Recent Atlas developments for Earth System Modelling

7th ENES HPC Workshop
Monday, May 9 2022 - Wednesday, May 11 2022
Barcelona Supercomputing Center (BSC), Spain

Willem Deconinck
with presented contributions by Slavko Brdar, Pedro Maciel (ECMWF), Oliver Lomax, Marek Wlasak, Toby Searle (UK Met Office)

ECMWF
willem.Deconinck@ecmwf.int

Atlas, a library for NWP and climate modelling – *Deconinck et al. 2017, J-CPC*

Atlas is an open-source library providing grids, mesh generation, and parallel data structures targeting Numerical Weather Prediction or Climate Model developments.

- Modern C++ library implementation with modern Fortran 2008 (OOP) interfaces

  → integration in existing models
  → Fortran / C++ interoperable data structures

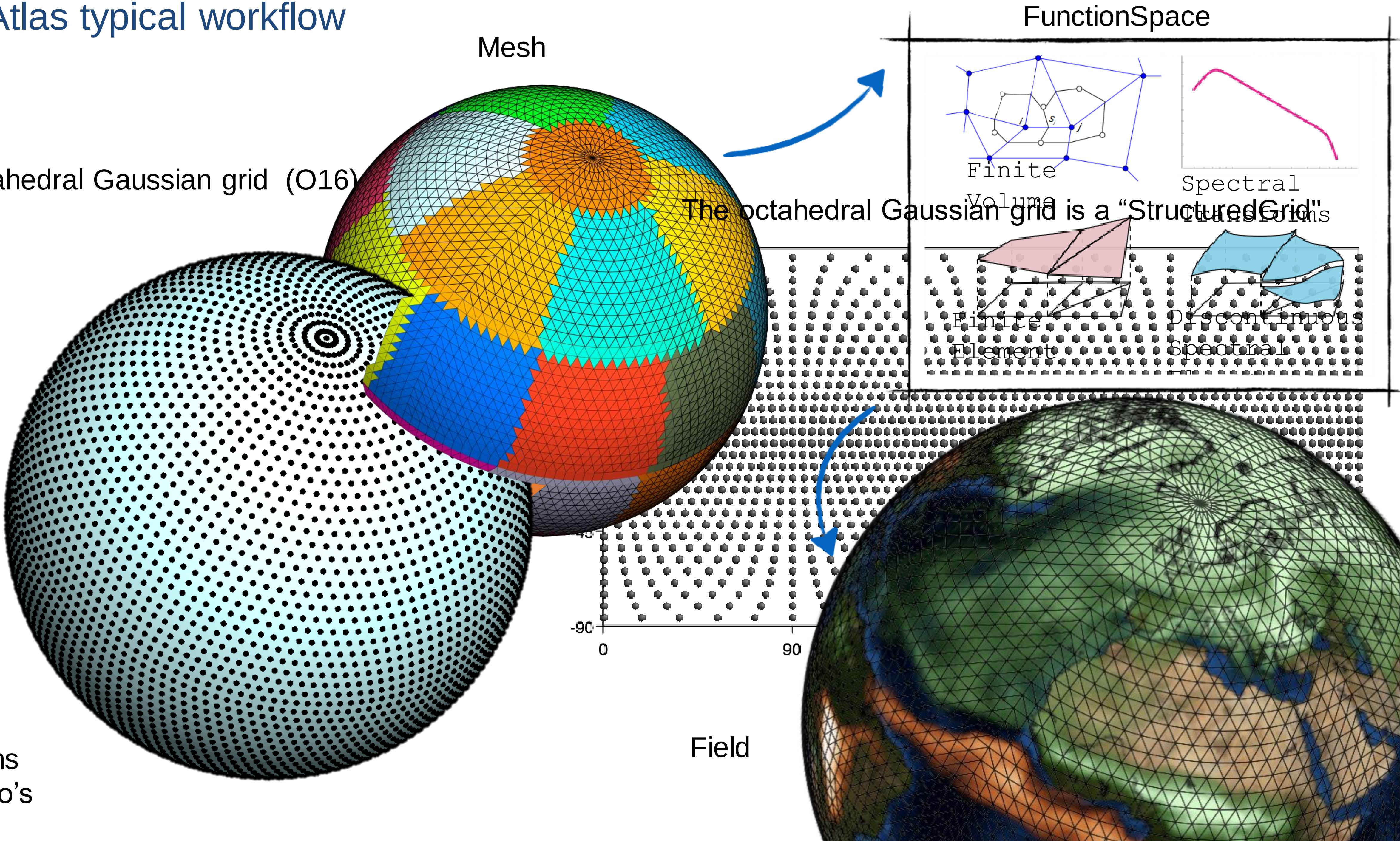
- Open-source (Apache 2.0),
  - Sources: http://github.com/ecmwf/atlas
  - Documentation: https://sites.ecmwf.int/docs/atlas
- Extensible design
  - Operators based on base Atlas concepts via Object oriented design.
  - Plugin architecture with self registration allowing for custom extensions

