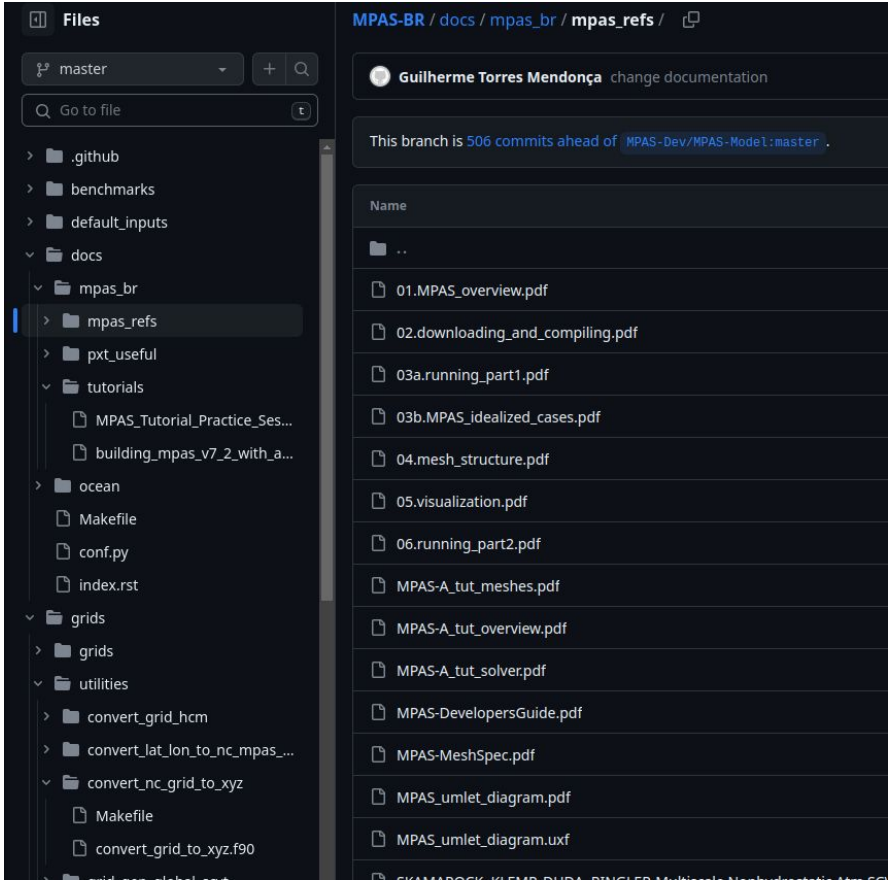


Documentation

MPAS-BR:

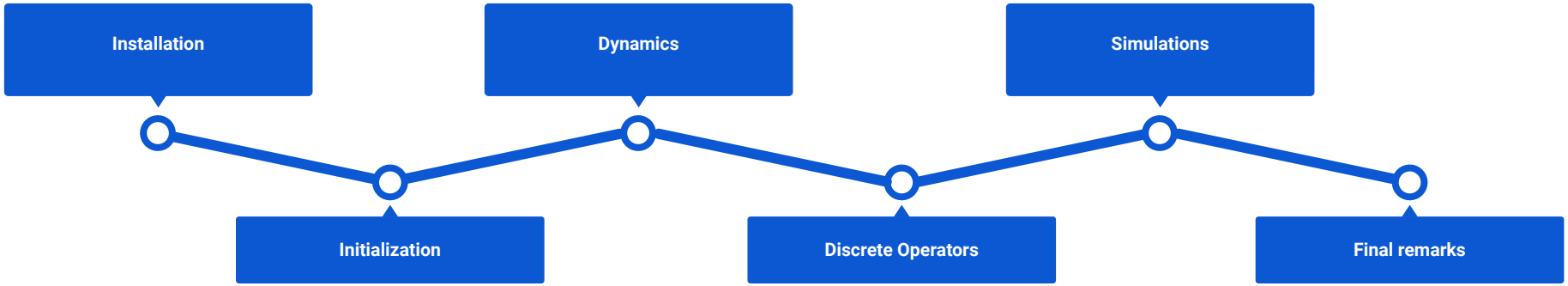
https://github.com/pedrospeixoto/MPAS-BR

Research code for MPAS Brazilian MONAN initiative.



Overview

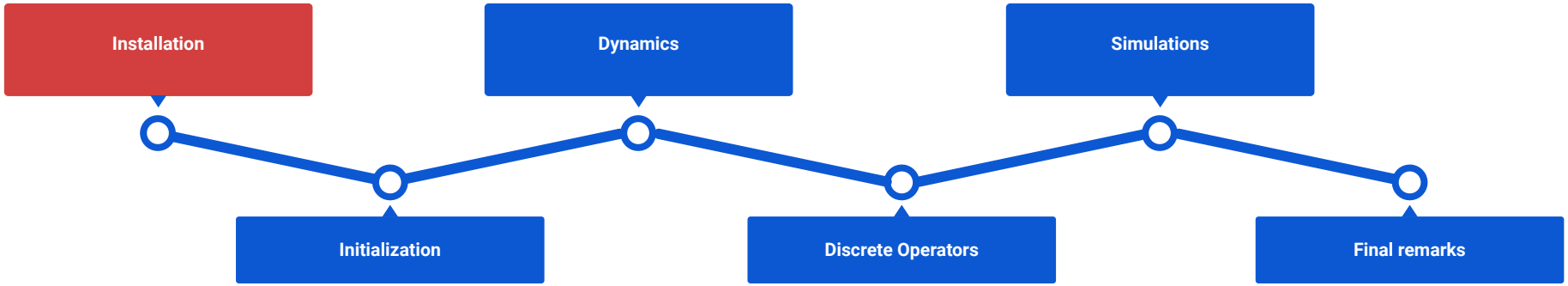
Today's class:



Disclaimer: Only idealized cases will be discussed (no real data).

Overview

Today's class:



Disclaimer: Only idealized cases will be discussed (no real data).

Installation

- MPAS source code can be downloaded directly at:
  - https://github.com/pedrospeixoto/MPAS-BR/archive/refs/heads/master.zip

- Alternatively, the repository can be cloned using git in a terminal (you may wish to FORK to your account):
  - `git clone https://github.com/pedrospeixoto/MPAS-BR.git`

- To compile the code fortran mpi compilers are needed. Working options are described in the file "Makefile". The most straightforward option is to use gfortran and OpenMPI, which can be installed on a Debian system with the terminal command:
  - `sudo apt install gfortran libopenmpi-dev`

- MPAS uses the NetCDF file format for its input and output. Software libraries to work with those files are needed. MPAS accepts a few different libraries for that purpose. We've found the most robust option to be using the PNetCDF library together with new built-in SMIOL code. PNetCDF can be installed on a Debian system with the terminal command:
  - `sudo apt install pnetcdf-bin libpnetcdf-dev`

SMIOL comes bundled with MPAS source code and needs not to be installed.

Installation

- Alternatively, if you use a conda environment (for instance, if you use “maps-tools” environment), you can install gfortran, mpi, netcdf and pnetcdf with:
  - `conda install -c conda-forge gfortran`
  - `conda install -c conda-forge openmpi`
  - `conda install -c conda-forge netcdf4`
  - `conda install -c "e3sm/label/compass" libpnetcdf`

These installations should be make based on your system!!!
Please check what is best strategy with your local IT support.

Compilation

- The MPAS Atmosphere model is divided into two components (also called cores): the ‘init_atmosphere’ core, used just for initialization of a problem, and the ‘atmosphere’ core, which actually runs the model.

- To compile the ‘init_atmosphere’ core, in a terminal inside the source code directory, issue the following command:
  - `make gfortran CORE=init_atmosphere PNETCDF=/usr`

- The PNETCDF environment variable should point to the directory where PNetCDF was installed, which, on Debian systems, is “/usr” by default. If PNetCDF was installed by other means, the env var should be changed accordingly. If compilation works, the following banner should be visible:

```
*********************************************************************************
MPAS was built with default double-precision reals.
Debugging is off.
Parallel version is on.
Papi libraries are off.
TAU Hooks are off.
MPAS was built without OpenMP support.
MPAS was built without OpenMP-offload GPU support.
MPAS was built without OpenACC accelerator support.
Position-dependent code was generated.
MPAS was built with .F files.
The native timer interface is being used
Using the SMIOL library.
*********************************************************************************
```

Compilation

- Now, to compile the ‘atmosphere’ core, in a terminal inside the source code directory, issue the following command:
  - `make gfortran CORE=atmosphere AUTOCLEAN=true PNETCDF=/usr`

- If compilation was successful, the same banner as before should be displayed. After compiling both cores, the contents of the directory should be:

```
atmosphere_model*       Makefile                        SOILPARM.TBL@
azure-pipelines.yml     met_data/                       src/
benchmarks/             MPAS-BR-contributions.md        stream_list.atmosphere.diagnostics
build_tables*           MPAS-BR-contributions-PXT.md    stream_list.atmosphere.output
CAM_ABS_DATA.DBL@       namelist.atmosphere             stream_list.atmosphere.surface
CAM_AEROPT_DATA.DBL@    namelist.init_atmosphere        stream_list.atmosphere.validate
default_inputs/         namelist.sw                     streams.atmosphere
doc/                    OZONE_DAT.TBL@                  streams.init_atmosphere
docs/                   OZONE_LAT.TBL@                  streams.sw
GENPARM.TBL@            OZONE_PLEV.TBL@                 target_domain.Catarina
grids/                  post_proc/                      target_domain.Petropolis
init_atmosphere_model*  README.md                       testing_and_setup/
INSTALL                 RRTMG_LW_DATA@                  validate/
LANDUSE.TBL@            RRTMG_LW_DATA.DBL@              variables_list
LICENSE                 RRTMG_SW_DATA@                  VEGPARM.TBL@
local_software/         RRTMG_SW_DATA.DBL@
```

Grid Recall

Load environment

- `$conda activate maps-tools`

Create an example grid with Jigsaw

- `(mpas-tools) /…/MPAS-BR/grids/utilities/jigsaw$ python3 spherical_grid.py -g unif -o unif240km -r 240 -l 240`

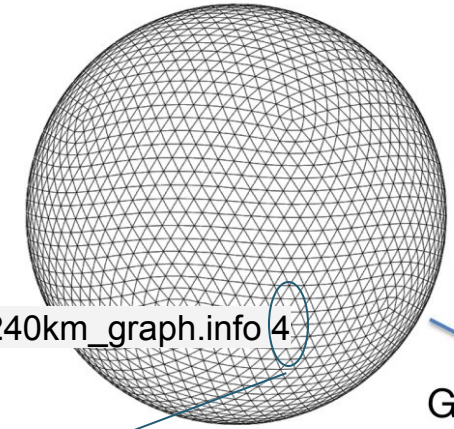
Check the grid created (resolutions) - Plot.

- `(mpas-tools) /…/MPAS-BR/post_proc/py/grid_maps$ python3 mpas_plot_grid.py -g ../../../grids/utilities/jigsaw/unif240km/unif240km_mpas.nc -o ../../../grids/utilities/jigsaw/unif240km/unif240km_mpas.jpg`

Grid partitions

Grid partitions for parallel processing are done using Metis

```
sudo apt-get install metis
```



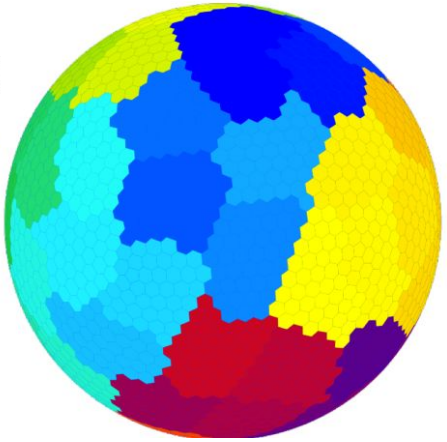
- The *dual* mesh of a Voronoi tessellation is a Delaunay triangulation – essentially the connectivity graph of the cells
- Parallel decomposition of an MPAS mesh then becomes a graph partitioning problem: ***equally distribute nodes among partitions (give each process equal work) while minimizing the edge cut (minimizing parallel communication)***

Create partitions. Ex:

```
gpmetis -minconn -contig -niter=200 unif240km_graph.info 4
```

N: number of partitions

Graph partitioning

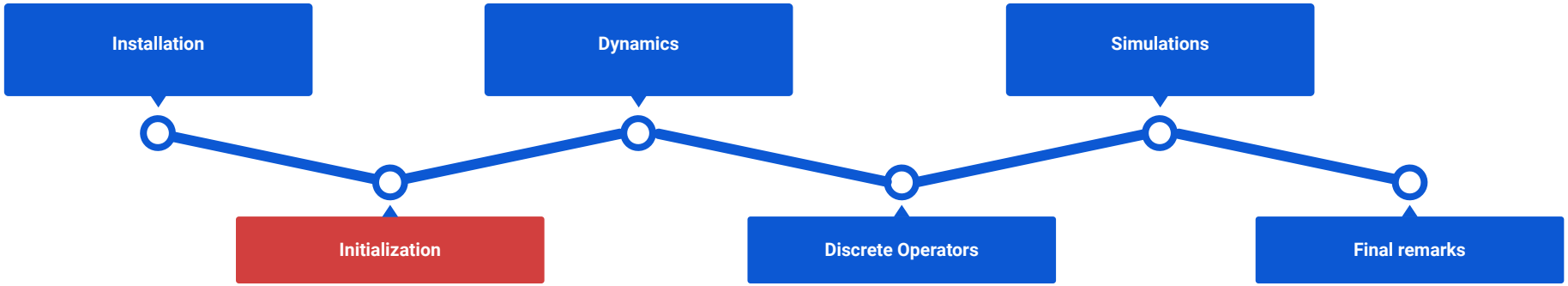


We use the Metis package for parallel graph decomposition

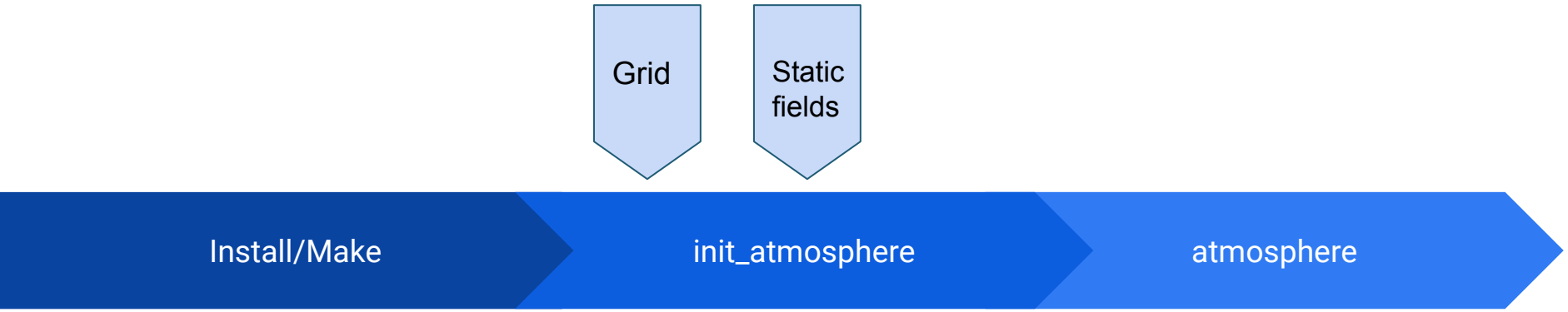
- Currently done as a pre-processing step, but could be done “on-line”
- Fortunately, Metis runs quickly, and a partitioning into *n* pieces only needs to be done once for a given mesh

Overview

Today's class:



Summary

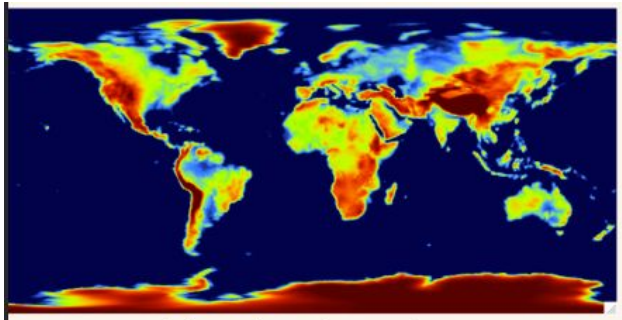


- Clone MPAS repository
- Install gfortran, libopenmpi, libpnetcdf
- Make

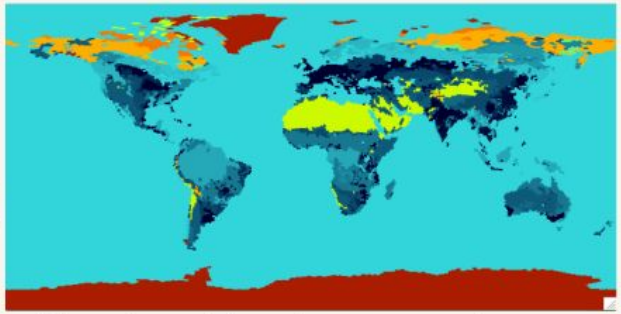
- Configure an initial condition (idealized or real data)
- Initialize the atmosphere model
- Set grid and static fields

- Run the atmospheric model based on the ini_atmosphere output
- Post process

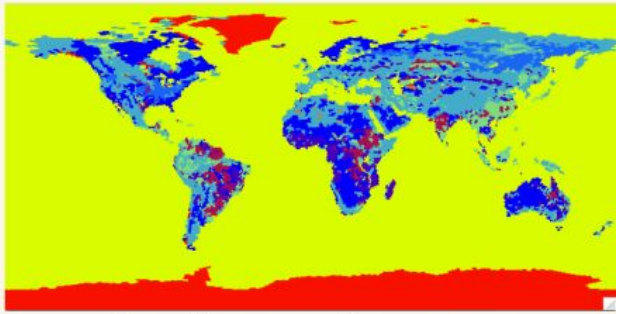
Static fields (examples)



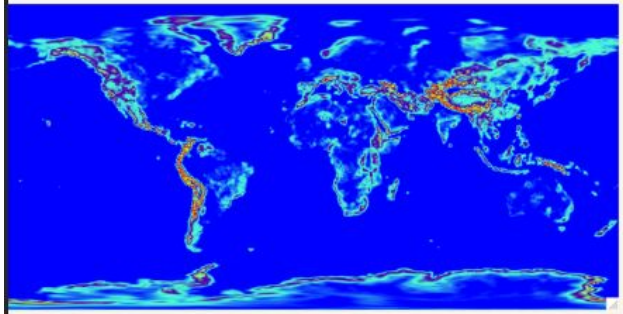
Terrain elevation



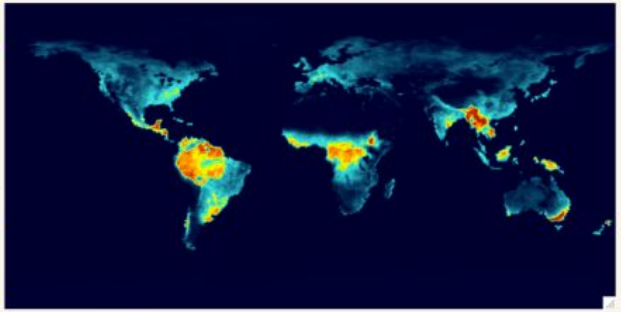
Dominant land cover category



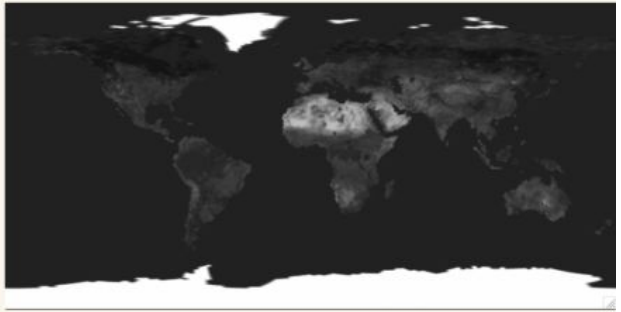
Dominant soil category



Sub-grid-scale terrain variance



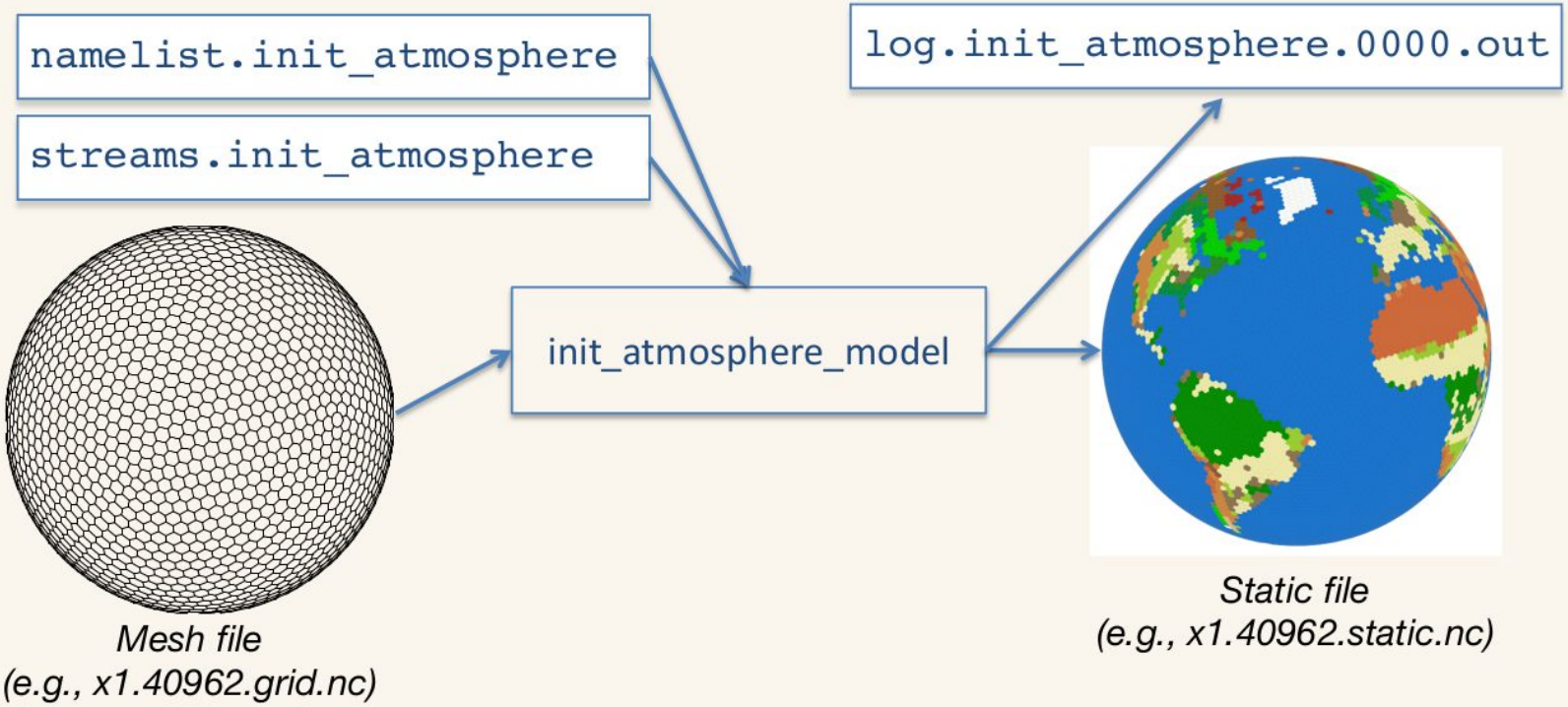
Climatological monthly vegetation fraction



Climatological monthly surface albedo

Pre-processing

Input and output files when producing a “static” file:

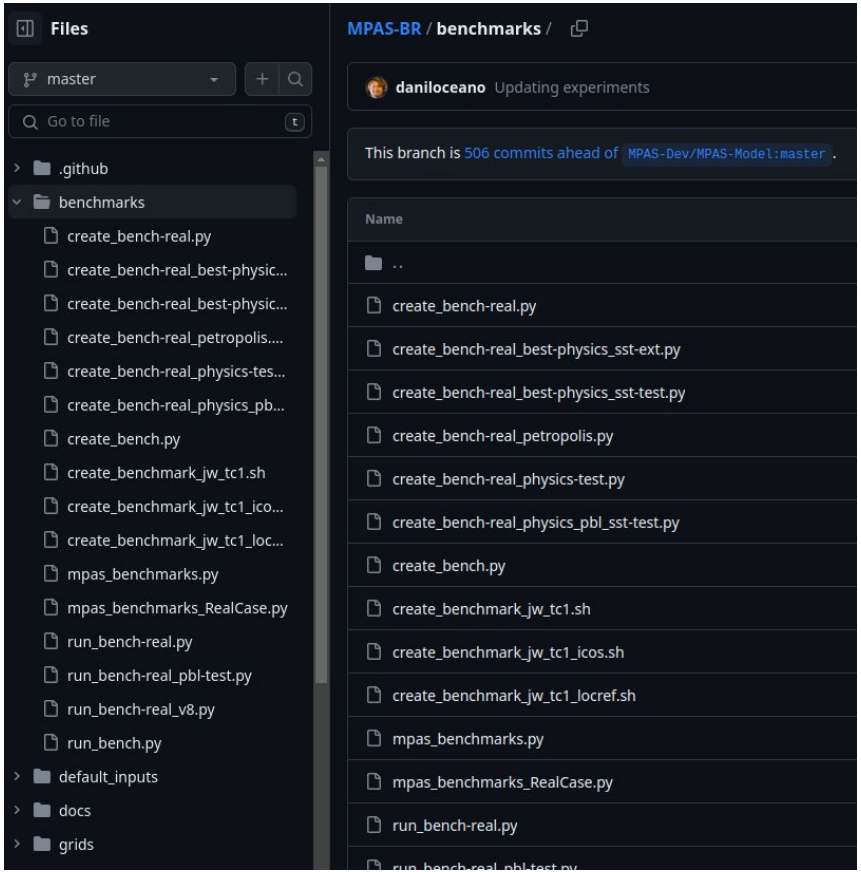


Config init_atmosphere

You can hand tune the namelists/streams and streams of MPAS ....

or

… we can use Python based scripts to create necessary files to initialize MPAS-A.



