Numerical Stability Condition

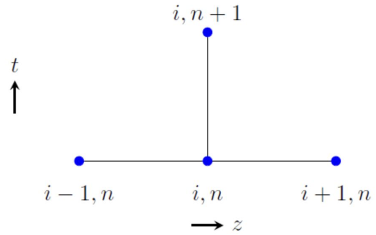
Wind speed vs time step

Intuition of numerical stability condition:

For Explicit Finite Volume (and Finite Difference) method,

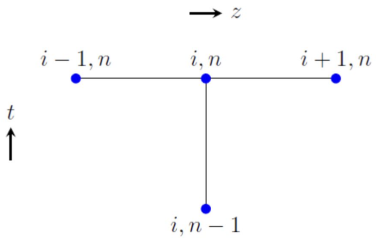
The simulation will be numerically unstable if the time step is too long that an air parcel passes through a cell in a time step.

Explicit scheme:



Stencil for the explicit method

Implicit scheme:



Stencil for the implicit method

https://yaredwb.github.io/FDM1D/

CFL condition

https://en.wikipedia.org/wiki/Courant%E2%80%93Friedrichs%E2%80%93Lewy_condition

Courant–Friedrichs–Lewy condition

From Wikipedia, the free encyclopedia

In mathematics, the **convergence condition by Courant–Friedrichs–Lewy** is a necessary condition for convergence while solving certain partial differential equations (usually hyperbolic PDEs) numerically. It arises in the numerical analysis of explicit time integration schemes, when these are used for the numerical solution. As a consequence, the time step must be less than a certain time in many explicit time-marching computer simulations, otherwise the simulation produces incorrect results. The condition is named after Richard Courant, Kurt Friedrichs, and Hans Lewy who described it in their 1928 paper.[1]

Contents [show]

Heuristic description [edit]

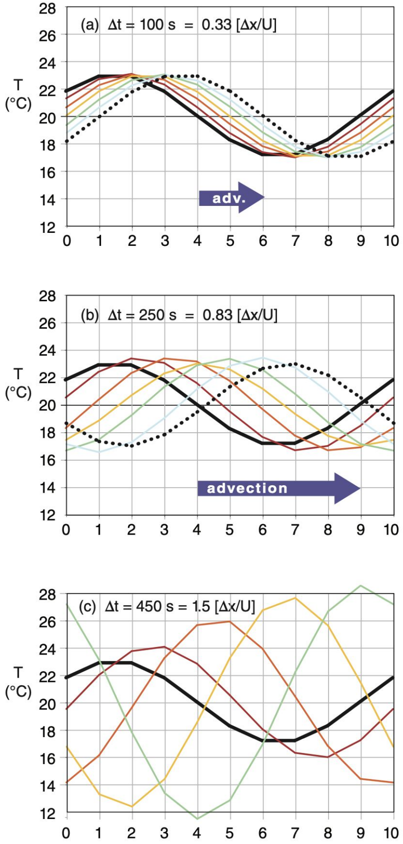
The principle behind the condition is that, for example, if a wave is moving across a discrete spatial grid and we want to compute its amplitude at discrete time steps of equal duration,[2] then this duration must be less than the time for the wave to travel to adjacent grid points. As a corollary, when the grid point separation is reduced, the upper limit for the time step also decreases. In essence, the numerical domain of dependence of any point in space and time (as determined by initial conditions and the parameters of the approximation scheme) must include the analytical domain of dependence (wherein the initial conditions have an effect on the exact value of the solution at that point) to assure that the scheme can access the information required to form the solution.

The wave illustration

Stable

Still stable

Unstable because of large time step



$$\Delta t \le \frac{\Delta X}{|U|} \qquad \bullet(20.18)$$

with similar requirements in the y and z directions. This is known as the **Courant-Friedrichs-Lewy (CFL) stability criterion**, or the **Courant condition**. When modelers use finer mesh grids with smaller ΔX values, they must also reduce Δt to preserve numerical stability. The combined effect

Figure 20.11 (at left)

Examples of numerical stability for advection, with ΔX = 3 km and U = 10 m s^{-1}. Thick black line is initial condition, and the forecast after each time step is shown as rainbow colors, with the last (6th) step dotted. A temperature signal of wavelength 10·ΔX is numerically stable for time steps Δt of (a) 100 s and (b) 250 s, but (c) = 450 s exceeds the CFL criterion, and the solution blows up (i.e., the wave amplitude increases without bound).

Stull 2016 Practical Meteorology: An Algebra-based Survey of Atmospheric Science
https://www.eoas.ubc.ca/books/Practical_Meteorology/
Chapter 20 Numerical Weather Prediction (NWP)

Time step determination practices

Time step used $\propto$ grid spacing.

- Assumed a max wind speed in the domain over whole simulation duration.
- Safety buffer
- Experience from old popular models (MM5, WRF)

Usually use the factor:

6 s/km or smaller

3.3.1 RK3 Time Step Constraint

The RK3 time step is limited by the advective Courant number $u\Delta t/\Delta x$ and the user's choice of advection schemes— users can choose 2^{nd} through 6^{th} order discretizations for the advection terms. The time-step limitations for 1D advection in the RK3 scheme using these advection schemes is given in Wicker and Skamarock (2002), and is reproduced here.

Time Scheme	Spatial order			
	3rd	4th	5th	6th
Leapfrog	*Unstable*	0.72	*Unstable*	0.62
RK2	0.88	*Unstable*	0.30	*Unstable*
RK3	1.61	1.26	1.42	1.08

Table 3.1: Maximum stable Courant numbers for one-dimensional linear advection. From Wicker and Skamarock (2002).

As is indicated in the table, the maximum stable Courant numbers for advection in the RK3 scheme are almost a factor of two greater than those for the leapfrog time-integration scheme. For advection in three spatial dimensions, the maximum stable Courant number is $1/\sqrt{3}$ times the Courant numbers given in Table 3.1. For stability, the time step used in the ARW should produce a maximum Courant number less than that given by theory. Thus, for 3D applications, the time step should satisfy the following equation:

$$\Delta t_{max} < \frac{Cr_{theory}}{\sqrt{3}} \cdot \frac{\Delta x}{u_{max}}, \qquad (3.55)$$

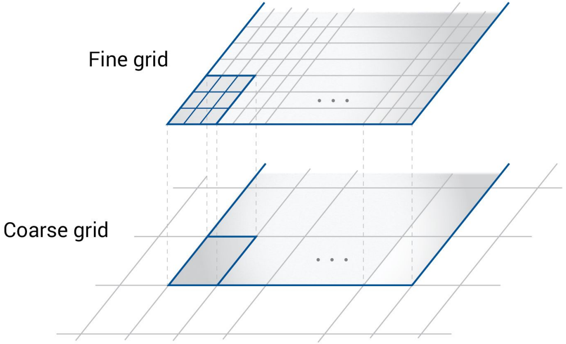
where Cr_{theory} is the Courant number taken from the RK3 entry in Table 3.1 and u_{max} is the maximum velocity expected in the simulation. For example in real-data applications, where jet stream winds may reach as high as 100 ms^{-1}, the maximum time step would be approximately 80 s on a Δx = 10 km grid using 5^{th} order advection. For convection-permitting resolutions (typically $\Delta x \leq$ 5 km), the vertical velocities in convective updrafts produce the stability-limiting Courant numbers. Given the additional constraint from the time splitting, and to provide a safety buffer, we usually choose a time step that is approximately 25% less than that given by (3.55). This time step is typically a factor of two greater than that used in leapfrog-based models. For those users familiar with the MM5 model, the rule of thumb for choosing a time step is that the time step, in seconds, should be approximately 3 times the horizontal grid distance, in kilometers. For the ARW, the time step (in seconds) should be approximately 6 times the grid distance (in kilometers).

Difficult paper, just a reference that the 6 s/km factor has some justification behind.

Skamarock, W. C., Klemp, J. B., Dudhia, J., Gill, D. O., Liu, Z., Berner, J., … Huang, X. -yu. (2021). A Description of the Advanced Research WRF Model Version 4.3 (No. NCAR/TN-556+STR). doi:10.5065/1dfh-6p97

Importance of time-stepping

Nesting in WRF



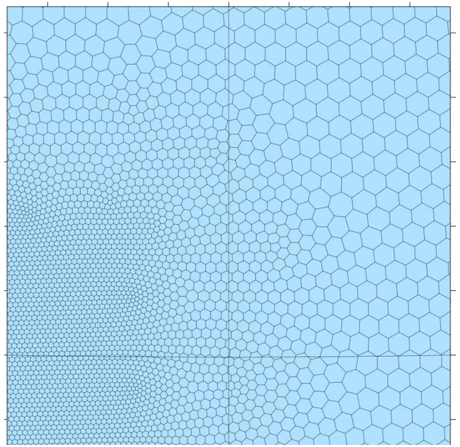
Δt

3Δt

Coarse grid uses large time step

- Computationally lite

MPAS-A resolution transition



open-source MPAS-A: whole mesh uses Δt

Coarse grid uses small time step

- Computationally intensive
- Unnecessary

Horizontally explicit time-integration scheme

MPAS
Model for Prediction Across Scales

Time Integration

3rd Order Runge-Kutta time integration

LHS:
Time-derivatives

RHS:
Spatial operators

advance $\phi^t \rightarrow \phi^{t+\Delta t}$

$$\frac{\partial U}{\partial t} = RHS_u$$

$$\frac{\partial W}{\partial t} = RHS_w$$

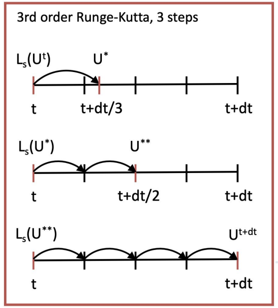
⋮

$$\phi^* = \phi^t + \frac{\Delta t}{3} RHS(\phi^t)$$

$$\phi^{**} = \phi^t + \frac{\Delta t}{2} RHS(\phi^*)$$

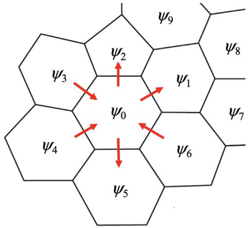
$$\phi^{t+\Delta t} = \phi^t + \Delta t \, RHS(\phi^{**})$$

