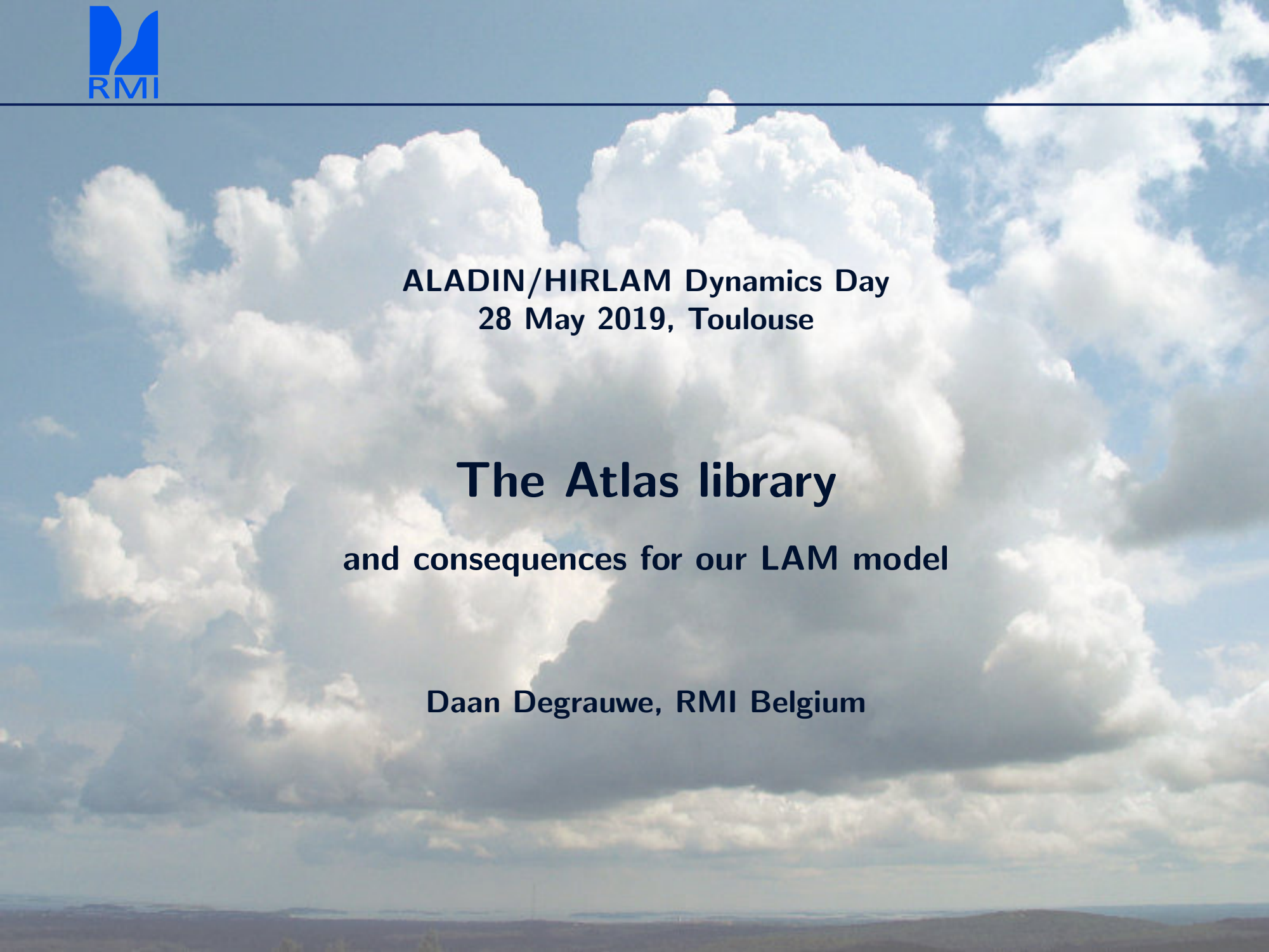


ALADIN/HIRLAM Dynamics Day
28 May 2019, Toulouse

The Atlas library

and consequences for our LAM model

Daan Degrauwe, RMI Belgium

Content

- Background
- Generalities of the Atlas library
- LAM features in Atlas
- Future plans

Background

Triggers for Atlas development at ECMWF:

- Need for a common framework for EULAG model (finite volume-based, possibly unstructured grids) and IFS;
- Modernization of IFS in view of scalability challenges;
- Platform for NWP dwarfs of the ESCAPE project. This project focused on scalability and energy-efficiency on heterogeneous hardware.

