HIRLAM All Staff Meeting/ALADIN Workshop
3–6 April 2017, Helsinki

The Atlas library and LAM features therein

Daan Degrauwe, RMI Belgium

with much help from Willem Deconinck, ECMWF

Content

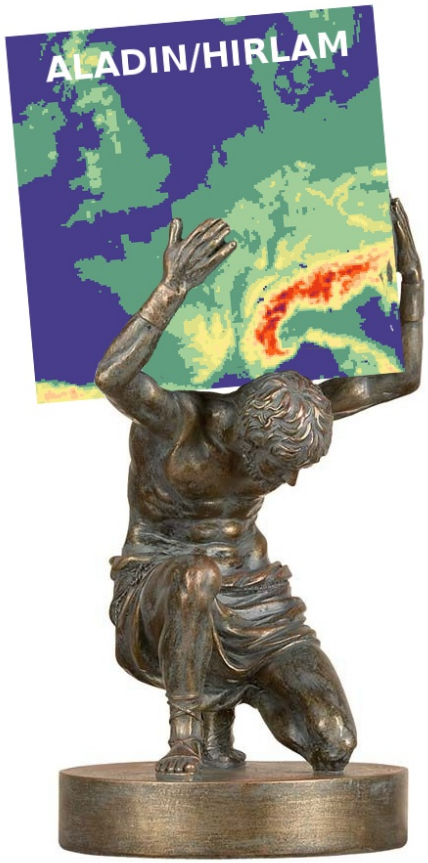
- Background
- Generalities of the Atlas library
- LAM features in Atlas
- Future plans

Background

- In Greek mythology, Atlas is the titan that supports the sky.
- Atlas is a software library developed at ECMWF for parallel data structures

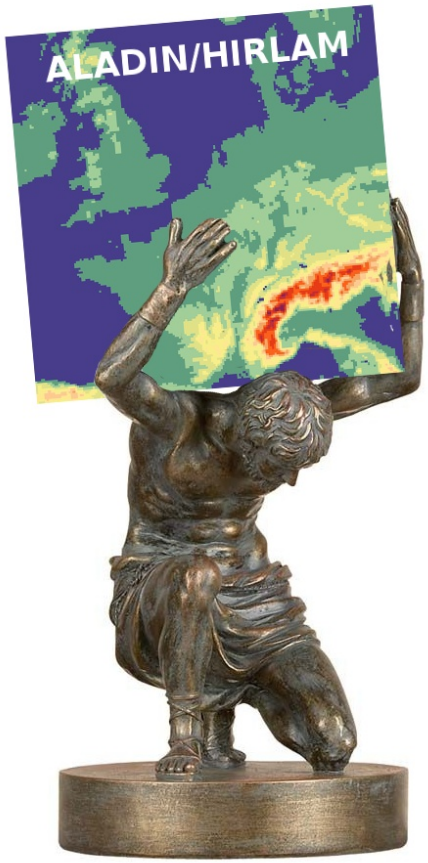
Background

- In Greek mythology, Atlas is the titan that supports the sky.
- Atlas is a software library developed at ECMWF for parallel data structures
- We would like Atlas to also support limited area models!



Background

- In Greek mythology, Atlas is the titan that supports the sky.
- Atlas is a software library developed at ECMWF for parallel data structures
- We would like Atlas to also support limited area models!
- Make sure to do this at an early stage in the development!



Background

Triggers for Atlas development:

- 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 focuses on energy-efficiency and heterogeneous hardware.

ECMWF decided to take an evolutionary approach, rather than a disruptive one:
"Atlas is a flexible parallel framework for unstructured hybrid meshes and structured grids".