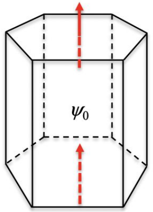
Amplification factor $\phi_t = ik\phi; \quad \phi^{n+1} = A\phi^n; \quad |A| = 1 - \frac{(k\Delta t)^4}{24}$



MPAS Tutorial - Dynamics
https://www2.mmm.ucar.edu/projects/mpas/tutorial/Boulder2019/slides/07.MPAS_solver.pdf

HEVI Solver





Horizontally Explicit
Δx affects numerical stability!

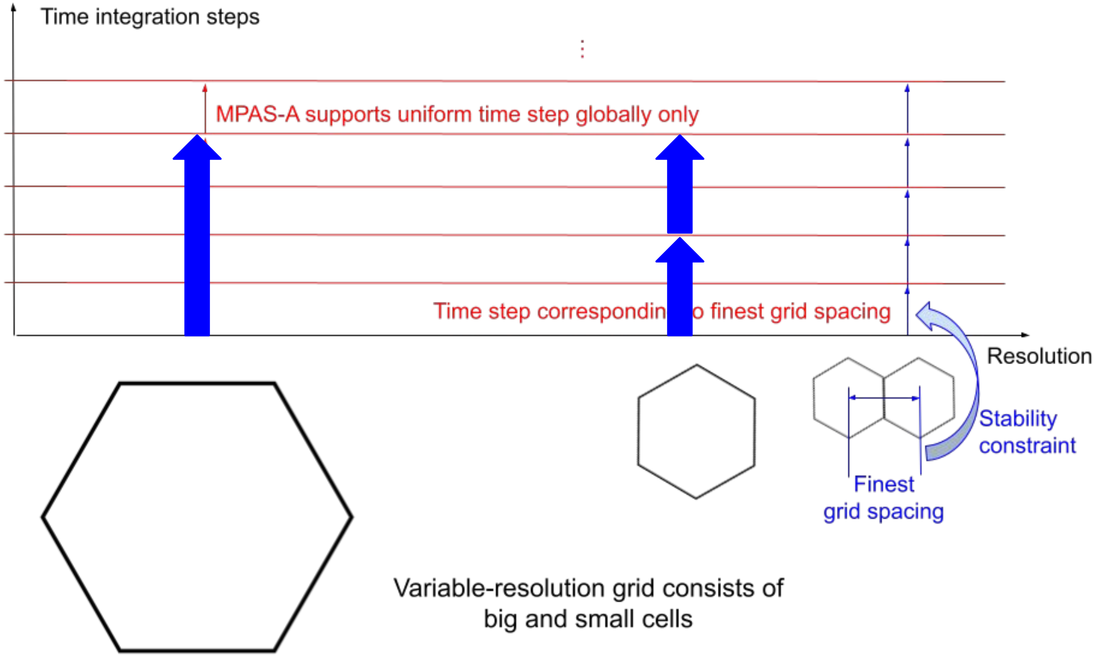
Vertically Implicit
Δz is safe in the dynamical core

Courant-Friedrichs-Lewy (CFL) Stability Condition

$$C = \frac{u\Delta t}{\Delta x} \leq C_{max}$$

For RK3 scheme, C_{max} = 1.73

Uniform time-stepping in MPAS-A



Computational cost:

- A lot of time steps for big cells
  - high computational cost!
  - not necessary for its own stability condition

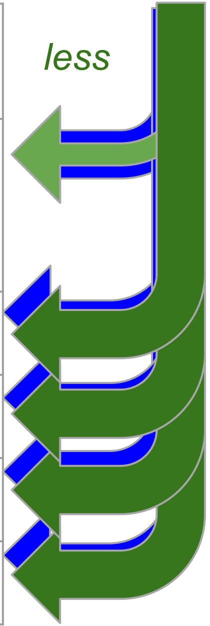
Can we "jump steps" in unstructured mesh?

Aspects competing for computation resource

(Computer resources are always in high demand for NWP no matter how big a cluster one has)

At fixed computational resource

Dimension	**Range of choices**	**Assigned resources**
Horizontal domain size / resolution distribution *Make this more resource-efficient*	e.g. high-res domain size transition zone width	*Redistribute saved resource to other aspects*
Vertical resolution	50 to > 100 vertical layers	*More*
Length of forecast	3 days - 9 days	*More*
Update frequency	daily - hourly	*More*
Number of ensemble members	20 - 50	*More*

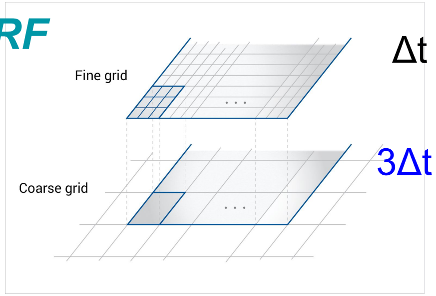


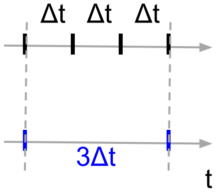
Flexible time discretization technique

CPAS Hierarchical Time-Stepping

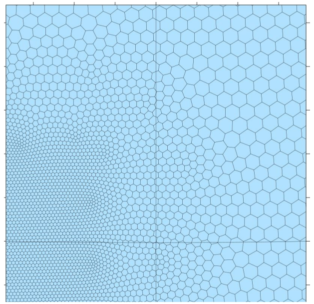
Hierarchical Time-Stepping (HTS) in CPAS

WRF





MPAS-A

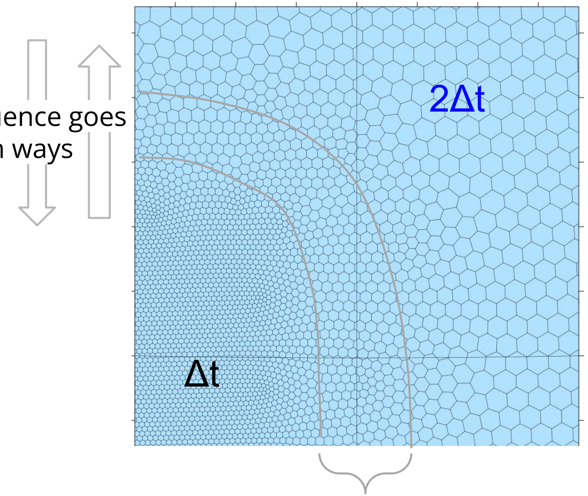


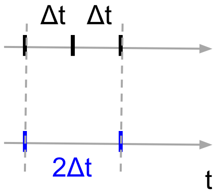
Refinement region:

- any shape
- smooth transition

CPAS Hierarchical Time-stepping

Influence goes both ways





illustrative diagram only

- Both time integration computation using 2Δt and Δt in this **buffer region** => overhead.
- freeing the outer region to use 2x Δt, hierarchically