ECMWF

Atlas, a library for NWP and climate modelling – *Deconinck et al. 2017, J-CPC*

- Data structures to enable new numerical algorithms, e.g. based on unstructured meshes
  - Support structured and unstructured grids
  - Support **global** as well as **regional** grids
  - Grids with projections including general Proj support (proj.org) – by Pedro Maciel
- Separation of concerns
  - Grids / Mesh generation
  - Parallelisation (domain decomposition, halo exchanges)
  - Accelerator-aware data structures (GPU/CPU/…)
- Readily available operators
  - Remapping and interpolation
  - Gradient, divergence, laplacian
  - Spherical Harmonics transforms

Atlas typical workflow



Mesh

Octahedral Gaussian grid (O16)

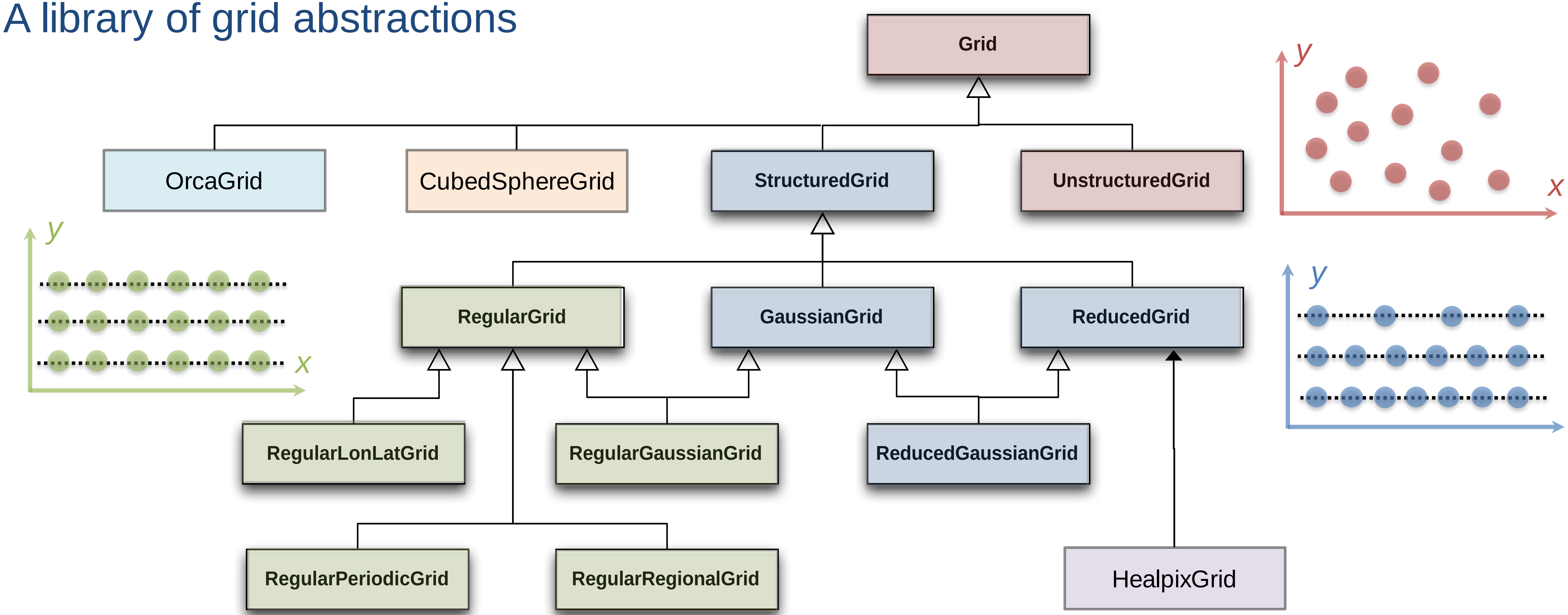
Grid

The octahedral Gaussian grid is a “StructuredGrid"

Field

Partitions
with halo’s

A library of grid abstractions



Grid

OrcaGrid

CubedSphereGrid

StructuredGrid

UnstructuredGrid

RegularGrid

GaussianGrid

ReducedGrid

RegularLonLatGrid

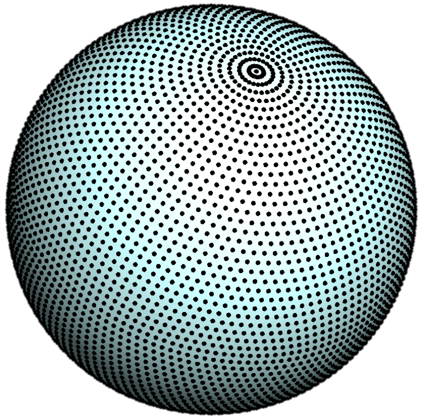
RegularGaussianGrid

ReducedGaussianGrid

RegularPeriodicGrid

RegularRegionalGrid

HealpixGrid



Example creation of operational octahedral reduced Gaussian grid using unique identifier