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

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
  - I/O support
  - Logging

  . . . and provide canonical test cases for these features

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

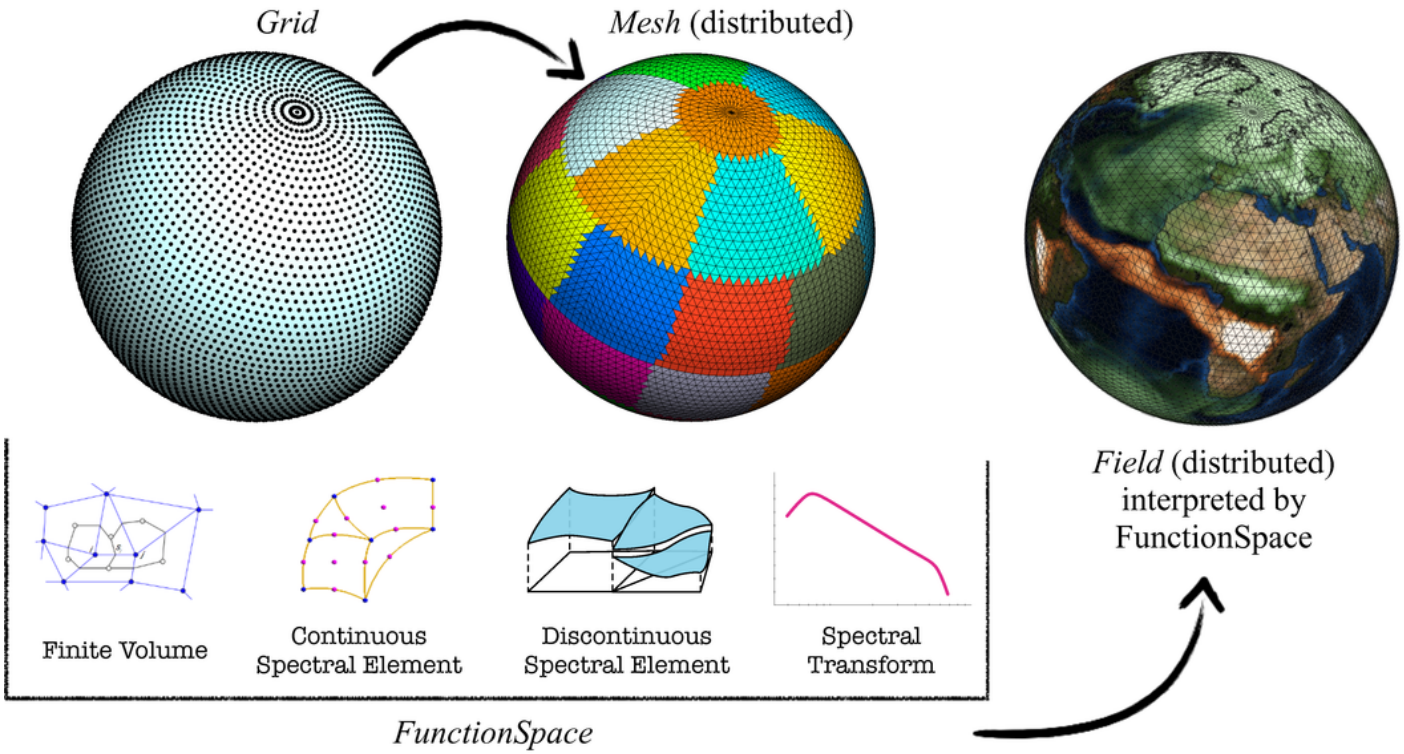
- Tasks of Atlas:
  - Mesh generation
  - Parallelization
  - Data structures
  - Interpolation algorithms
  - I/O support
  - Logging

  ...and provide canonical test cases for these features

- Atlas is not OOPS!
  - OOPS is blind to LAM, Atlas is not
  - OOPS provides an entire DA and forecasting system, Atlas provides a supporting framework