CPAS = Clustertech Platform for Atmospheric Simulation, https://cpas.earth/

Flexible space discretization technique

CPAS Customizable Unstructured Mesh Generation
Before Lab

User-friendly Web Graphical Interface

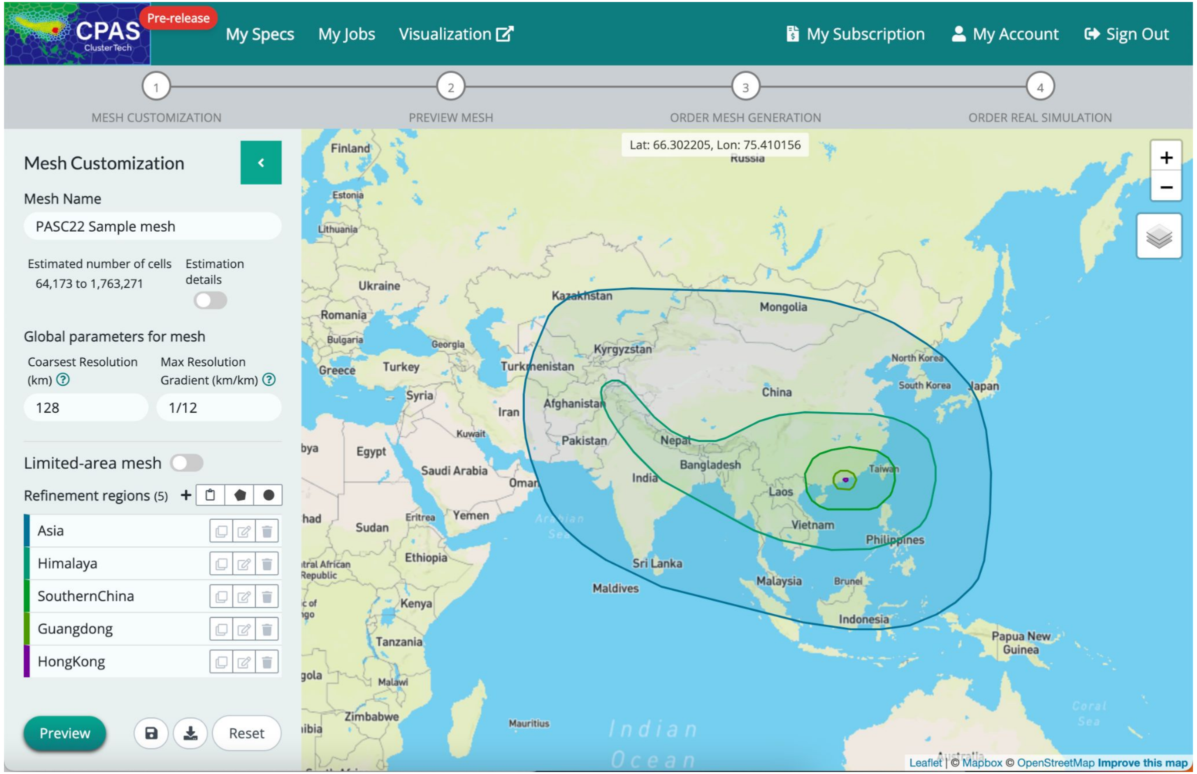
User defines refinement regions of arbitrary shapes and properties

- GEOJSON file for mesh specification.

PASC22 Sample mesh.json


```
{
  "id": "20220427-064300-d91868b0-c4f6-4126-8473-d47d4a2f6228",
  "name": "PASC22 Sample mesh",
  "schema_version": "2019-05-09",
  "global_options": {
    "coarsest_resolution_km": 128,
    "max_resolution_gradient": 0.08333333333333333
  },
  "regional_options": {
    "type": "FeatureCollection",
    "features": [
      {
        "type": "Feature",
        "geometry": {
          "type": "Polygon",
          "coordinates": [
            [
              [
                136.625977,
                4.056056
              ],
              [
                137.15332,
                4.740675
              ],
```


"Spend resolution at where you want!"

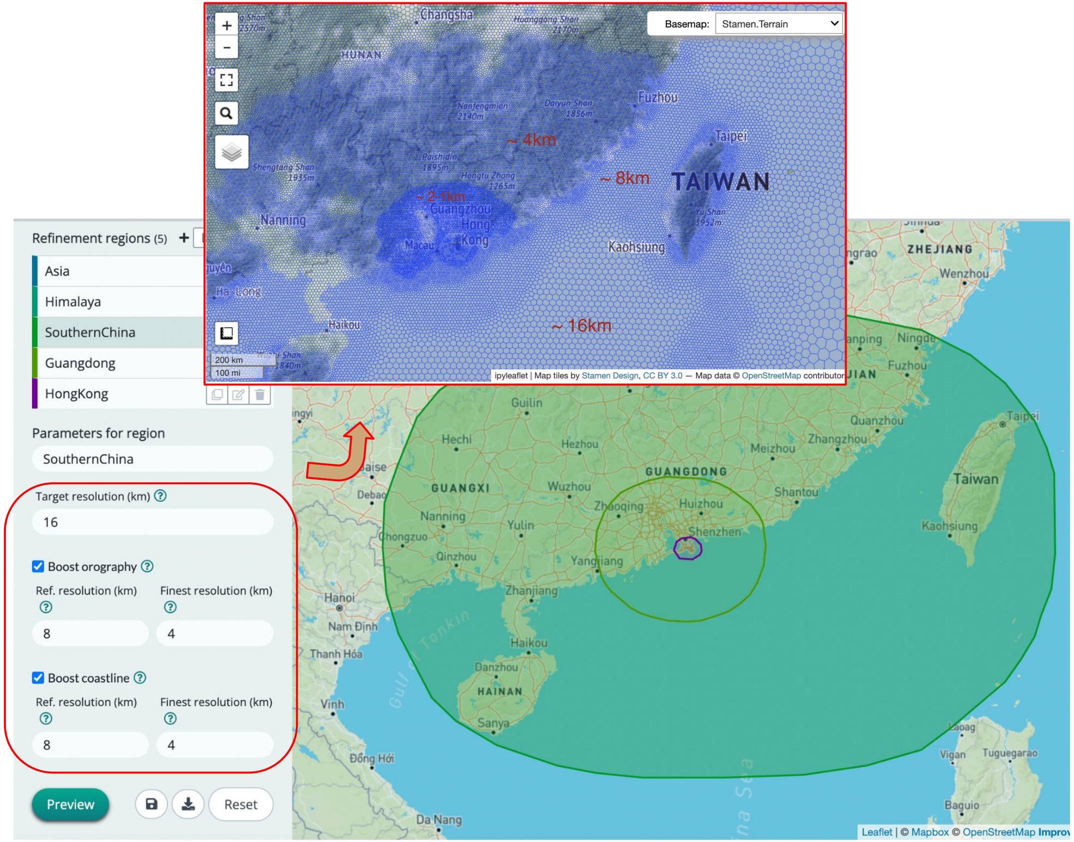


Geographical features that needs higher resolution

The system looks up geographical data automatically

- Orography
  - Wind dragging / blocking effect
  - Temperature at raised altitude
- Coastline
  - Land-sea breeze
  - Diurnal temperature change on land vs water

- Significant change of values over short horizontal distance:
  - Terrain height
  - Water / land land-cover
- Needs high resolution to resolve the variation



Flexible space discretization technique

CPAS Customizable Unstructured Mesh Generation
After Lab

OLAM-based mesh construction algorithm

CPAS' Customized Unstructured Mesh Generation
is based on this
Walko & Avissar (2011) paper

A Direct Method for Constructing Refined Regions in Unstructured Conforming Triangular–Hexagonal Computational Grids: Application to OLAM

ROBERT L. WALKO AND RONI AVISSAR

University of Miami, Miami, Florida

(Manuscript received 31 January 2011, in final form 12 May 2011)

Reimplemented in the cloud-computing platform

- Taking the GEOJSON specification as input

3926 MONTHLY WEATHER REVIEW VOLUME 139

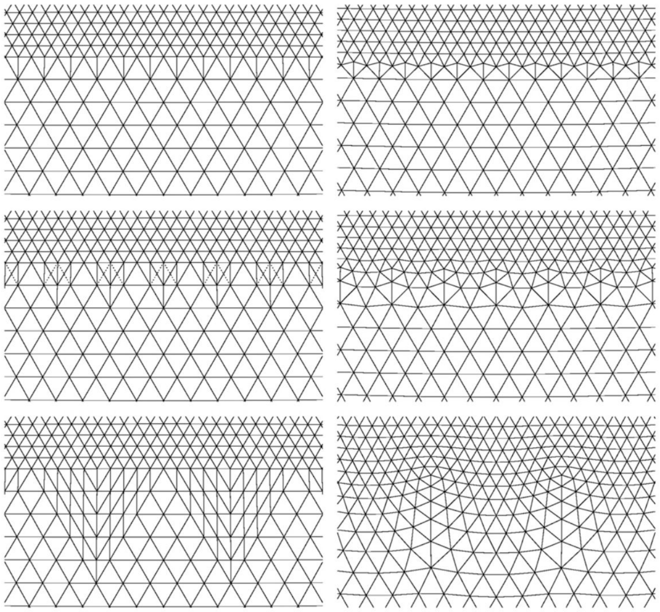
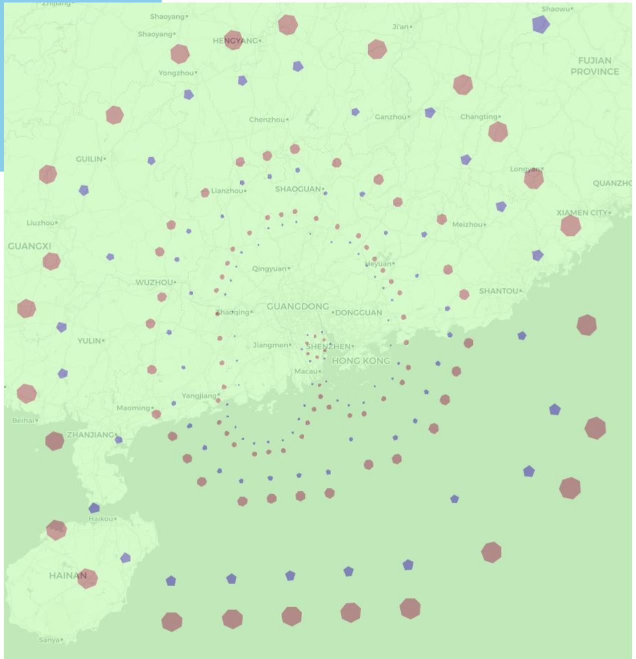
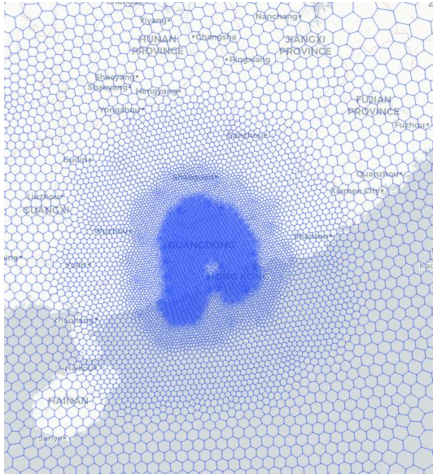


