MAX PLANCK GESELLSCHAFT

Towards a native format for storing DGGS data

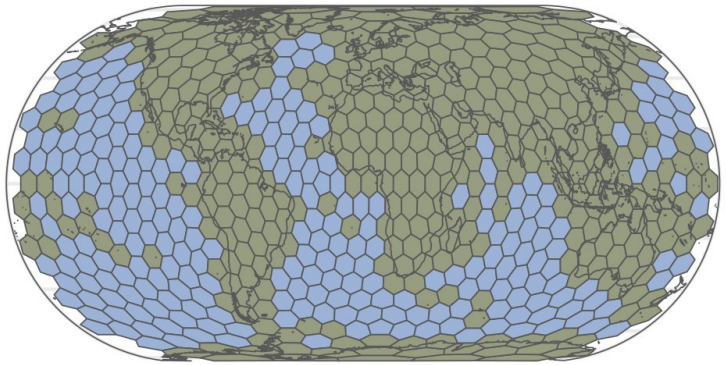
Daniel Loos

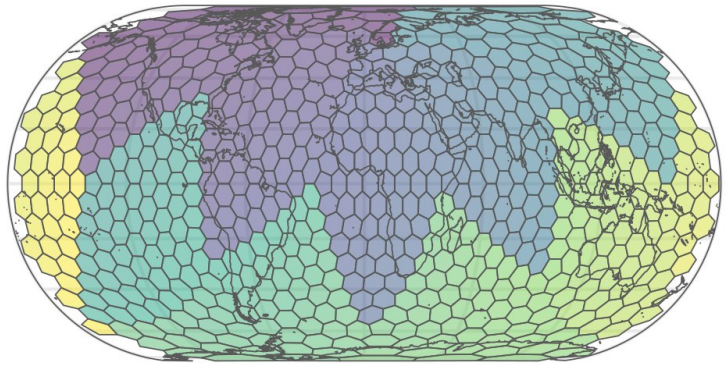
Max Planck Institute for Biogeochemistry

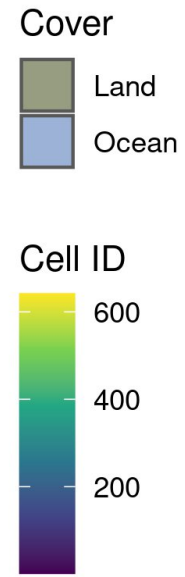
Motivation

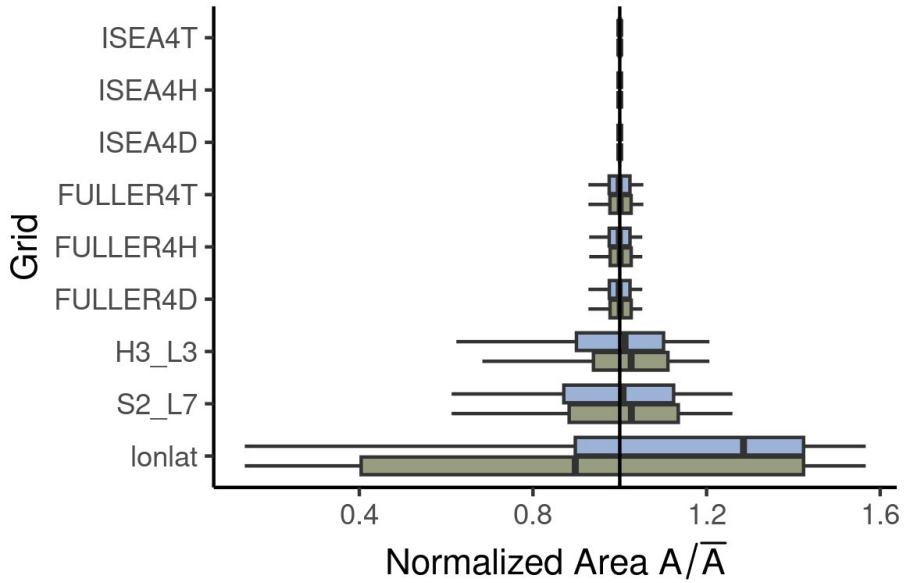
- want to minimize distortions of area and shape in both analysis and storage
- We don’t want to reproject between geographical and dggs coordinates every single time
- Up to know: DGGS grids with little visualizations and no file format (planned in future parts of ISO 19170)
- Goal: Create a DGGS native file format and data structure for efficient and accurate storage of spatial data