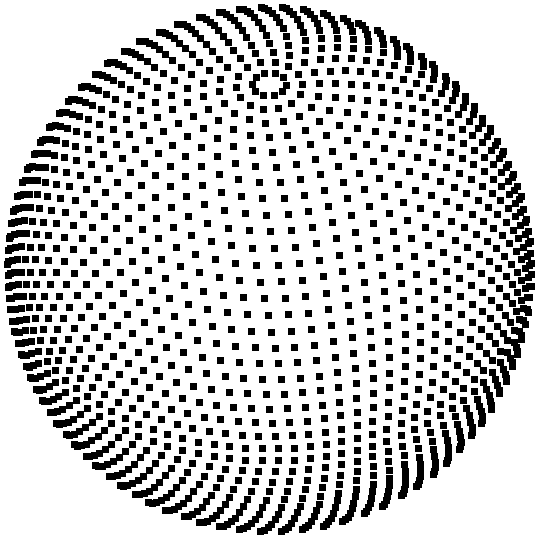
Atlas generalities: design

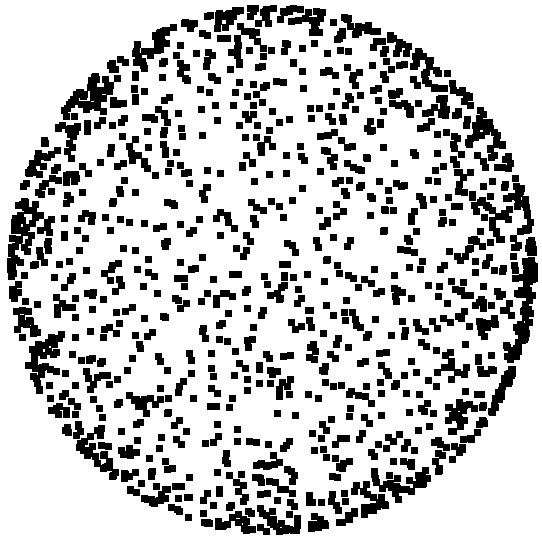
The workflow in Atlas is the following:



Atlas generalities: grid object

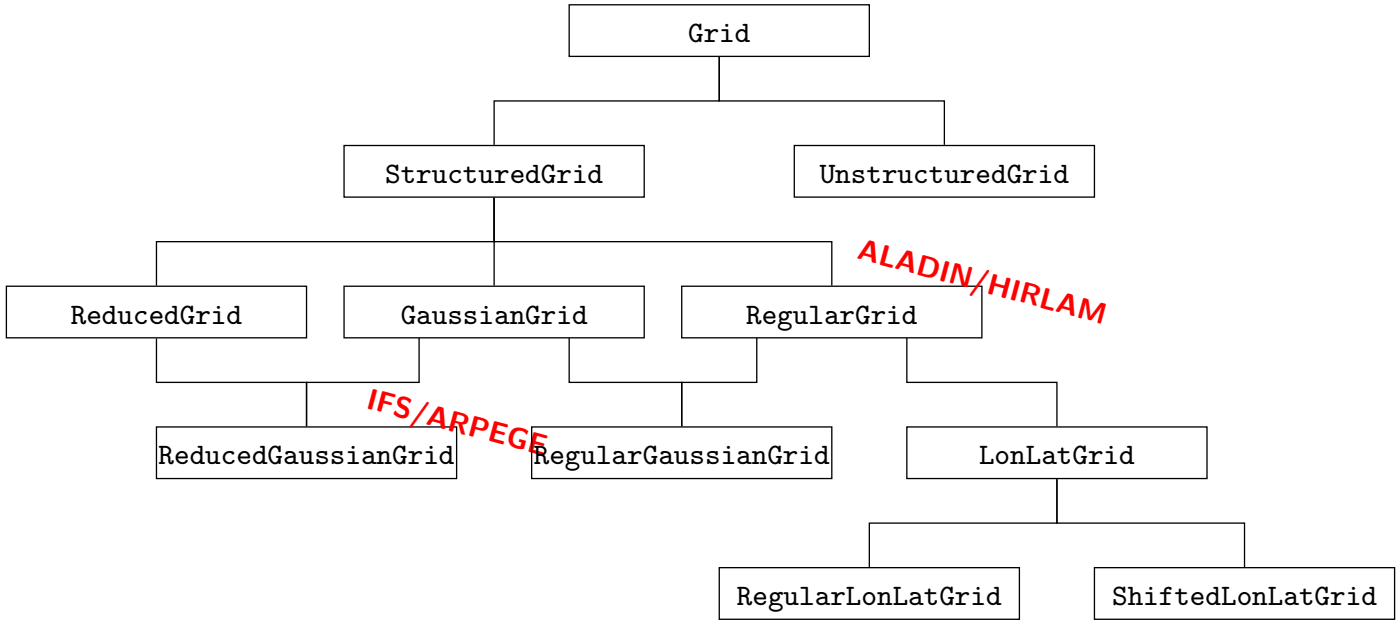
- In Atlas, a `Grid` object is just a collection of gridpoints (structured or unstructured)