C++

```
atlas::Grid grid;
grid = atlas::Grid ( “O1280” )
```

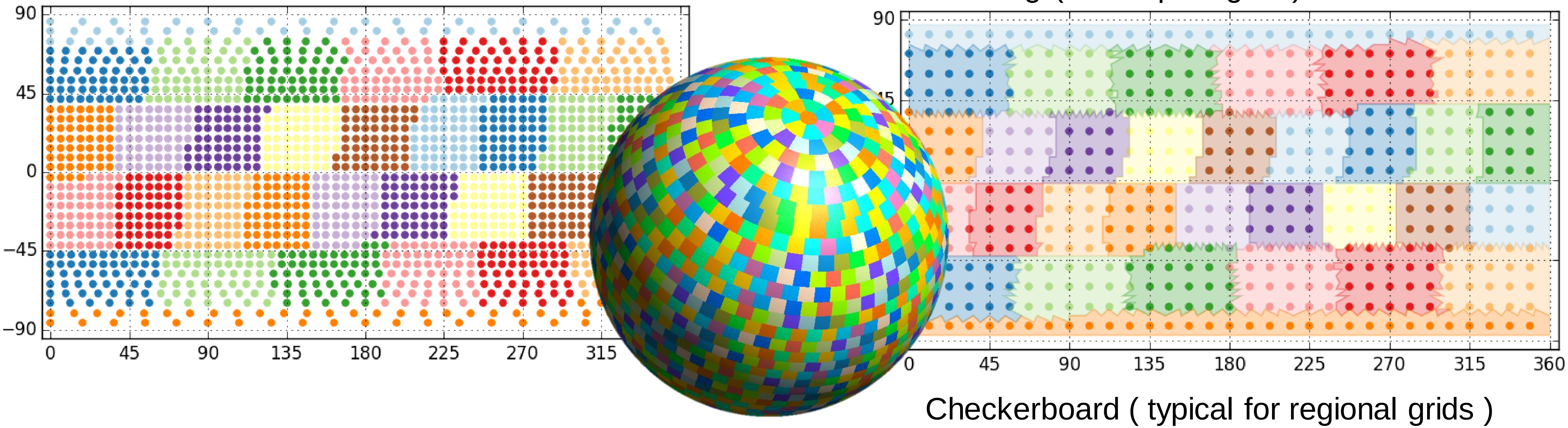
Fortran

```
type(atlas_Grid) :: grid
grid = atlas_Grid( “O1280” )
```

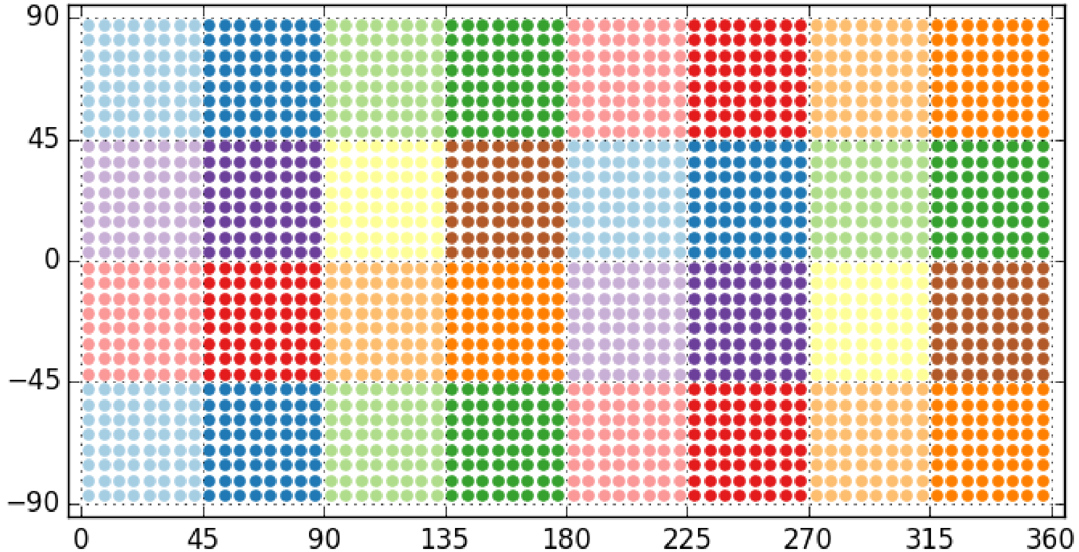
Multiple domain decomposition strategies