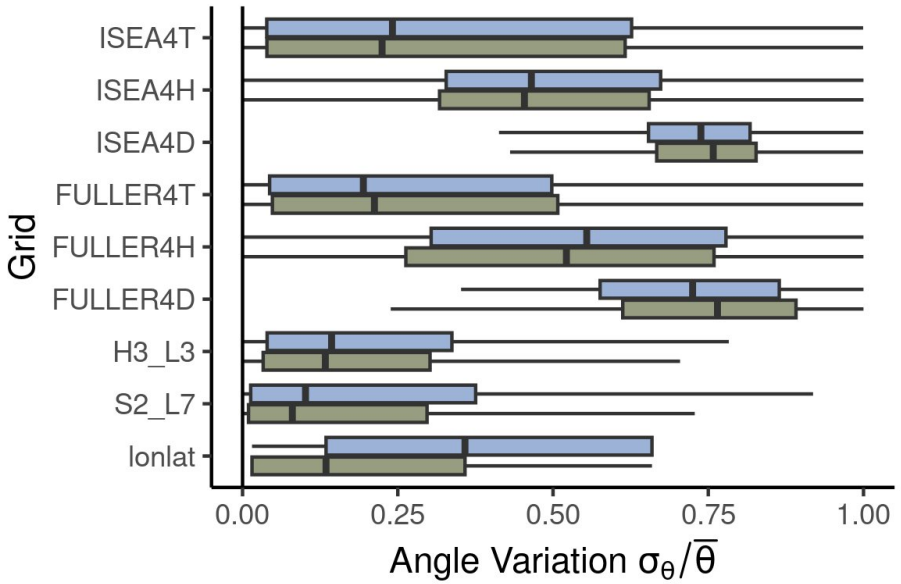
DGGS: Distortions



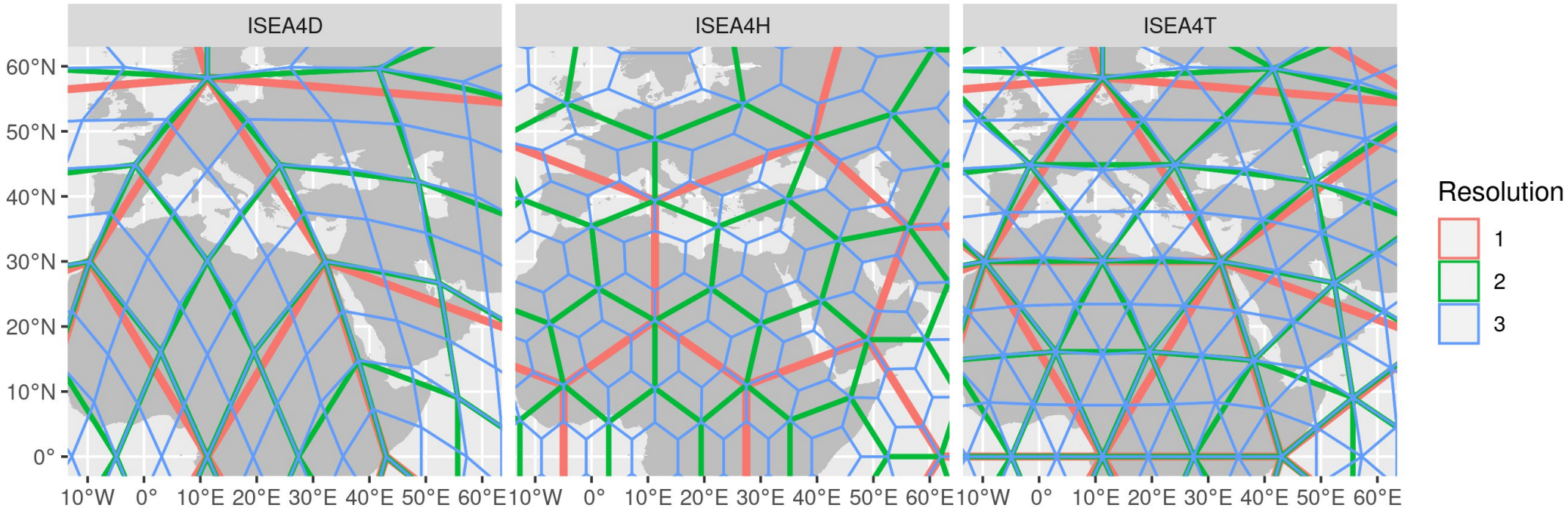






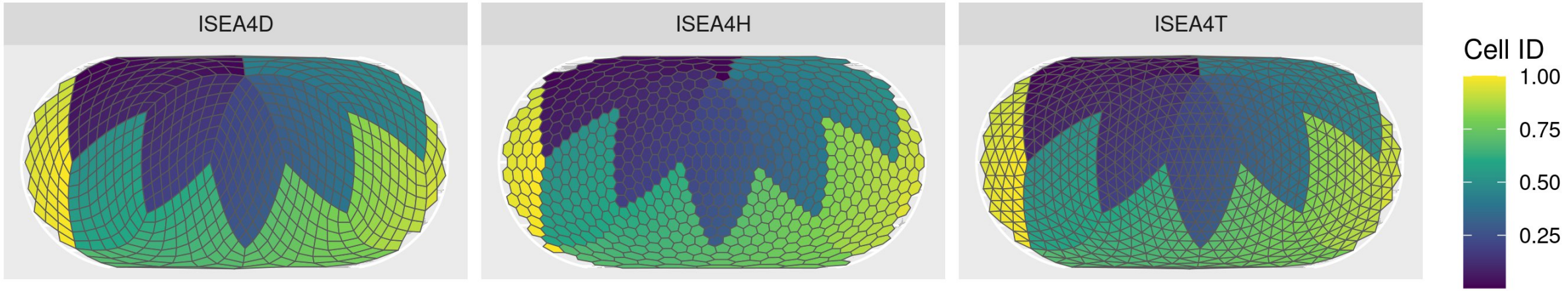


DGGS: Shapes


ISEA4D
ISEA4H
ISEA4T
60°N
50°N
40°N
30°N
20°N
10°N
0°
10°W 0° 10°E 20°E 30°E 40°E 50°E 60°E
Resolution
1
2
3

DGGS: Zone index

- DGGRID cells were already aligned nicely
- Places within each icosahedron face have similar cell ids

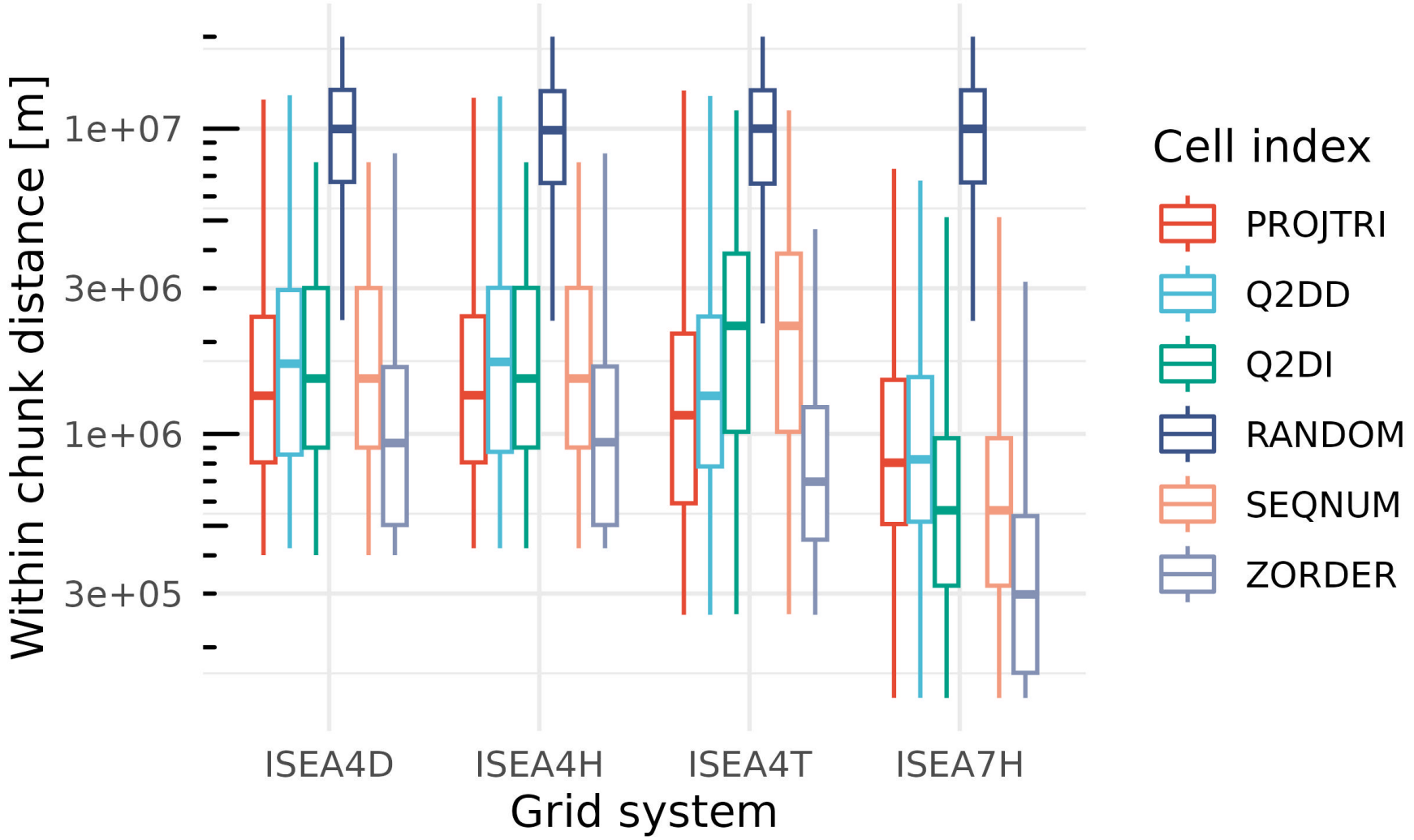


DGGS zone index

- OGC specifies DGGS in ISO 19170-1:2021
- zone identifier: spatio-temporal reference (Clause 4.42) in the form of a label or code that identifies a zone
- Zone identifier are just an label for a chunk of space-time according to OGC. Not specified in terms of dimensionality or sorting