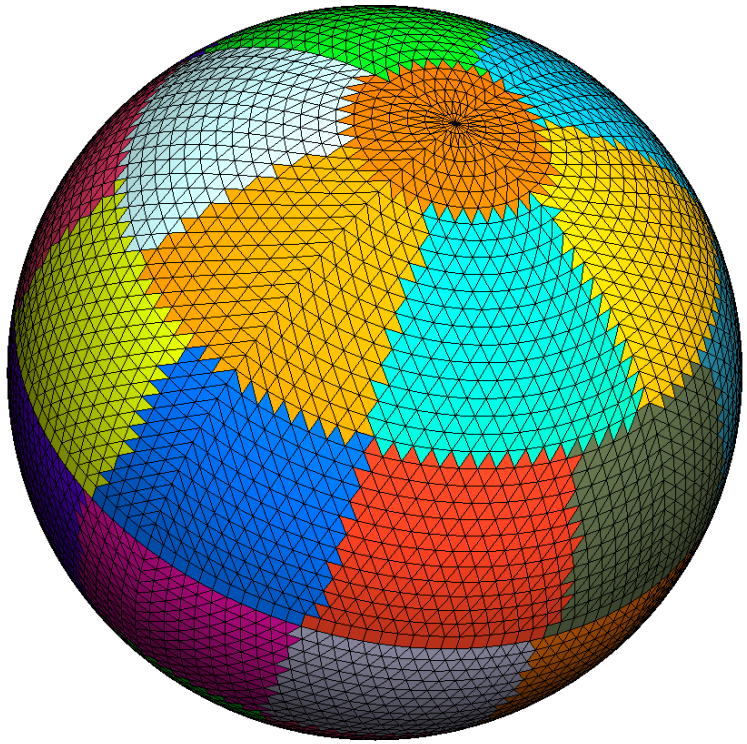
Atlas generalities: grid object

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



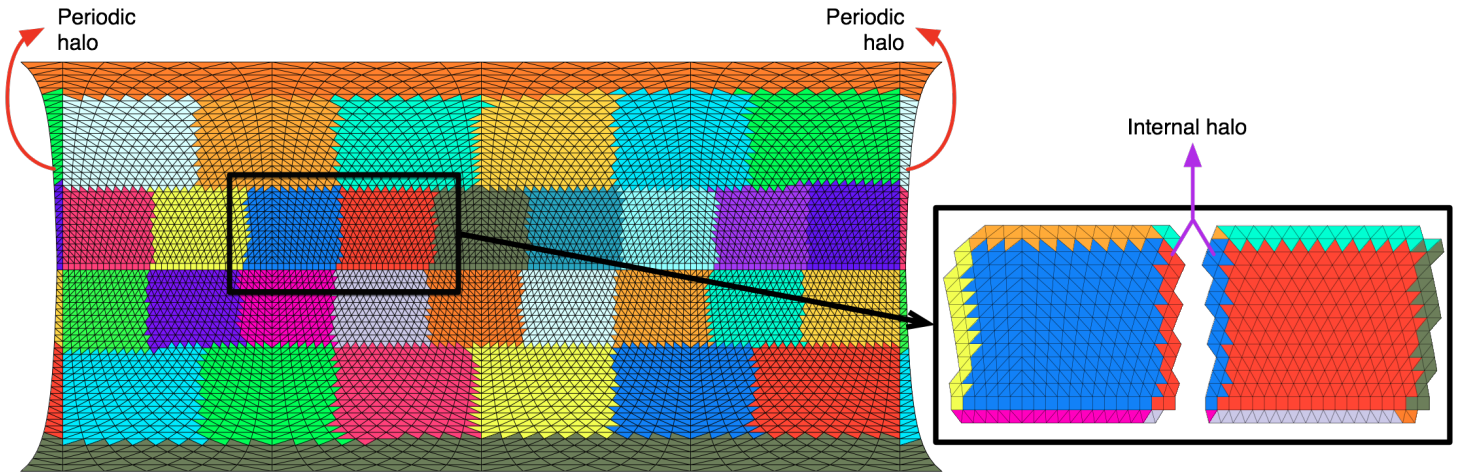
Atlas generalities: mesh object