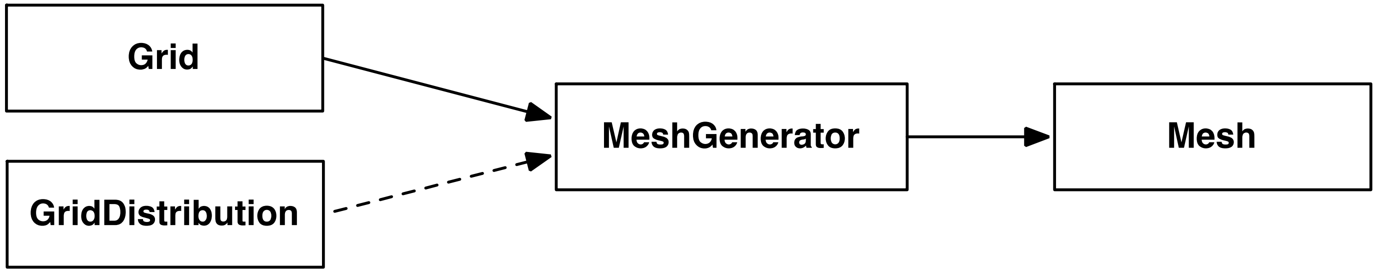
Equal Regions (typical for IFS grids)

Matching (for coupled grids)



Checkerboard (typical for regional grids)

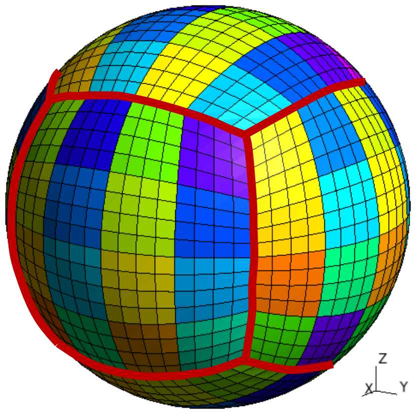


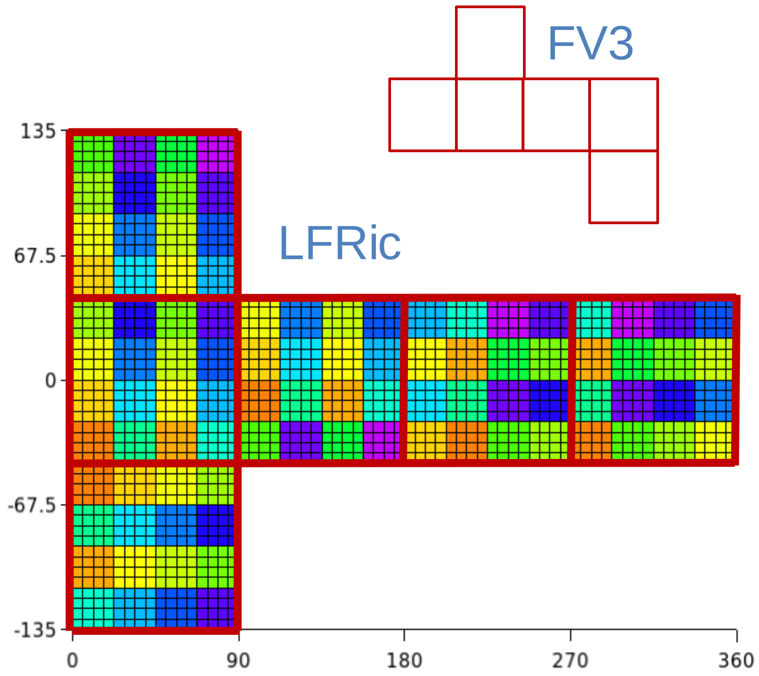


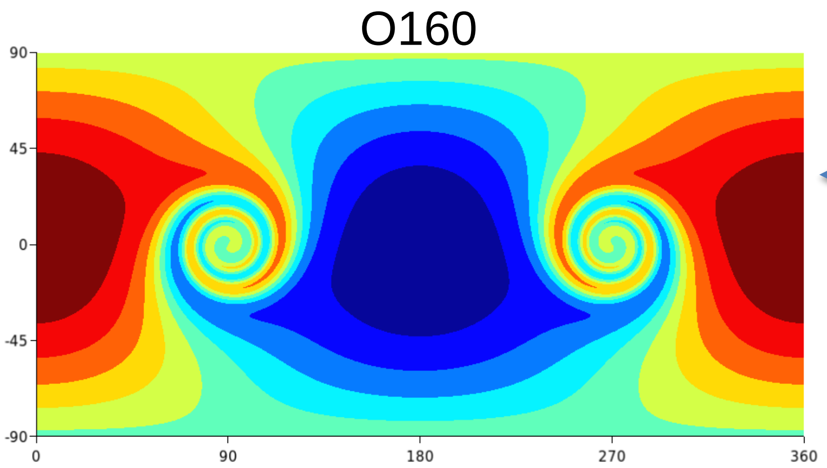
New Atlas support for cubed sphere grids