1D indices

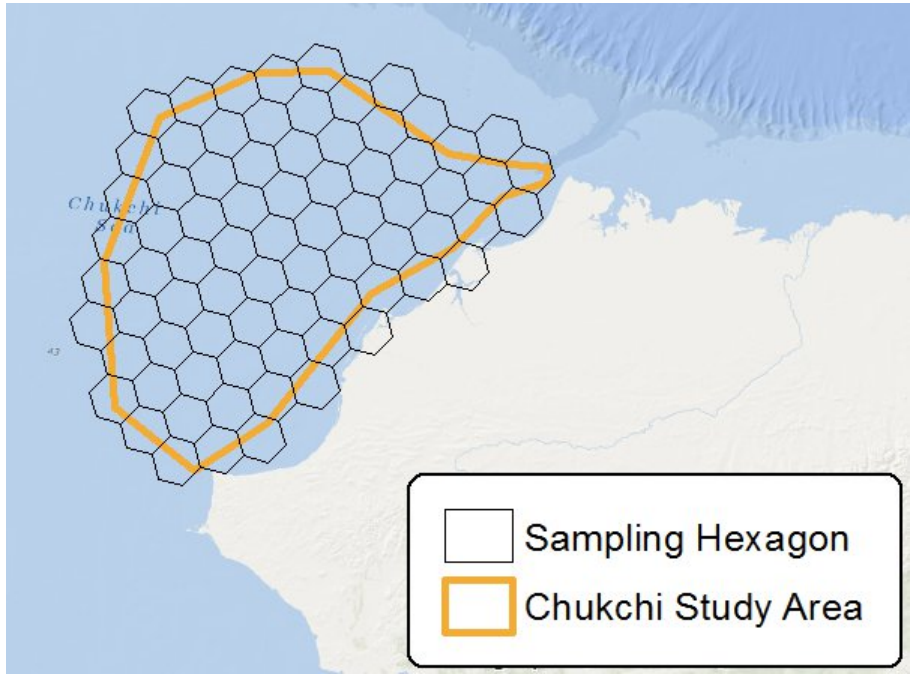
Zones nearby in geographical space should be also nearby in memory space at the disk



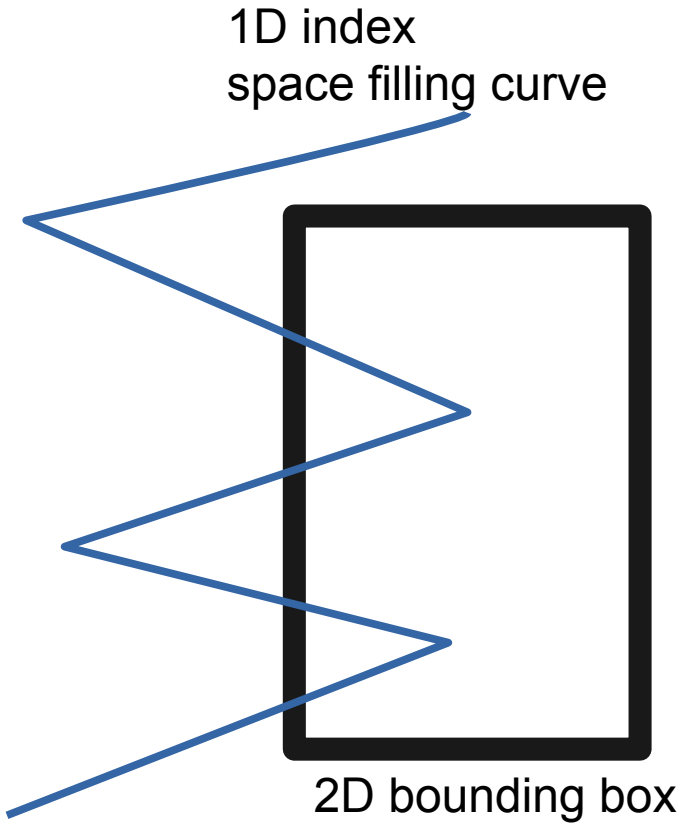
2D indices

- Bounding boxes are 2D
- Neural network kernels are 2D
- The surface of the earth is 2D
- Image sensors are 2D

- Why not just use 2D index for DGGS zones?