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



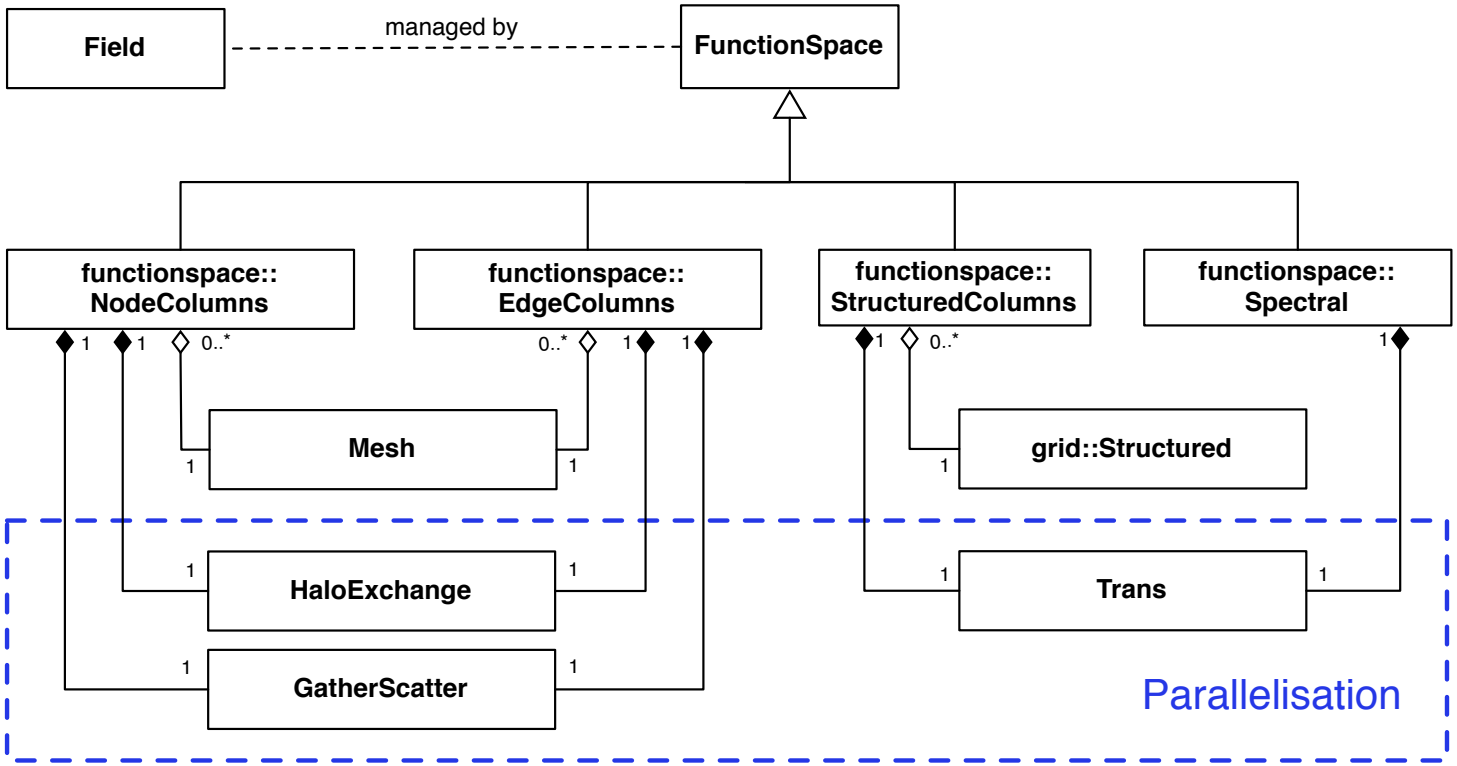
Atlas generalities: mesh object

- 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.