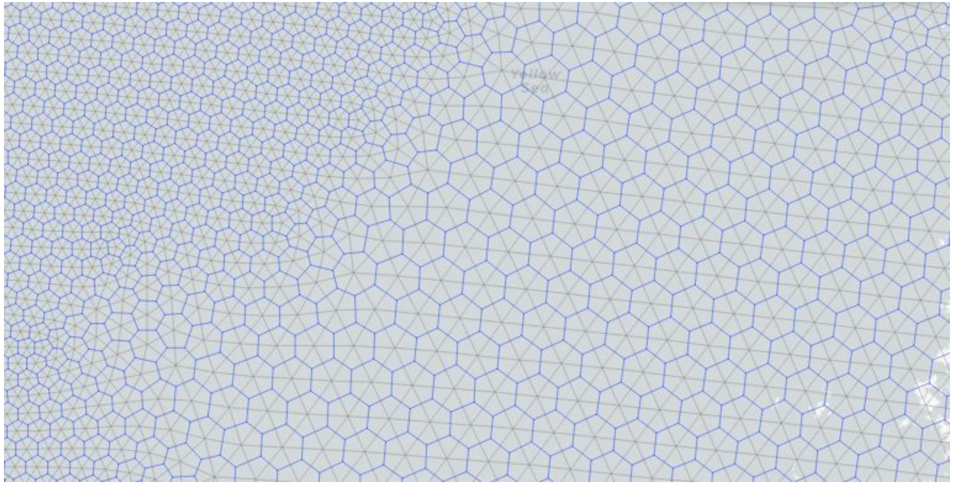
FIG. 2. Construction of transition rows outside refined mesh region for method A. (top) $M = 1$ (one transition row). (middle) $M = 2$. (bottom) $M = 5$. Each case (left) before and (right) after spring dynamics adjustment.

Example with concavity





Voronoi Tessellation and its
Delaunay triangulation duality



pentagon and
heptagon pairs

Resolution and computational cost consideration in NWP

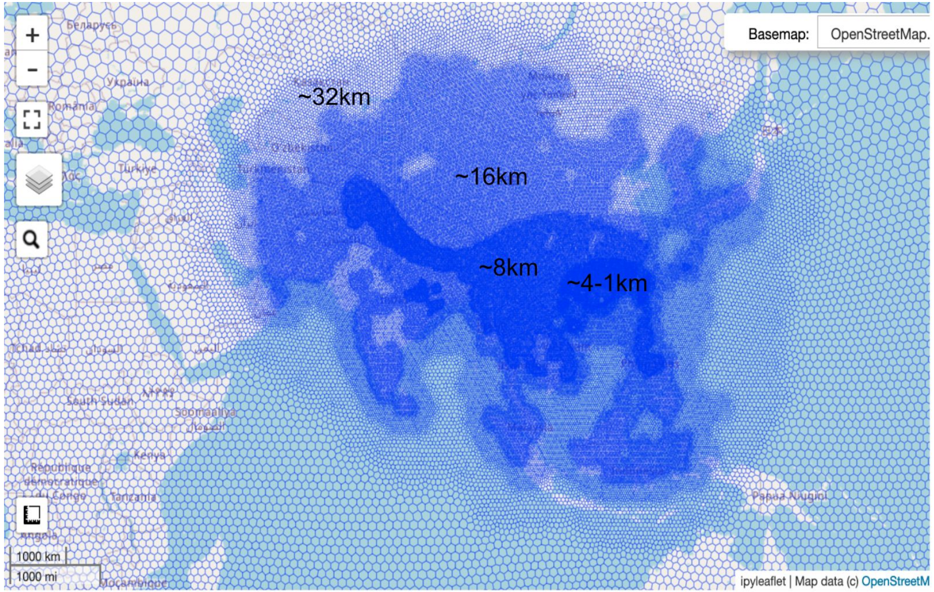
Flexible space and time discretization techniques together

Spend only the computational cost you want

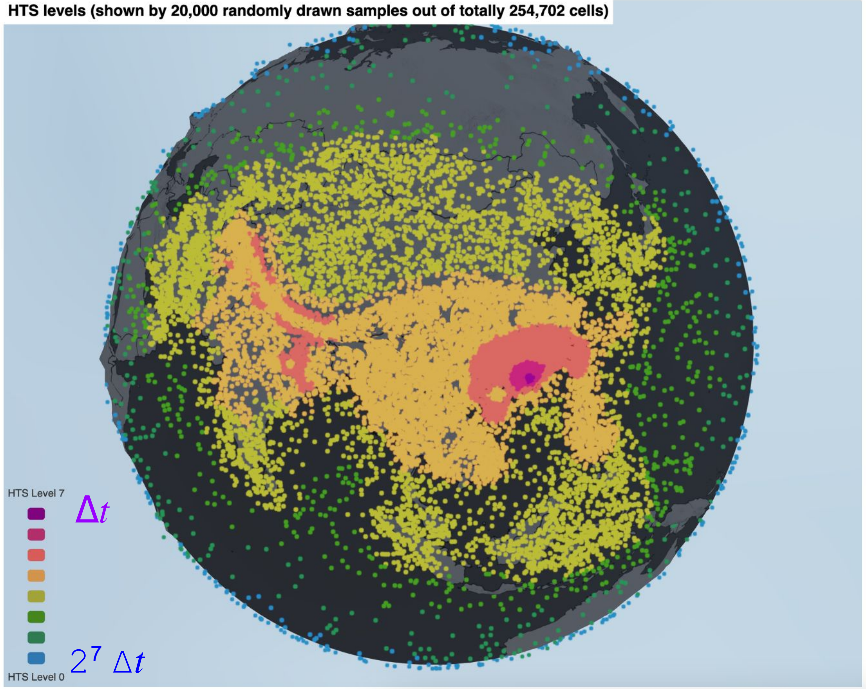
Flexible space discretization
Denser mesh spend more

Outside the designated regions:
saves a lot of cost

Flexible time discretization
Purple & red spend more



An example showing large resolution variation is feasible



Computational cost considerations

Flexible time discretization: HTS
Large time steps count fewer steps

Flexible space discretization: CUMG
Large cells count fewer cells

Made practical; computational cost saved relative to MPAS-A

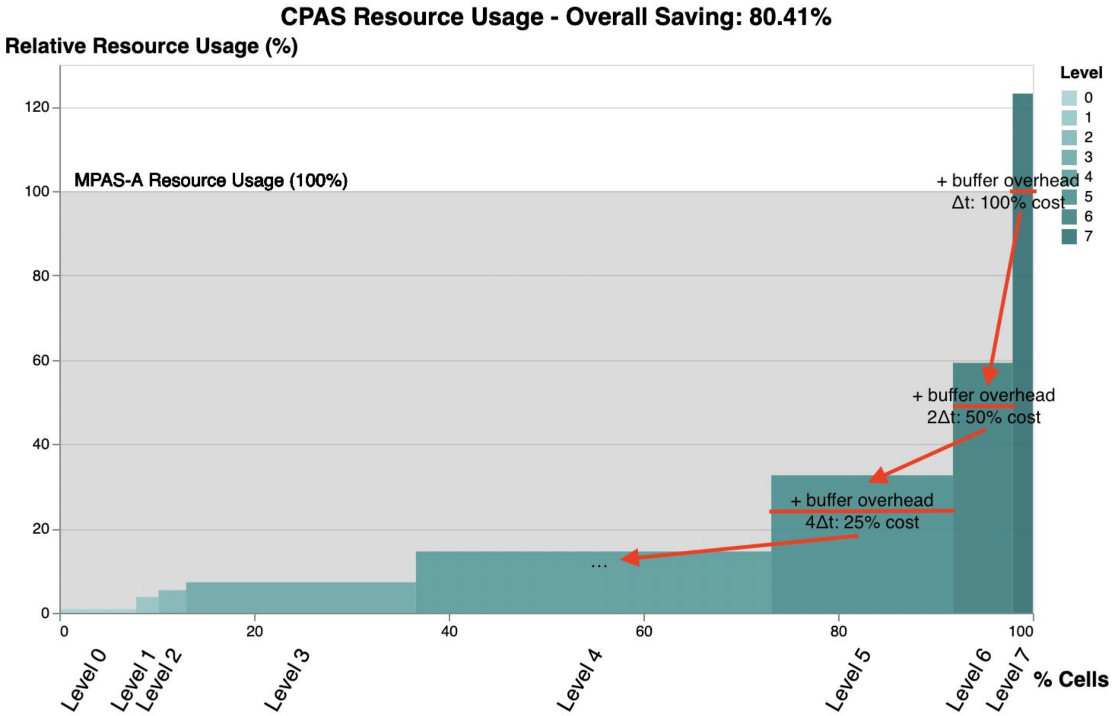
Level	Resolution distance between cells	Time step	#Cell (%)	CPU core resource in giga cell-step per simulation day (%)	Saving
0	127.555km - 152.183km	600.000s	20,115 (7.90%)	0.003 (0.36%)	99.12%
1	58.448km - 154.548km	300.000s	5,850 (2.30%)	0.004 (0.44%)	96.27%
2	25.312km - 118.735km	150.000s	7,222 (2.84%)	0.007 (0.78%)	94.64%
3	12.520km - 70.719km	75.000s	60,104 (23.60%)	0.080 (8.72%)	92.76%
4	6.284km - 32.520km	37.500s	92,999 (36.51%)	0.249 (27.12%)	85.45%
5	3.450km - 15.822km	18.750s	47,542 (18.67%)	0.286 (31.07%)	67.39%
6	1.753km - 6.505km	9.375s	15,607 (6.13%)	0.171 (18.54%)	40.73%
7	0.899km - 2.746km	4.688s	5,263 (2.07%)	0.119 (12.99%)	-23.14%
				Total 0.920	Overall 80.41%

Resolution variation for a large range (e.g. 128km-1km)

How is computational cost saved from coarsening time steps

Grey area not covered by green bars is the saving.