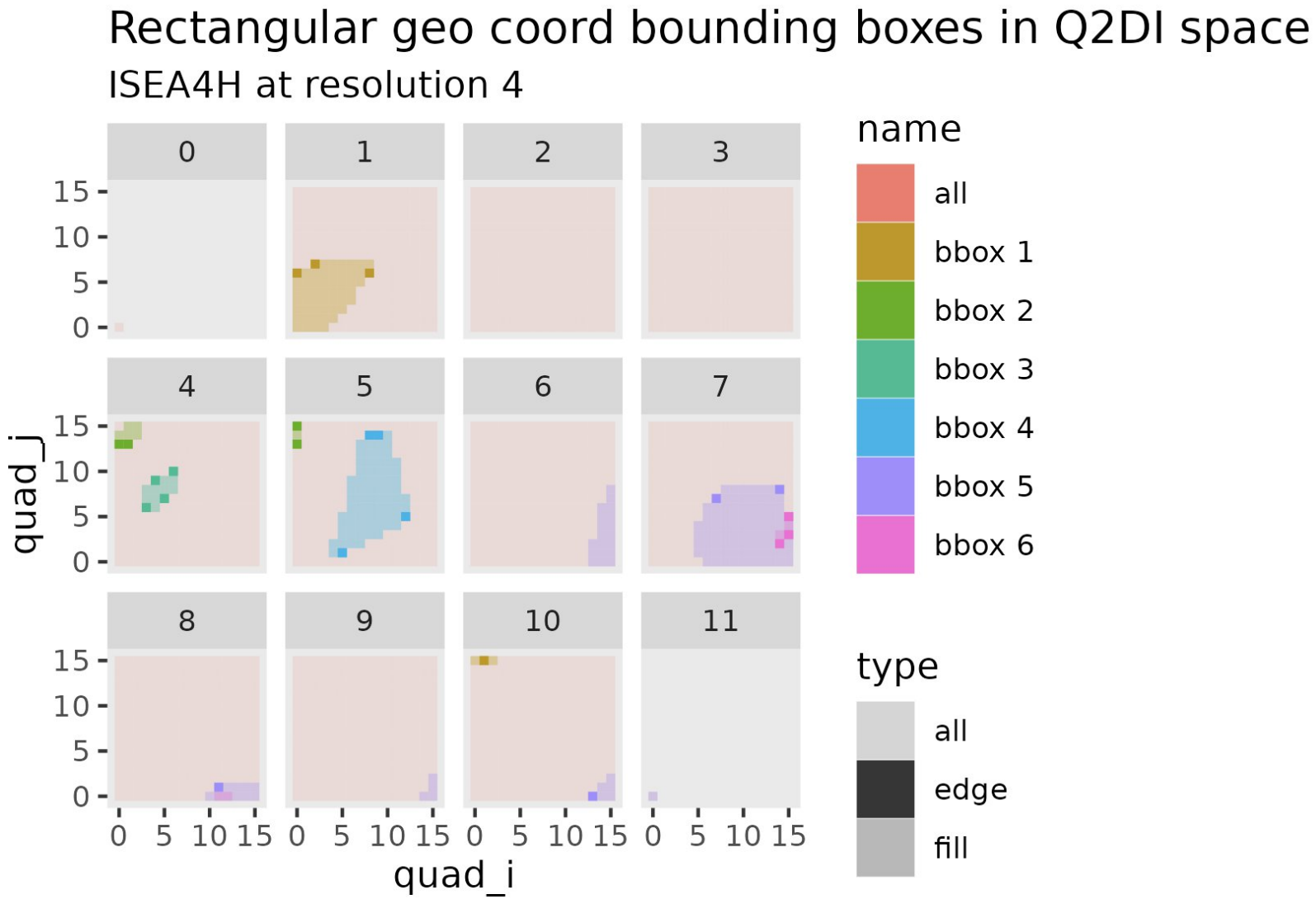


A 2D Polygon is a 1D list of DGGS zones



Multiple 1D intervals needed in just one 2D bounding box

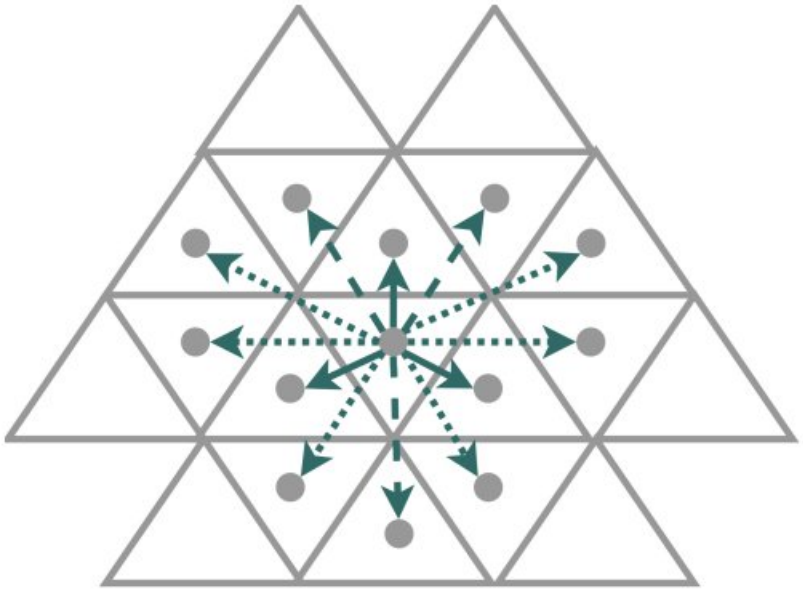
Bounding boxes are convex quads in 2D index

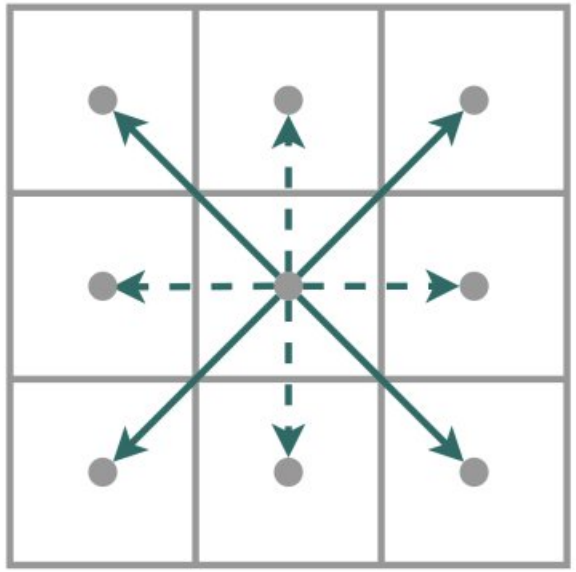


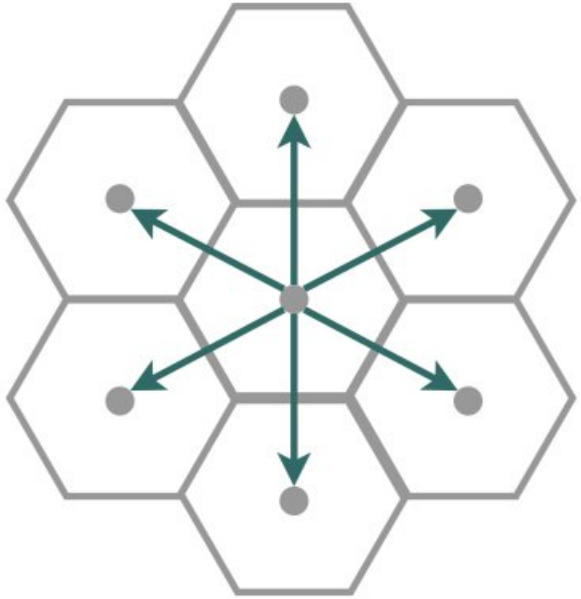
Axes i and j are not cartesian, but 120° resulting in distorted polygons

Hexagonal indices

Better representation of:
- Neighbors: Equal distances
- Angles: More edges
- Rotation: Higher degree of symmetry in group convolutions