Idealized test case

Namelist

benchmarks > monan-class-example > jw_baroclinic_wave > namelist.init_atmosphere

```
&nhyd_model
   config_start_time = '0000-01-01_00:00:00'
   config_init_case = 2
/

&dimensions
   config_nvertlevels = 26
/

&decomposition
   config_block_decomp_file_prefix = 'x1.40962.graph.info.part.'
/
```

src/core_init_atmosphere/mpas_init_atm_cases.F

```
if ((config_init_case == 1) .or. (config_init_case == 2) .or. (config_init_case == 3)) then

   call mpas_log_write(' Jablonowski and Williamson baroclinic wave test case ')
   if (config_init_case == 1) call mpas_log_write(' no initial perturbation ')
   if (config_init_case == 2) call mpas_log_write(' initial perturbation included ')
   if (config_init_case == 3) call mpas_log_write(' normal-mode perturbation included ')
   block_ptr => domain % blocklist
```

Idealized test case

Streams
(related files)

Input grid file

Output initialization file
For MPAS run

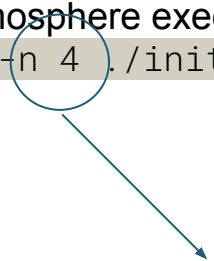
benchmarks > monan-class-example > jw_baroclinic_wave > streams.init_atmosphere

```
<streams>
<immutable_stream name="input"
                  type="input"
                  filename_template="x1.40962.grid.nc"
                  input_interval="initial_only" />

<immutable_stream name="output"
                  type="output"
                  filename_template="x1.40962.init.nc"
                  packages="initial_conds"
                  output_interval="initial_only" />

<immutable_stream name="surface"
                  type="output"
                  filename_template="sfc_not_needed_for_jw"
                  filename_interval="none"
                  packages="sfc_update"
                  output_interval="86400" />

<immutable_stream name="lbc"
                  type="output"
                  filename_template="lbc_not_needed_for_jw"
                  filename_interval="output_interval"
                  packages="lbcs"
                  output_interval="none" />

</streams>
```

Run init_amosphere

Run the init_atmosphere executable:

- `mpirun -n 4 ./init_atmosphere_model`



Must match the number of partitions set with metis!

Netcdf output (mpas input file)

Use either `ncdump -h` or `ncmpidump -h` to quickly inspect the file created:

```
(mpas-tools) pedrosp@ppeixoto:~/ppstorage/Simulations/MPAS-BR/benchmarks/monan-class-example/jw_baroclinic_wave$ ncdump x1.40962.init.nc -h
netcdf x1.40962.init {
dimensions:
        nVertLevels = 26 ;
        nCells = 40962 ;
        Time = UNLIMITED ; // (1 currently)
        StrLen = 64 ;
        nEdges = 122880 ;
        nVertices = 81920 ;
        TWO = 2 ;
        maxEdges = 10 ;
        maxEdges2 = 20 ;
        vertexDegree = 3 ;
        R3 = 3 ;
        nMonths = 12 ;
        FIFTEEN = 15 ;
        nVertLevelsP1 = 27 ;
        nSoilLevels = 4 ;
variables:
        double qv(Time, nCells, nVertLevels) ;
                qv:long_name = "Water vapor mixing ratio" ;
                qv:units = "kg kg^{-1}" ;
        double qc(Time, nCells, nVertLevels) ;
                qc:long_name = "Cloud water mixing ratio" ;
                qc:units = "kg kg^{-1}" ;
        double qr(Time, nCells, nVertLevels) ;
```

Log ini_atmosphere

benchmarks > monan-class-example > jw_baroclinic_wave > log.init_atmosphere.0000.out

```
 1  ------------------------------------------------------------------------
 2  Beginning MPAS-init_atmosphere Output Log File for task       0 of       1
 3      Opened at 2023/10/30 10:29:56
 4  ------------------------------------------------------------------------
 5
 6
 7  MPAS Init-Atmosphere Version 8.0.1
 8
 9
10  Output from 'git describe --dirty': v7.3-1045-g31c4868f
11
12  Compile-time options:
13    Build target: gfortran
14    OpenMP support: no
15    OpenACC support: no
16    Default real precision: double
17    Compiler flags: optimize
18    I/O layer: SMIOL
19
20  Run-time settings:
21    MPI task count: 1
22
23  Reading namelist from file namelist.init_atmosphere
24  *** Encountered an issue while attempting to read nam
25      The following values will be used for variables i
26
27          config_geog_data_path = /glade/work/wrfhelp/W
28          config_met_prefix = CFSR
29          config_sfc_prefix = SST
30          config_fg_interval = 86400
```

```
   timer_name                                 total       calls        mi
 1 total time                                 6.41626         1        6.4
 2  initialize                                0.78789         1        0.7

----------------------------------------
Total log messages printed:
   Output messages =                  233
   Warning messages =                  11
   Error messages =                     0
   Critical error messages =            0
----------------------------------------
Logging complete.  Closing file at 2023/10/30 10:30:02
```

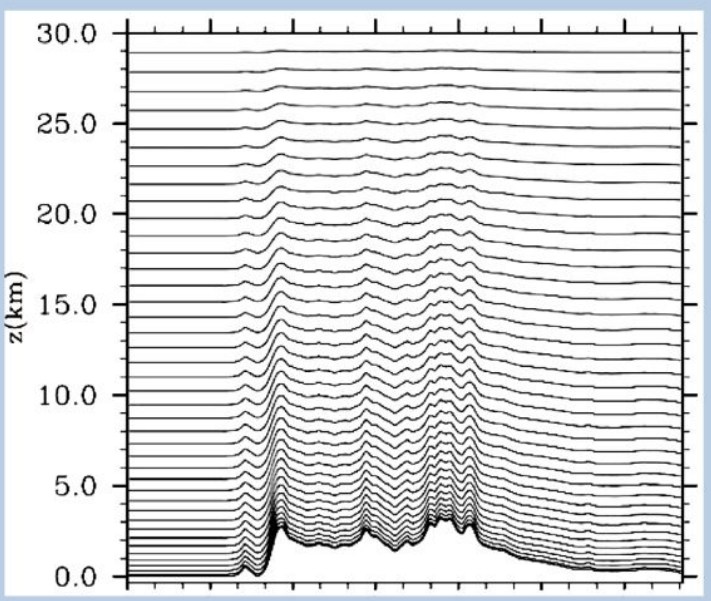
Vertical Coordinates

```
Jablonowski and Williamson baroclinic wave test case
initial perturbation included
calling test case setup
point 1 in test case setup
hx computation complete
k, sh, zw, ah = 1 0.00000000000000 0.00000000000000 0.00000000000000
k, sh, zw, ah = 2 0.754292827454554E-02 1730.76923076923 0.108968669152331E-01
k, sh, zw, ah = 3 0.213346229317396E-01 3461.53846153846 0.429570573912
k, sh, zw, ah = 4 0.391942050280822E-01 5192.30769230769 0.943427718327
k, sh, zw, ah = 5 0.603434261963643E-01 6923.07692307692 0.162163576704
k, sh, zw, ah = 6 0.843325018564451E-01 8653.84615384615 0.242716511050
k, sh, zw, ah = 7 0.110857952634291 10384.6153846154 0.331786982582272
k, sh, zw, ah = 8 0.139696986601515 12115.3846153846 0.424980127306677
k, sh, zw, ah = 9 0.170676983453917 13846.1538461538 0.518049998084810
k, sh, zw, ah = 10 0.203659063412730 15576.9230769231 0.607195305432139
k, sh, zw, ah = 11 0.238528335748478 17307.6923076923 0.689295225282555
k, sh, zw, ah = 12 0.275187691979535 19038.4615384615 0.762066370245209
k, sh, zw, ah = 13 0.313553640224657 20769.2307692308 0.824131424617781
k, sh, zw, ah = 14 0.353553390593274 22500.0000000000 0.875000000000000
k, sh, zw, ah = 15 0.395122746149031 24230.7692307692 0.914971756916988
k, sh, zw, ah = 16 0.438204533834279 25961.5384615385 0.944979639374745
k, sh, zw, ah = 17 0.482747409570915 27692.3076923077 0.966396305281607
k, sh, zw, ah = 18 0.528704930040957 29423.0769230769 0.980828974592044
k, sh, zw, ah = 19 0.576034819156969 31153.8461538462 0.989926834556141
k, sh, zw, ah = 20 0.624698379655273 32884.6153846154 0.995221127789069
k, sh, zw, ah = 21 0.674660014851561 34615.3846153846 0.998011762321982
k, sh, zw, ah = 22 0.725886835374279 36346.1538461538 0.999306656308190
k, sh, zw, ah = 23 0.778348332391215 38076.9230769231 0.999812143270829
k, sh, zw, ah = 24 0.832016103534503 39807.6923076923 0.999965697603075
k, sh, zw, ah = 25 0.886863621074329 41538.4615384615 0.999996932988832
k, sh, zw, ah = 26 0.942866034318192 43269.2307692308 0.999999951549997
k, sh, zw, ah = 27 1.00000000000000 45000.0000000000 1.00000000000000
```

src/core_init_atmosphere/mpas_init_atm_cases.F

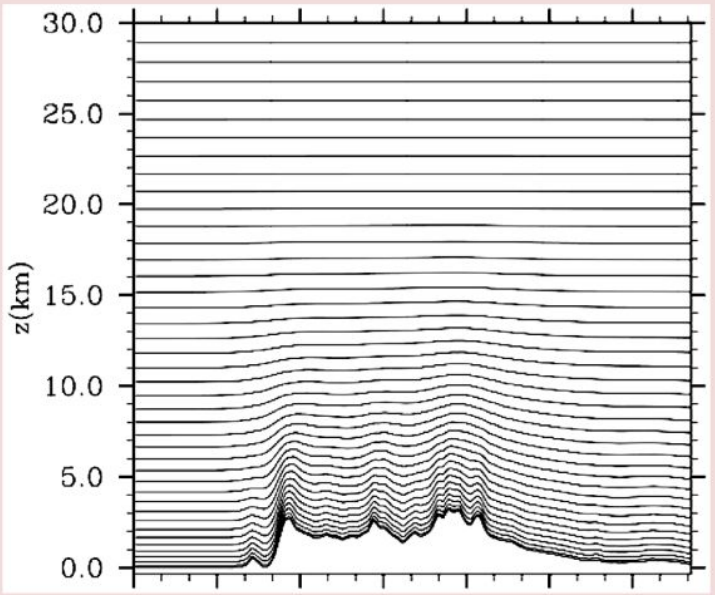
```
!     Metrics for hybrid coordinate and vertical stretching

str = 1.5
zt = 45000.
dz = zt/float(nz1)

call mpas_log_write(' hx computation complete ')

do k=1,nz

      sh(k) is the stretching specified for height surfaces

      sh(k) = (real(k-1)*dz/zt)**str

      to specify specific heights zc(k) for coordinate surfaces,
      input zc(k) and define sh(k) = zc(k)/zt
      zw(k) is the hieght of zeta surfaces
           zw(k) = (k-1)*dz yields constant dzeta
                   and nonconstant dzeta/dz
           zw(k) = sh(k)*zt yields nonconstant dzeta
                   and nearly constant dzeta/dz

      zw(k) = float(k-1)*dz
       zw(k) = sh(k)*zt

      ah(k) governs the transition between terrain-following
      and pureheight coordinates
           ah(k) = 0 is a terrain-following coordinate
           ah(k) = 1 is a height coordinate

      ah(k) = 1.-cos(.5*pii*(k-1)*dz/zt)**6
       ah(k) = 0.
      call mpas_log_write(' k, sh, zw, ah = $i $r $r $r', intArgs=(/k/), realArgs=(/sh(k),zw(k),ah(k)/))
end do
```

Vertical Coordinates



WRF

Pressure-based terrain-following sigma vertical coordinate



MPAS

Height-based hybrid smoothed terrain-following vertical coordinate

- Improved numerical accuracy

Vertical Coordinate

$$z = \zeta + A(\zeta) h_s(x, y, \zeta)$$

- Height hybrid terrain-following

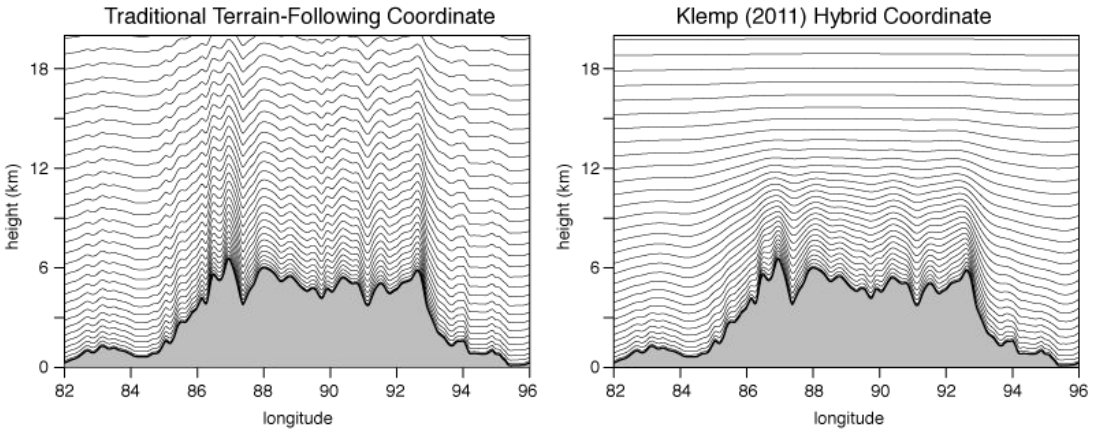
$A = 0$ Pure height coordinate.

$A = 1 - \zeta / z_t$ Pure terrain following

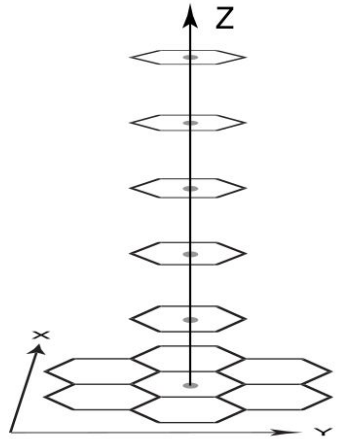
Smoothed terrain height

Factor Height/Terrain Following

Nominal heights



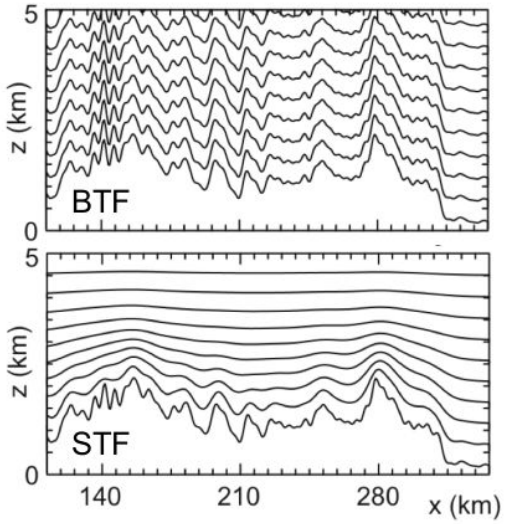
MPAS cross section through the Himalayas at 28 degrees N latitude for a 15 km (mean cell-center spacing) uniform mesh. The model top is at 30 km. The vertical coordinate in the MPAS-Atmosphere solver allows for both the traditional terrain following coordinate (right) and a general hybrid coordinate (left). From https://mpas-dev.github.io/ .



Specification of terrain:

- High resolution terrain data (30 arcsec) averaged over grid-cell area
- Terrain smoothing with one pass of a 4th order Laplacian

Smoothed Terrain-Following (STF) hybrid Coordinate



$$z(x, y, \zeta) = \zeta + A(\zeta) h_s(x, y, \zeta)$$

$A(\zeta)$ Controls rate at which terrain influences are attenuated with height

$h_s(x, y, \zeta)$ Terrain influence that represents increased smoothing of the actual terrain with height

Multiple passes of simple Laplacian smoother at each ζ level:

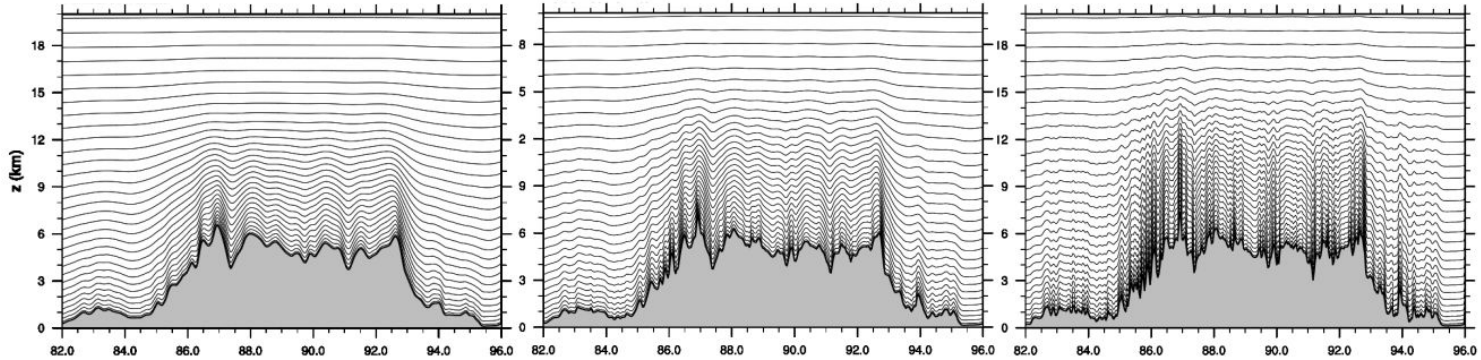
$$h_s^{(n)} = h_s^{(n-1)} + \beta(\zeta) d^2 \nabla_\zeta^2 h_s^{(n-1)}$$

STF progressively smooths coordinate surfaces while transitioning to a height coordinate

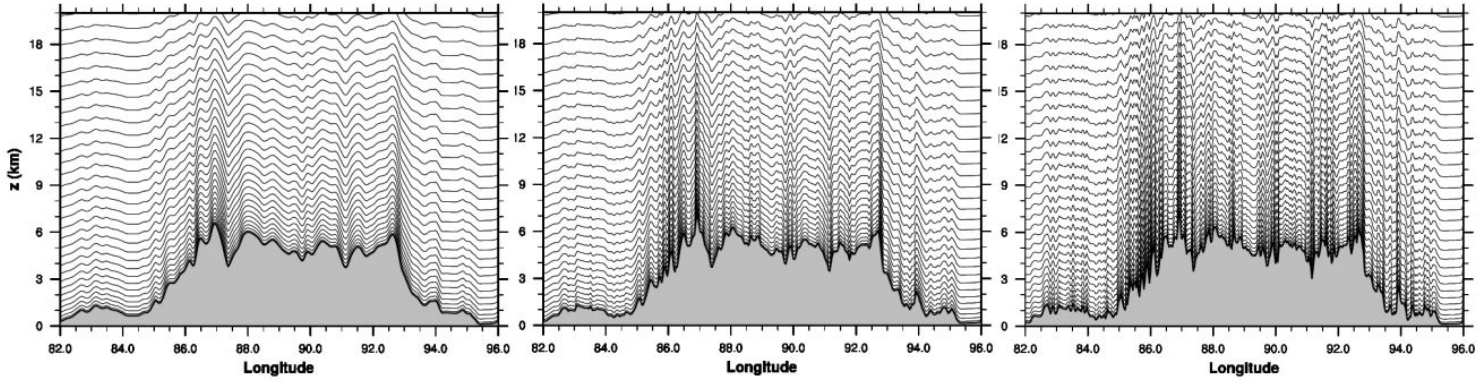
15, 7.5, & 3 km MPAS - Tibetan Plateau, 28° N

15 km grid | 7.5 km grid | 3 km grid

Smoothed hybrid terrain-following (STF) coordinate

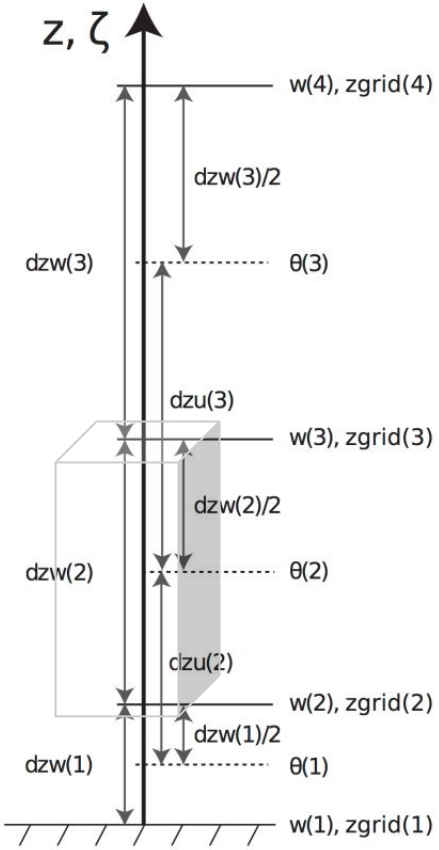


Basic terrain-following (BTF) coordinate



(Model top is at 30 km)

Vertical grid



The MPAS-Atmosphere vertical grid is also staggered:

- vertical velocities on *w* levels
- all other fields on Θ levels

zgrid gives geometric height at w levels

Θ levels lie at the midpoints of bracketing w levels

To vertically interpolate field *F* from theta levels to w levels:

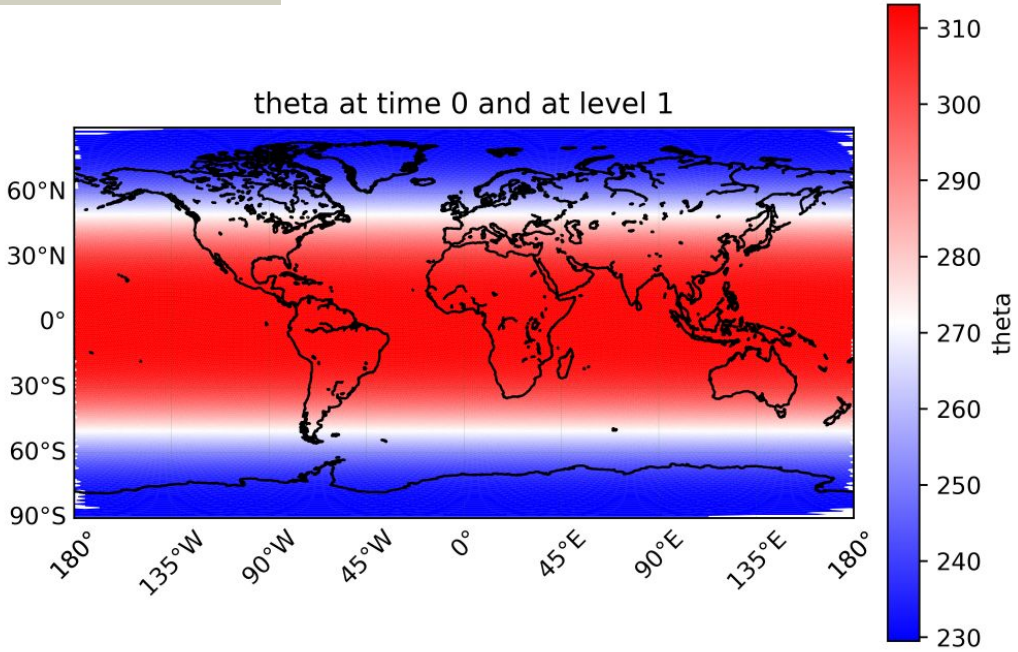
fzp(k) = 0.5 * dzw(k) / dzu(k)
fzm(k) = 0.5 * dzw(k-1) / dzu(k)

$F_w(k) = fzm(k) * F_\Theta(k) + fzp(k) * F_\Theta(k-1)$

Plot init fields

- `MPAS-BR/benchmarks/monan-class-example/jw_baroclinic_wave$ python ../../../post_proc/py/scalar_lat_lon_2d_plot/mpas_plot_scalar.py -v theta x1.40962.init.nc`

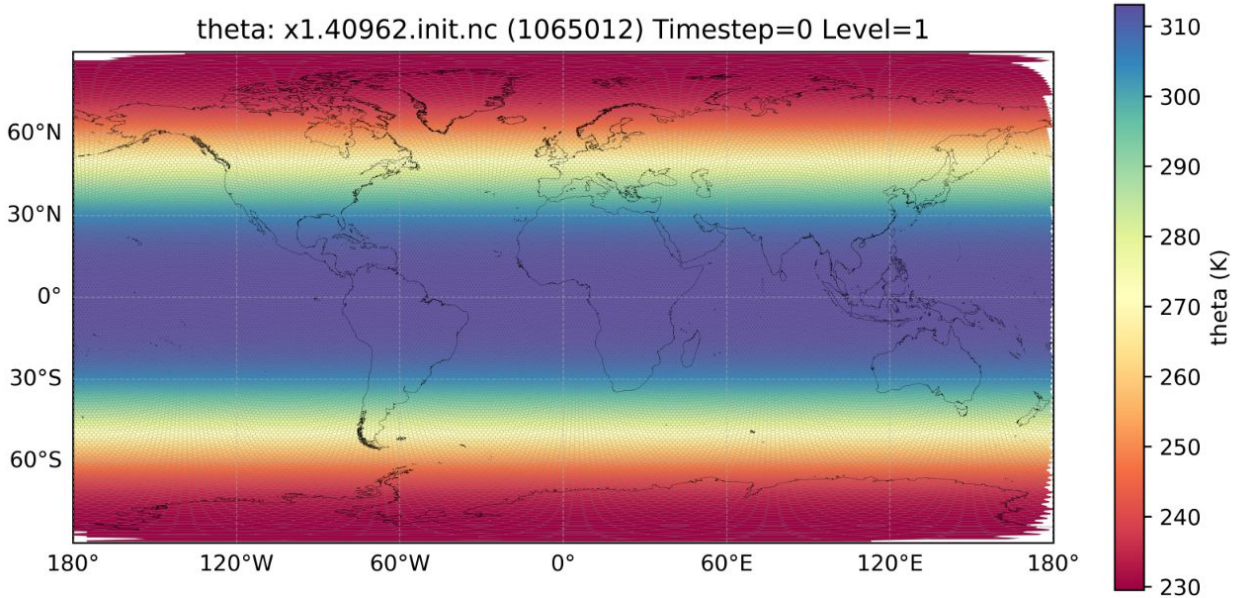
Uses “basemap” Python library



Plot init fields

- `MPAS-BR/benchmarks/monan-class-example/jw_baroclinic_wave$ python3 ../../../post_proc/py/plot_scalar_on_native_grid/mpas_plot.py -f x1.40962.init.nc -o x1.40962.init.theta.jpg -v theta -l 0 -t 0`

Uses “cartopy” Python library

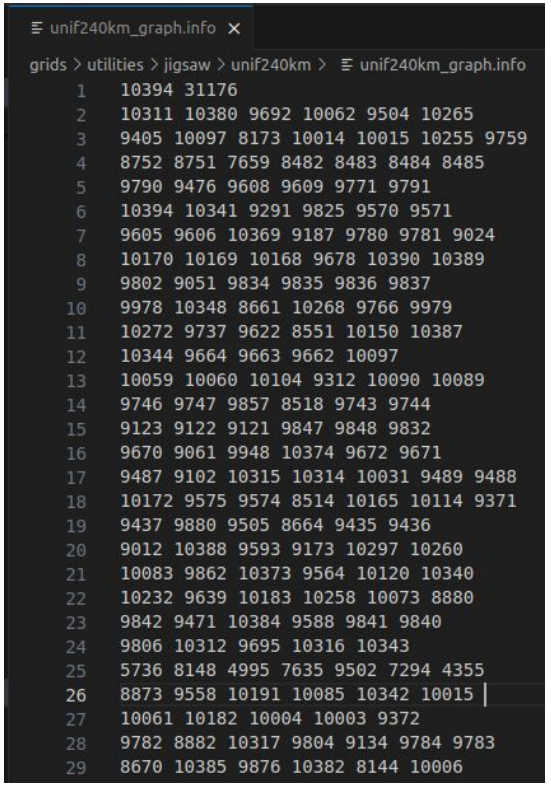


Using Jigsaw grids

With the grid created, we need to create a partition for parallel runs.

- We have a file xx.graph.info from JIGSAW
- Install METIS software for grid partitioning (http://glaros.dtc.umn.edu/gkhome/views/metis)
  - sudo apt-get install metis
- Run metis partitioning:
  - gpmetis -minconn -contig -niter=200 xx.graph.info N
  - N = number of partitions (nodes/cores to be used)

- Adjust namelist.init_atmosphere and streams.init_atmosphere to point to jigsaw grid.
- Run init_atmosphere_model



```
(mpas-tools) pedrosp@ppeixoto:~/ppstorage/Simulations/MPAS-BR/grids/utilities/jigsaw/unif240km$ gpmetis -minconn -contig -niter=200 unif240km_graph.info
**********************************************************************
METIS 5.0 Copyright 1998-13, Regents of the University of Minnesota
 (HEAD: , Built on: Mar 24 2022, 16:16:52)
 size of idx_t: 32bits, real_t: 32bits, idx_t *: 64bits

Graph Information ---------------------------------------------------------
 Name: unif240km_graph.info, #Vertices: 10394, #Edges: 31176, #Parts: 4

Options -------------------------------------------------------------------
 ptype=kway, objtype=cut, ctype=shem, rtype=greedy, iptype=metisrb
 dbglvl=0, ufactor=1.030, no2hop=NO, minconn=YES, contig=YES, nooutput=NO
 seed=-1, niter=200, ncuts=1

Direct k-way Partitioning -------------------------------------------------
 - Edgecut: 696, communication volume: 702.

 - Balance:
     constraint #0:  1.003 out of 0.000

 - Most overweight partition:
     pid: 0, actual: 2606, desired: 2598, ratio: 1.00.

 - Subdomain connectivity: max: 3, min: 3, avg: 3.00

 - Each partition is contiguous.

Timing Information --------------------------------------------------------
  I/O:                          0.002 sec
  Partitioning:                 0.005 sec   (METIS time)
  Reporting:                    0.001 sec

Memory Information --------------------------------------------------------
  Max memory used:              1.086 MB
**********************************************************************
(mpas-tools) pedrosp@ppeixoto:~/ppstorage/Simulations/MPAS-BR/grids/utilities/jigsaw/unif240
tmpqtmnn28t          unif240km_graph.info.part.4  unif240km.jig  unif240km-MESH.msh  unif24
unif240km_graph.info  unif240km-HFUN.msh          unif240km.log  unif240km_mpas.jpg  unif24
```

grids > utilities > jigsaw > unif240km > unif240km_graph.info.part.4

1	1
2	1
3	2
4	2
5	3
6	3
7	3
8	3
9	1
10	0
11	1
12	0
13	3
14	0
15	1
16	1
17	1
18	1
19	1
20	0
21	0
22	2
23	1
24	3
25	1

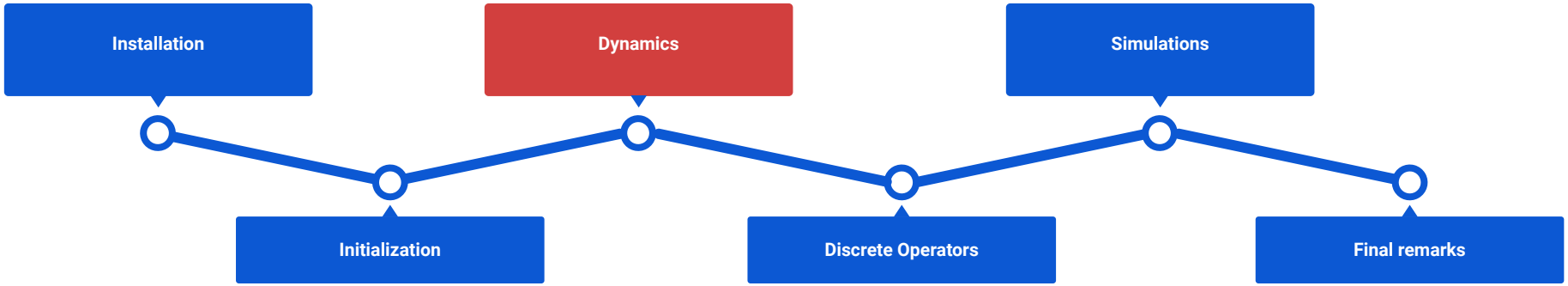
Convert MPAS data to lat-lon

It is possible to interpolate the MPAS output data to latitude-longitude data, so you can work on your preferred visualization tool….

…. we will not discuss this topic in this class.

Overview

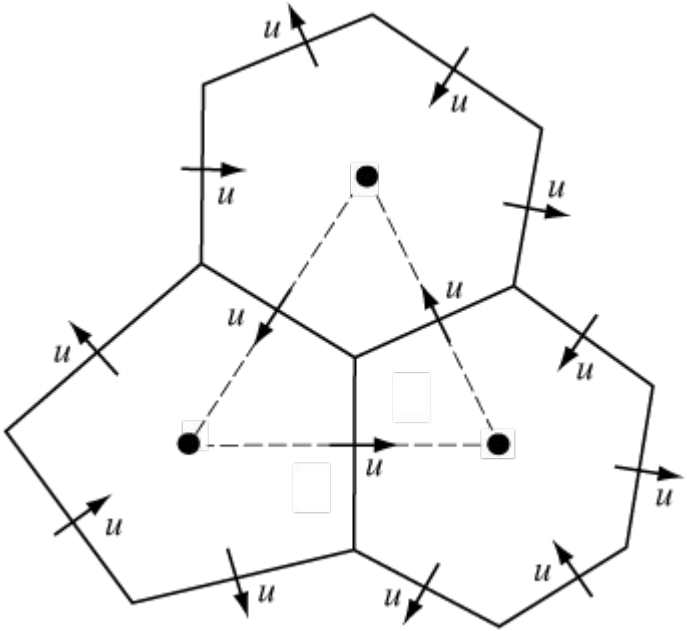
Today's class:



Horizontal Coordinates

Staggered Voronoi Spherical Grid

C-grid staggered variables on the horizontal Voronoi mesh. Normal velocities are defined on the cell faces and all other scalar variables are defined at the cell centers. Vertical vorticity is defined at the cell vertices.



Arbitrary cell shapes (Voronoi polygons)

Nonhydrostatic formulation

Equations

- Prognostic equations for coupled variables.
- Generalized height coordinate.
- Horizontally vector invariant eqn set.
- Continuity equation for dry air mass.
- Thermodynamic equation for coupled potential temperature.

Time integration scheme

As in Advanced Research WRF - Split-explicit Runge-Kutta (3rd order)

Variables:

$$(U,V,\Omega,\Theta,Q_j) = \tilde{\rho}_d \cdot (u,v,\dot{\eta},\theta,q_j)$$

Vertical coordinate:

$$z = \zeta + A(\zeta)\, h_s(x,y,\zeta)$$

Prognostic equations:

$$\frac{\partial \boldsymbol{V}_H}{\partial t} = -\frac{\rho_d}{\rho_m}\left[\boldsymbol{\nabla}_\zeta\left(\frac{p}{\zeta_z}\right) - \frac{\partial \boldsymbol{z}_H p}{\partial \zeta}\right] - \eta\, \boldsymbol{k} \times \boldsymbol{V}_H$$
$$- \boldsymbol{v}_H \boldsymbol{\nabla}_\zeta \cdot \boldsymbol{V} - \frac{\partial \Omega \boldsymbol{v}_H}{\partial \zeta} - \rho_d \boldsymbol{\nabla}_\zeta K - eW\cos\alpha_r - \frac{uW}{r_e} + \boldsymbol{F}_{V_H},$$

$$\frac{\partial W}{\partial t} = -\frac{\rho_d}{\rho_m}\left[\frac{\partial p}{\partial \zeta} + g\tilde{\rho}_m\right] - \left(\boldsymbol{\nabla}\cdot\boldsymbol{v}W\right)_\zeta$$
$$+ \frac{uU + vV}{r_e} + e\left(U\cos\alpha_r - V\sin\alpha_r\right) + F_W,$$

$$\frac{\partial \Theta_m}{\partial t} = -\left(\boldsymbol{\nabla}\cdot\boldsymbol{V}\,\theta_m\right)_\zeta + F_{\Theta_m},$$

$$\frac{\partial \tilde{\rho}_d}{\partial t} = -\left(\boldsymbol{\nabla}\cdot\boldsymbol{V}\right)_\zeta,$$

$$\frac{\partial Q_j}{\partial t} = -\left(\boldsymbol{\nabla}\cdot\boldsymbol{V}\,q_j\right)_\zeta + \rho_d S_j + F_{Q_j},$$

Diagnostics and definitions:

$$\theta_m = \theta[1 + (R_v/R_d)q_v] \qquad p = p_0\left(\frac{R_d \zeta_z \Theta_m}{p_0}\right)^\gamma$$

$$\frac{\rho_m}{\rho_d} = 1 + q_v + q_c + q_r + \ldots$$

Prognostic Variables

$$(U, V, W, \Theta_m, Q_j) = \tilde{\rho}_d \cdot (u, v, w, \theta_m, q_j)$$

Horizontal Velocities

Vertical velocity

Moist potential temperature

Mixing ratio of water species (vapor, cloud, rain)

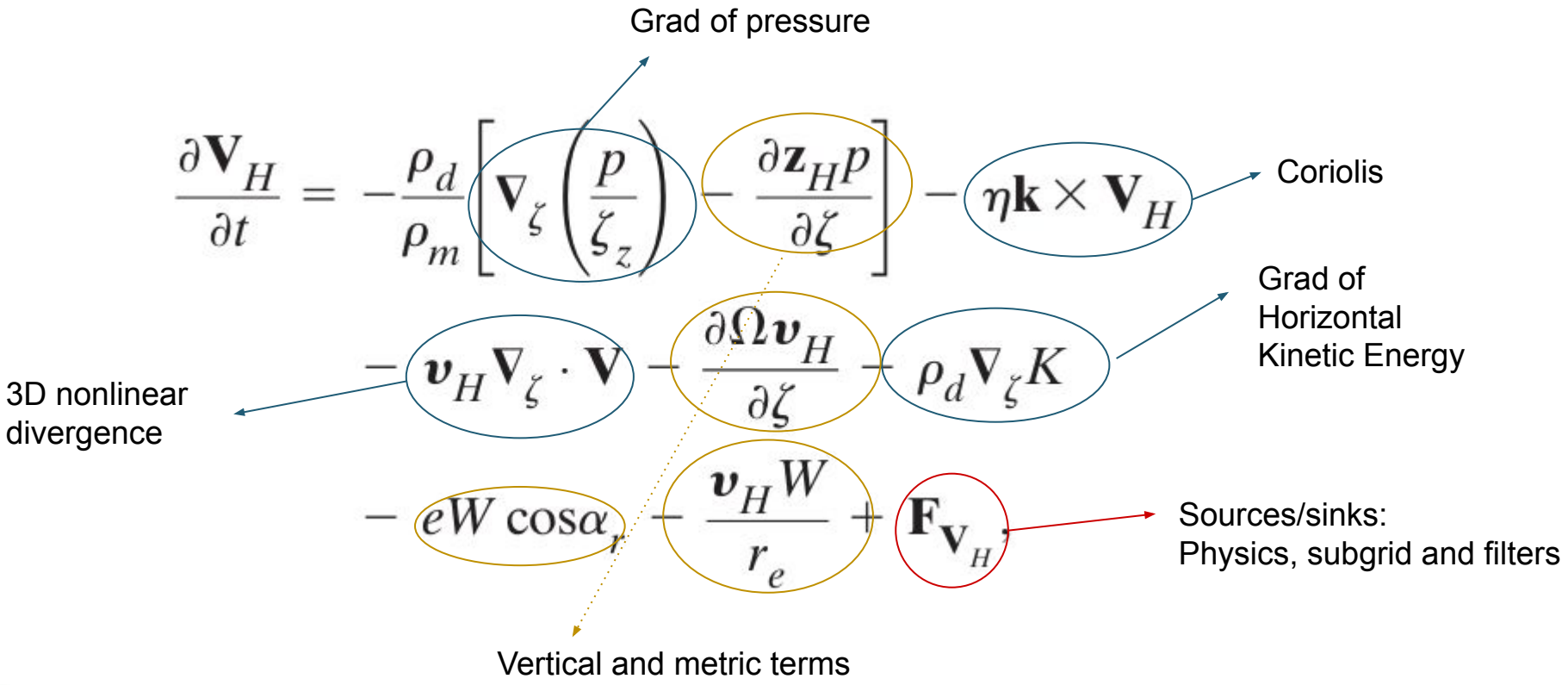
$$\tilde{\rho}_d = \rho_d / \zeta_z$$

Dry air density

Vertical derivative of height coord

Nonhydrostatic fluid-flow equations on the sphere

Horizontal Velocity Dynamics $\mathbf{v}_H = (U, V)$



Mass Dynamics

$$\frac{\partial \tilde{\rho}_d}{\partial t} = -(\nabla \cdot \mathbf{V})_\zeta$$

$$\frac{\partial Q_j}{\partial t} = -(\nabla \cdot \mathbf{V} q_j)_\zeta + F_{Q_j}$$

$$\frac{\partial \Theta_m}{\partial t} = -(\nabla \cdot \mathbf{V} \theta_m)_\zeta + F_{\Theta_m}$$

Flux form divergence
(Ready for Finite Volumes!)

Summary

$$\frac{\partial \boldsymbol{V}_H}{\partial t} = -\frac{\rho_d}{\rho_m}\left[\boldsymbol{\nabla}_\zeta\left(\frac{p}{\zeta_z}\right) + \frac{\partial z_H p}{\partial \zeta}\right] - \eta\,\boldsymbol{k}\times\boldsymbol{V}_H$$
$$- \boldsymbol{v}_H\boldsymbol{\nabla}_\zeta\cdot\boldsymbol{V} - \frac{\partial \Omega \boldsymbol{v}_H}{\partial \zeta} - \rho_d\boldsymbol{\nabla}_\zeta K + eW\cos\alpha_r - \frac{uW}{r_e} + \boldsymbol{F}_{V_H},$$

$$\frac{\partial W}{\partial t} = -\frac{\rho_d}{\rho_m}\left[\frac{\partial p}{\partial \zeta} + g\tilde{\rho}_m\right] - \left(\boldsymbol{\nabla}\cdot\boldsymbol{v}W\right)_\zeta$$
$$+ \frac{uU+vV}{r_e} + e\left(U\cos\alpha_r - V\sin\alpha_r\right) + F_W,$$

$$\frac{\partial \Theta_m}{\partial t} = -\left(\boldsymbol{\nabla}\cdot\boldsymbol{V}\,\theta_m\right)_\zeta + F_{\Theta_m},$$

$$\frac{\partial \tilde{\rho}_d}{\partial t} = -\left(\boldsymbol{\nabla}\cdot\boldsymbol{V}\right)_\zeta,$$

$$\frac{\partial Q_j}{\partial t} = -\left(\boldsymbol{\nabla}\cdot\boldsymbol{V}\,q_j\right)_\zeta + \rho_d S_j + F_{Q_j},$$

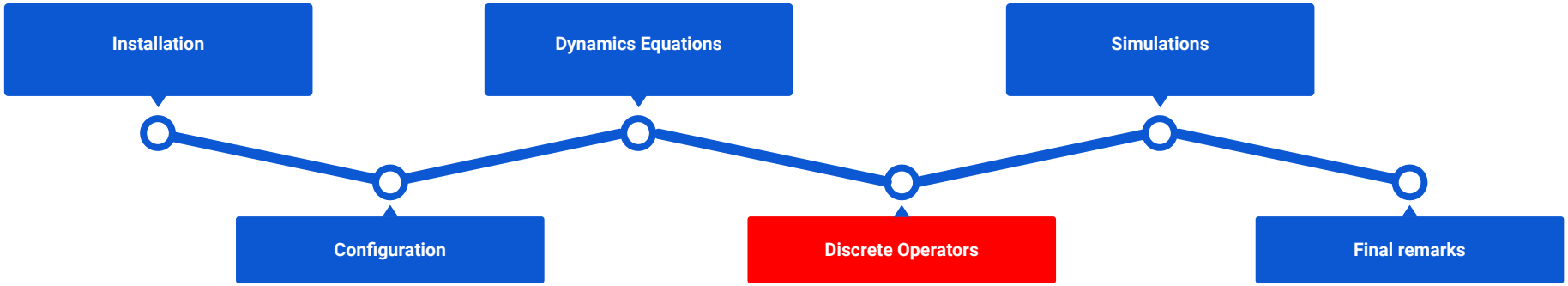
(1) Gradient operators

(2) Flux divergence operators

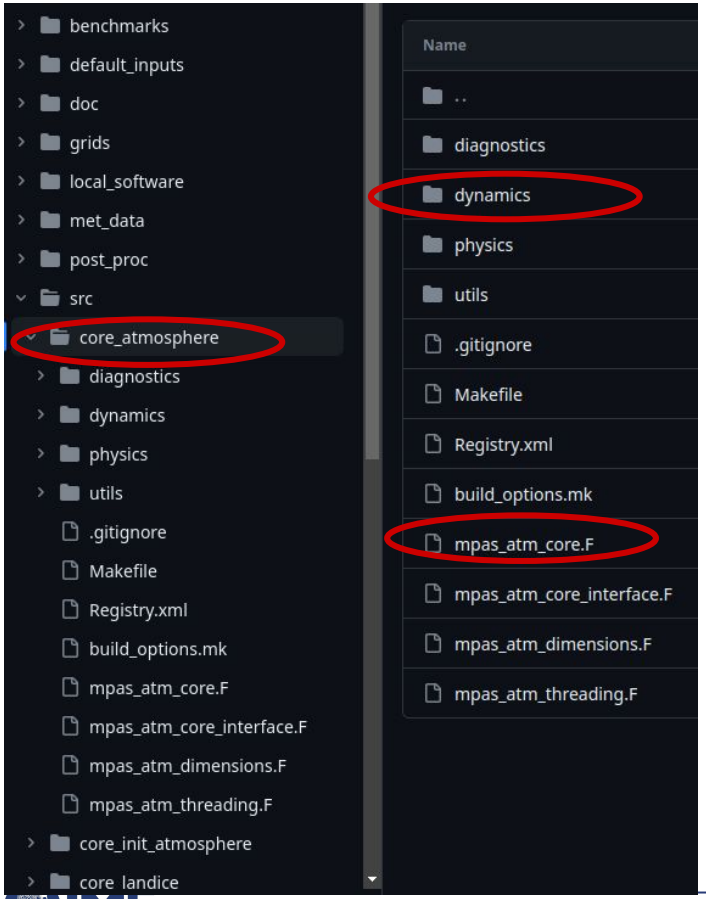
(3) Nonlinear Coriolis term

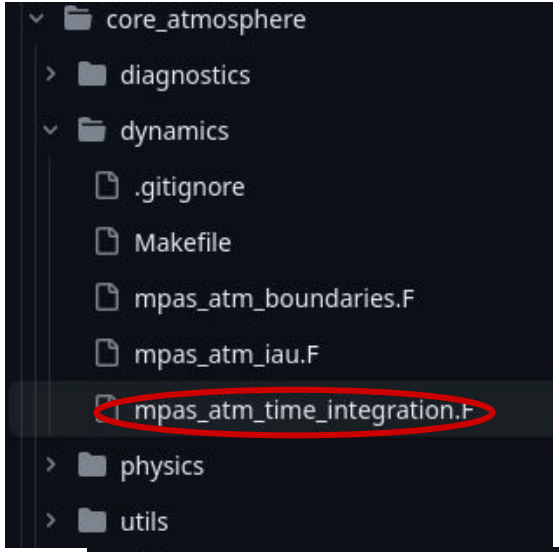
Overview

Today's class:



Discrete Operators / Code

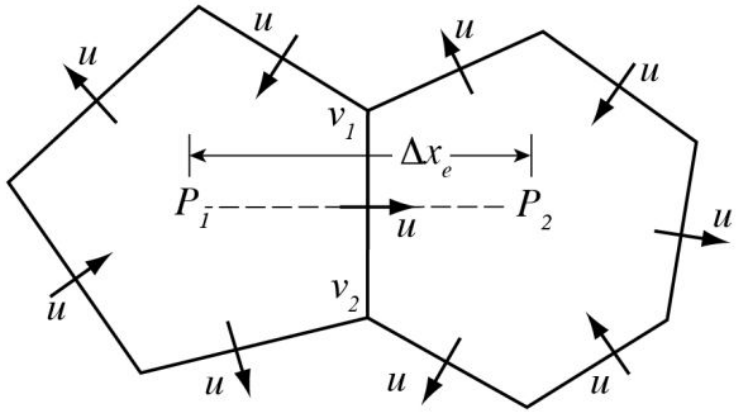





```
4216
4217      subroutine atm_compute_dyn_tend(tend, tend_physics, state, diag, mesh, configs, nVertL
4218                                      cellStart, cellEnd, vertexStart, vertexEnd, edgeStart,
4219                                      cellSolveStart, cellSolveEnd, vertexSolveStart, vertex
4220      !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
4221      ! Compute height and normal wind tendencies, as well as diagnostic variables
4222      !
4223      ! Input: state - current model state
4224      !        mesh - grid metadata
4225      !        diag - some grid diagnostics
4226      !
4227      ! Output: tend - tendencies: tend_u, tend_w, tend_theta and tend_rho
4228      !               these are all coupled-variable tendencies.
```

Discrete Gradient

$$\frac{\partial \mathbf{V}_H}{\partial t} = -\frac{\rho_d}{\rho_m}\left[\nabla_\zeta\left(\frac{p}{\zeta_z}\right) - \frac{\partial \mathbf{z}_H p}{\partial \zeta}\right] - \eta\,\mathbf{k}\times\mathbf{V}_H$$
$$-\mathbf{v}_H\nabla_\zeta\cdot\mathbf{V} - \frac{\partial \Omega \mathbf{v}_H}{\partial \zeta} - \rho_d\nabla_\zeta K - eW\cos\alpha_r - \frac{uW}{r_e} + \mathbf{F}_{V_H},$$



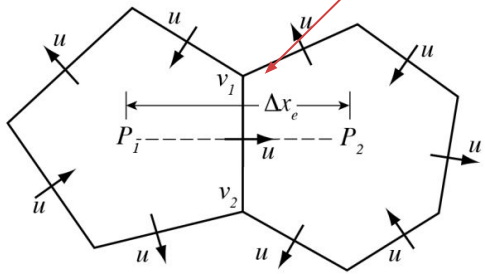
On the Voronoi mesh, P_1P_2 is perpendicular to v_1v_2 and is bisected by v_1v_2, hence $P_x \sim (P_2\text{-}P_1)\Delta x_e^{-1}$ is 2nd order accurate.

Code example

MPAS-BR / src / core_atmosphere / dynamics / **mpas_atm_time_integration.F**

Code | Blame | 6607 lines (5328 loc) · 293 KB | Code 55% faster with GitHub Copilot | Raw

```
4380
4381            ! horizontal pressure gradient
4382
4383            if(rk_step == 1) then
4384    !DIR$ IVDEP
4385               do k=1,nVertLevels
4386                  tend_u_euler(k,iEdge) =  - cqu(k,iEdge)*( (pp(k,cell2)-pp(k,cell1))*invDcEdge(iEdge)/(.5*(zz(k,cell2)+zz(k,cell1))) &
4387                                           -0.5*zxu(k,iEdge)*(dpdz(k,cell1)+dpdz(k,cell2)) )
4388               end do
4389
4390            end if
```



Flux Divergence and Transport

Transport equation, conservative form:

$$\frac{\partial(\rho\psi)}{\partial t} = -\nabla \cdot \mathbf{V}(\rho\psi)$$

Finite-Volume formulation, Integrate over cell:

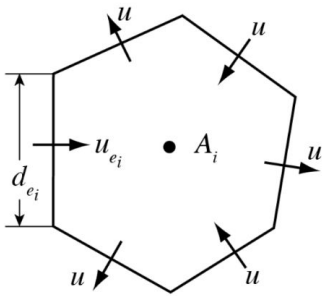
$$\int_D \left[\frac{\partial}{\partial t}(\rho\psi) = -\nabla \cdot \mathbf{V}(\rho\psi)\right] dV$$

Apply divergence theorem:

$$\frac{\partial(\overline{\rho\psi})}{\partial t} = -\frac{1}{V}\int_\Sigma (\rho\psi)\,\mathbf{V}\cdot\mathbf{n}\; d\sigma$$

Discretize in time and space:

$$(\rho\psi)_i^{t+\Delta t} = (\rho\psi)_i^t - \Delta t\,\frac{1}{A_i}\sum_{n_{e_i}} d_{e_i}\overline{(\rho\mathbf{V}\cdot\mathbf{n}_{e_i})\psi}$$



Velocity divergence operator is 2nd-order accurate for edge-centered velocities.

Flux Divergence and Transport

MPAS uses a Runge-Kutta time-integration scheme.

$$\frac{\partial(\rho\psi)}{\partial t} = L(\mathbf{V}, \rho, \psi)$$

$$(\rho\psi)^{*} = (\rho\psi)^{t} + \frac{\Delta t}{3} L(\mathbf{V}, \rho, \psi^{t})$$

$$(\rho\psi)^{**} = (\rho\psi)^{t} + \frac{\Delta t}{2} L(\mathbf{V}, \rho, \psi^{*})$$

$$(\rho\psi)^{t+\Delta t} = (\rho\psi)^{t} + \Delta t\, L(\mathbf{V}, \rho, \psi^{**})$$

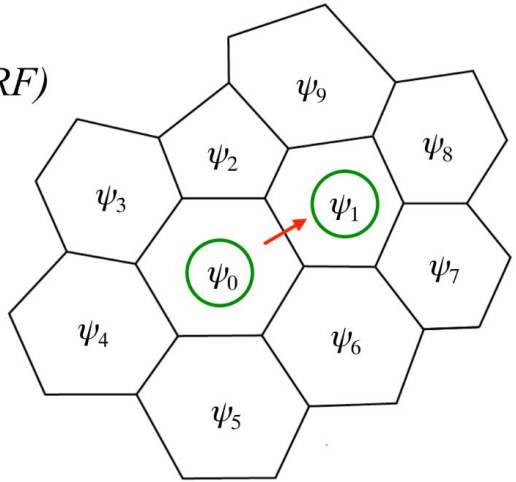
$$(\rho\psi)_i^{t+\Delta t} = (\rho\psi)_i^{t} - \Delta t \left(\frac{1}{A_i} \sum_{n_{e_i}} d_{e_i} (\rho\mathbf{V} \cdot \mathbf{n}_{e_i}) \psi\right)$$

Instantaneous flux divergence in RK-based scheme

Computing the flux - consider 1D transport (e.g. from WRF)

$$\frac{\partial(u\psi_i)}{\partial x} = \frac{1}{\Delta x}\left[F_{i+1/2}(u\psi) - F_{i-1/2}(u\psi)\right] + O(\Delta x^{p}).$$

2nd-order flux: $F(u,\psi)_{i+1/2} = u_{i+1/2}\left[\frac{1}{2}(\psi_{i+1} + \psi_i)\right]$



Flux Divergence and Transport

3rd and 4th-order fluxes:

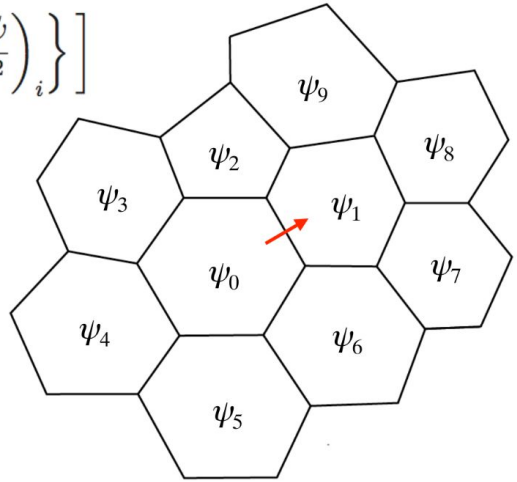
$$F(u,\psi)_{i+1/2} = u_{i+1/2}\left[\frac{1}{2}\left(\psi_{i+1}+\psi_i\right) - \frac{1}{12}\left(\delta_x^2\psi_{i+1}+\delta_x^2\psi_i\right) + sign(u)\,\frac{\beta}{12}\left(\delta_x^2\psi_{i+1}-\delta_x^2\psi_i\right)\right]$$

where $\delta_x^2\psi_i = \psi_{i-1} - 2\psi_i + \psi_{i+1}$ *(Hundsdorfer et al, 1995; Van Leer, 1985)*

Recognizing $\delta_x^2\psi = \Delta x^2\frac{\partial^2\psi}{\partial x^2} + O(\Delta x^4)$ we recast the 3rd and 4th order flux as

$$F(u,\psi)_{i+1/2} = u_{i+1/2}\left[\frac{1}{2}\left(\psi_{i+1}+\psi_i\right) - \Delta x_e^2\frac{1}{12}\left\{\left(\frac{\partial^2\psi}{\partial x^2}\right)_{i+1} + \left(\frac{\partial^2\psi}{\partial x^2}\right)_i\right\}\right.$$
$$\left. + sign(u)\,\Delta x_e^2\frac{\beta}{12}\left\{\left(\frac{\partial^2\psi}{\partial x^2}\right)_{i+1} - \left(\frac{\partial^2\psi}{\partial x^2}\right)_i\right\}\right]$$

where x is the direction normal to the cell edge and i and $i+1$ are cell centers. We use the least-squares-fit polynomial to compute the second derivatives.



Flux Divergence and Transport

3rd and 4th-order fluxes:

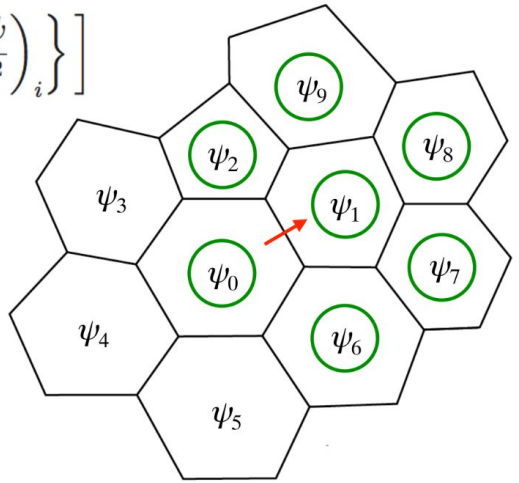
$$F(u,\psi)_{i+1/2} = u_{i+1/2}\left[\frac{1}{2}\left(\psi_{i+1}+\psi_i\right) - \frac{1}{12}\left(\delta_x^2\psi_{i+1}+\delta_x^2\psi_i\right) + sign(u)\,\frac{\beta}{12}\left(\delta_x^2\psi_{i+1}-\delta_x^2\psi_i\right)\right]$$

where $\delta_x^2\psi_i = \psi_{i-1} - 2\psi_i + \psi_{i+1}$ *(Hundsdorfer et al, 1995; Van Leer, 1985)*

Recognizing $\delta_x^2\psi = \Delta x^2 \dfrac{\partial^2\psi}{\partial x^2} + O(\Delta x^4)$ we recast the 3rd and 4th order flux as

$$F(u,\psi)_{i+1/2} = u_{i+1/2}\left[\frac{1}{2}\left(\psi_{i+1}+\psi_i\right) - \Delta x_e^2\,\frac{1}{12}\left\{\left(\frac{\partial^2\psi}{\partial x^2}\right)_{i+1} + \left(\frac{\partial^2\psi}{\partial x^2}\right)_i\right\}\right.$$
$$\left. + sign(u)\,\Delta x_e^2\,\frac{\beta}{12}\left\{\left(\frac{\partial^2\psi}{\partial x^2}\right)_{i+1} - \left(\frac{\partial^2\psi}{\partial x^2}\right)_i\right\}\right]$$

where x is the direction normal to the cell edge and i and $i+1$ are cell centers. We use the least-squares-fit polynomial to compute the second derivatives.



Flux Divergence and Transport

3rd and 4th-order fluxes:

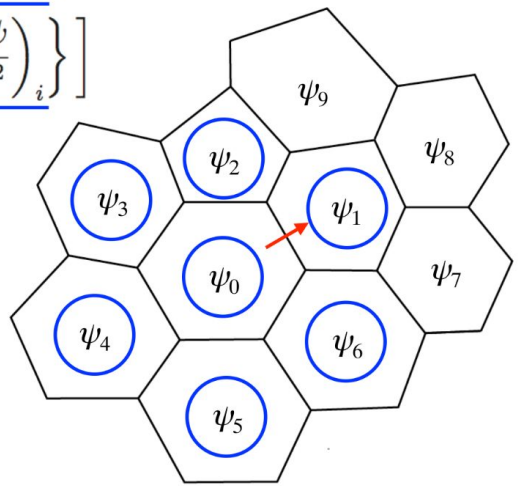
$$F(u,\psi)_{i+1/2} = u_{i+1/2}\left[\frac{1}{2}\left(\psi_{i+1}+\psi_i\right) - \frac{1}{12}\left(\delta_x^2\psi_{i+1}+\delta_x^2\psi_i\right) + sign(u)\,\frac{\beta}{12}\left(\delta_x^2\psi_{i+1}-\delta_x^2\psi_i\right)\right]$$

where $\delta_x^2\psi_i = \psi_{i-1} - 2\psi_i + \psi_{i+1}$ *(Hundsdorfer et al, 1995; Van Leer, 1985)*

Recognizing $\delta_x^2\psi = \Delta x^2 \dfrac{\partial^2\psi}{\partial x^2} + O(\Delta x^4)$ we recast the 3rd and 4th order flux as

$$F(u,\psi)_{i+1/2} = u_{i+1/2}\left[\frac{1}{2}\left(\psi_{i+1}+\psi_i\right) - \Delta x_e^2\,\frac{1}{12}\left\{\left(\frac{\partial^2\psi}{\partial x^2}\right)_{i+1} + \left(\frac{\partial^2\psi}{\partial x^2}\right)_{i}\right\}\right.$$
$$\left. + sign(u)\,\Delta x_e^2\,\frac{\beta}{12}\left\{\left(\frac{\partial^2\psi}{\partial x^2}\right)_{i+1} - \left(\frac{\partial^2\psi}{\partial x^2}\right)_{i}\right\}\right]$$

where x is the direction normal to the cell edge and i and $i+1$ are cell centers. We use the least-squares-fit polynomial to compute the second derivatives.



Código (init_atmosphere)

MPAS-BR / src / core_init_atmosphere / **mpas_atm_advection.F**

Code | Blame | 941 lines (740 loc) · 31.5 KB | Code 55% faster with GitHub Copilo

```
210
211            if (polynomial_order == 2) then
212               na = 6
213               ma = ma+1
214
215               amatrix(1,1) = 1.
216               wmatrix(1,1) = 1.
217               do i=2,ma
218                  amatrix(i,1) = 1.
219                  amatrix(i,2) = xp(i-1)
220                  amatrix(i,3) = yp(i-1)
221                  amatrix(i,4) = xp(i-1)**2
222                  amatrix(i,5) = xp(i-1) * yp(i-1)
223                  amatrix(i,6) = yp(i-1)**2
224
225                  wmatrix(i,i) = 1.
226               end do
227
228            else if (polynomial_order == 3) then
229               na = 10
230               ma = ma+1
231
232               amatrix(1,1) = 1.
233               wmatrix(1,1) = 1.
234               do i=2,ma
235                  amatrix(i,1) = 1.
236                  amatrix(i,2) = xp(i-1)
237                  amatrix(i,3) = yp(i-1)
238
239                  amatrix(i,4) = xp(i-1)**2
240                  amatrix(i,5) = xp(i-1) * yp(i-1)
241                  amatrix(i,6) = yp(i-1)**2
242
243                  amatrix(i,7) = xp(i-1)**3
244                  amatrix(i,8) = yp(i-1) * (xp(i-1)**2)
245                  amatrix(i,9) = xp(i-1) * (yp(i-1)**2)
```

Polynomial fit

Pre-calculations for high-order flux

```
368   !              do j=1,n
369   !
370   !                 deriv_two(j,1,iEdge) =   2.*xe(iEdge)*xe(iEdge)*bmatrix(4,j)  &
371   !                                        + 2.*xe(iEdge)*ye(iEdge)*bmatrix(5,j)  &
372   !                                        + 2.*ye(iEdge)*ye(iEdge)*bmatrix(6,j)
373   !              end do
374
375                  if (iCell == cellsOnEdge(1,iEdge)) then
376                     do j=1,n
377                        deriv_two(j,1,iEdge) =   2.*cos2t*bmatrix(4,j)  &
378                                               + 2.*costsint*bmatrix(5,j)  &
379                                               + 2.*sin2t*bmatrix(6,j)
380                     end do
381                  else
382                     do j=1,n
383                        deriv_two(j,2,iEdge) =   2.*cos2t*bmatrix(4,j)  &
384                                               + 2.*costsint*bmatrix(5,j)  &
385                                               + 2.*sin2t*bmatrix(6,j)
386                     end do
387                  end if
```

Código (atmosphere)

Flux calculation

MPAS-BR / src / core_atmosphere / dynamics / **mpas_atm_time_integration.F**

Code | Blame | 6607 lines (5328 loc) · 293 KB | Code 55% faster with GitHub Copilot | Raw

```
2939          flux4(q_im2, q_im1, q_i, q_ip1, ua) =                    &
2940              ua*( 7.*(q_i + q_im1) - (q_ip1 + q_im2) )/12.0
2941
2942          flux3(q_im2, q_im1, q_i, q_ip1, ua, coef3) =             &
2943                    flux4(q_im2, q_im1, q_i, q_ip1, ua) +          &
2944                    coef3*abs(ua)*((q_ip1 - q_im2)-3.*(q_i-q_im1))/12.0
2945
2946          local_advance_density = advance_density
```

$$F(u,\psi)_{i+1/2} = u_{i+1/2}\left[\frac{1}{2}(\psi_{i+1}+\psi_i) - \frac{1}{12}\left(\delta_x^2\psi_{i+1}+\delta_x^2\psi_i\right) + sign(u)\,\frac{\beta}{12}\left(\delta_x^2\psi_{i+1}-\delta_x^2\psi_i\right)\right]$$

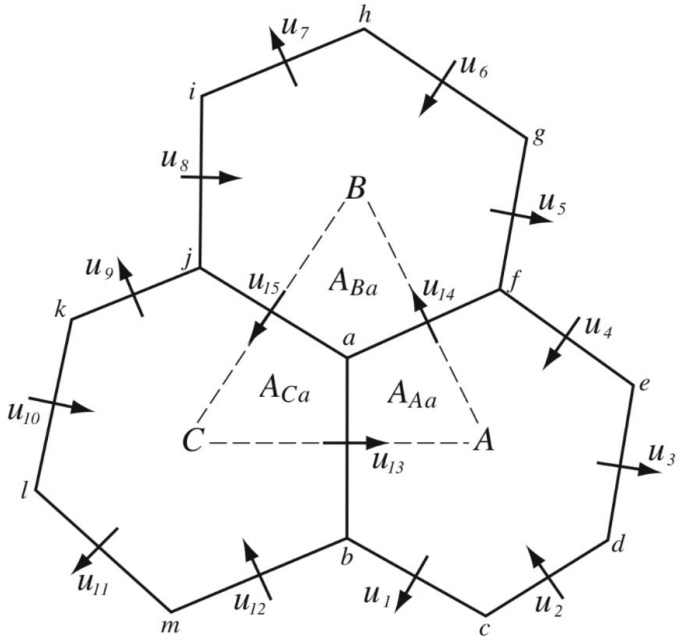
where $\delta_x^2\psi_i = \psi_{i-1} - 2\psi_i + \psi_{i+1}$ *(Hundsdorfer et al, 1995; Van Leer, 1985)*

Nonlinear Coriolis Term

$$\frac{\partial \boldsymbol{V}_H}{\partial t} = -\frac{\rho_d}{\rho_m}\left[\nabla_\zeta\left(\frac{p}{\zeta_z}\right) - \frac{\partial z_H p}{\partial \zeta}\right] - \eta\,\boldsymbol{k}\times\boldsymbol{V}_H$$
$$-\nu_H\nabla_\zeta\cdot\boldsymbol{V} - \frac{\partial \Omega \nu_H}{\partial \zeta} - \rho_d\nabla_\zeta K - eW\cos\alpha_r - \frac{uW}{r_e} + \boldsymbol{F}_{V_H},$$

Vorticity is computed by evaluating the circulation around the triangles.
Vorticity *lives* on the vertices.

First, the linear piece: $f\ \boldsymbol{k}\ x\ \boldsymbol{V_H}$



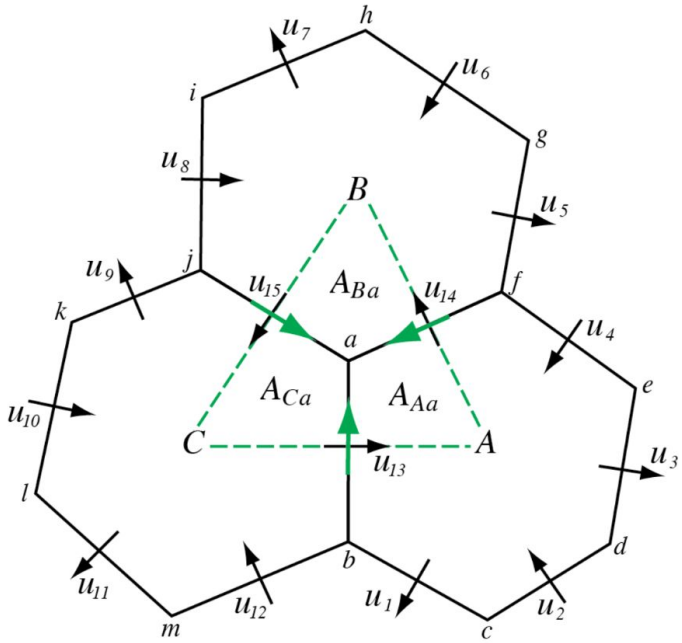
Nonlinear Coriolis Term

$$\frac{\partial \boldsymbol{V}_H}{\partial t} = -\frac{\rho_d}{\rho_m}\left[\nabla_\zeta\left(\frac{p}{\zeta_z}\right) - \frac{\partial z_H p}{\partial \zeta}\right] - \eta\,\boldsymbol{k}\times\boldsymbol{V}_H$$
$$-\boldsymbol{v}_H\nabla_\zeta\cdot\boldsymbol{V} - \frac{\partial \Omega \boldsymbol{v}_H}{\partial \zeta} - \rho_d\nabla_\zeta K - eW\cos\alpha_r - \frac{uW}{r_e} + \boldsymbol{F}_{V_H},$$

Vorticity is computed by evaluating the circulation around the triangles.
Vorticity *lives* on the vertices.