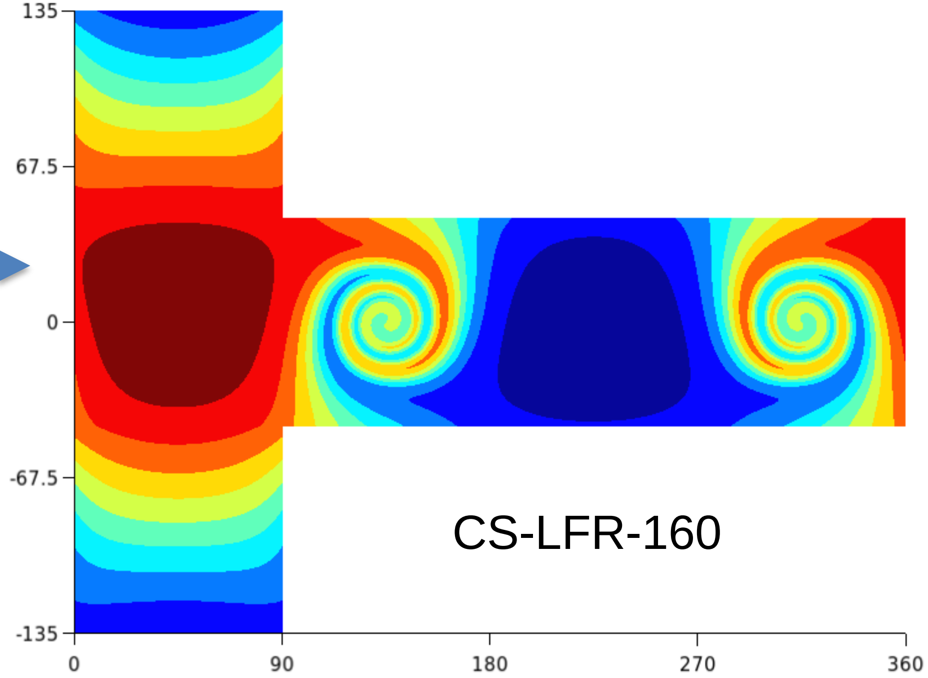
by Oliver Lomax, Marek Wlasak (UK Met Office) and Dan Holdaway (NASA)

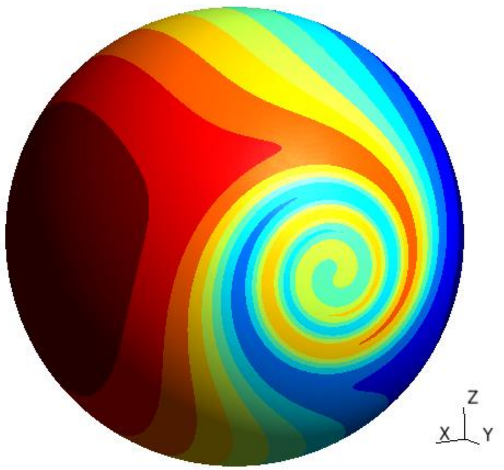
- CubedSphereGrid
  - Flexibility in grid tiling (ordering, orientation, …)
  - Flexibility in projection (equiangular, equidistant, stretching)
- CubedSpherePartitioner
- CubedSphereMeshGenerator
- CubedSphereFunctionSpaces (NodeColumns,CellColumns)
- CubedSphereBilinearInterpolation (uses structure)