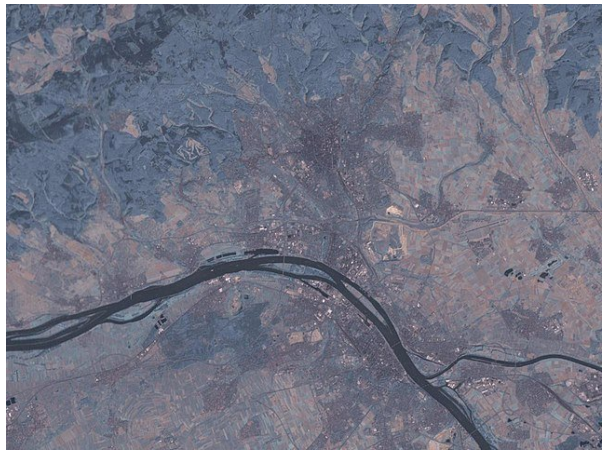


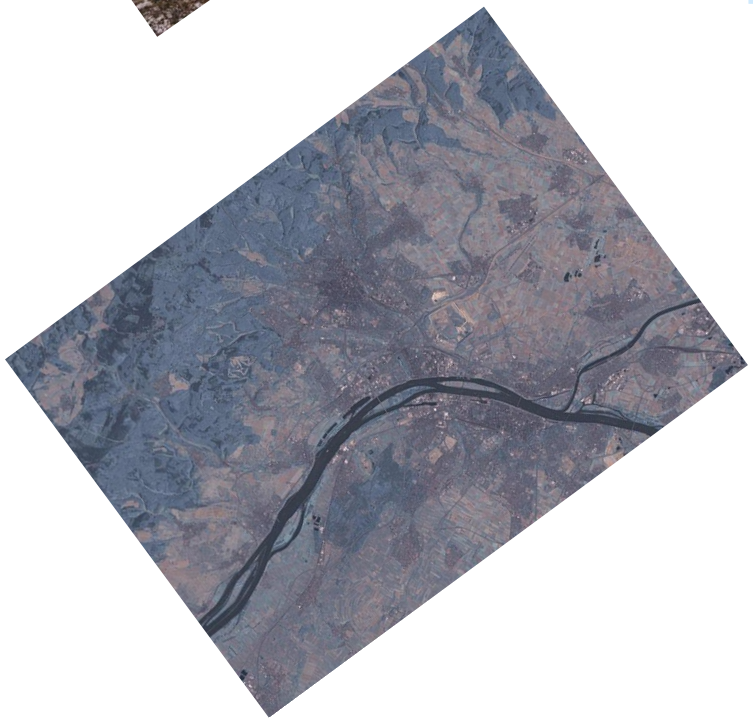


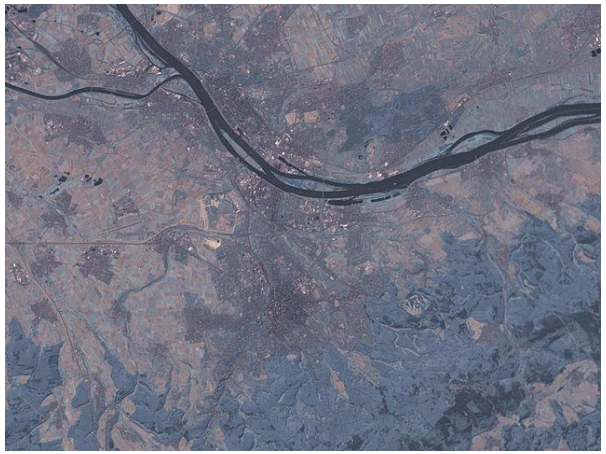


Hexagonal indices are more suited for rotational equivariant convolutions

Not rotational equivariant

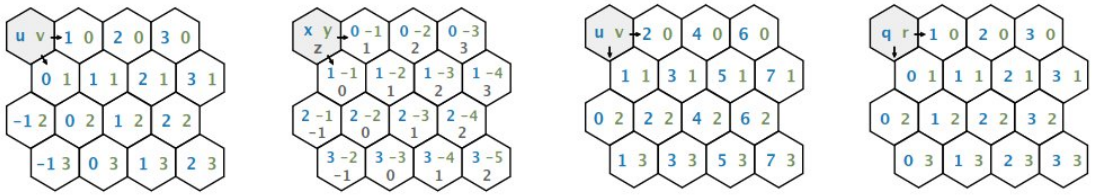






rotational equivariant

2D indices for planes tesselated with hexagons



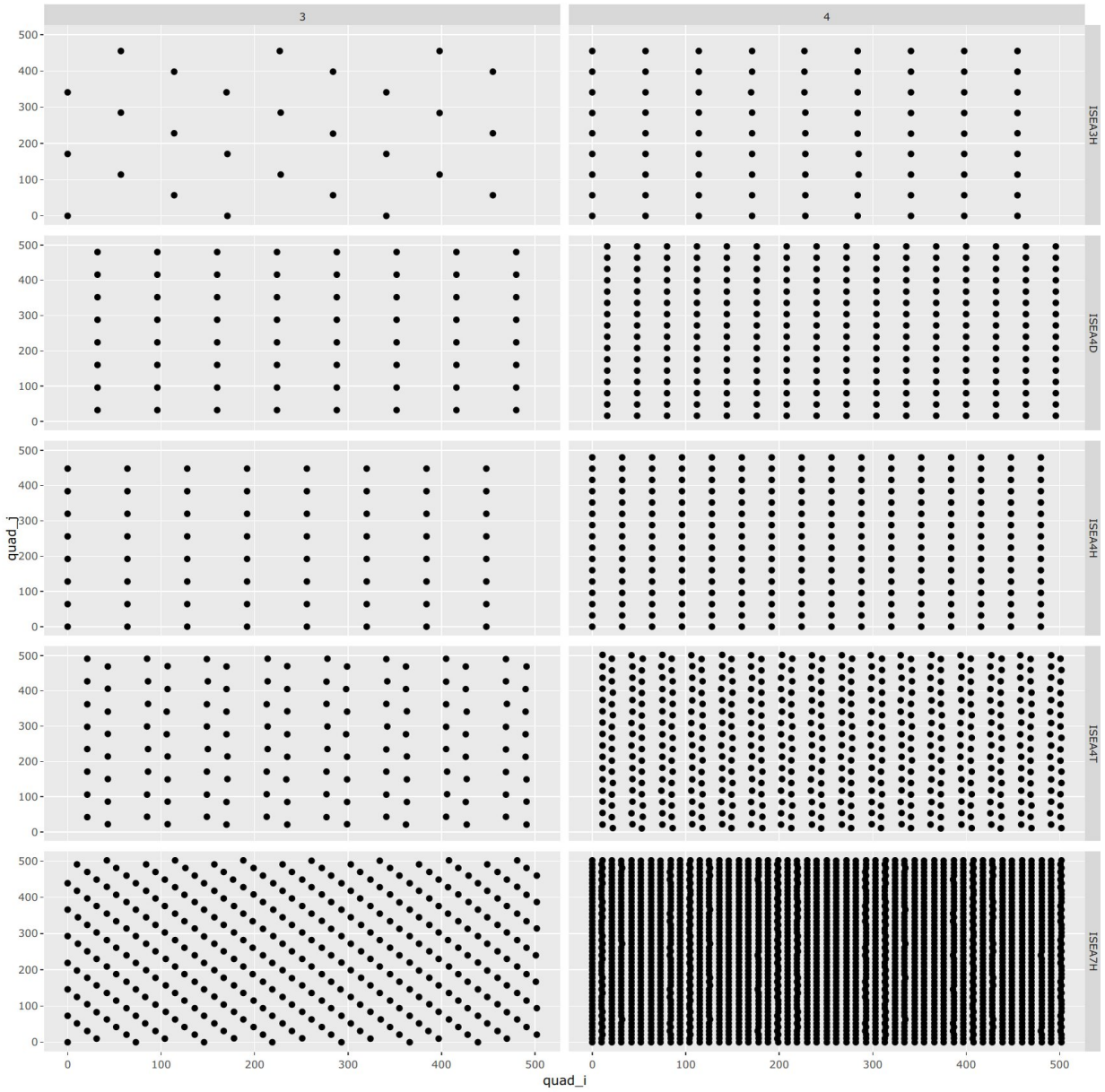
(a) Axial (b) Cube (c) Double Width (d) Offset

Figure 2: Four candidate coordinate systems for a hexagonal grid. Notice that the cube coordinate system uses three integer indexes and both the axial and cube coordinate system may have negative indices when using a top left origin.

	Offset	Doubled	Axial	Cube
Pointy rotation	evenr, oddr	doublewidth		
Flat rotation	evenq, oddq	double-height	axial	cube
Other rotations	no		yes	
Vector operations (add, subtract, scale)	no	yes	yes	yes
Array storage	rectangular	no*	rhombus*	no*
Hash storage	any shape		any shape	
Hexagonal symmetry	no	no	no	yes
Easy algorithms	few	some	most	most

https://www.redblobgames.com/grids/hexagons/
https://arxiv.org/pdf/1803.02108.pdf