Doubling time step halves the cost, with the overhead in the buffer region.

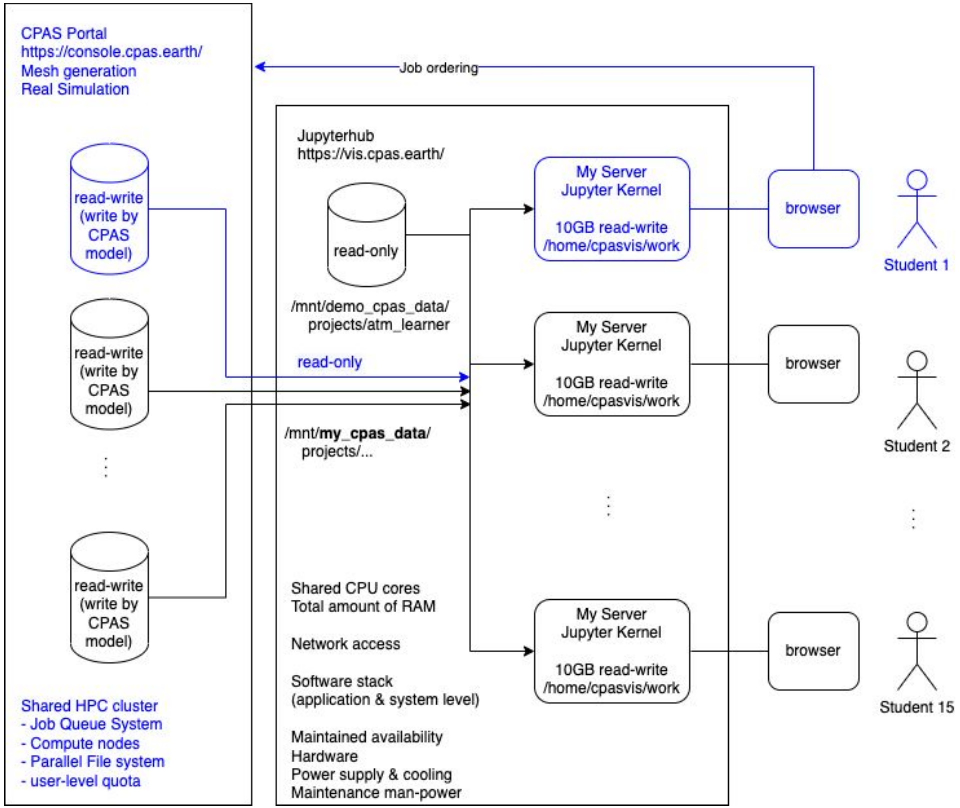


Level	Resolution distance between cells	Time step		Saving
0	127.555km - 152.183km	600.000s		99.12%
1	58.448km - 154.548km	300.000s		96.27%
2	25.312km - 118.735km	150.000s		94.64%
3	12.520km - 70.719km	75.000s		92.76%
4	6.284km - 32.520km	37.500s	...	85.45%
5	3.450km - 15.822km	18.750s	4Δt	67.39%
6	1.753km - 6.505km	9.375s	2Δt	40.73%
7	0.899km - 2.746km	4.688s	Δt	-23.14%
				Overall 80.41%

Your
First CPAS jobs

Have you done that already?

Where are the result data of my jobs?



Result data for your:

- Mesh generation jobs
- Real Simulation jobs

Short answer:

- Some complicated read-only directories
  - Specific to CPAS system architecture
    - Not user’s concern

CPAS provides a GUI tool for you to get the long path.

Lab 1
Finding result files

5 minutes

CPAS-specific tools for finding files

```
from cpas.ui import UI

ui = UI().select_job()
```

Generated mesh:

```
ui.grid_ncfile

ds = xr.open_dataset(ui.grid_ncfile)
```

Real Simulation:

```
ui.output_ncfile_dict
```

Lab 1
Finding result files

Time's up

Geometry of unstructured grid

Spherical Centroidal Voronoi Tessellations
Dual triangular Delaunay tessellation

SCVT

MPAS Tutorial

https://www2.mmm.ucar.edu/projects/mpas/tutorial/Boulder2019/index.html

Mesh structure

Analogy of Voronoi Tessellations:

- A castle with a King at a generating point
- Farmer pay tax to the nearest King
- The Voronoi cell is the King's land.

MPAS
Model for Prediction Across Scales

A defining feature of MPAS models is their use of **centroidal Voronoi tessellations** (CVTs) with a C-grid staggering

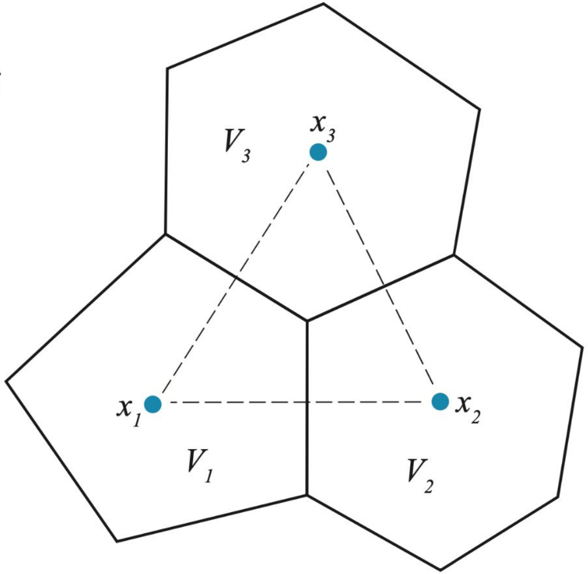
– When constrained to lie on the surface of a sphere, we often call them spherical centroidal Voronoi tessellations (SCVTs)

Voronoi = each grid volume (cell) V_i is uniquely associated with a *generating point* $\boldsymbol{x}_i$ such that all points within V_i are closer to $\boldsymbol{x}_i$ than to any other $\boldsymbol{x}_j$

- Lines joining generating points of adjacent cells are

1. bisected by the shared cell face; and
2. intersect the shared cell face at a right angle.

Centroidal = the generating point for each Voronoi cell is also the mass centroid of that cell (**w.r.t. some density function**)



MPAS-Atmosphere Tutorial
9 – 11 September 2019, Boulder
3

Cell, Vertex, Edge

```
import xarray as xr

ds = xr.open_dataset("atm_learner/data_for_tutorials/demo_kompasu/grid.nc")
ds
```

executed in 716ms, finished 08:59:41 2022-05-13

xarray.Dataset

▸ Dimensions: (nCells: 105986, nEdges: 317952, nVertices: 211968, TWO: 2, maxEdges: 7, maxEdges2: 14, vertexDegree: 3)

▸ Coordinates: (0)

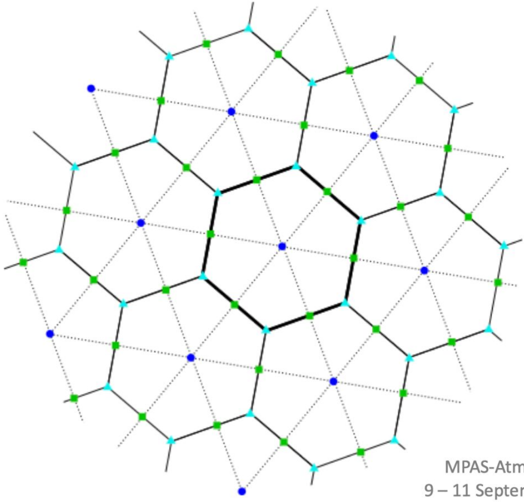
▾ Data variables:

```
latCell        (nCells)      float64 ...
[105986 values with dtype=float64]
lonCell        (nCells)      float64 ...
[105986 values with dtype=float64]
meshDensity    (nCells)      float64 ...
xCell          (nCells)      float64 ...
yCell          (nCells)      float64 ...
zCell          (nCells)      float64 ...
indexToCellID  (nCells)        int32 ...
latEdge        (nEdges)      float64 ...
[317952 values with dtype=float64]
lonEdge        (nEdges)      float64 ...
[317952 values with dtype=float64]
xEdge          (nEdges)      float64 ...
yEdge          (nEdges)      float64 ...
zEdge          (nEdges)      float64 ...
indexToEdgeID  (nEdges)        int32 ...
latVertex      (nVertices)   float64 ...
[211968 values with dtype=float64]
lonVertex      (nVertices)   float64 ...
[211968 values with dtype=float64]
```

MPAS
Model for Prediction Across Scales

How do MPAS, *and you,* keep track of this unstructured Voronoi mesh?