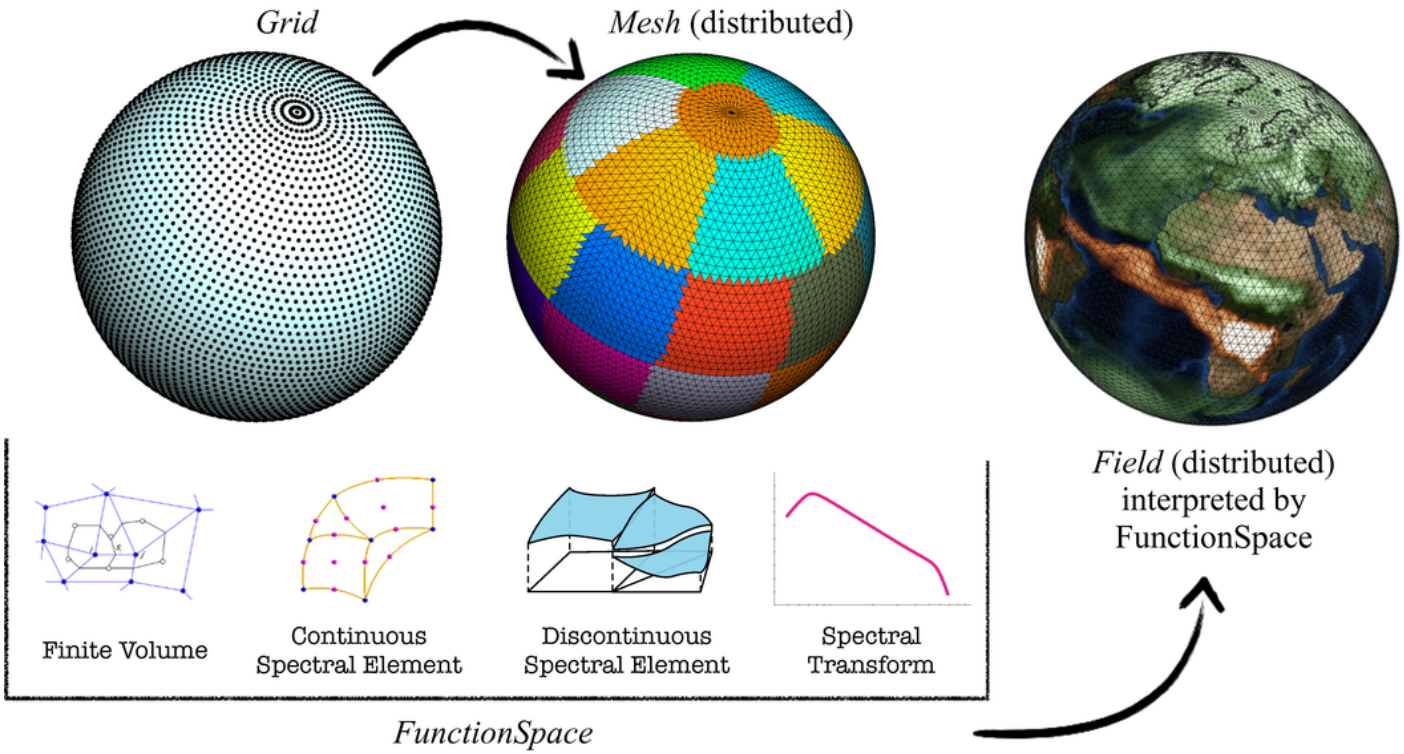
Atlas generalities

- Object-Oriented design enabling runtime flexibility and abstraction from implementation
- Mostly written in C++, but with complete Fortran interface
- Modern coding, e.g. including unit testing
- Tasks of Atlas:
  - Mesh generation
  - Parallelization
  - Data structures
  - Interpolation algorithms
  - Logging
  - I/O support