DGGRID Q2DI: A 2D index



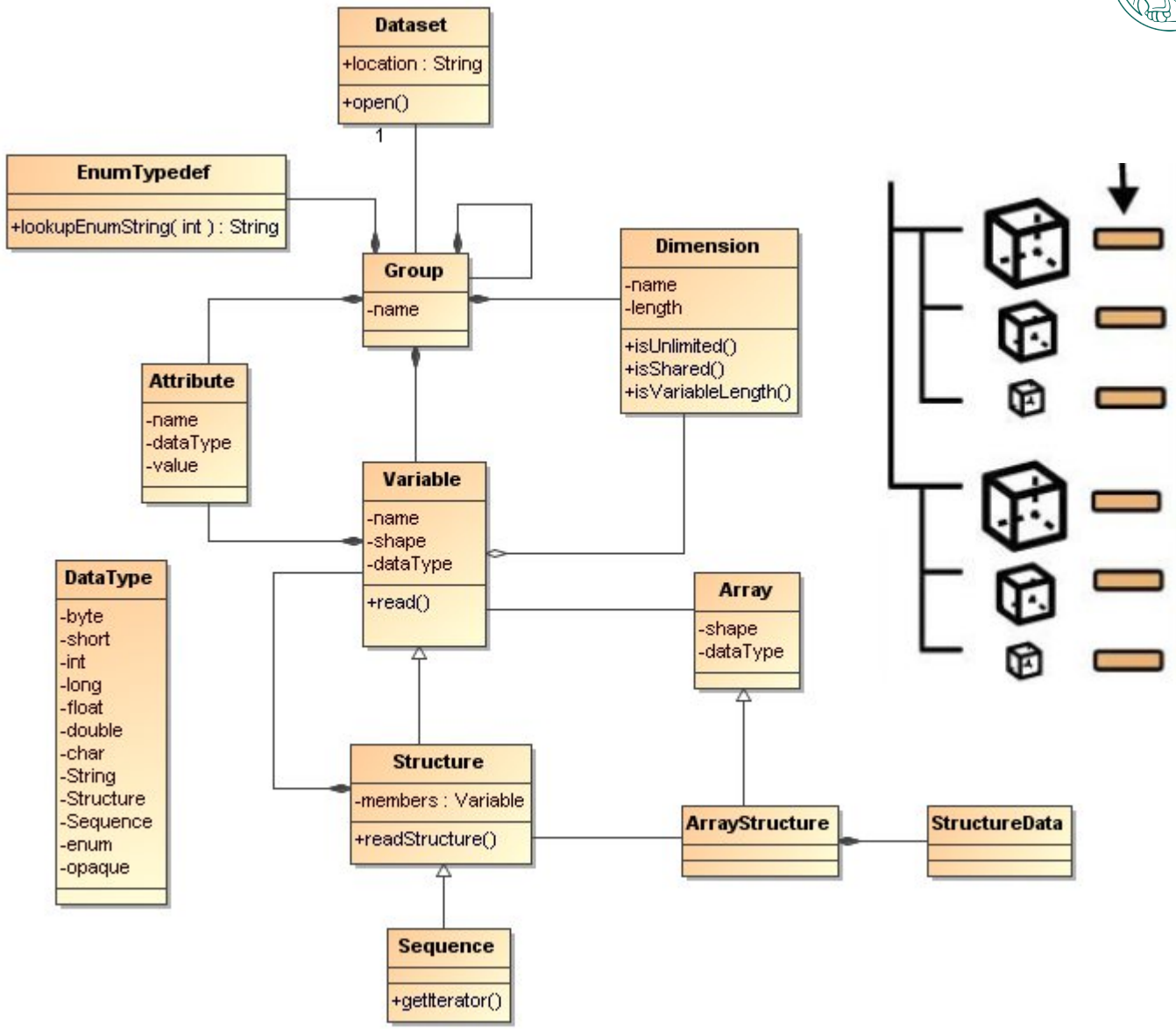
- Q2DI index based on axial coordinates
- Need a simple way to store the data in tensors with rectangular shape
- simple pattern for aperture 4 grids
- complex patterns for ISEA7H

DGGS in the Common Data Model

- DGGS data are just multiple arrays in different resolutions

- Array: A variable (e.g. temperature) is stored in one array
- Dataset: All Variables must share the same axes for space and time
- Groups: All different resolutions for space and time (cross product) are stored as groups
- By using the Common Data Model, we can store DGGS is used in Zarr, NetCDF, HDF5

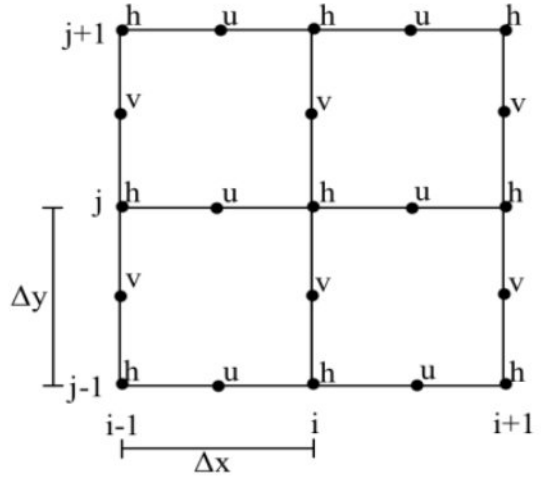
temporal resolution	spatial resolution	Variable	Coordinates	Value
daily	res_spec 5	NIR	(2023, 12, 24, 42)	5.1337
group	group	dataset	array	element



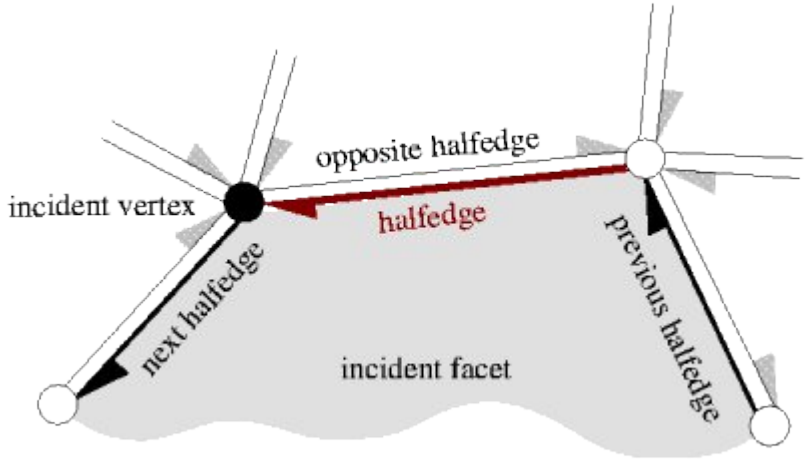
https://docs.unidata.ucar.edu/netcdf-java/current/userguide/common_data_model_overview.html

DGGS for fluxes using a staggered grid

- Staggered grid: Values not only for cell centers but also at other points e.g. wind speed components at vertices and temperature at centers
- Arakawa and Lamb 1977
- Enables to encode direction of fluxes, i.e. vectors from cell center to a vertex
- Often used for dynamic systems e.g. ICON climate simulation to store masses at vertices and wind speeds at centers
- Can be used to store other vector fields as well
- Staggering can be also applied to other dimensions e.g. time and altitude
- Halfedge i.e. doubly-connected edge list, implemented in CGAL
- Atlas of Connectivity maps



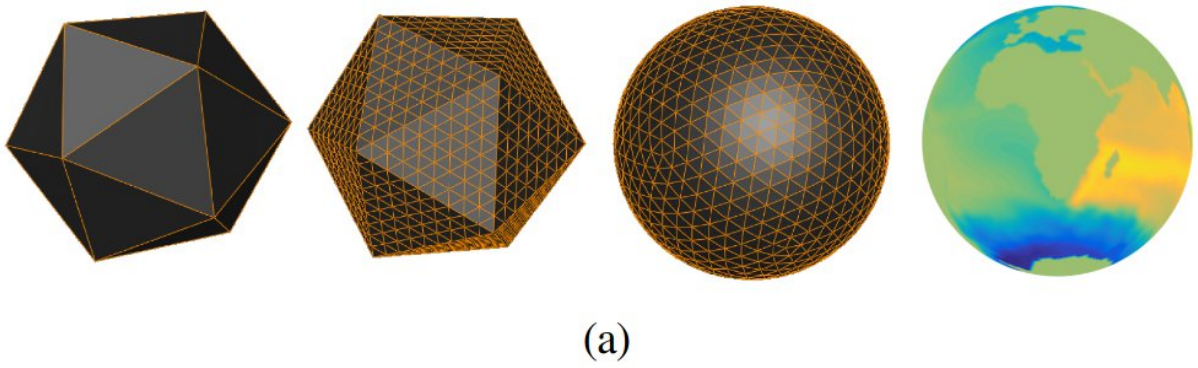
Arakawa Type C horizontal staggering in the ICON grid