First, the linear piece: $f\ \boldsymbol{k}\ x\ \boldsymbol{V_H}$

How do we compute the tangential velocity on the cell faces needed in the Coriolis term?



Nonlinear Coriolis Term

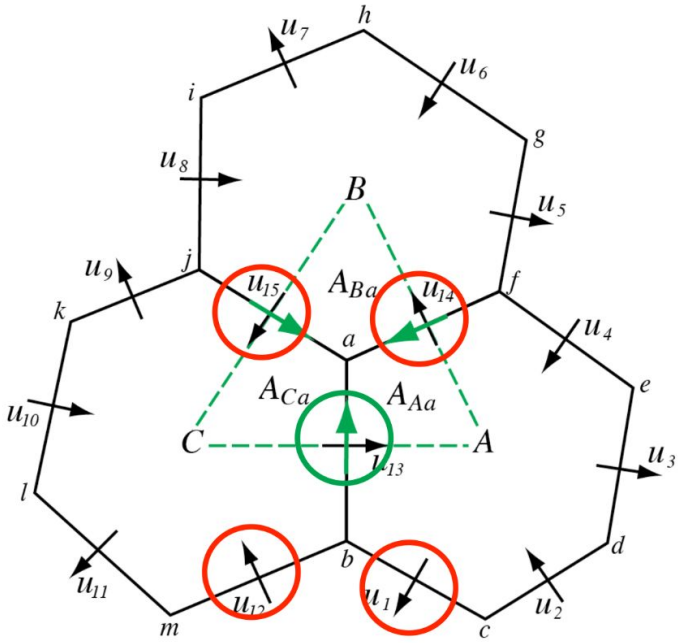
$$\frac{\partial \boldsymbol{V}_H}{\partial t} = -\frac{\rho_d}{\rho_m}\left[\nabla_\zeta\left(\frac{p}{\zeta_z}\right) - \frac{\partial z_H p}{\partial \zeta}\right] - \eta\, \boldsymbol{k} \times \boldsymbol{V}_H$$
$$- \boldsymbol{v}_H \nabla_\zeta \cdot \boldsymbol{V} - \frac{\partial \Omega \boldsymbol{v}_H}{\partial \zeta} - \rho_d \nabla_\zeta K - eW\cos\alpha_r - \frac{uW}{r_e} + \boldsymbol{F}_{V_H},$$

Linear piece: $f\ \boldsymbol{k}\ x\ \boldsymbol{V_H}$

Simplest approach: Construct tangential velocities from weighted sum of the four nearest neighbors.

Result: physically stationary geostrophic modes (geostrophically-balanced flow) will not be stationary in the discrete system; the solver is unusable.

(Nickovic et al, MWR 2002)



Nonlinear Coriolis Term

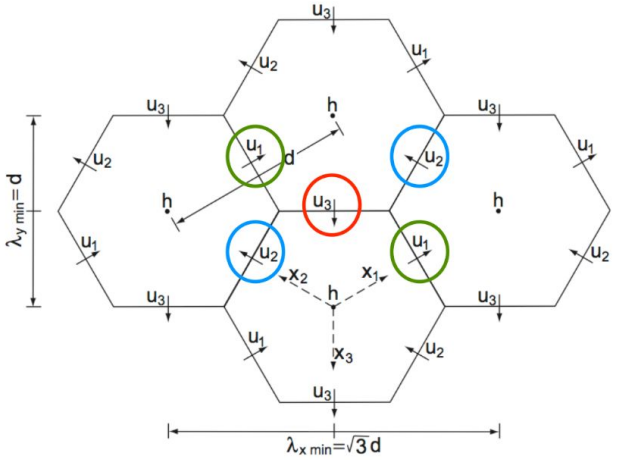
Linear piece: $f\ \boldsymbol{k}\ x\ \boldsymbol{V_H}$

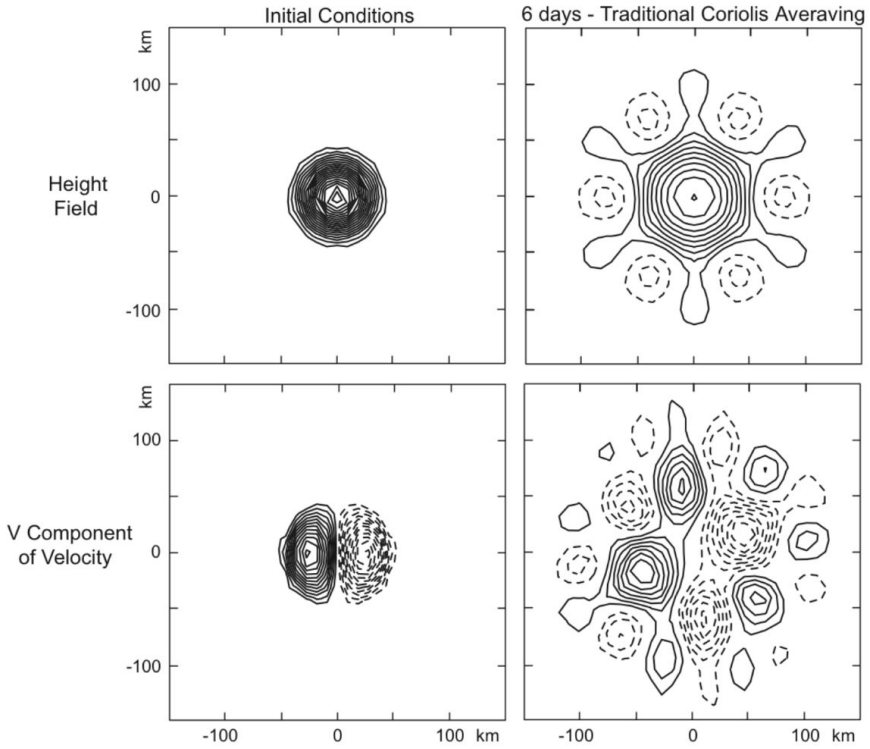
$$\partial_t u_1 + g\delta_{x_1} h + \frac{f}{\sqrt{3}}(u_{31} - u_{21}) = 0$$

$$\partial_t u_2 + g\delta_{x_2} h + \frac{f}{\sqrt{3}}(u_{12} - u_{32}) = 0$$

$$\partial_t u_3 + g\delta_{x_3} h + \frac{f}{\sqrt{3}}(u_{23} - u_{13}) = 0$$

$$\partial_t h + \tfrac{2}{3} H(\delta_{x_1} u_1 + \delta_{x_2} u_2 + \delta_{x_3} u_3) = 0$$





Nonlinear Coriolis Term

Linear piece: $f\ \boldsymbol{k}\ x\ \boldsymbol{V_H}$

$$\partial_t u_1 + g\delta_{x_1} h + \frac{f}{\sqrt{3}}(u_{31} - u_{21}) = 0$$

$$\partial_t u_2 + g\delta_{x_2} h + \frac{f}{\sqrt{3}}(u_{12} - u_{32}) = 0$$

$$\partial_t u_3 + g\delta_{x_3} h + \frac{f}{\sqrt{3}}(u_{23} + u_{13}) = 0$$

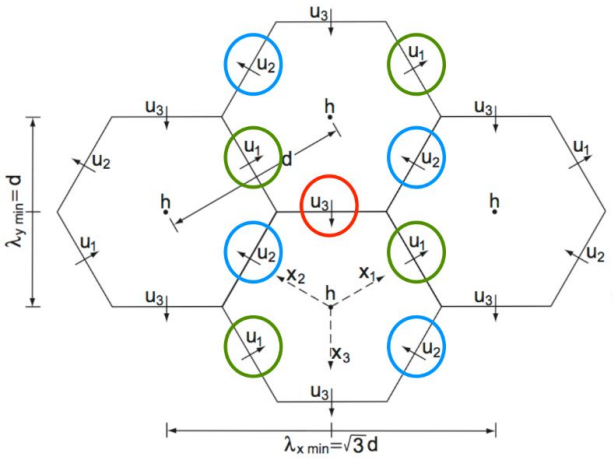
$$\partial_t h + \frac{2}{3}H(\delta_{x_1} u_1 + \delta_{x_2} u_2 + \delta_{x_3} u_3) = 0$$

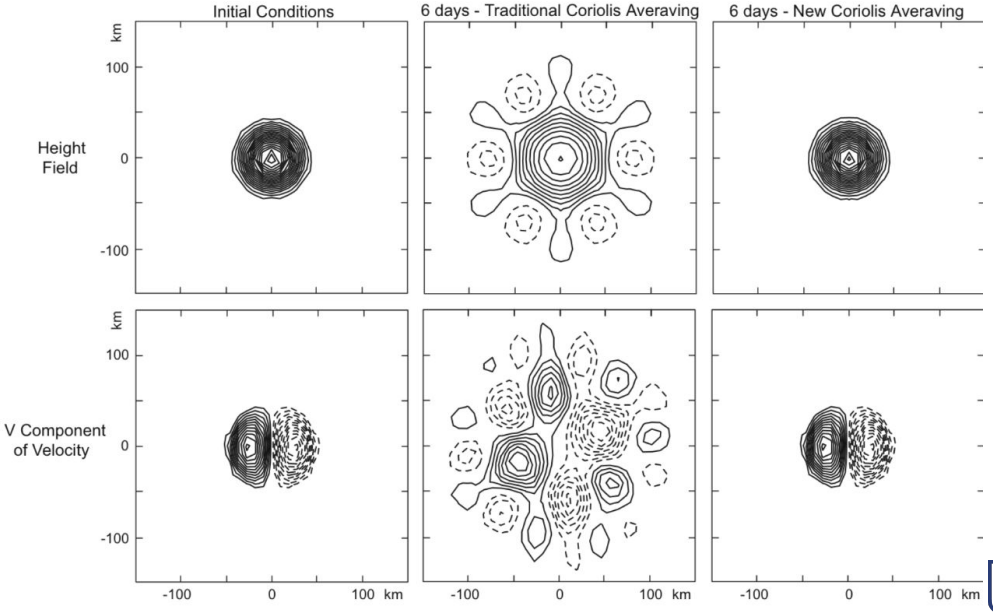
(Thuburn et al, 2009 JCP)

$$u_{21} = \frac{1}{3}\overline{u_2}^{x_3} + \frac{2}{3}\overline{\overline{u_2}^{x_1}}^{x_2}, \quad u_{31} = \frac{1}{3}\overline{u_3}^{x_2} + \frac{2}{3}\overline{\overline{u_3}^{x_1}}^{x_3},$$

$$u_{12} = \frac{1}{3}\overline{u_1}^{x_3} + \frac{2}{3}\overline{\overline{u_1}^{x_1}}^{x_2}, \quad u_{32} = \frac{1}{3}\overline{u_3}^{x_1} + \frac{2}{3}\overline{\overline{u_3}^{x_2}}^{x_3},$$

$$u_{13} = \frac{1}{3}\overline{u_1}^{x_2} + \frac{2}{3}\overline{\overline{u_1}^{x_1}}^{x_3}, \quad u_{23} = \frac{1}{3}\overline{u_2}^{x_1} + \frac{2}{3}\overline{\overline{u_2}^{x_2}}^{x_3}$$





Nonlinear Coriolis Term

In the discrete analogue of vorticity equation (ξ_τ=-fδ_a), the divergence δ_a on the Delaunay triangulation is identical to the divergence δ_A on the Voronoi hexagons used in the height equation (h_t=-Hδ_A) integrated over the triangle.

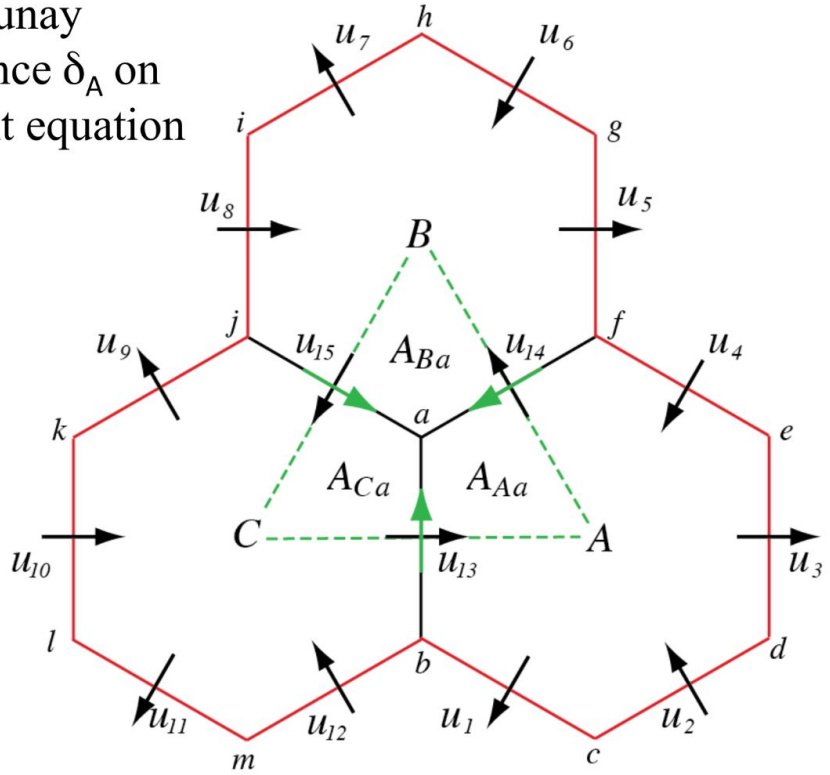
$$A_a\delta_a = \frac{A_A\delta_A + A_B\delta_B + A_C\delta_C}{6}$$

Divergence δ_A in hexagon A:

$$A_A\delta_A = \sum_{i=1}^{6} l_i u_i \cdot \mathbf{n}_i$$

Divergence δ_a in triangle ABC:

$$A_a\xi_t = -f A_a\delta_a = f\sum_{j=1}^{3} d_j u_j^{\perp} \cdot \mathbf{n}_j$$



Nonlinear Coriolis Term

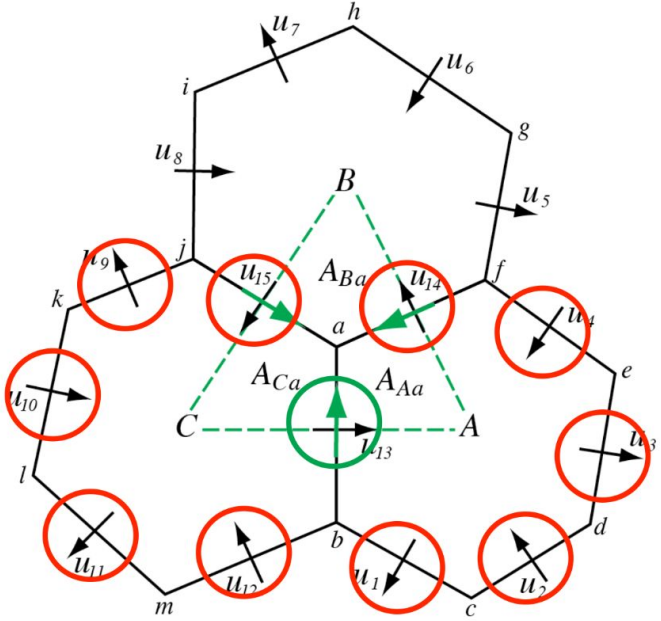
Linear piece: $f\ \boldsymbol{k}\ x\ \boldsymbol{V_H}$

Generalization for the Voronoi mesh:

Construct tangential velocities from weighted sum of normal velocities on edges of adjacent hexagons.

$$d_e\, u_e^{\perp} = \sum_j w_e^j\, l_j\, u_j$$

Result: geostrophic modes are stationary; local and global mass and PV conservation is satisfied on the dual (triangular) mesh (for the SW equations).



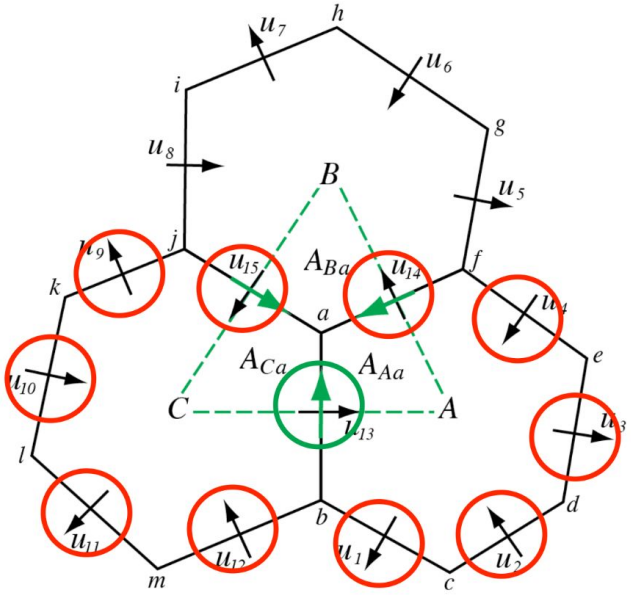
Nonlinear Coriolis Term

$$\frac{\partial \boldsymbol{V}_H}{\partial t} = -\frac{\rho_d}{\rho_m}\left[\nabla_\zeta\left(\frac{p}{\zeta_z}\right) - \frac{\partial z_H p}{\partial \zeta}\right] - \eta\,\boldsymbol{k}\times\boldsymbol{V}_H$$
$$- \boldsymbol{v}_H\nabla_\zeta\cdot\boldsymbol{V} - \frac{\partial \Omega \boldsymbol{v}_H}{\partial \zeta} - \rho_d\nabla_\zeta K - eW\cos\alpha_r - \frac{uW}{r_e} + \boldsymbol{F}_{V_H},$$

Nonlinear term:

$$\boldsymbol{v}_{e_i} = \sum_{j=1}^{n_{e_i}} w_{e_{i,j}}\boldsymbol{u}_{e_{i,j}}$$

$$[\eta\,\boldsymbol{k}\times\boldsymbol{V}_H]_{e_i} = \sum_{j=1}^{n_{e_i}} \frac{1}{2}(\eta_{e_i} + \eta_{e_{i,j}})\, w_{e_{i,j}}\rho_{e_{i,j}}\boldsymbol{u}_{e_{i,j}}$$



The general tangential velocity reconstruction produces a consistent divergence on the primal and dual grids, and allows for PV, enstrophy and energy* conservation in the nonlinear SW solver.

Exemplo código (atmosphere)

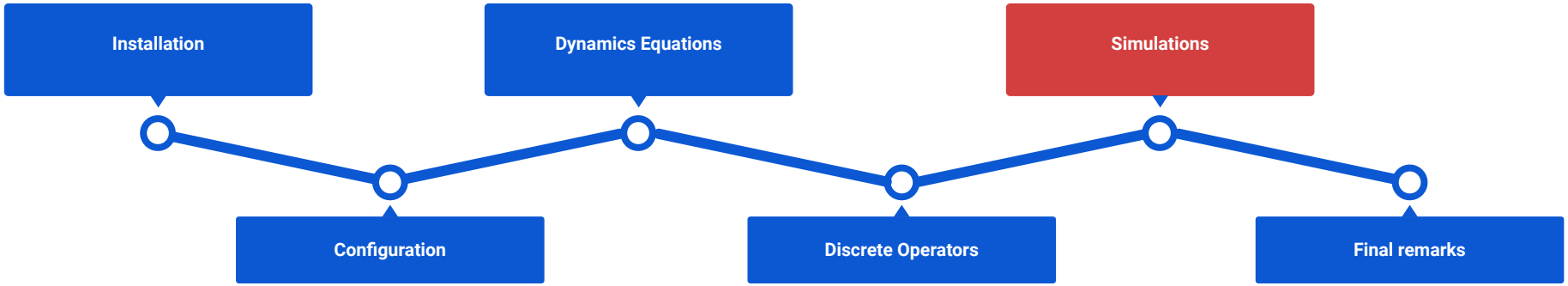
MPAS-BR / src / core_atmosphere / dynamics / **mpas_atm_time_integration.F**

Code | Blame | 6607 lines (5328 loc) · 293 KB | Code 55% faster with GitHub Copilot

```
4410
4411            ! Next, nonlinear Coriolis term (q) following Ringler et al JCP 2009
4412
4413            q(:) = 0.0
4414            do j = 1,nEdgesOnEdge(iEdge)
4415               eoe = edgesOnEdge(j,iEdge)
4416               do k=1,nVertLevels
4417                  workpv = 0.5 * (pv_edge(k,iEdge) + pv_edge(k,eoe))
4418     !  the original definition of pv_edge had a factor of 1/density.  We have removed that factor
4419     !  given that it was not integral to any conservation property of the system
4420                  q(k) = q(k) + weightsOnEdge(j,iEdge) * u(k,eoe) * workpv
4421               end do
4422            end do
```

Overview

Today's class:



Initial setup / configuration

First, let's create a 200km uniform mesh using Jigsaw.
In the MPAS source code directory:

- Activate the maps-tools conda environment: `conda activate mpas-tools`
- Step into de grids/grids directory: `cd grids/grids`
- Create the grid: `python3 ../utilities/jigsaw/spherical_grid.py -g unif -r 200 -o unif200km`

This will create the *unif200km* directory with the following data:

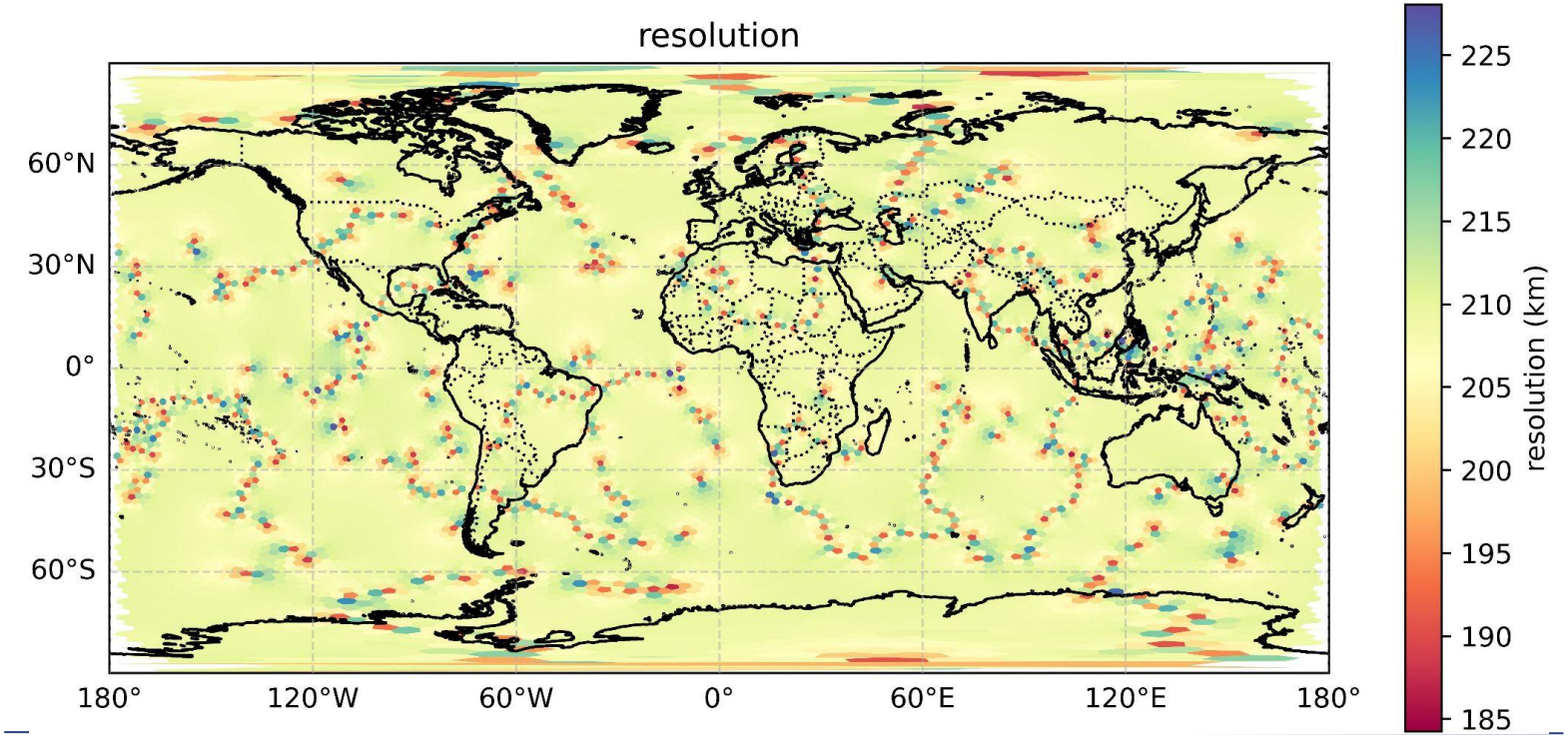
```
tmpyc_kkvna/          unif200km.jig       unif200km_mpas.nc
unif200km_graph.info  unif200km.log       unif200km.msh
unif200km-HFUN.msh    unif200km-MESH.msh  unif200km_triangles.nc
```

Inside this new directory we partition the grid using *metis* and the *unif200km_graph.info*, for example, for 4 processes: `gpmetis -contig -minconn unif200km_graph.info 4`
This will create the *unif200km_graph.info.part.4* file.