  . . . and provide canonical test cases for these features

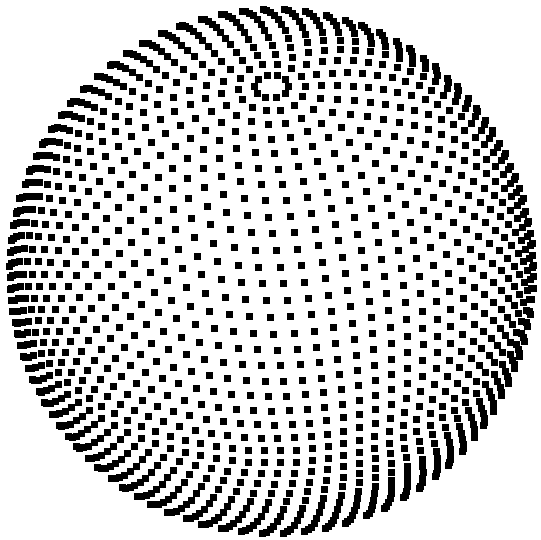
Atlas generalities: design

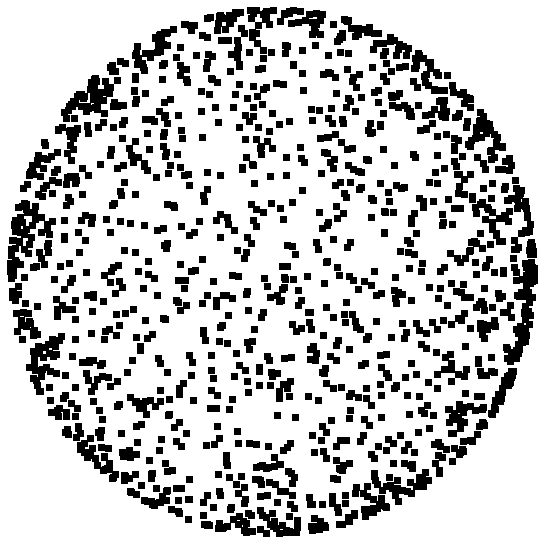
The workflow in Atlas is the following:



Atlas generalities: grid object

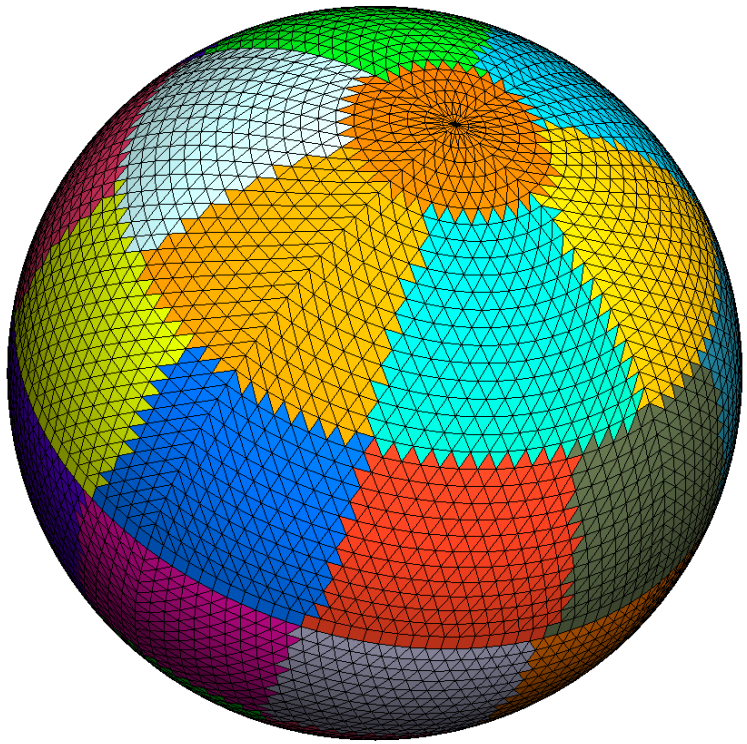
- In Atlas, a `Grid` object is just a collection of gridpoints (structured or unstructured)