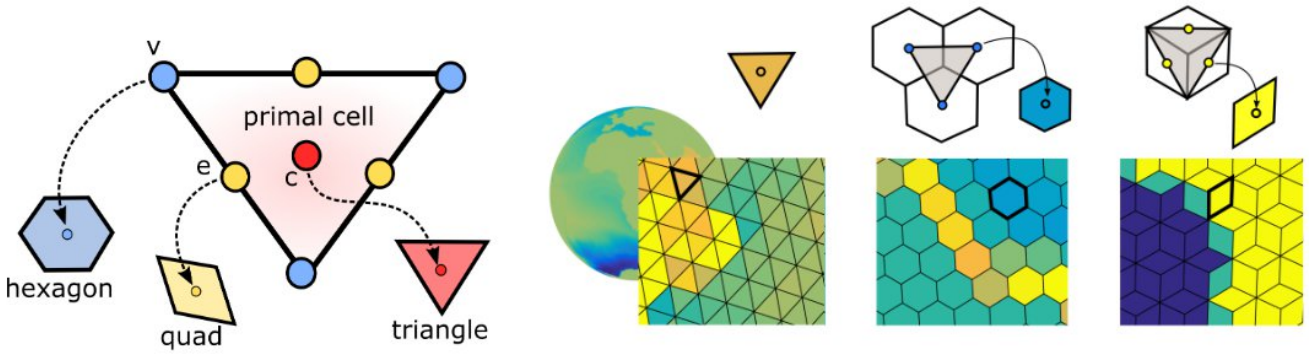
A halfedge stores values for one face, one vertex and one edge

https://doi.org/10.1016/B978-0-12-460817-7.50009-4
http://dx.doi.org/10.5772/55922
https://doc.cgal.org/latest/HalfedgeDS/index.html
https://doc.cgal.org/latest/Polyhedron/index.html
https://doi.org/10.1016/j.cag.2013.09.003
http://dx.doi.org/10.2312/vmv.20161355

ICON grid: A staggered DGGS at centers and vertices



(a)



(b) (c)

Figure 1: (a) (Left to right) Icosahedron as base polyhedron, its refinement, projection and data assignment. (b) Data at the centroid (red), vertices (blue) and edge midpoints (yellow) of the triangles are associated with (c) triangular, hexagonal and quadrilateral cells respectively.

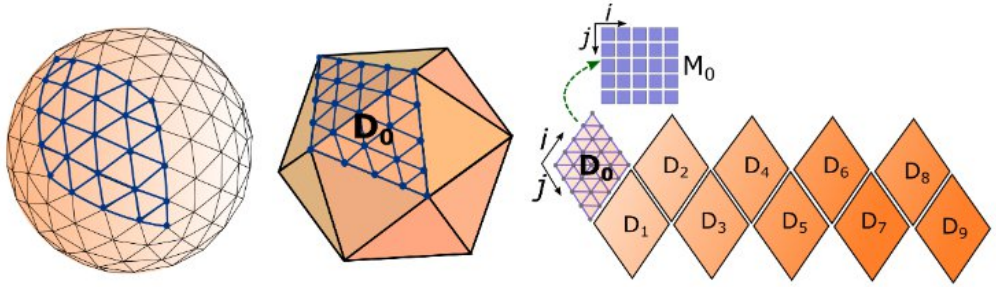
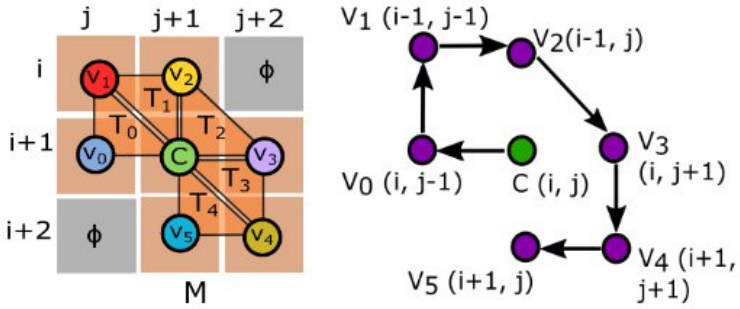


Figure 5: (Left) The Earth's surface unfolded into a net consisting of ten diamonds; each diamaond refers to a paired face of the base icosahedron. (Right) The vertex information of each diamond is stored in a rectangular 2D grid that corresponds to a hexagonal lattice associated with the diamond.



(a) (b)

Figure 7: (a) A hexagonal fan around a vertex. (b) The order in which vertices are placed into the grid.

https://imruljubair.github.io/papers/visagg_presentation.pdf
http://dx.doi.org/10.2312/vmv.20161355