Schemes for implicitly finding the indices/identities (the "IDs") of neighboring mesh elements (i.e., cells, edges, vertices) are bound to fail...
... so we must find them explicitly through connectivity fields that are the foundation of the MPAS mesh representation.



Three types of mesh elements are tracked in the mesh representation:

- **Cell** locations (blue circles) - the generating points of the Voronoi mesh
- **Vertex** locations (cyan triangles) - the corners of primal mesh cells
- **Edge** locations (green squares) - the points where the dual mesh edges intersect the primal mesh edges

MPAS-Atmosphere Tutorial
9 – 11 September 2019, Boulder

8

Neighborhood

Connectivity of cell centers:

“dual mesh”

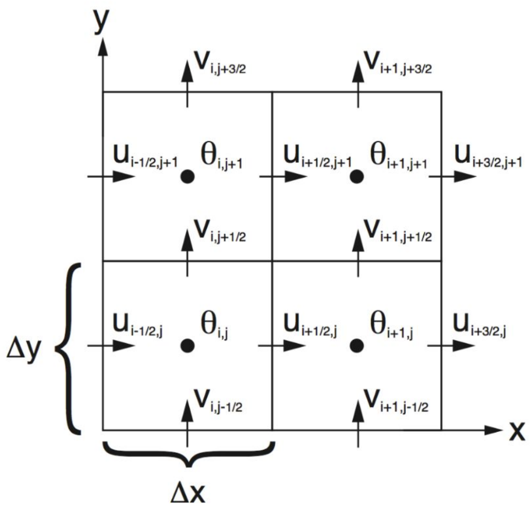
Delaunay triangulation

MPAS
Model for Prediction Across Scales

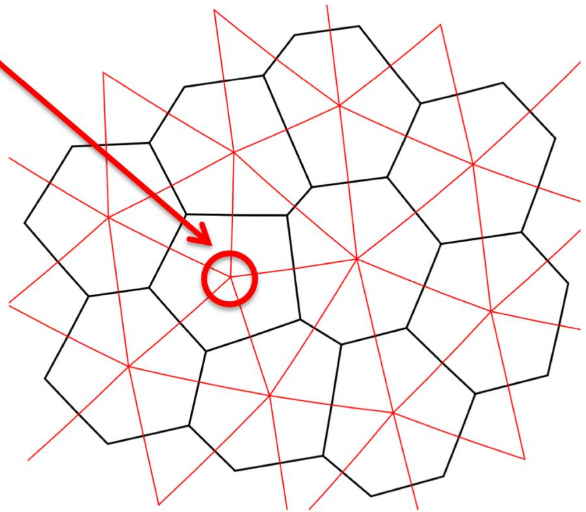
How do MPAS, *and you,* keep track of this unstructured Voronoi mesh?

One can start to imagine way to identify neighboring cells implicitly based on the index or location of each cell

- In a rectangular mesh, our neighbors are at (i+1, j), (i-1, j), (i, j+1), (i, j-1)
- Who is the “next” cell after this one in any direction?



Above: A region from the ARW C-staggered grid, stored in a 2-d array.



Above: A region from an MPAS mesh showing Voronoi regions (black) and Delaunay triangles (red).

MPAS-Atmosphere Tutorial 7
9 – 11 September 2019, Boulder

Indices

All entities of the unstructured grid are put to arrays in a 1-dimensional way.

Every element is referred by

- index

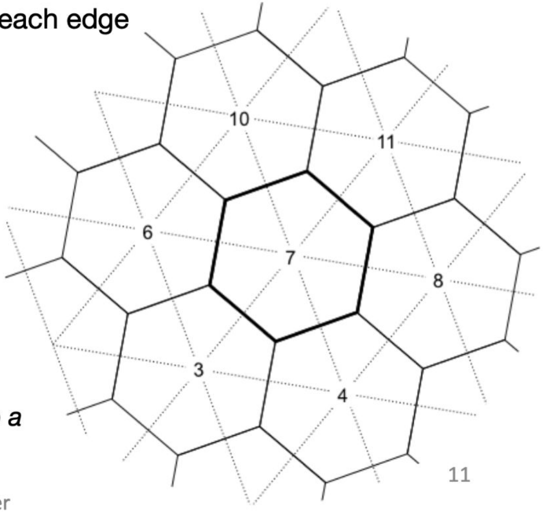
Connectivities also represented by indices

MPAS
Model for Prediction Across Scales

Explicit connectivity fields describe the structure of a mesh

- **nEdgesOnCell(nCells)** – the number of neighbors for each cell
- **cellsOnCell(maxEdges, nCells)** – the indices of neighboring cells for each cell
- **edgesOnCell(maxEdges, nCells)** – the indices of bounding edges for each cell
- **verticesOnCell(maxEdges, nCells)** – the indices of corner vertices for each cell
- **edgesOnVertex(vertexDegree,nVertices)** – the indices of edges incident with each vertex
- **verticesOnEdge(2,nEdges)** – the indices of endpoint vertices for each edge
- **cellsOnVertex(vertexDegree,nVertices)** – the indices of cells meeting at each vertex
- **cellsOnEdge(2,nEdges)** – the indices of cells separated by each edge

```
nEdgesOnCell(7)=6   cellsOnCell(1,7)=8
                    cellsOnCell(2,7)=11
                    cellsOnCell(3,7)=10
                    cellsOnCell(4,7)=6
                    cellsOnCell(5,7)=3
                    cellsOnCell(6,7)=4
```



At model start-up, all indices in these arrays are re-numbered to a local indexing scheme.

MPAS-Atmosphere Tutorial
9 – 11 September 2019, Boulder

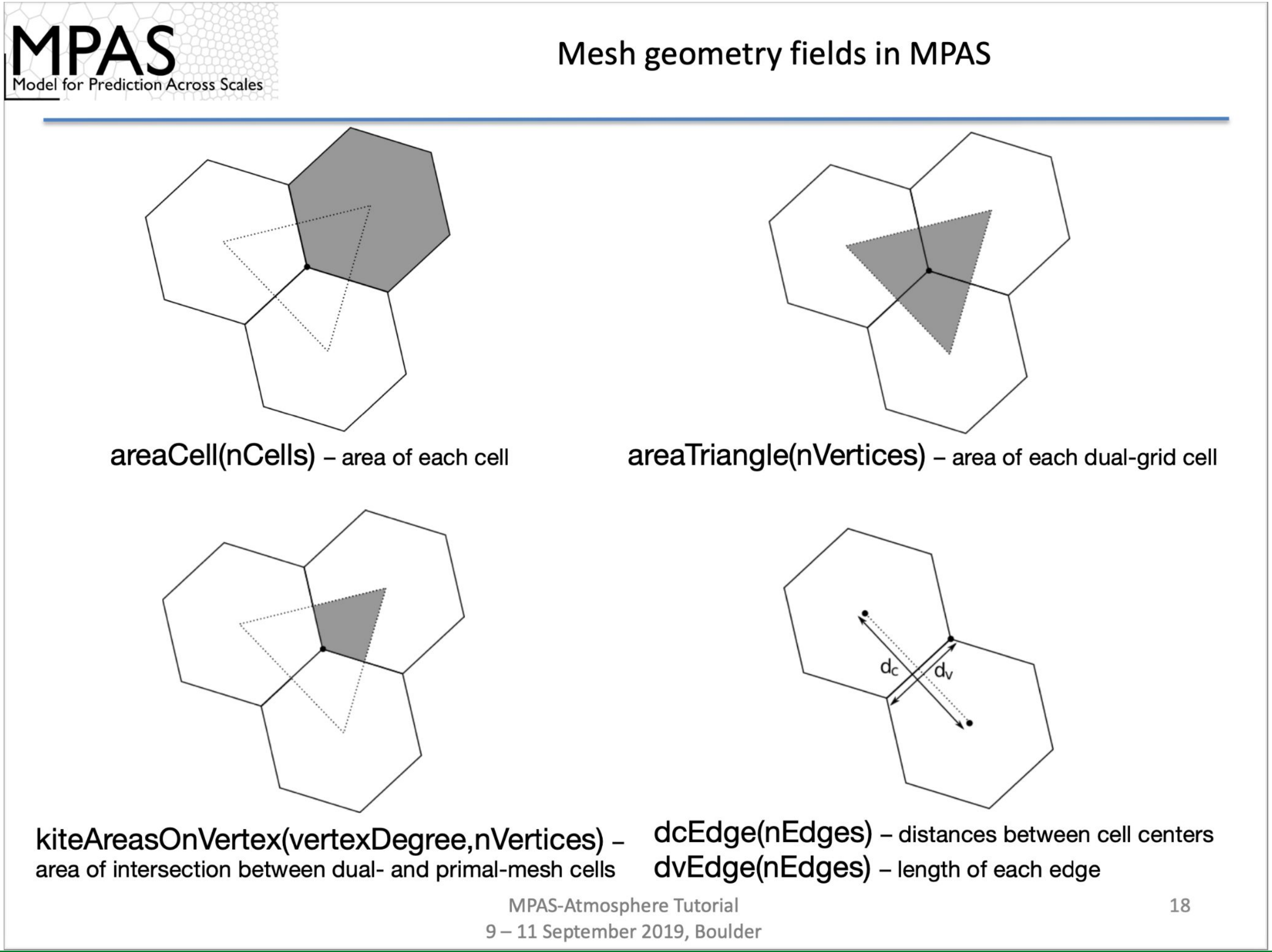
11

Fields

Fields used in dynamical core computation

Most often concerned:

Grid spacing: **dcEdge**



Dimensions

MPAS
Model for Prediction Across Scales

What do MPAS meshes look like in netCDF files?