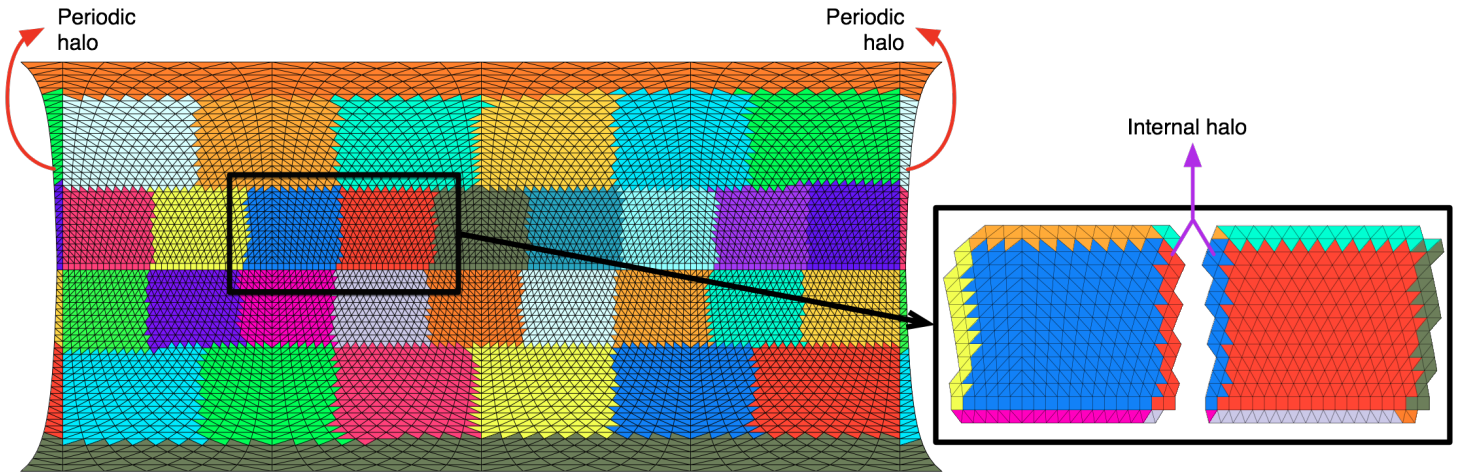
Atlas generalities: mesh object

- A `Mesh` is distributed by a `Partitioner`, and describes the connectivity between gridpoints by defining triangular and quadrangular cells.