ICON grid: Unifying cell shapes and build pyramid using wavelets

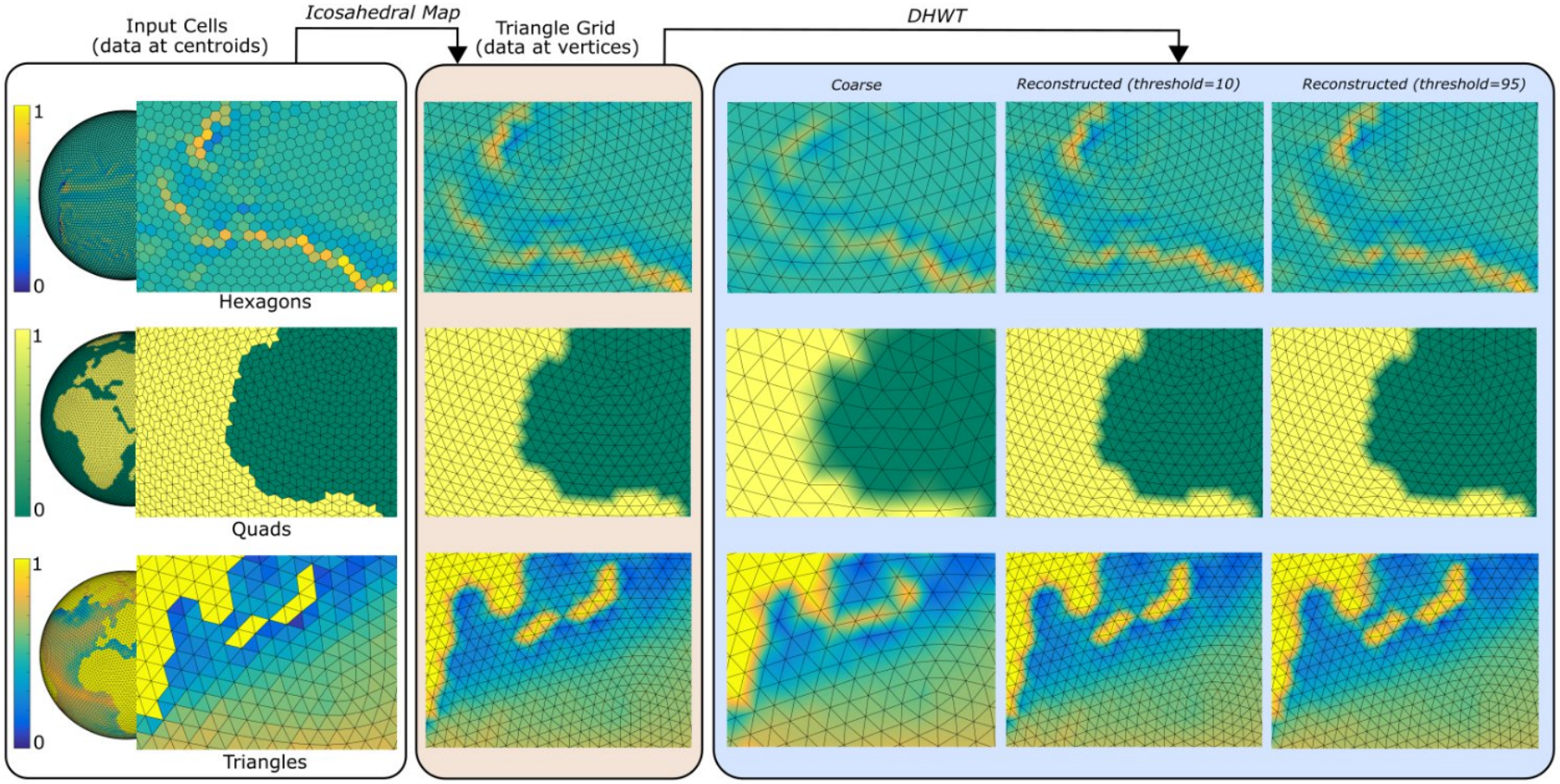


Figure 12: Results of applying icosahedral maps and the DHWT on ICON data at the centroids of hexagons (top), edge midpoints in one direction (middle), and the centroids of triangles (bottom). The grid resolutions per diamond from top to bottom are 65×65, 65×65 and 99×67 respectively. Each row (left to right) illustrates the original cell-type in ICON (data at centroids), result of converting to a triangular grid (data at vertices) via the icosahedral map, applying DHWT on the converted grid to obtain coarse data, and reconstructed data using two different thresholds: 10 and 95. The figure also focuses on the borders of the diamonds around an extraordinary vertex to show the smoothness along boundaries due to padding.

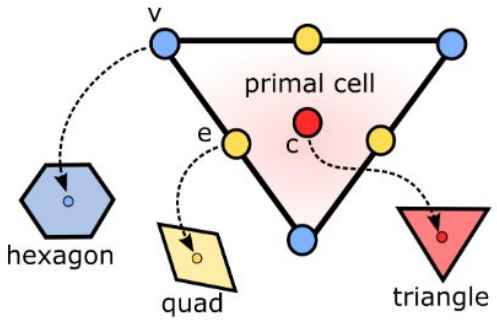
- Abstract layer to model triangles, diamonds and hexagons (similar to DGGRID ISEA)
- Discrete wavelet transformation for downsampling and convolution

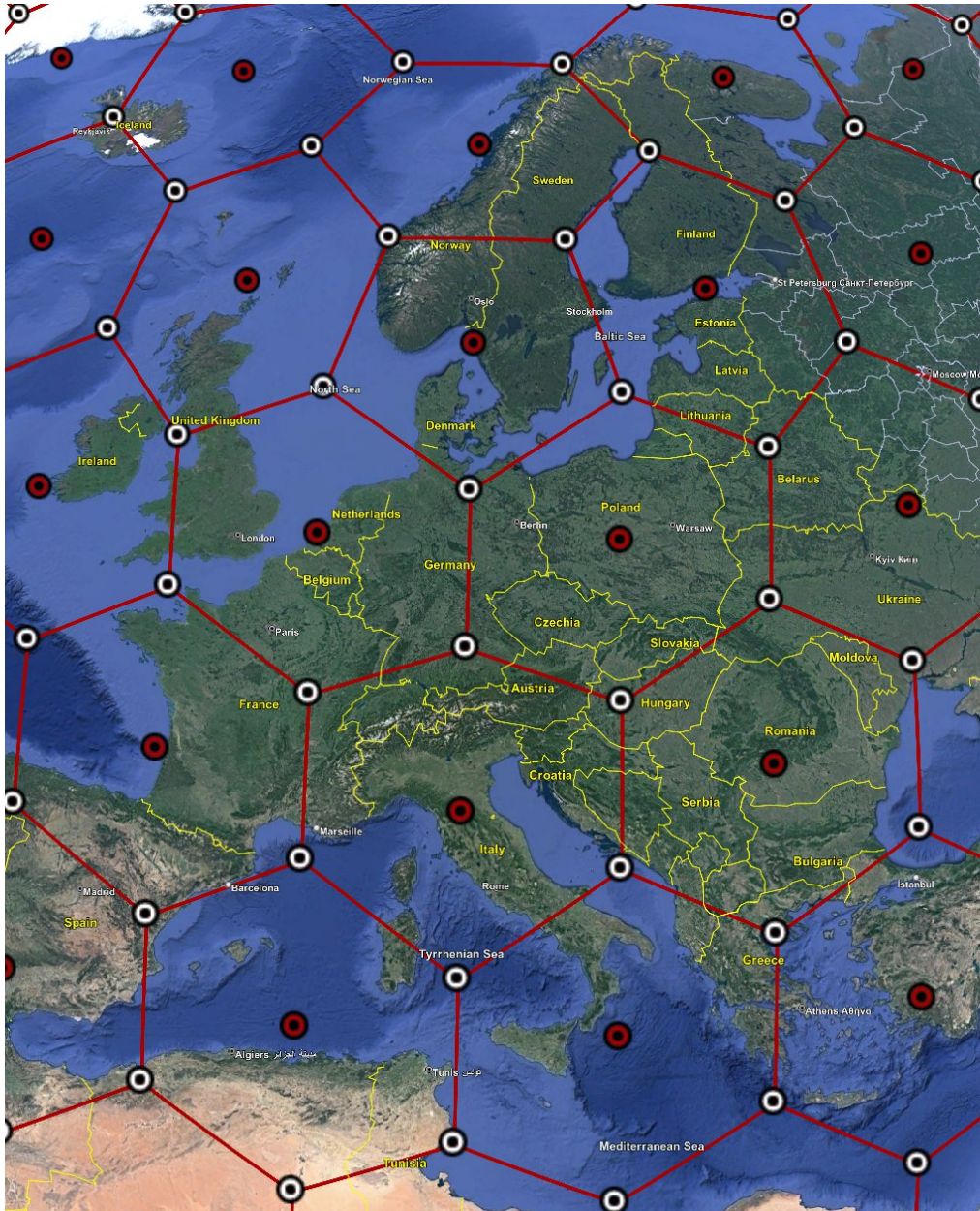
https://imruljubair.github.io/papers/visagg_presentation.pdf
http://dx.doi.org/10.2312/vmv.20161355

Staggering grid in DGGRID

- Hexagon vertices are just triangle centers in the ISEA grid at aperture 4
- The same idea is applicable to edges ($N_{edges} = N_{vertices}$)
- Add a staggering layer in the common data model to store edges and vertices in separate triangular grids
- Must use aperture 4

Position	Grid	N_{points} at resolution 3
center	ISEA4H	642
vertex	ISEA4T	1280
edge	ISEA4T	1280





ISEA grid created by DGGRID at resolution 3 and aperture 4. Hexagons and triangles are shown in red and white, respectively.