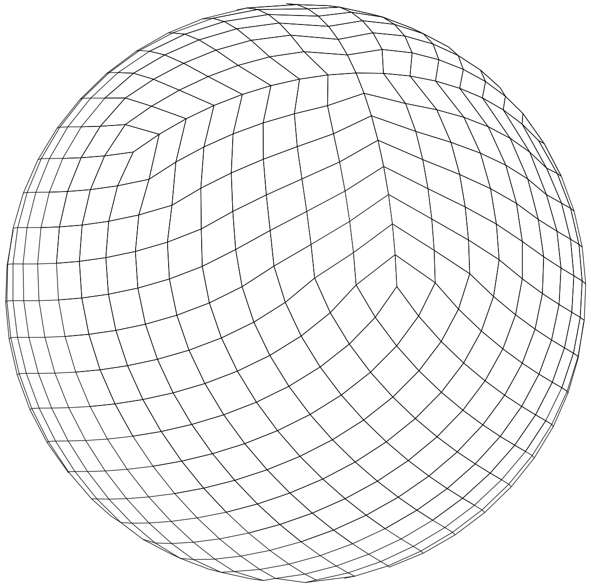


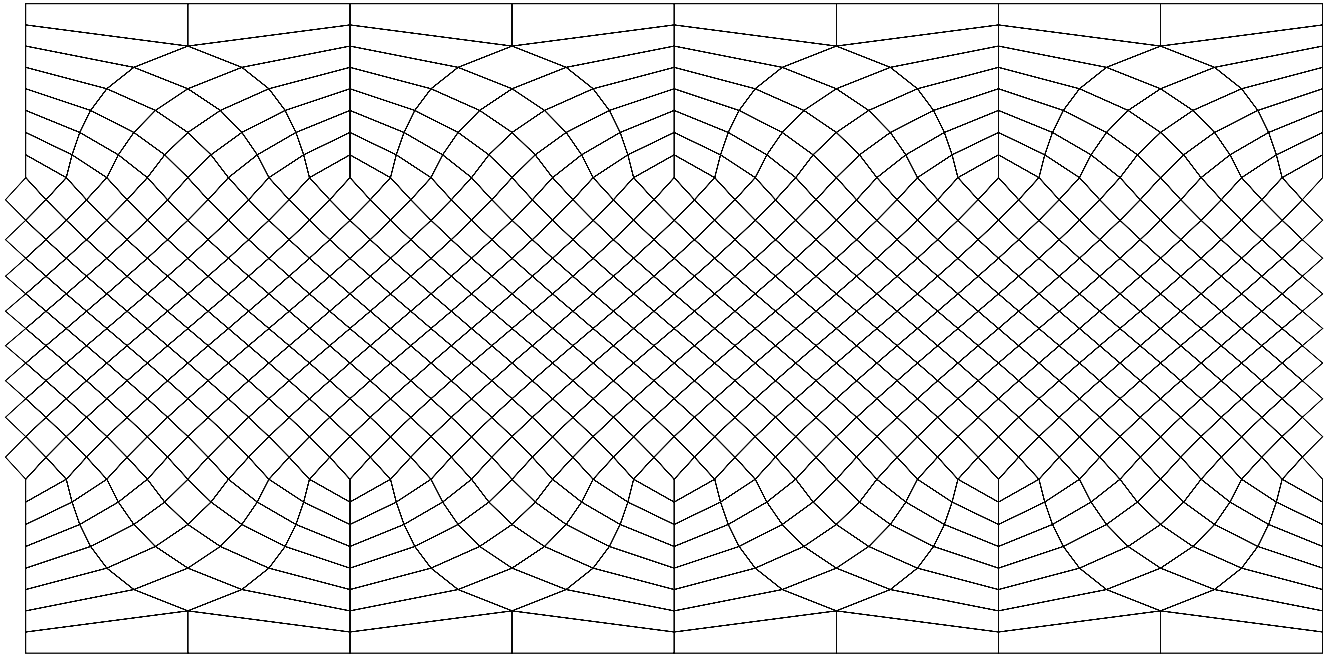
Most existing interpolation methods work

New Atlas support for HEALPix grids by Slavko Brdar (ECMWF)

Hierarchical Equal Area isoLatitude Pixelization of a sphere (https://healpix.jpl.nasa.gov)

- Grid points are centres of quadrilateral pixels
- Hierarchical: benefits for remapping between multiple resolutions
- All quadrilateral mesh (splitting 4 pentagons in half at each Pole in LonLat projection for visualization)
- iso-latitude: efficient spherical harmonics transforms possible (Spectral Model)

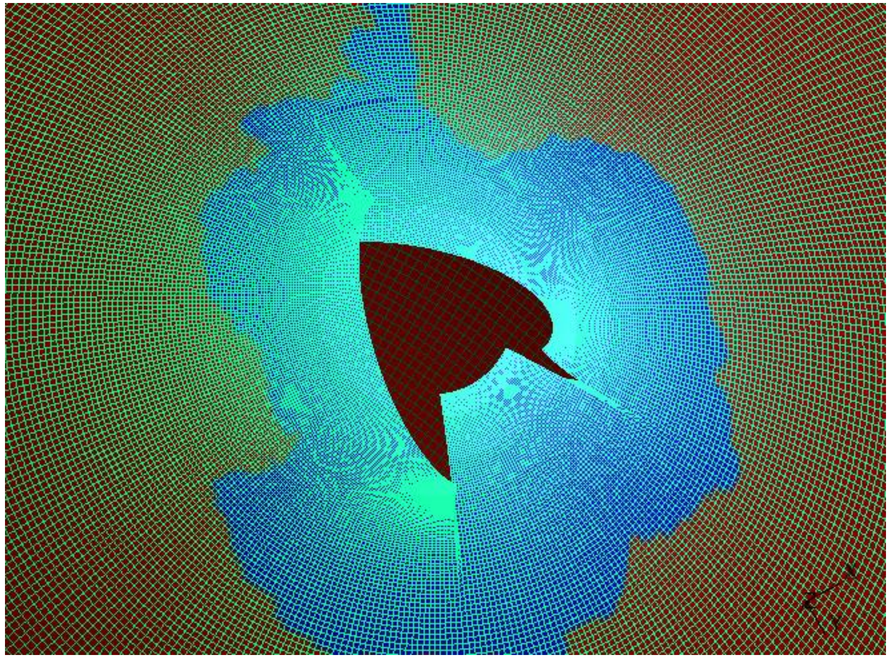




Intended use in Panther (p-adaptive Discontinuous Galerkin dynamical core), requiring all-quad mesh.