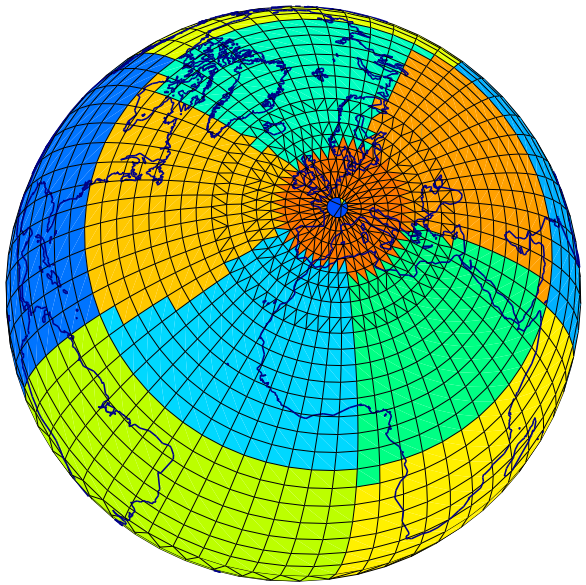


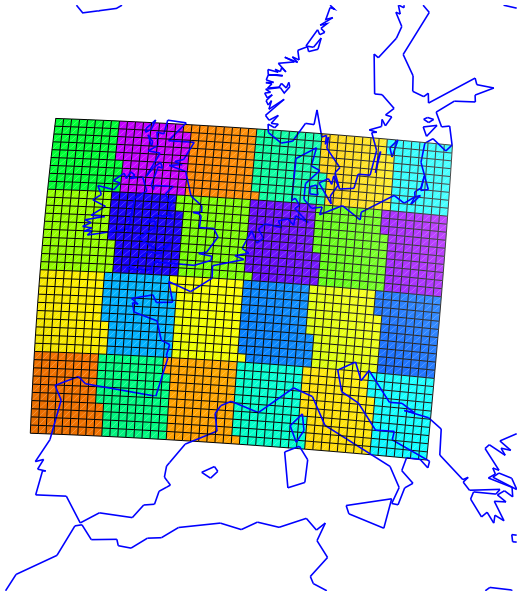
Atlas generalities: field and functionspace object

- 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

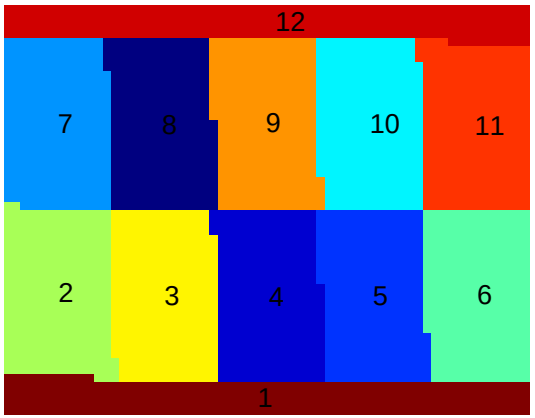
- LAM's require the concept of a geographic `Projection`, to distinguish between grid coordinates (x, y) and geographic coordinates (λ, ϕ).
- Following projections were introduced in Atlas:
  - (rotated) longitude-latitude
  - conformal Lambert
  - (rotated) Schmidt (ARPEGE stretching)
  - (rotated)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