When stored in netCDF files ("*grid.nc*"), MPAS meshes have at least the following dimensions:

```
dimensions:
    nCells = 40962 ;
    nEdges = 122880 ;
    nVertices = 81920 ;
    maxEdges = 10 ;
    maxEdges2 = 20 ;
    TWO = 2 ;
    vertexDegree = 3 ;
```

The number of cells, edges, and vertices in the mesh.

For global, spherical meshes:
nVertices = 2 * (nCells - 2)
nEdges = 3 * (nCells - 2)

For *doubly-periodic* planar meshes:
nEdges = nCells + nVertices

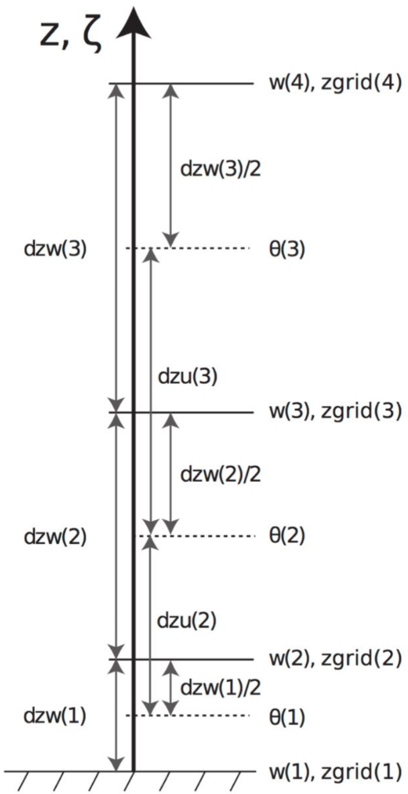
For *limited-area* meshes:
nEdges + 1 = nCells + nVertices

Scalars at center;

Vertical wind on faces

MPAS
Model for Prediction Across Scales

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:

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

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

MPAS-Atmosphere Tutorial
9 – 11 September 2019, Boulder

Vertical layers

MPAS tutorial

Dynamics: Overview and Configuration

https://www2.mmm.ucar.edu/projects/mpas/tutorial/Boulder2019/slides/07.MPAS_solver.pdf

CPAS uses terrain data sources with multiple resolutions (2 arc minute - 15 arc second)

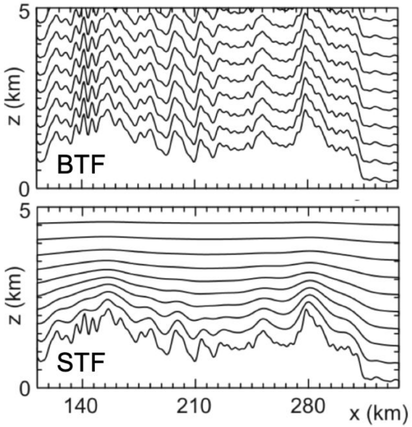
MPAS
Model for Prediction Across Scales

MPAS Vertical Mesh

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 https://github.com/MPAS-Dev/MPAS-Model/blob/master/src/core_init_atmosphere/Registry.xml

smooth 1/4

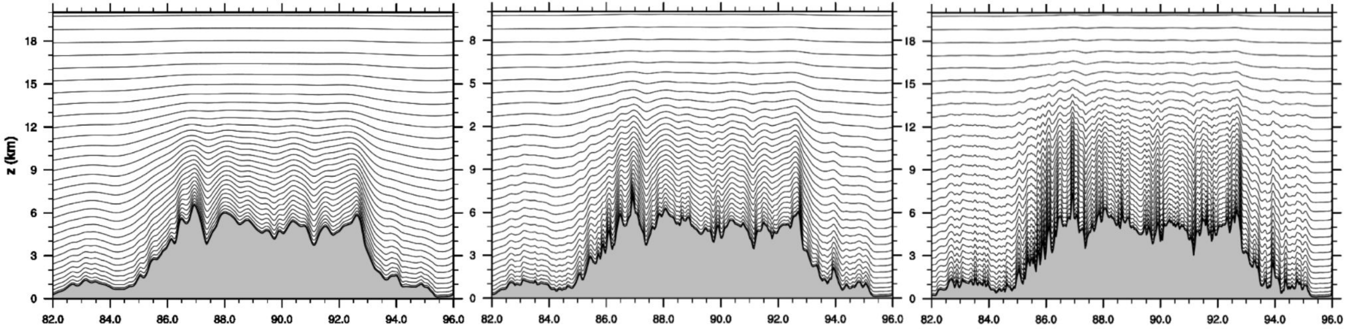
```
<nml_record name="vertical_grid" in_defaults="true">

        <nml_option name="config_ztop"                    type="real"          default_value="30000.0"
             units="m"
             description="Model top height"
             possible_values="Positive real values"/>

        <nml_option name="config_nsmterrain"              type="integer"       default_value="1"
             units="-"
             description="Number of smoothing passes to apply to the interpolated terrain field"
             possible_values="Non-negative integer values"/>

        <nml_option name="config_smooth_surfaces"         type="logical"       default_value="true"
             units="-"
             description="Whether to smooth zeta surfaces"
             possible_values="true or false"/>

        <nml_option name="config_dzmin"                   type="real"          default_value="0.3"
             units="-"
             description="Minimum thickness of layers as a fraction of nominal thickness"
             possible_values="Real values in the interval (0,1)"/>

        <nml_option name="config_nsm"                     type="integer"       default_value="30"
             units="-"
             description="Maximum number of smoothing passes for coordinate surfaces"
             possible_values="Positive integer values"/>
```

MPAS
Model for Prediction Across Scales

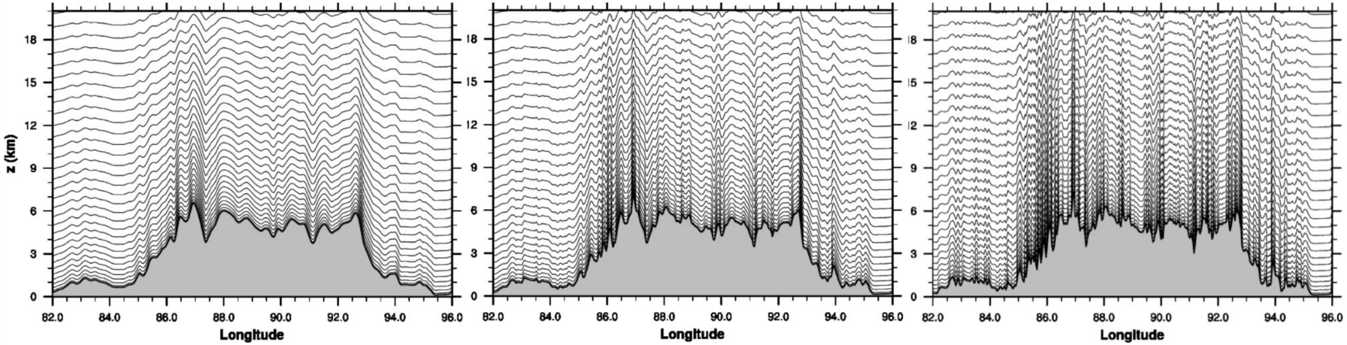
MPAS -Tibetan Plateau, 28º N

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)

Lab 2
Inspecting grid.nc

30 minutes

Where are a cell's neighbour cells?

```
import xarray as xr
ds = xr.open_dataset("atm_learner/data_for_tutorials/demo_kompasu/grid.nc")
ds
```

```
ds.nEdgesOnCell.isel(nCells=999)
```

```
cell1000_neighbours = ds.cellsOnCell.isel(nCells=999)
cell1000_neighbours
```

```
ds.latCell.isel(nCells=cell1000_neighbours-1)
```

What are these numbers?

```
import numpy as np
np.rad2deg(ds.latCell.isel(nCells=cell1000_neighbours-1))
```

```
np.rad2deg(ds.lonCell.isel(nCells=cell1000_neighbours-1))
```

```
cell1_neighbours = ds.cellsOnCell.isel(nCells=0)
cell1_neighbours
```

Inspect cell1's neighbours' longitude. Surprise? Why?

```
cell1000_neighbours = ds.cellsOnCell.isel(nCells=999)
cell1000_neighbours
```

executed in 15ms, finished 09:22:43 2022-05-13

xarray.DataArray 'cellsOnCell' (maxEdges: 7)

```
array([17764, 17766, 17595, 17680, 17759, 17762, 17762], dtype=int32)
```

▸ Coordinates: (0)

▸ Attributes: (0)

nEdgesOnCell ignore

```
import numpy as np
np.rad2deg(ds.latCell.isel(nCells=cell1000_neighbours-1))
```

executed in 22ms, finished 09:23:10 2022-05-13

xarray.DataArray 'latCell' (maxEdges: 7)

```
array([-11.15173089, -11.11398784, -10.00561016,  -8.91706896,
        -8.94593913, -10.0715039 , -10.0715039 ])
```

▸ Coordinates: (0)

▸ Attributes: (0)

```
np.rad2deg(ds.lonCell.isel(nCells=cell1000_neighbours-1))
```

executed in 34ms, finished 09:23:37 2022-05-13

xarray.DataArray 'lonCell' (maxEdges: 7)

```
array([-110.12075953, -108.83134853, -108.21958661, -108.89914003,
       -110.19893936, -110.80919304, -110.80919304])
```

▸ Coordinates: (0)

▸ Attributes: (0)

Explore your own generated mesh

```
ds = xr.open_dataset(ui.grid_ncfile)
ds
```

Where are a cell’s edges?