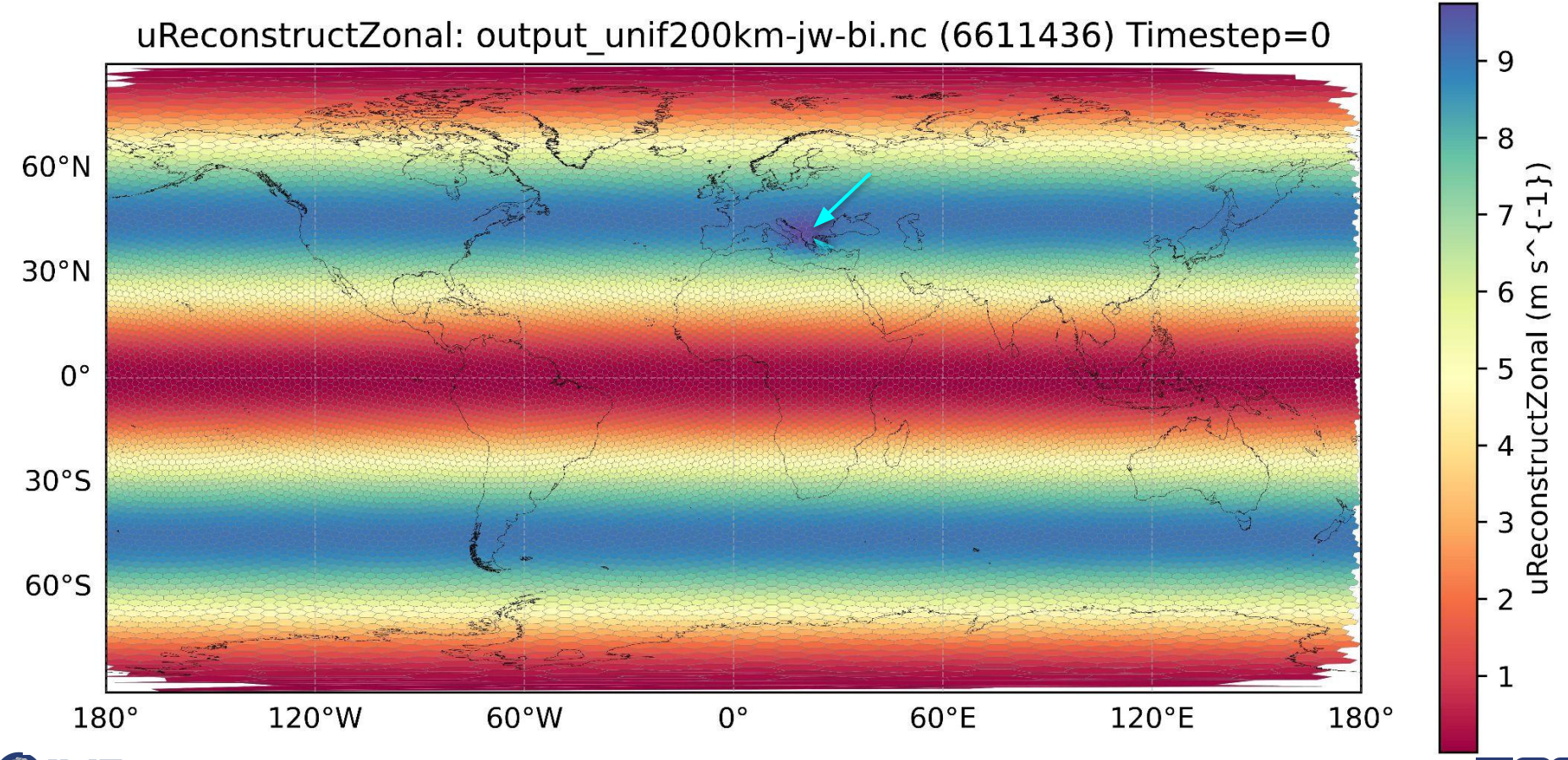
Initial setup / configuration

We can check the grid resolution using one of the post-processing scripts from the MPAS-BR repository:

- `python ../../../post_proc/py/grid_maps/mpas_plot_grid.py -g unif200km_mpas.nc -o unif200km-grid.jpg`



Example Simulations: JW Baroclinic Instability



Example Simulations: JW Baroclinic Instability

In the MPAS root directory create a new runs directory and a directory for this case:

- `mkdir -p runs/jw-bi-200km ; cd runs/jw-bi-200km`

Create a link to the grid files:

- `ln -s ../../grids/grids/unif200km/unif200km_mpas.nc ./`
- `ln -s ../../grids/grids/unif200km/unif200km_graph.info.part.4 ./`

Create a link to the init_atmosphere executable:

- `ln -s ../../init_atmosphere ./`

Then, create the *namelist.init_atmosphere* and *streams.init_atmosphere* files

Example Simulations: JW Baroclinic Instability

namelist.init_atmosphere

```
&nhyd_model
    config_start_time = '0000-01-01_00:00:00'
    config_init_case = 2
/

&dimensions
    config_nvertlevels = 26
/

&decomposition
    config_block_decomp_file_prefix = 'unif200km_graph.info.part.'
/
```

streams.init_atmosphere

```
<streams>
<immutable_stream name="input"
                  type="input"
                  filename_template="unif200km_mpas.nc"
                  input_interval="initial_only" />

<immutable_stream name="output"
                  type="output"
                  filename_template="unif200km_jw-bi.init.nc"
                  packages="initial_conds"
                  output_interval="initial_only" />
</streams>
```

Example Simulations: JW Baroclinic Instability

Run the init_atmosphere executable:

- `mpirun -n 4 ./init_atmosphere_model`

Check that the *unif200km_jw-bi.init.nc* file was created. The end of the *log.init_atmosphere.0000.out* should look like:

```
*********************************************************
   Finished running the init_atmosphere core
*********************************************************


 Timer information:
    Globals are computed across all threads and processors

 Columns:
    total time: Global max of accumulated time spent in timer
    calls: Total number of times this timer was started / stopped.
    min: Global min of time spent in a single start / stop
    max: Global max of time spent in a single start / stop
    avg: Global max of average time spent in a single start / stop
    pct_tot: Percent of the timer at level 1
    pct_par: Percent of the parent timer (one level up)
    par_eff: Parallel efficiency, global average total time / global max total time


   timer_name                                  total       calls        min          max          avg      pct_tot   pct_par     par_eff
 1 total time                                 0.94919         1        0.94870      0.94919      0.94882   100.00      0.00       1.00
 2  initialize                                0.26813         1        0.26759      0.26813      0.26782    28.25     28.25       1.00

----------------------------------------
Total log messages printed:
   Output messages =                 218
   Warning messages =                  7
   Error messages =                    0
   Critical error messages =           0
----------------------------------------
Logging complete.  Closing file at 2023/11/11 15:04:11
```

namelist.atmosphere

```
&nhyd_model
    config_dt = 720.0
    config_start_time = '0000-01-01_00:00:00'
    config_run_duration = '16_00:00:00'
    config_split_dynamics_transport = false
    config_number_of_sub_steps = 6
    config_dynamics_split_steps = 1
    config_h_mom_eddy_visc2    = 0.0
    config_h_mom_eddy_visc4    = 0.0
    config_v_mom_eddy_visc2    = 0.0
    config_h_theta_eddy_visc2  = 0.0
    config_h_theta_eddy_visc4  = 0.0
    config_v_theta_eddy_visc2  = 0.0
    config_horiz_mixing        = '2d_smagorinsky'
    config_len_disp            = 200000.
    config_u_vadv_order      = 3
    config_w_vadv_order      = 3
    config_theta_vadv_order  = 3
    config_scalar_vadv_order = 3
    config_theta_adv_order   = 3
    config_scalar_adv_order  = 3
    config_scalar_advection  = false
    config_positive_definite = false
    config_coef_3rd_order = 1.0
    config_monotonic = false
    config_epssm = 0.1
    config_smdiv = 0.1
/

&damping
    config_zd = 22000.0
    config_xnutr = 0.0
/

&decomposition
    config_block_decomp_file_prefix = 'unif200km_graph.info.part.'
/

&restart
    config_do_restart = false
/

&printout
    config_print_global_minmax_vel = true
    config_print_global_minmax_sca = false
/

&physics
    config_physics_suite = 'none'
/
```

streams.atmosphere

```
<streams>

<immutable_stream name="input"
                  type="input"
                  filename_template="unif200km_jw-bi.init.nc"
                  input_interval="initial_only"/>

<immutable_stream name="restart"
                  type="input;output"
                  filename_template="restart.$Y-$M-$D_$h.$m.$s.nc"
                  input_interval="initial_only"
                  output_interval="5_00:00:00"/>

<stream name="output"
        type="output"
        filename_template="output_unif200km-jw-bi.nc"
        filename_interval="none"
        output_interval="24:00:00">

    <file name="stream_list.atmosphere.output"/>

</stream>

</streams>
```

stream_list.atmosphere.output

```
latCell
lonCell
xCell
yCell
zCell
indexToCellID
latEdge
lonEdge
xEdge
yEdge
zEdge
indexToEdgeID
latVertex
lonVertex
xVertex
yVertex
zVertex
indexToVertexID
cellsOnEdge
nEdgesOnCell
nEdgesOnEdge
edgesOnCell
edgesOnEdge
weightsOnEdge
dvEdge
dcEdge
angleEdge
areaCell
areaTriangle
cellsOnCell
verticesOnCell
verticesOnEdge
edgesOnVertex
cellsOnVertex
kiteAreasOnVertex
u
w
pressure
rho
theta
vorticity
ke
uReconstructZonal
uReconstructMeridional
```

namelist.atmosphere options

The available options for the namelist.atmosphere file can be inspected in *src/core_atmosphere/Registry.xml*

src > core_atmosphere > Registry.xml > registry > dims

```
<!-- ********************************************************************************************** -->
<!-- ************************************ Namelists *********************************************** -->
<!-- ********************************************************************************************** -->

        <nml_record name="nhyd_model" in_defaults="true">
                <nml_option name="config_time_integration" type="character" default_value="SRK3" in_defaults="false"
                     units="-"
                     description="Time integration scheme"
                     possible_values="`SRK3'"/>

                <nml_option name="config_time_integration_order" type="integer" default_value="2"
                     units="-"
                     description="Order for RK time integration"
                     possible_values="2 or 3"/>

                <nml_option name="config_dt" type="real" default_value="720.0"
                     units="s"
                     description="Model time step, seconds"
                     possible_values="Positive real values"/>

                <nml_option name="config_calendar_type" type="character" default_value="gregorian" in_defaults="false"
                     units="-"
                     description="Simulation calendar type"
                     possible_values="`gregorian',`gregorian_noleap'"/>

                <nml_option name="config_start_time" type="character" default_value="2010-10-23_00:00:00"
                     units="-"
                     description="Starting time for model simulation"
                     possible_values="`YYYY-MM-DD_hh:mm:ss'"/>

                <nml_option name="config_stop_time" type="character" default_value="none" in_defaults="false"
                     units="-"
                     description="Stopping time for model simulation"
                     possible_values="`YYYY-MM-DD_hh:mm:ss'"/>

                <nml_option name="config_run_duration" type="character" default_value="5_00:00:00"
                     units="-"
                     description="Length of model simulation"
                     possible_values="[DDD_]hh:mm:ss"/>
```

Namelist and Streams (links)

Init_atmosphere
- Namelist
- Streams

Atmosphere
- Namelist
- Streams

Run the atmosphere executable:
- `mpirun -n 4 ./atmosphere_model`

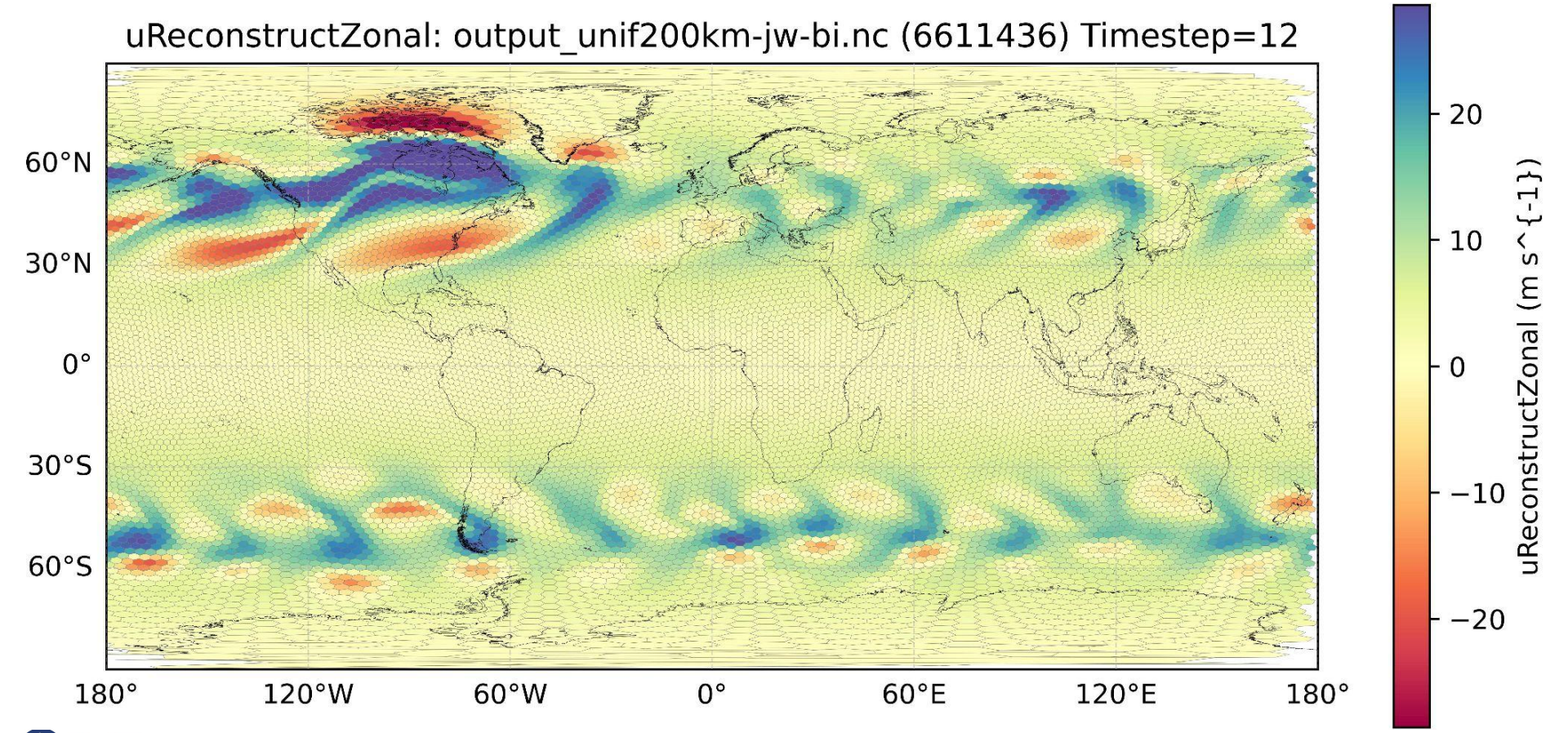
namelist.atmosphere

```
&nhyd_model
    config_dt = 720.0
    config_start_time = '0000-01-01_00:00:00'
    config_run_duration = '16_00:00:00'
    config_split_dynamics_transport = false
    config_number_of_sub_steps = 6
    config_dynamics_split_steps = 1
    config_h_mom_eddy_visc2    = 0.0
    config_h_mom_eddy_visc4    = 0.0
    config_v_mom_eddy_visc2    = 0.0
    config_h_theta_eddy_visc2  = 0.0
    config_h_theta_eddy_visc4  = 0.0
    config_v_theta_eddy_visc2  = 0.0
    config_horiz_mixing        = '2d_smagorinsky'
    config_len_disp            = 200000.
    config_u_vadv_order      = 3
    config_w_vadv_order      = 3
    config_theta_vadv_order  = 3
    config_scalar_vadv_order = 3
    config_theta_adv_order   = 3
    config_scalar_adv_order  = 3
    config_scalar_advection  = false
    config_positive_definite = false
    config_coef_3rd_order = 1.0
    config_monotonic = false
    config_epssm = 0.1
    config_smdiv = 0.1
/

&damping
    config_zd = 22000.0
    config_xnutr = 0.0
/

&decomposition
    config_block_decomp_file_prefix = 'unif200km_graph.info.part.'
/

&restart
    config_do_restart = false
/

&printout
    config_print_global_minmax_vel = true
    config_print_global_minmax_sca = false
/

&physics
    config_physics_suite = 'none'
/
```

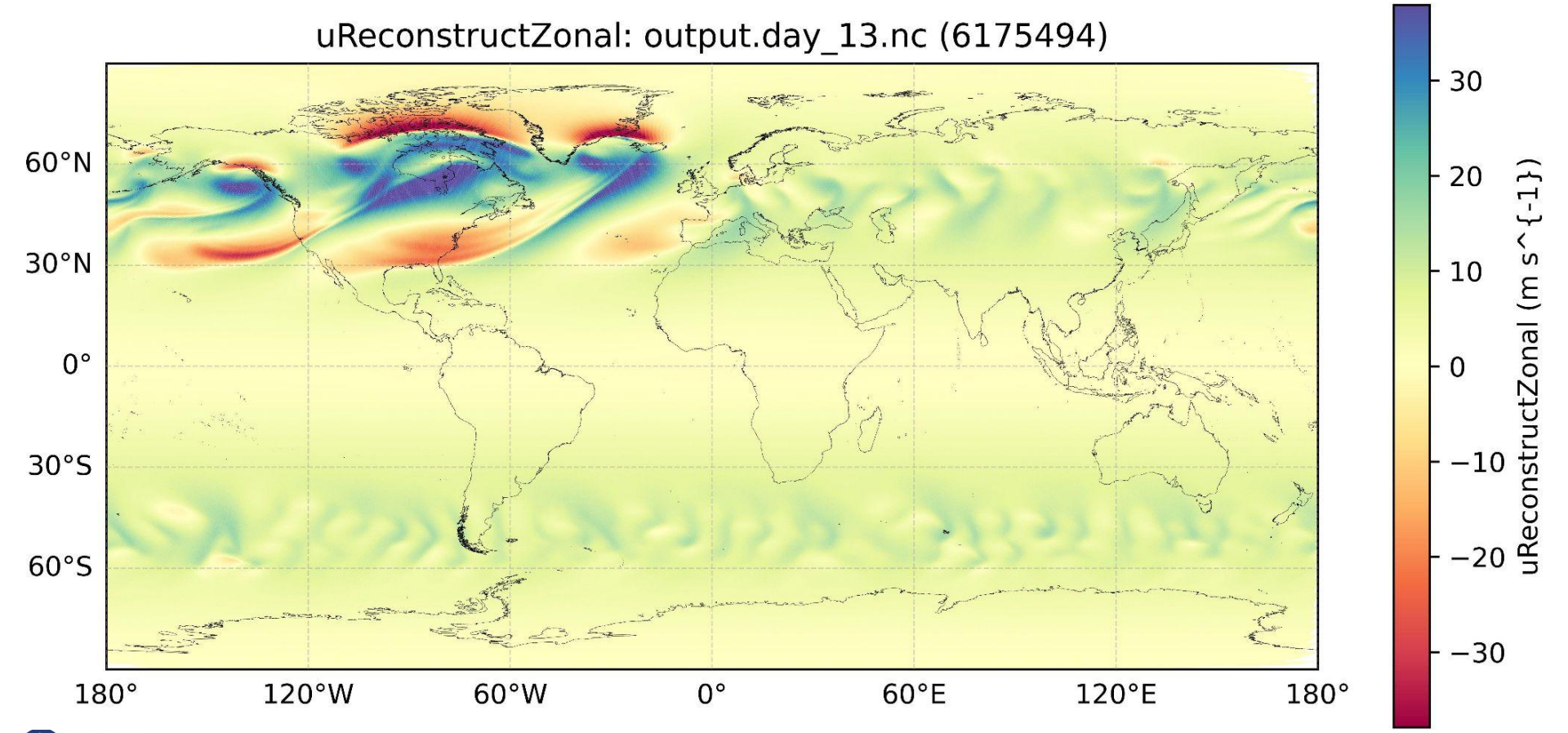
streams.atmosphere

```
<streams>

<immutable_stream name="input"
                  type="input"
                  filename_template="unif200km_jw-bi.init.nc"
                  input_interval="initial_only"/>

<immutable_stream name="restart"
                  type="input;output"
                  filename_template="restart.$Y-$M-$D_$h.$m.$s.nc"
                  input_interval="initial_only"
                  output_interval="5_00:00:00"/>

<stream name="output"
        type="output"
        filename_template="output_unif200km-jw-bi.nc"
        filename_interval="none"
        output_interval="24:00:00">

    <file name="stream_list.atmosphere.output"/>

</stream>

</streams>
```