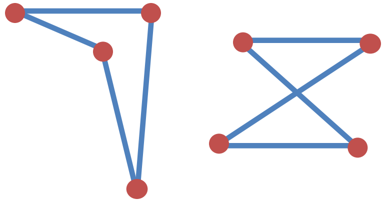
New Atlas support for ORCA grids

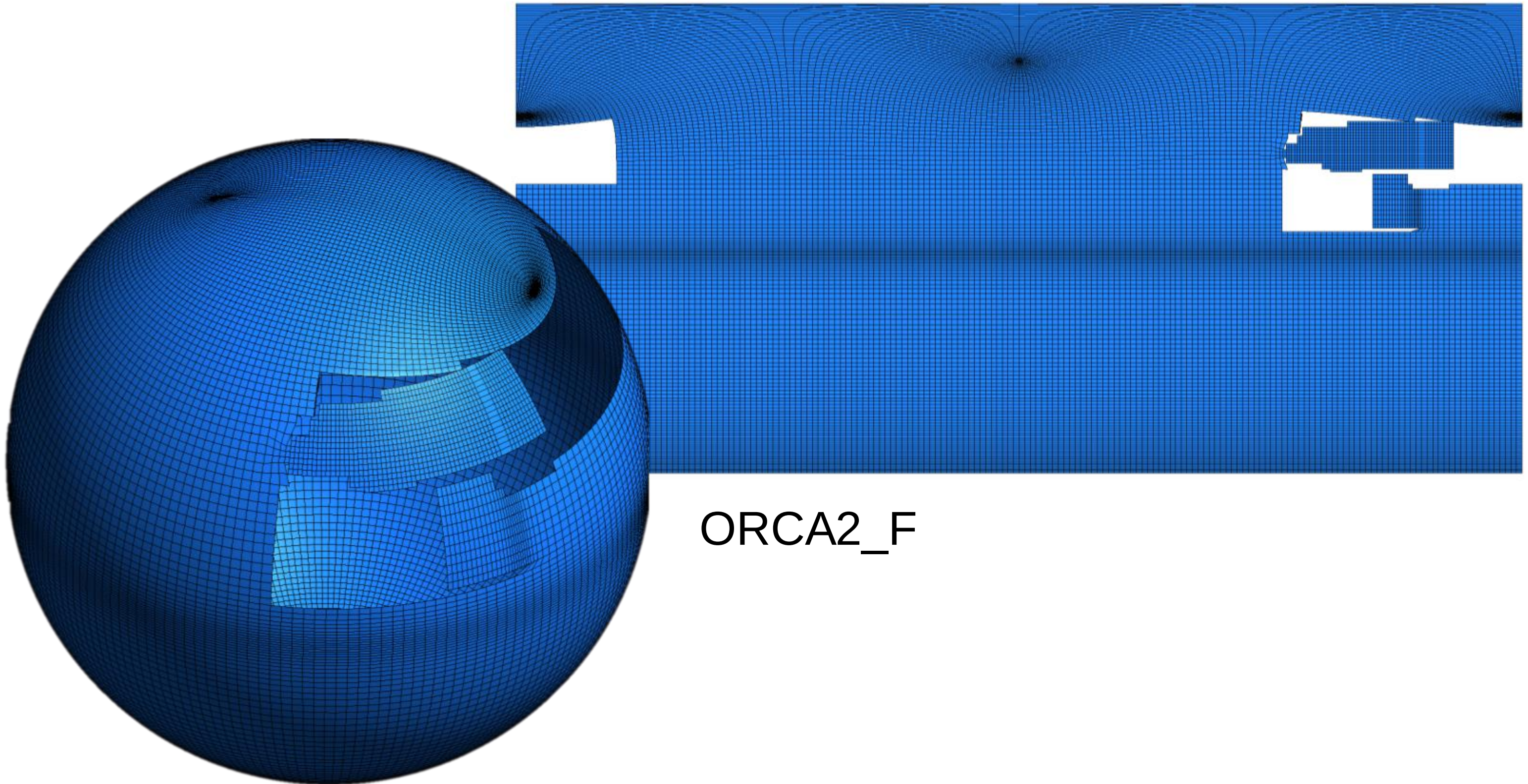
- Tripolar ocean grid used in NEMO ocean model
  - OrcaGrid
  - OrcaMeshGenerator
- Available grids: [e]ORCA<resolution>_<arrangement>
  - resolution: “2”, “1”, “025” (=0.25), “12” (=1/12)
  - arrangement: T,F,U,V,W
- Grid specific data curated and stored in “Atlas-IO” binary format
- MultIO / MARS / PGEN supports ORCA in GRIB via Atlas



Mesh around Antarctica self-intersects for extended grids.