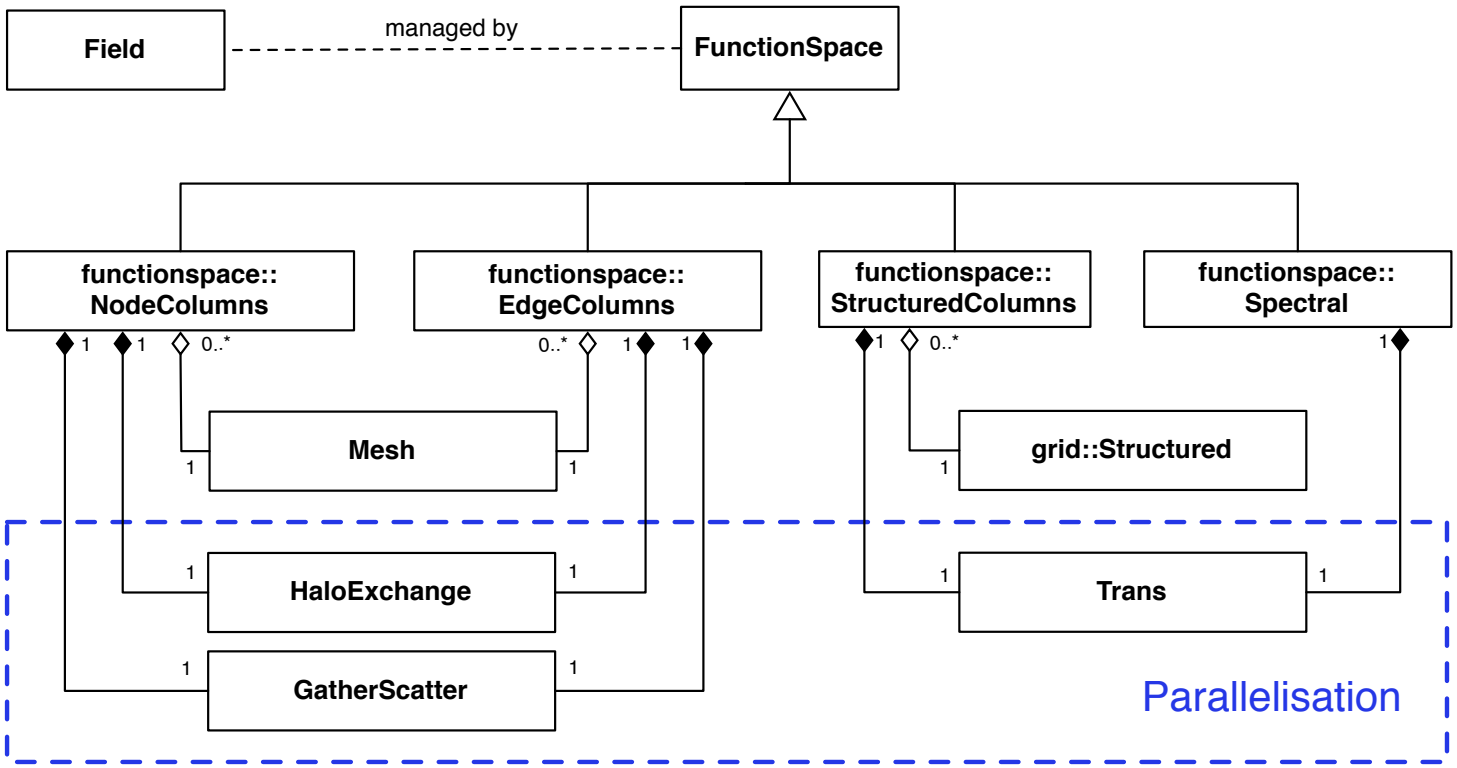
Atlas generalities: mesh object

- A `Mesh` is distributed by a `Partitioner`, and describes the connectivity between gridpoints by defining triangular and quadrangular cells.
- The mesh includes halo's for communications with neighboring processors



Atlas generalities: field and functionspace object

- A `Field` can be represented in several ways: spectral coefficients, grid point values, cell-center values, edge-center values, finite-element integration points, ...
- To cope with these different representations, the concept of a `FunctionSpace` is introduced.

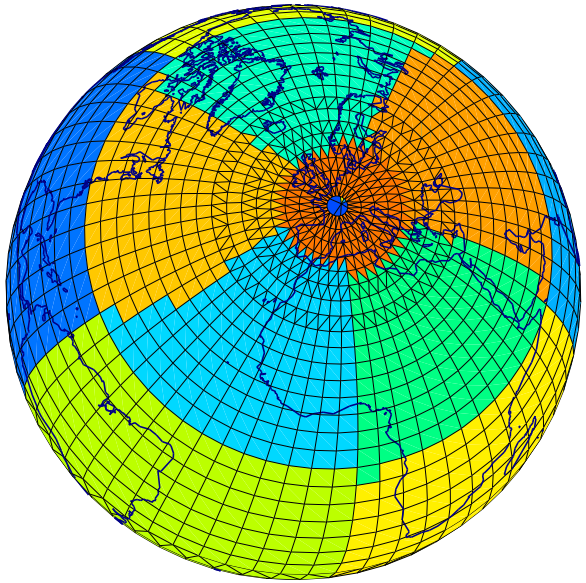


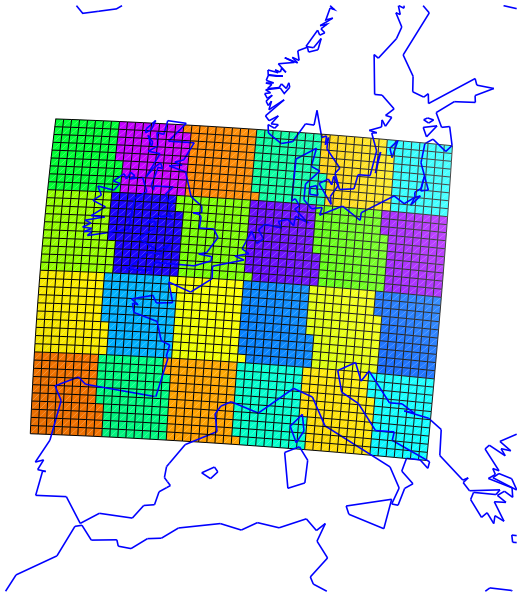
Atlas generalities: field and functionspace object

- A `Field` can be represented in several ways: spectral coefficients, grid point values, cell-center values, edge-center values, finite-element integration points, ...
- To cope with these different representations, the concept of a `FunctionSpace` is introduced.
- Numerical operators (e.g. nabla-operator) are defined within a `FunctionSpace`
- The actual storage (memory layout) of a Field is abstracted in such a way that it can be (and is!) optimized for accelerators such as GPU's.