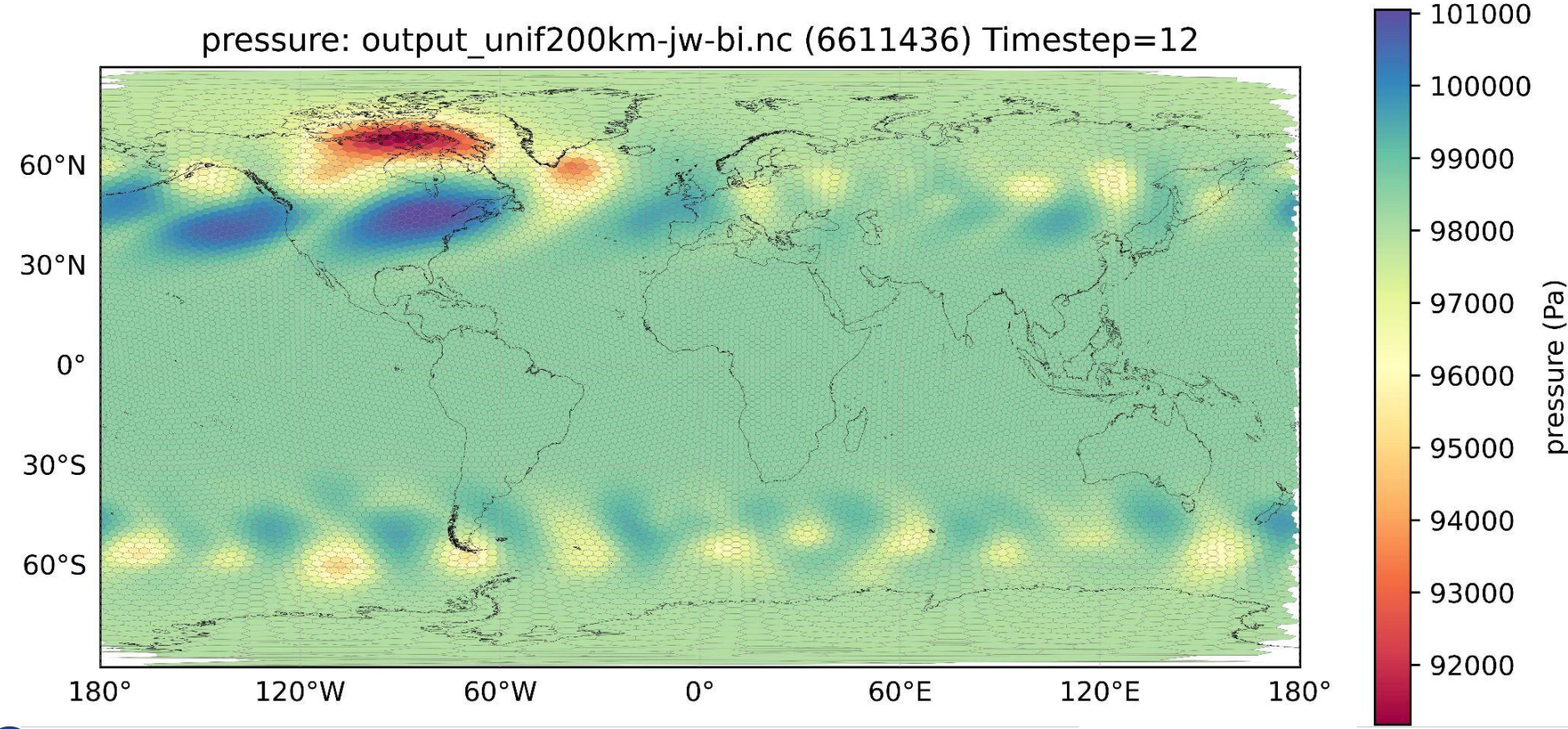
JW BI case with 200km grid resolution: Zonal wind field at day 13



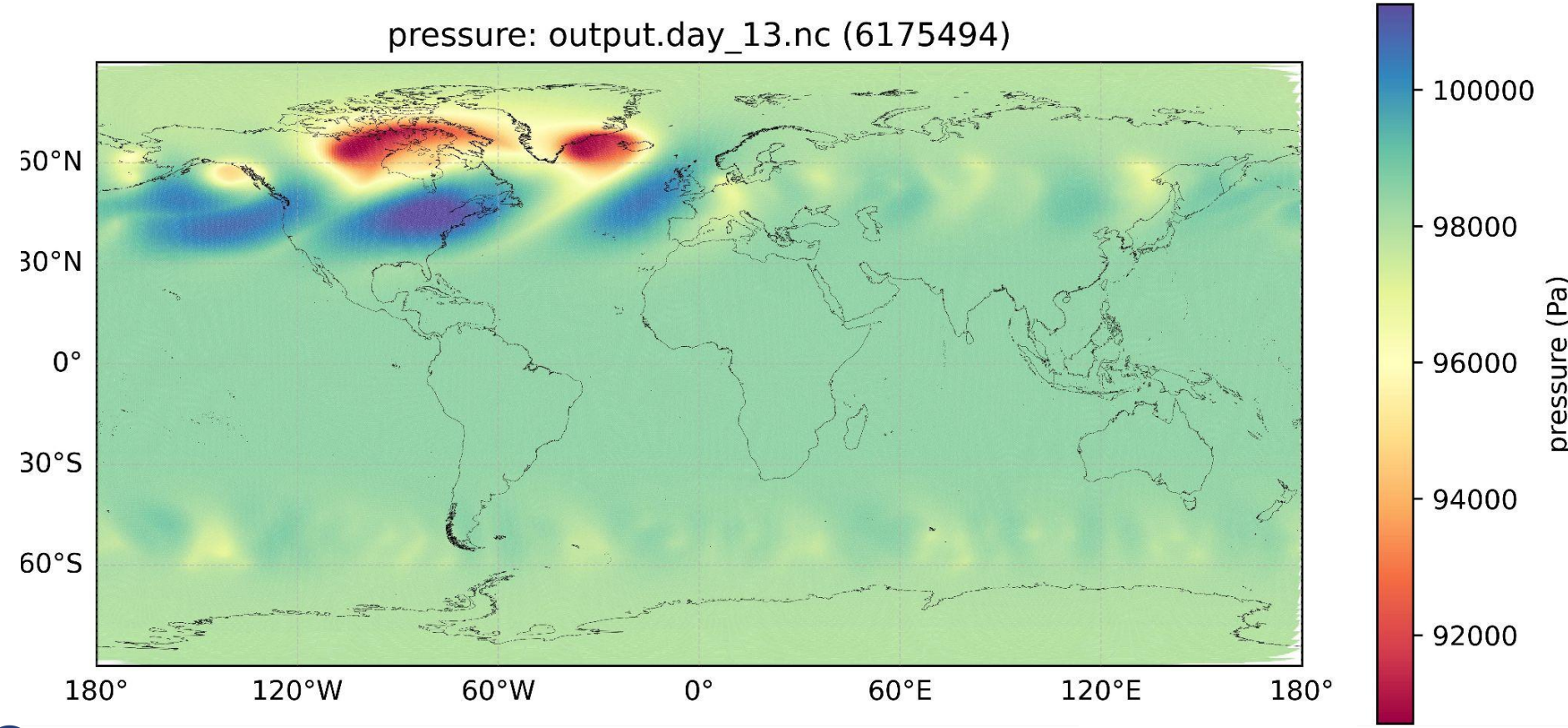
JW BI case with 50km grid resolution: Zonal wind field at day 13



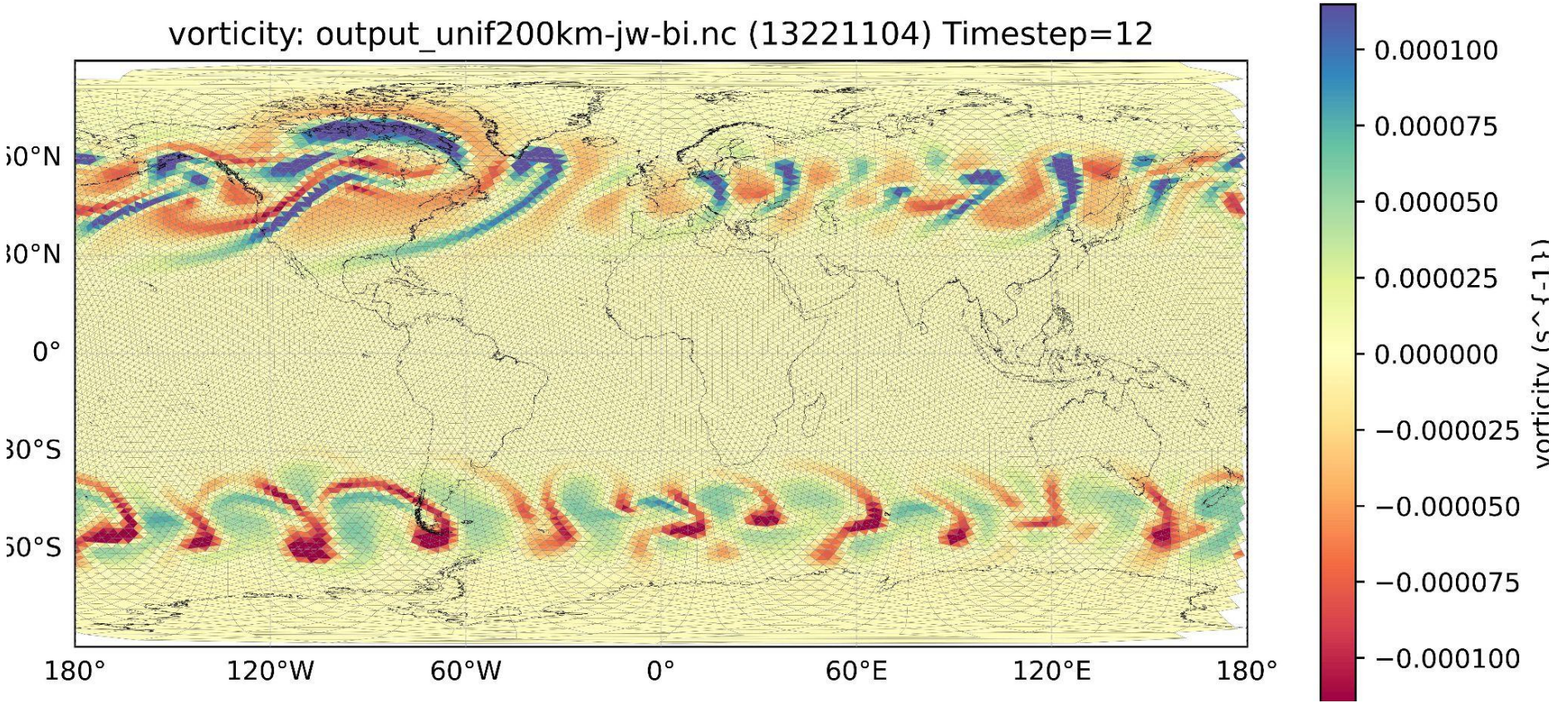
JW BI case with 200km grid resolution: Pressure field at day 13



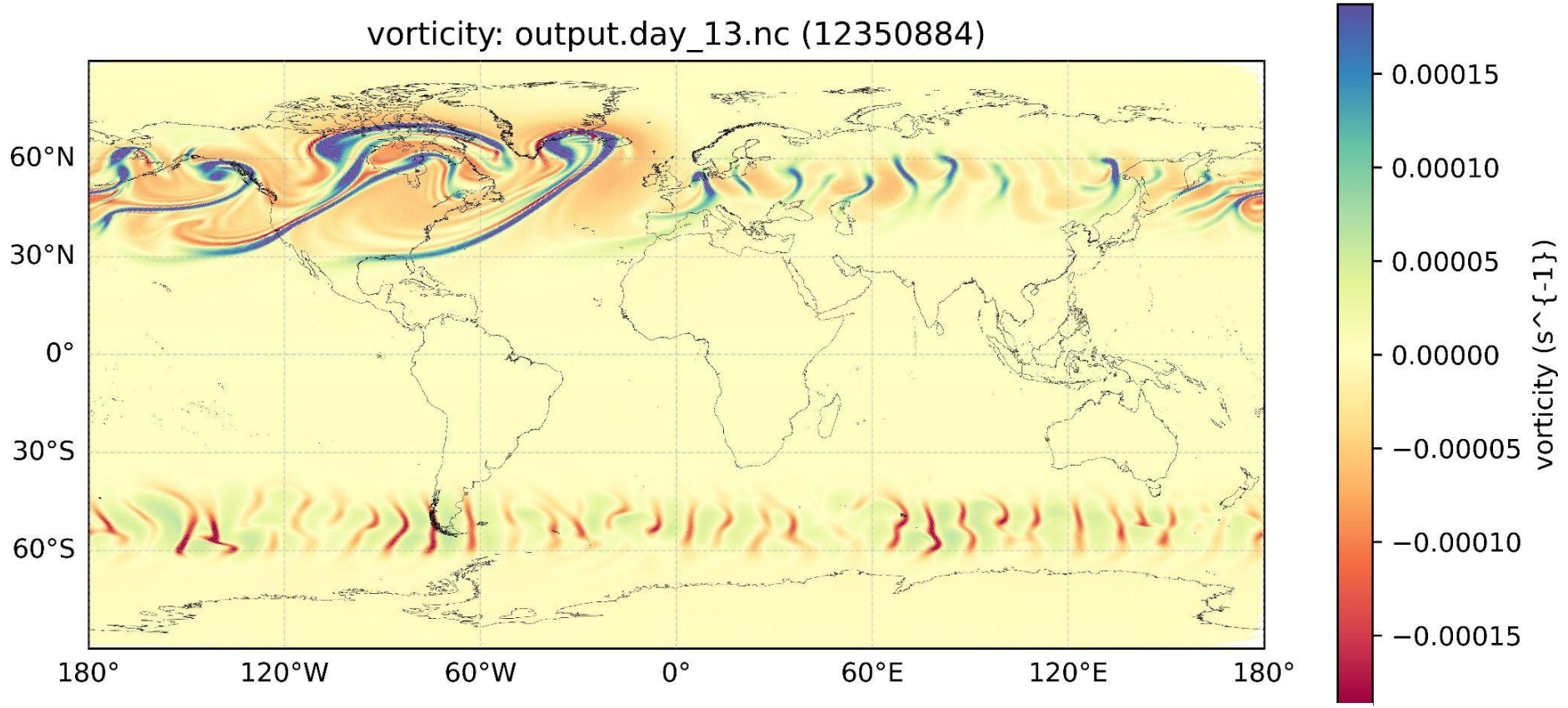
JW BI case with 50km grid resolution: Pressure field at day 13



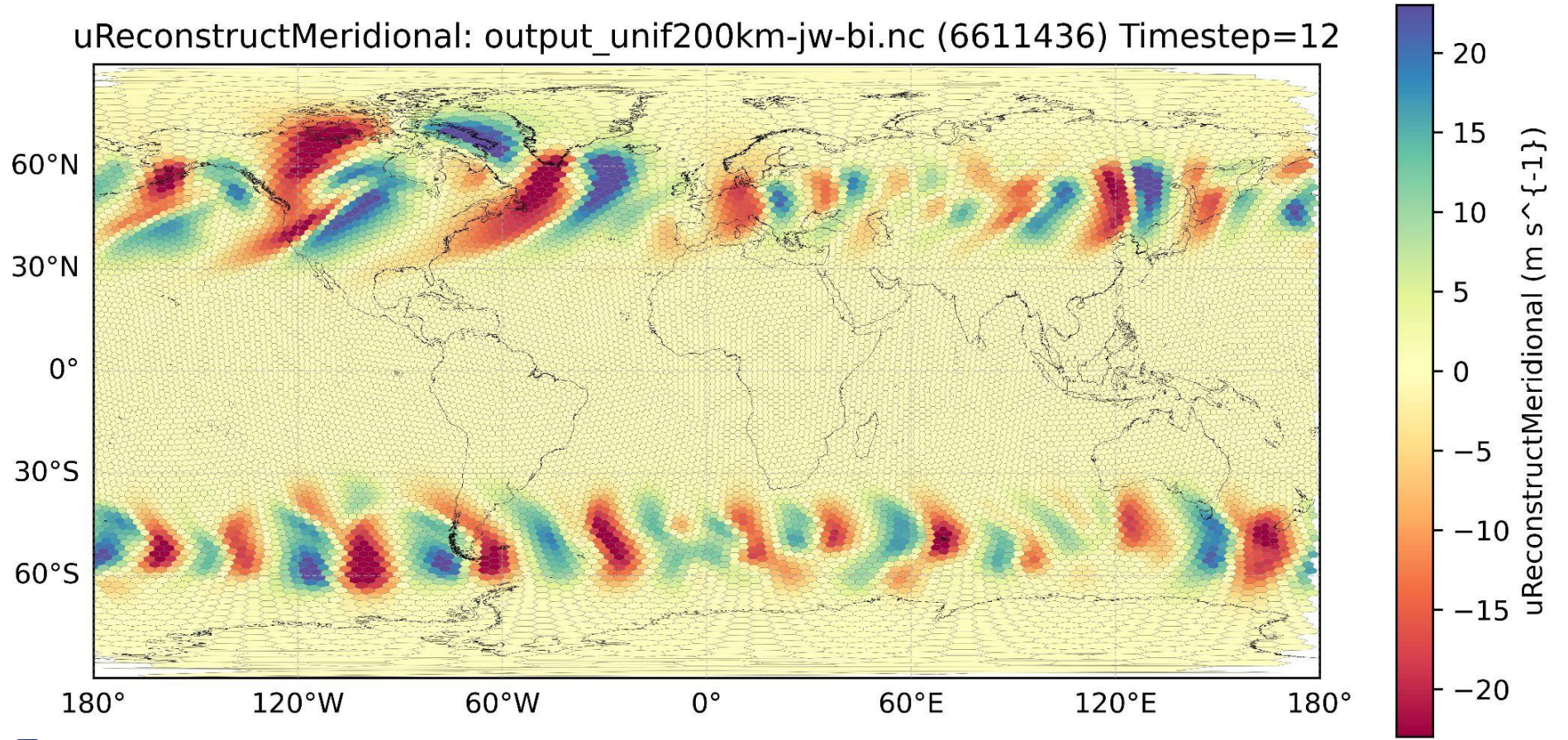
JW BI case with 200km grid resolution: Vorticity field at day 13



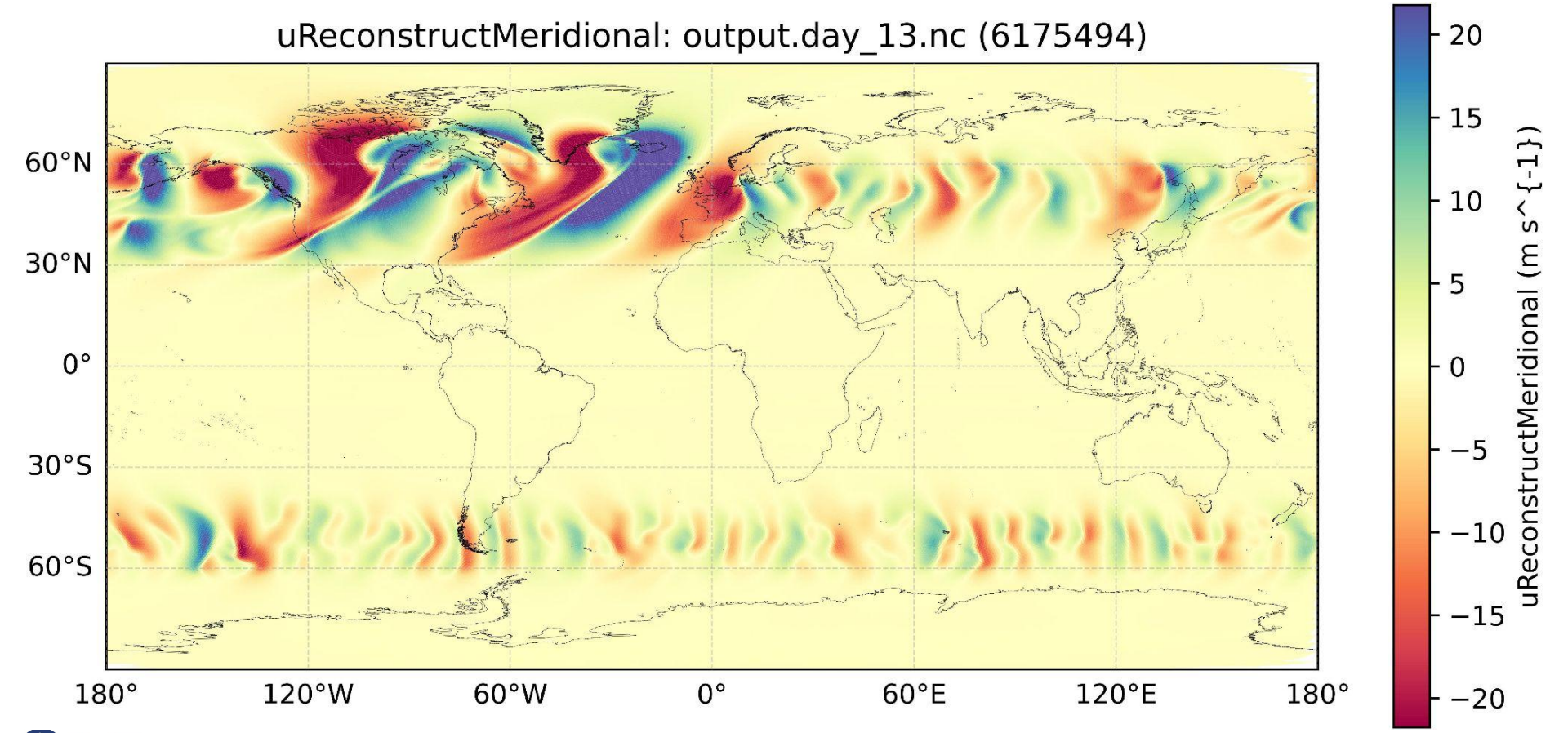
JW BI case with 50km grid resolution: Vorticity field at day 13



JW BI case with 200km grid resolution: Meridional wind field at day 13



JW BI case with 50km grid resolution: Meridional wind field at day 13



Animations

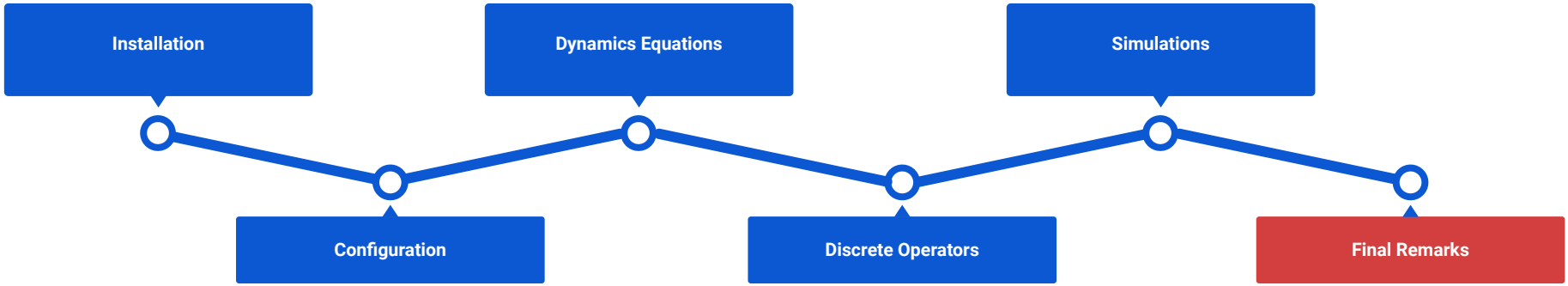
JW BI case
- 50km grid resolution (Jigsaw)
- Level 0 (ground)

Evolution of fields:

- Vorticity
- Zonal Wind
- Meridional Wind
- Pressure

Overview

Today's class:



DCMIP2016

Non-hydrostatic dynamical core design and intercomparison

https://www.earthsystemgrid.org/project/DCMIP_2016.html

Short name	Long name	Modeling center or group
ACME-A	Atmosphere model of the Accelerated Climate Model for Energy	Sandia National Laboratories and University of Colorado, Boulder, USA
CSU	Colorado State University Model	Colorado State University, USA
DYNAMICO	DYNAMical core on the ICOsahedron	Institut Pierre Simon Laplace (IPSL), France
FV^3	GFDL Finite-Volume Cubed-Sphere Dynamical Core	Geophysical Fluid Dynamics Laboratory, USA
FVM	Finite Volume Module of the Integrated Forecasting System	European Centre for Medium-Range Weather Forecasts
GEM	Global Environmental Multiscale model	Environment and Climate Change Canada, Canada
ICON	ICOsahedral Non-hydrostatic model	Max-Planck-Institut für Meteorologie, Germany
MPAS	Model for Prediction Across Scales	National Center for Atmospheric Research, USA
NICAM	Non-hydrostatic Icosahedral Atmospheric Model	AORI/JAMSTEC/AICS, Japan
OLAM	Ocean Land Atmosphere Model	Duke University/University of Miami, USA
Tempest	Tempest Non-hydrostatic Atmospheric Model	University of California, Davis, USA

https://gmd.copernicus.org/articles/10/4477/2017/

DCMIP2016

Non-hydrostatic dynamical core design and intercomparison

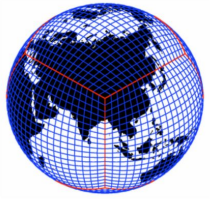
Short name	Equation set	Prognostic variables	Horizontal grid	Numerical method	Horizontal staggering
ACME-A	H/NH	$\boldsymbol{u}_\mathrm{h}$, w, ρ_s, $\rho_\mathrm{s}\theta$, Φ, $\rho_\mathrm{s}q_i$	Cubed sphere (Sect. 3.2)	SE	A grid
CSU	NH (unified)	ζ, D, w, p_s, θ_v, q_i	Geodesic (Sect. 3.4)	FV	Z grid
DYNAMICO	H/NH	$\boldsymbol{v}_\mathrm{h}$, $\rho_\mathrm{s}w$, ρ_s, $\rho_\mathrm{s}\theta_\mathrm{v}$, Φ, $\rho_\mathrm{s}q_i$	Geodesic (Sect. 3.4)	FV	C grid
FV^3	NH	$\boldsymbol{u}_\mathrm{h}$, w, ρ_s, $\rho_\mathrm{s}\theta_\mathrm{v}$, Φ, $\rho_\mathrm{s}q_i$	Cubed sphere (Sect. 3.2)	FV	D grid
FVM	NH (D)	ρ_d, $\boldsymbol{u}_\mathrm{h}$, w, θ', q_i	Octahedral (Sect. 3.6)	FV	A grid
GEM	NH	$\boldsymbol{u}_\mathrm{h}$, w, $\dot{\zeta}$, T_v, p, q_i	Yin–Yang (Sect. 3.7)	FD	C grid
ICON	NH (D)	$\boldsymbol{u}_\mathrm{h}$, w, ρ, θ_v, ρq_i	Icosahedral triangular (Sect. 3.3)	FV	C grid
MPAS	NH	$\rho_\mathrm{d}\boldsymbol{u}_\mathrm{h}$, $\rho_\mathrm{d}w$, ρ_d, $\rho_\mathrm{d}\theta_\mathrm{v}$, $\rho_\mathrm{d}q_i$	CCVT (Sect. 3.5)	FV	C grid
NICAM	NH	$\rho\boldsymbol{u}_\mathrm{h}$, ρw, ρ, ρe, ρq_i	Geodesic (Sect. 3.4)	FV	A grid
OLAM	NH (D)	$\rho\boldsymbol{u}_\mathrm{h}$, ρw, ρ, $\rho\theta_\mathrm{il}$, ρq_i	Geodesic (Sect. 3.4)	FV	C grid
Tempest	NH	$\boldsymbol{u}_\mathrm{h}$, w, ρ, $\rho\theta_\mathrm{v}$, ρq_i	Cubed sphere (Sect. 3.2)	SE	A grid

https://gmd.copernicus.org/articles/10/4477/2017/

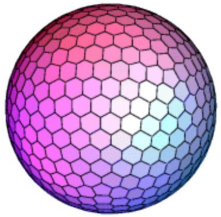
DCMIP2016

Non-hydrostatic dynamical core design and intercomparison

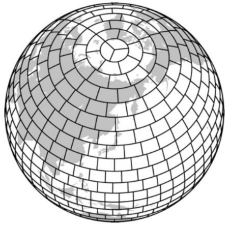
DCMIP-2016 Models



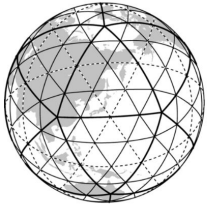
- ACME (E3SM) (DoE, CU)
- FV3 (GFDL)
- Tempest (UC Davis)



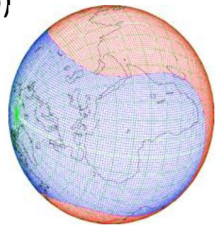
- CSU_LZ (CSU)
- OLAM (U. Miami)
- NICAM (Riken, U. Tokyo)
- MPAS (NCAR)



- FVM (ECMWF)



- ICON (DWD & MPI, Germany)
- DYNAMICO (LMD, IPSL, France), hydrostatic

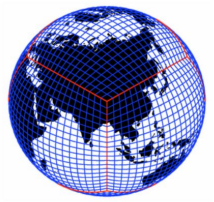


- GEM (Environment Canada)

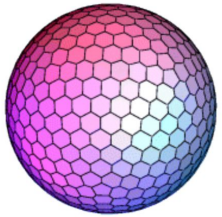
DCMIP2016

Non-hydrostatic dynamical core design and intercomparison

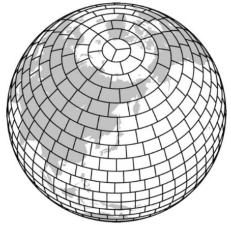
DCMIP-2016 Models



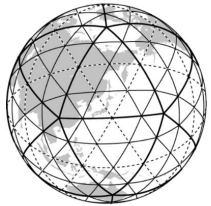
- ACME (E3SM) (DoE, CU)
- FV3 (GFDL)
- Tempest (UC Davis)



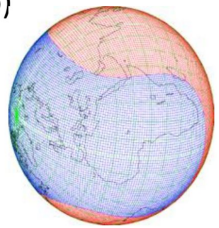
- CSU_LZ (CSU)
- OLAM (U. Miami)
- NICAM (Riken, U. Tokyo)
- MPAS (NCAR)



- FVM (ECMWF)



- ICON (DWD & MPI, Germany)
- DYNAMICO (LMD, IPSL, France), hydrostatic

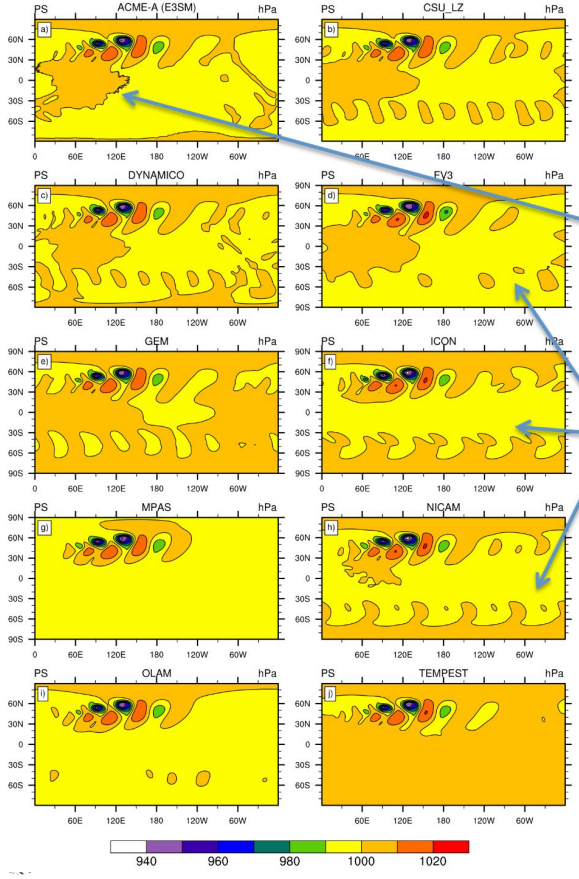


- GEM (Environment Canada)

https://www2.cesm.ucar.edu/events/wg-meetings/2018/presentations/amwg/jablonowski.pdf

DCMIP2016

Snapshots of the dry baroclinic wave



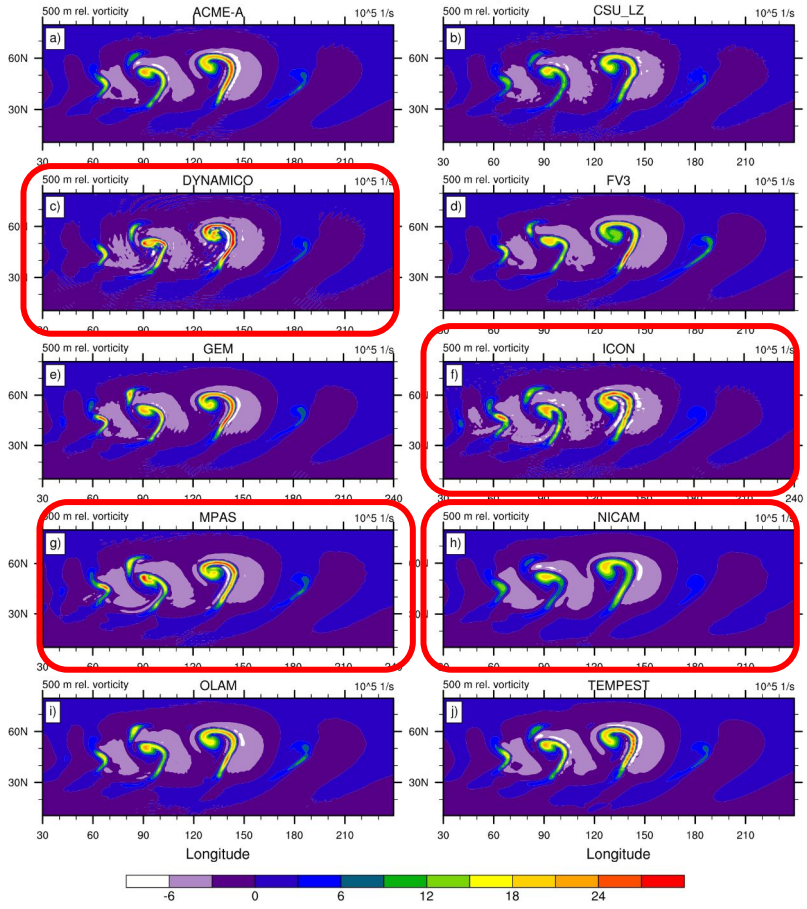
Surface pressure at day 10 (Δx=110 km): overall patterns similar, details differ

- Some Gibb's ringing in ACME
- Some grid imprinting (wave 4 and wave 5 signals) in CSU_LZ, DYNAMICO, FV3, ICON, NICAM, apparent in the Southern Hemispheres

https://www2.cesm.ucar.edu/events/wg-meetings/2018/presentations/amwg/jablonowski.pdf

DCMIP2016

Relative vorticity in the moist baroclinic wave

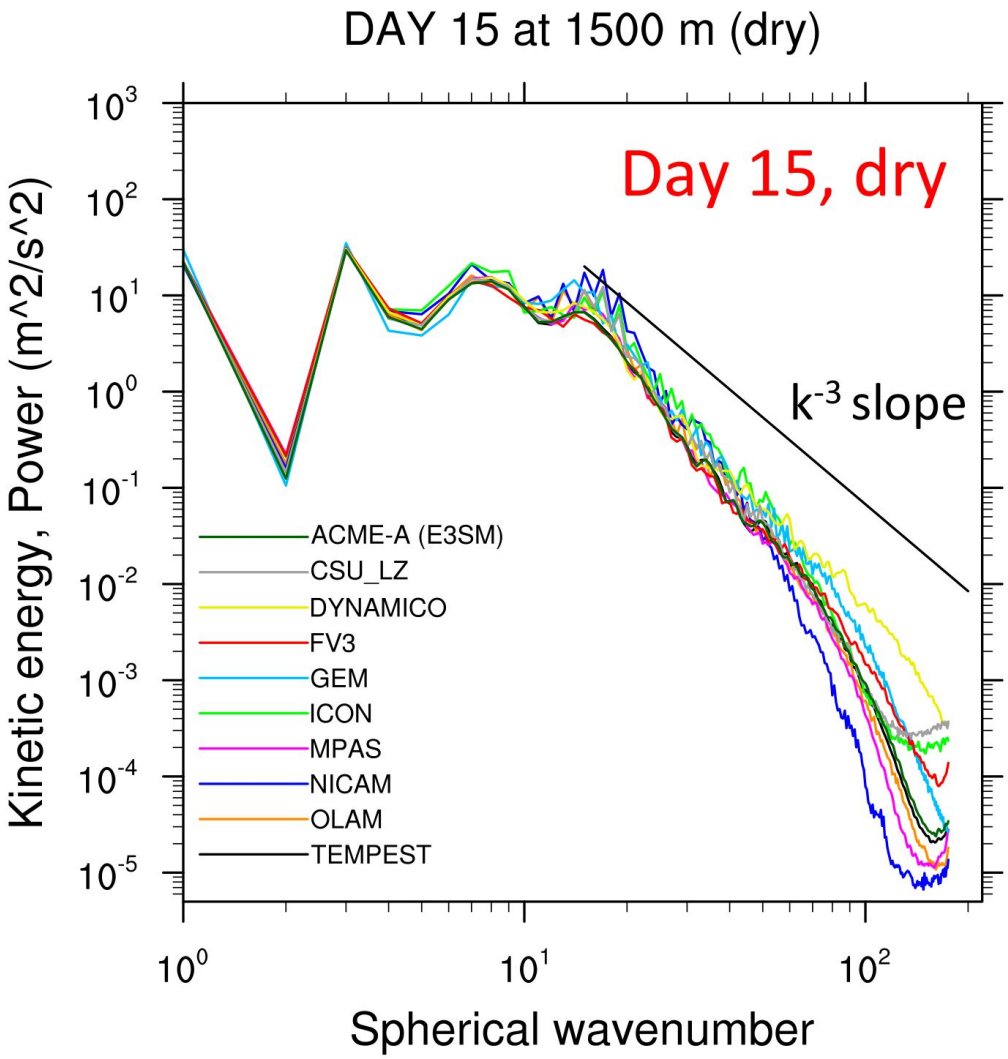


500 m relative vorticity at day 10 (Δx=110 km): overall patterns similar, details differ

- Maxima and minima differ (by about 30%) and are found in very narrow strips (challenges the 110 km grid spacing)
- Vorticity highlights noise and the diffusive properties of the model

https://www2.cesm.ucar.edu/events/wg-meetings/2018/presentations/amwg/jablonowski.pdf

DCMIP2016



https://www2.cesm.ucar.edu/events/wg-meetings/2018/presentations/amwg/jablonowski.pdf

HIWPP - High Impact Weather Prediction Project

Non-hydrostatic dynamical core tests: Results from idealized test cases

The “new” NCEP Global Model, replacing Spectral (2019)

Participating Dynamical Cores:

The five candidate dycores are listed below, with sponsors in parentheses.

- **FV3** (GFDL) – Cubed sphere grid, finite-volume discretization. (non-hydrostatic version of the hydrostatic core described in Lin, 2004).
- **MPAS** (NCAR) – Unstructured grid with C-grid variable staggering (Skamarock et al, 2012).
- **NEPTUNE** (NRL) – Flexible cubed sphere or icosahedral grid using a spectral element discretization with the Non-hydrostatic Unified Model of the Atmosphere (NUMA) core (Giraldo et al, 2014).
- **NIM** (ESRL) – Non-hydrostatic Icosahedral Model (unstaggered finite-volume A-grid implementation).
- **NMMUJ** (EMC) – Finite-difference cubed-sphere grid version of the B-grid lat/lon grid core described in Janjic and Gall (2012). The construction of the ‘uniform jacobian’ cubed sphere grid is described in NCEP office note 467, available at http://www.lib.ncep.noaa.gov/ncepofficenotes/2010s/.

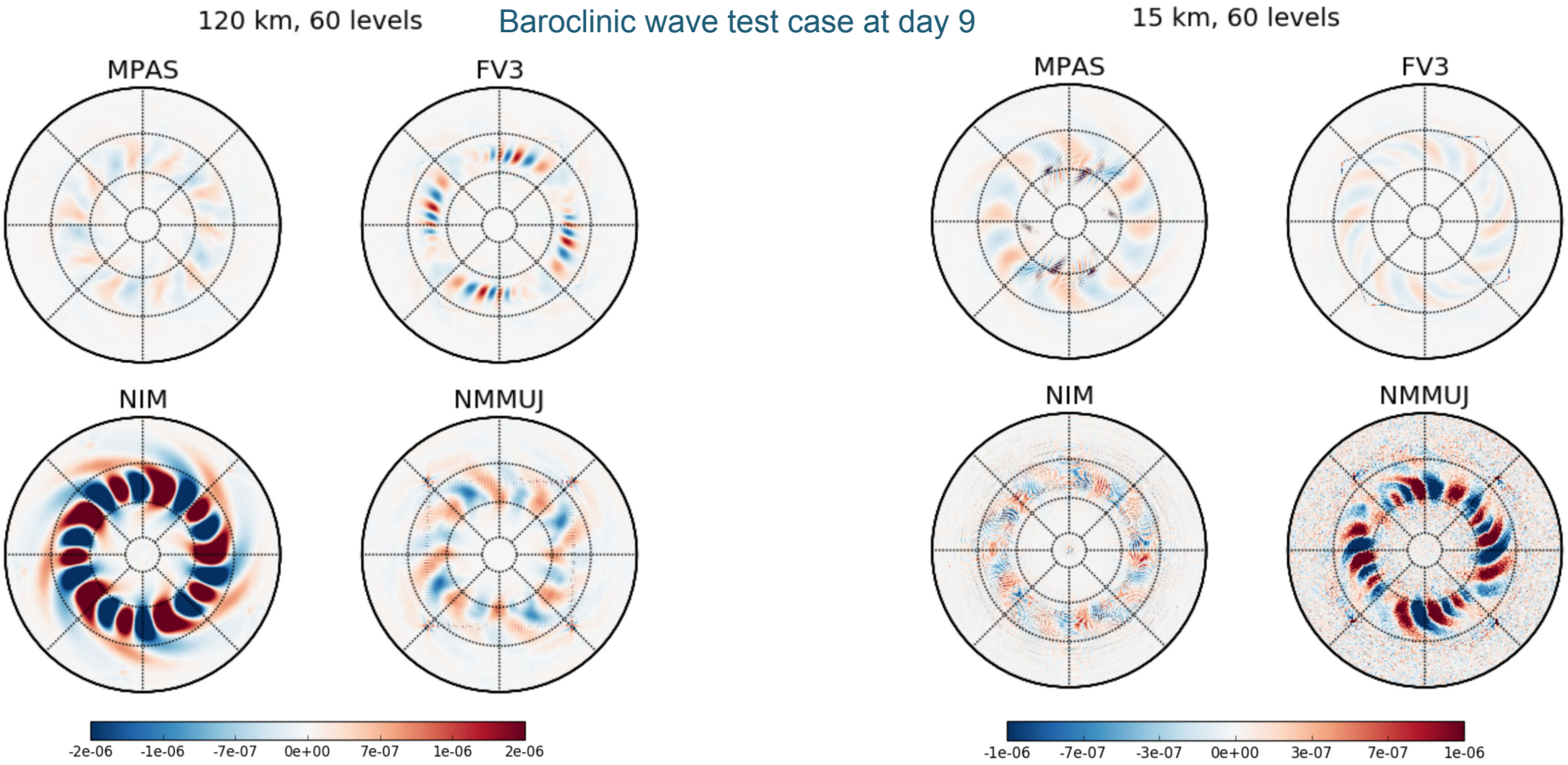
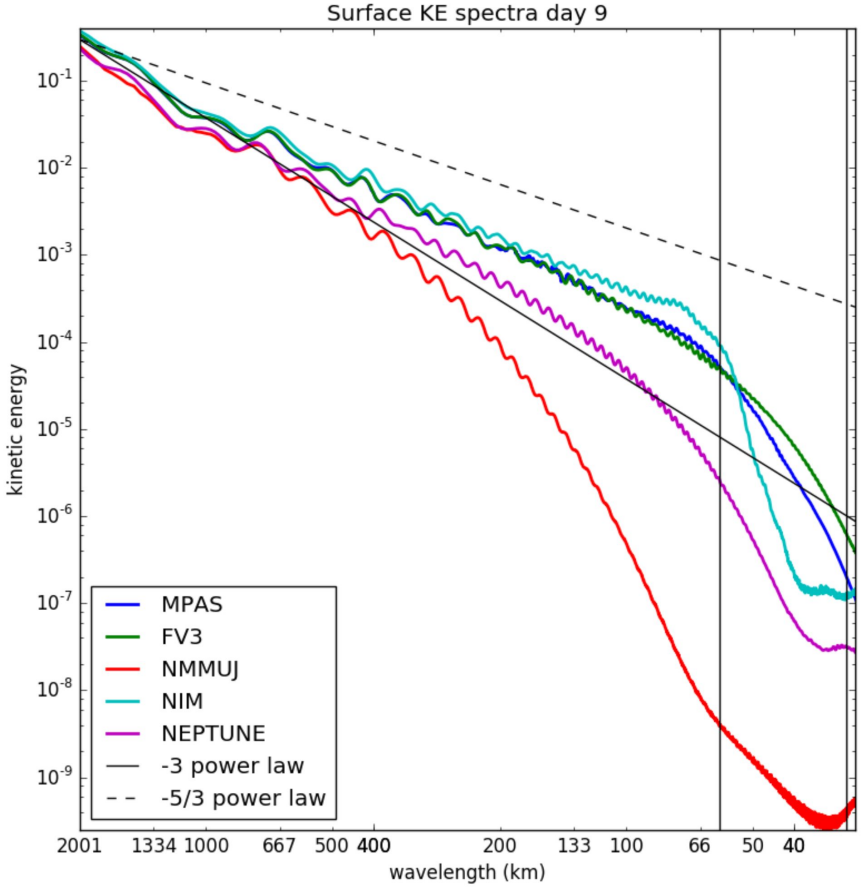


Figure 1: Plots of Southern Hemisphere relative vorticity at 850hPa (with the zonal mean removed) for the high-resolution (nominally 15-km, with 60 levels) baroclinic wave test case at day 9. The outer edge of the plot is 20 degrees south latitude.

Figure 8: Near surface global kinetic energy spectra (m^2/s^2) for day 9 of the 15-km/60 level baroclinic wave test case solutions. The x-axis is wavelength in km, with values ranging from total wavenumber 20 (~ 2000 km) to wavenumber 1440 (~ 28 km), with a log-scale in total wavenumber. Two reference lines are plotted, one with a slope corresponding to a -3 power-law



MONAN

Avaliação de Núcleos Dinâmicos

Software

Funcionalidade

DIMNT - INPE

Ariane Frassoni, Julio P. Fernandez, Marcelo Barbio Rosa, Bárbara Yamada

D. Arsego, C. Bastarz, J. G. Mattos, J. R. Rozante

Agradecimentos a Denis Eiras

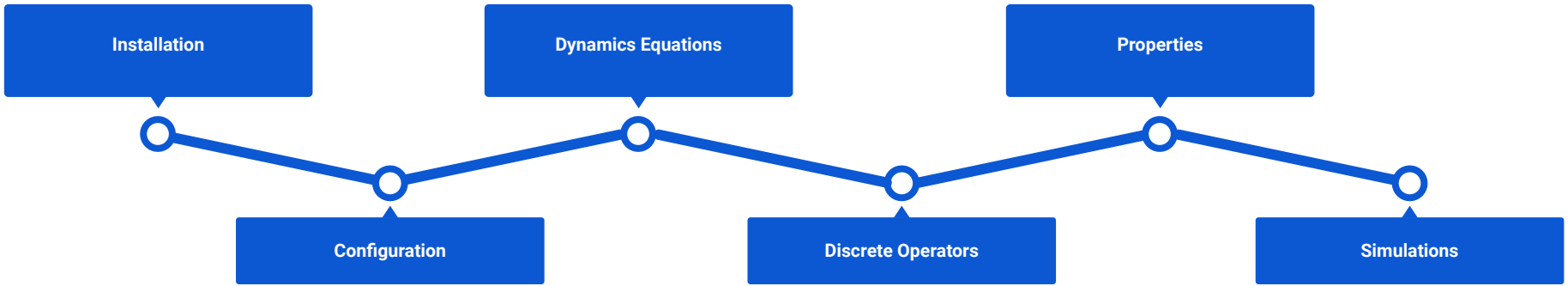
Escolha do MPAS para a base do MONAN

Detalhes em:

https://monanadmin.github.io/monan_cc_docs/

Overview

Today's class:



Final Activity!

Today's activity

Perform a simulation in MPAS using the idealized test case

Main goal:

Learn the main processing steps

Main steps:

- Install/Make
- Set parameter files
- Run
- Post-process
- Show 2-3 figures of the results (hand in)

That is all for today…

“All models are wrong, but some are useful”

— George Box

More at: www.ime.usp.br/~pedrosp

Thanks!