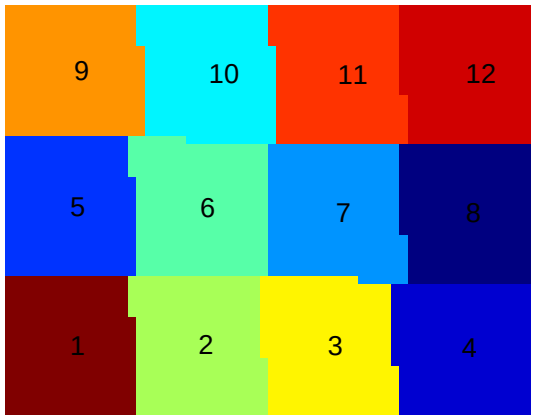


LAM features: partitioner

- 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



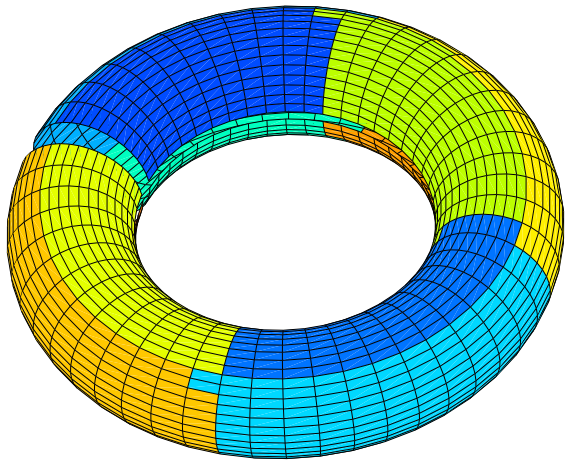
Equal-regions partitioner for global grid



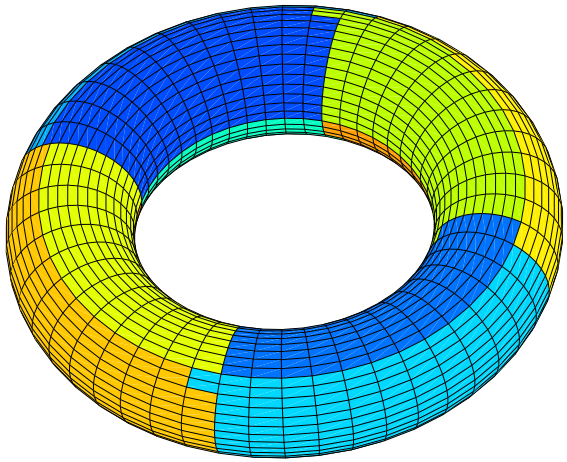
Checkerboard partitioner for LAM grid

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

Conclusions and future plans

- Introduction of Atlas in IFS: not immediately; probably only after OOPS
- Still, it's reassuring that LAM requirements are already part of Atlas from the beginning!

  (although some issues still need to be addressed, e.g. an interface to etrans)

Conclusions and future plans

- Introduction of Atlas in IFS: not immediately; probably only after OOPS
- Still, it's reassuring that LAM requirements are already part of Atlas from the beginning!

  (although some issues still need to be addressed, e.g. an interface to etrans)
- Future features in Atlas:
  - (advanced) interpolation schemes for multigrid systems and semi-Lagrangian advection; close integration with MIR (ECMWF's interpolation library).
  - more function spaces (finite elements, spectral elements) and numerical operators.

Conclusions and future plans

- Introduction of Atlas in IFS: not immediately; probably only after OOPS
- Still, it's reassuring that LAM requirements are already part of Atlas from the beginning!

  (although some issues still need to be addressed, e.g. an interface to etrans)
- Future features in Atlas:
  - (advanced) interpolation schemes for multigrid systems and semi-Lagrangian advection; close integration with MIR (ECMWF's interpolation library).
  - more function spaces (finite elements, spectral elements) and numerical operators.
- What will the future look like? Maybe it's a model with
  - spectral transforms on optical processors;
  - the radiation scheme on GPU's;
  - turbulence on quantum processors;
  - ???

  . . . but at least the Atlas library seems a development that may allow such a future.

Thank you