LAM features: grids and projections

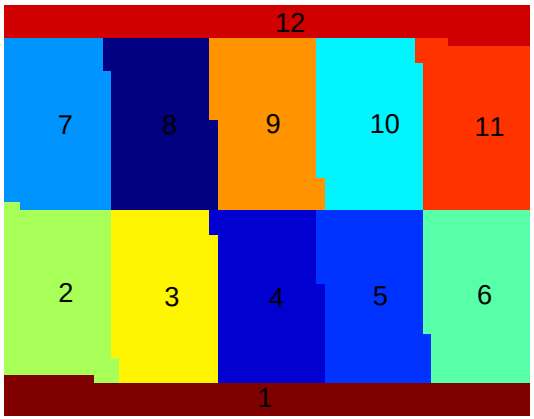
- LAM's require the concept of a geographic `Projection`, to distinguish between grid coordinates (x, y) and geographic coordinates (λ, ϕ).
- Following projections were introduced in Atlas:
  - (rotated) longitude-latitude
  - conformal Lambert
  - (rotated) Schmidt (ARPEGE stretching)
  - (rotated) Mercator



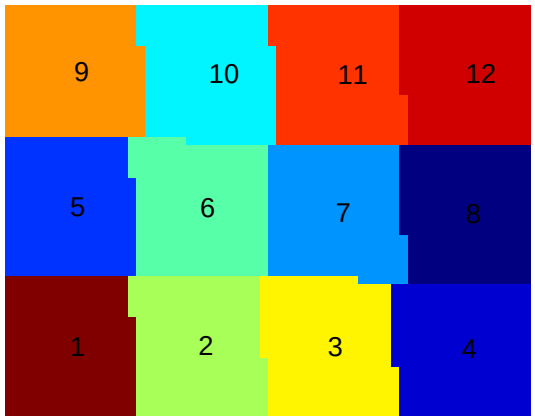


LAM features: partitioner

- LAM's require a dedicated partitioner to distribute a grid/mesh on a parallel machine.
- The `Checkerboard` partitioner (quite close to the one used in ALADIN/HIRLAM) was introduced in Atlas



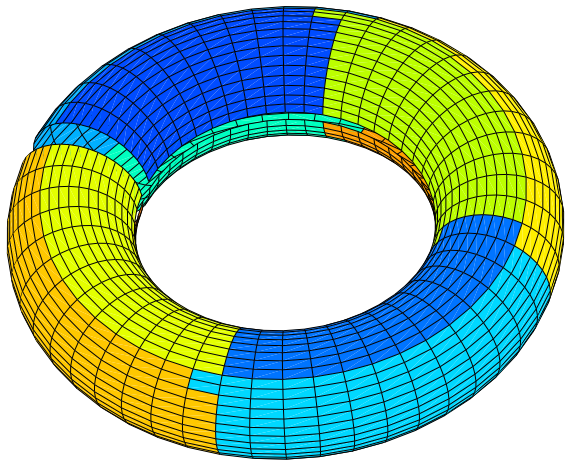
Equal-regions partitioner for global grid



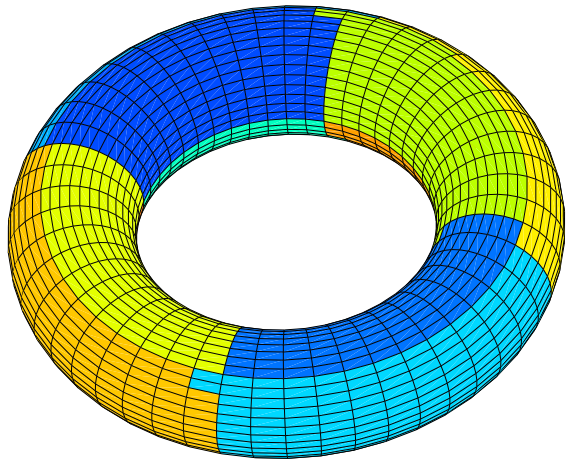
Checkerboard partitioner for LAM grid

LAM features: mesh

- In principle, LAM meshes do not differ substantially from global meshes.
- One exception: our spectral LAM is bi-periodic, so extra cells are required to connect opposite sides. This is now also supported in Atlas.



non-periodic LAM grid in Atlas



periodic LAM grid in Atlas

What's next?

- Introduction of Atlas in IFS: progressively, initially with wrappers so existing code isn't broken
- Some features haven't been tested/developed for LAM yet:
  - map factors and effect of geographic projection on differential operators
  - semi-Lagrangian advection dwarf and interpolation schemes
  - finite-volume functionspace
- Whether 'we are ready for Atlas' depends on our ambitions and on how our dynamics will evolve!
  - If we keep a semi-implicit semi-Lagrangian spectral model, Atlas contains most of what we need.
  - Going to ALADIN-FVM will require much more work.
  - Implementing a non-spectral Helmholtz solver in Atlas would be a useful exercise!
  - Should Davies relaxation be dealt with in Atlas? What about VFE? Or FullPos? ...

Thank you