Where are a cell’s vertices?

Demo: Plotting the mesh

atm_learner/lecture2/Lecture2_unstructured_mesh.ipynb

Lab 2
Inspecting grid.nc

30 minutes - Time’s up

Icosahedron

CPAS' mesh generation:

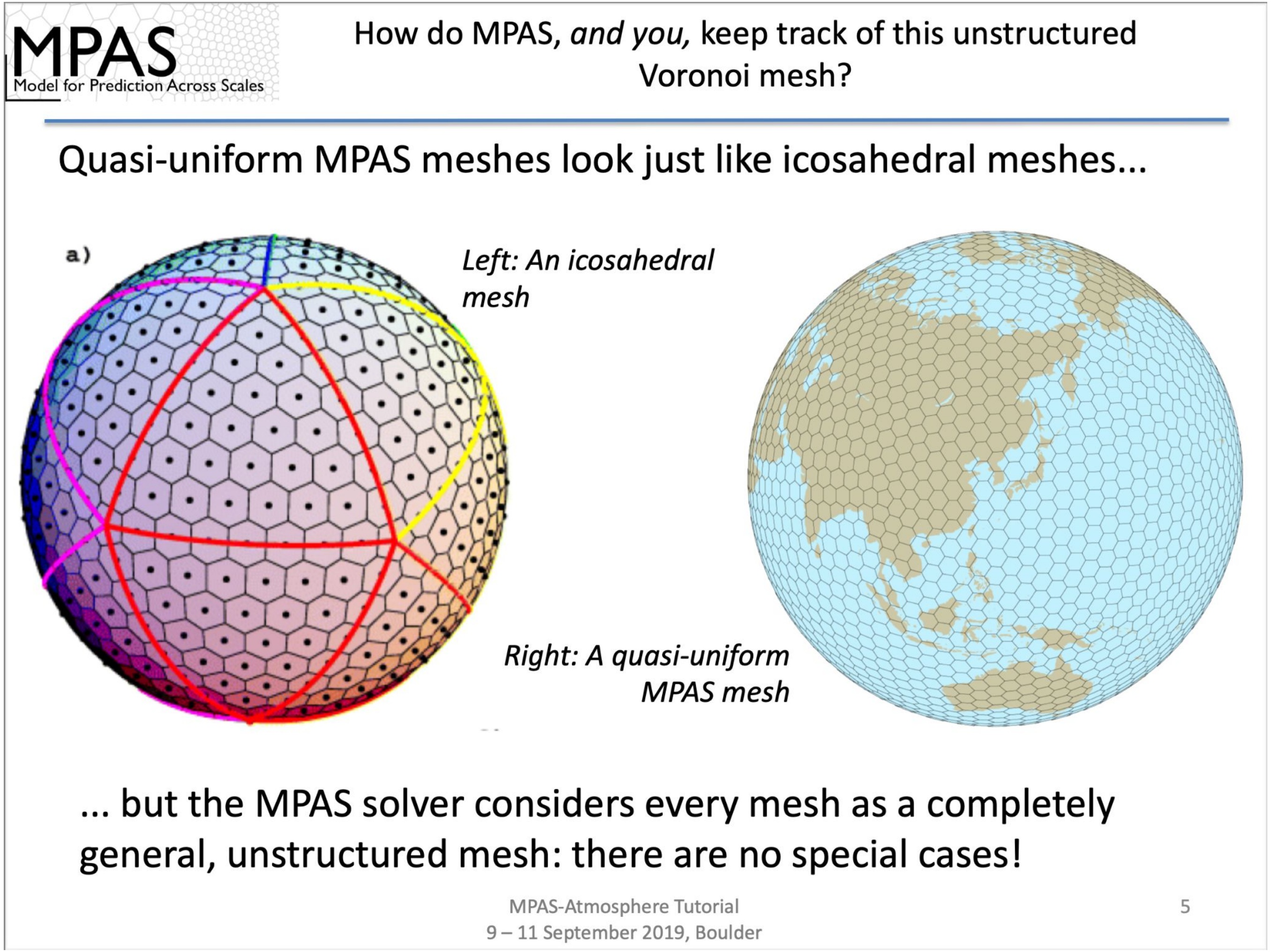
OLAM-based mesh construction algorithm

Starts with 12 generating points

- North pole
- South pole
- 5 on a North hemisphere latitude
- 5 on a South hemisphere latitude

± arctan 1/2 = ±26.57°

Still there after bisections and multisections

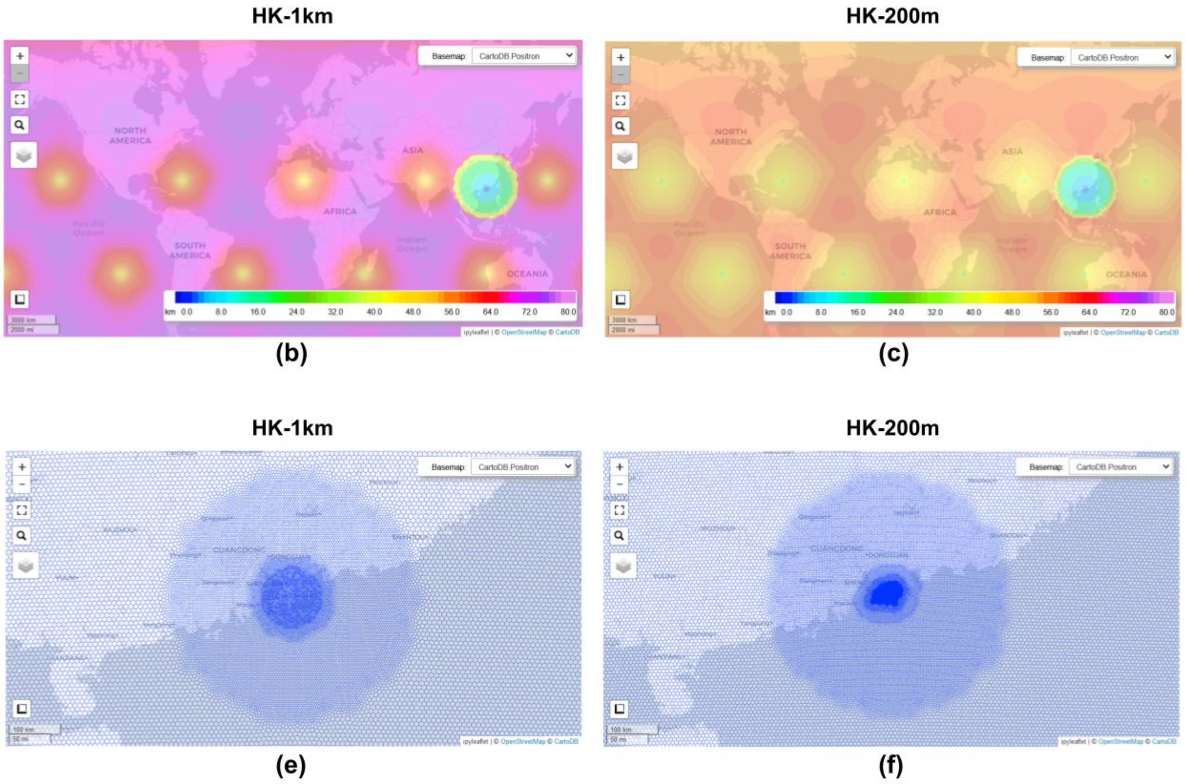


CPAS (current version) mesh drawback

The 12 original generating points have finer resolution than surrounding.

Easily seen in plotting.

To be improved in future versions.



Assignment - now you've got basic knowledges to *explore*

Synoptic circulation

1) Generate a simple unstructured mesh for simulating synoptic-scale atmospheric circulation in the North hemisphere
2) Order a simulation job
3) Ensure the resulting data is ready before the next session.

Project:

- Run some trial job for the severe weather event you intent to choose
- Show result and discuss with group-mates (25 May Wed break-out room)
- Make decision on the severe weather event your group will use.

For your project

- Do some trial of mesh generation & real simulation.
- If you don't know what options to use, just use the default.
    - Defaults are usually generic in engineering practices
- Expected iterative trials - "My Jobs" page
    - "Cancel job" button - kill job
    - "Delete job" button - release storage
    - No worry