Invalid elements need to be marked for mesh based operators (interpolation, …)





ORCA2_F

Atlas-IO binary format (first implementation, experimental use in atlas-orca)

Record format, like GRIB, but more flexible. Not a file format.

- Header section (version, endianness, length, offsets to Metadata and Data Index section)
- Metadata section (free format JSON/YAML string) which can describe arbitrary data structures with references to binary via following Data Index section.
- Data Index section: An index for following Data sections (offset, size, checksum)
- Multiple Data sections, containing binary compressed data described in Metadata section

What can be encoded?

- Basic types: (double, float, int, long, std::string) encoded as Base64-string within metadata
- Standard C++ types: std::vector, std::array
- Multidimensional arrays: metadata + data
- Arbitrary types if you write functions for encoding and decoding:

```
void encode_data(const MyType& in, atlas::io::Data& out) ;
size_t encode_metadata(const MyType& in, atlas::io::Metadata& out);
void decode(const atlas::io::Metadata&, const atlas::io::Data&, MyType& out);
```

Any compression algorithm available via eckit can be used: AEC, LZ4, Snappy, BZip2
and can be chosen independently for each data section.

Not to be used for archiving model output! There's GRIB for that.
Use for binary data internal to project, like restart files, or for serializing data and send over network

Atlas-IO binary format

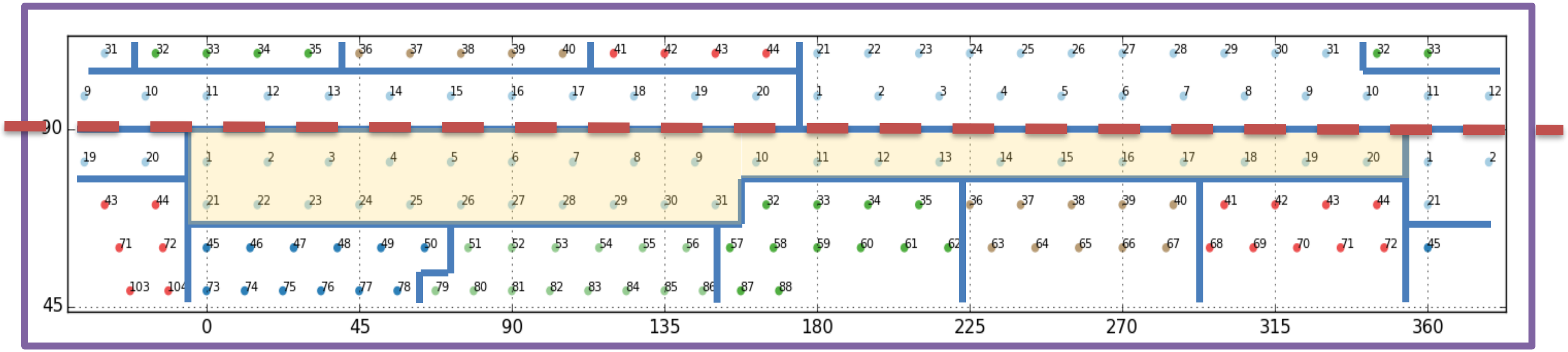
Example content of data file read by the ”OrcaGrid” class for a ORCA1_T grid.

```
>>> bin/atlas-io-list ORCA1_T.atlas --details

# ORCA1_T.atlas [0] { size: 401.375K, version: 0.2, created: 2021-04-27T17:39:00.434432887Z }

   name        type    description            ver.  comp.  size             checksum[:8]    endian  created
   ----------  ------  ----------------------  ----  -----  -----------------  --------------  ------  ----------------
   version     scalar  int32  : 0              0.2                                                      2021-04-27 17:39
   dimensions  array   int32  : {362,292}      0.2   lz4        9B <      8B   xxh64:247857c0  little  2021-04-27 17:39
   halo        array   int32  : {1,1,1,1}      0.2   lz4       13B <     16B   xxh64:53a66ab9  little  2021-04-27 17:39
   pivot       array   real64 : {180.5,290.5}  0.2   lz4       16B <     16B   xxh64:aa44229a  little  2021-04-27 17:39
   longitude   array   real64 [362,292]        0.2   lz4   223.42K < 825.81K   xxh64:491105db  little  2021-04-27 17:39
   latitude    array   real64 [362,292]        0.2   lz4   165.29K < 825.81K   xxh64:b26ea38e  little  2021-04-27 17:39
   flags       array   byte   [362,292]        0.2   lz4     9.77K < 103.23K   xxh64:1a2ff0c8  little  2021-04-27 17:39
```

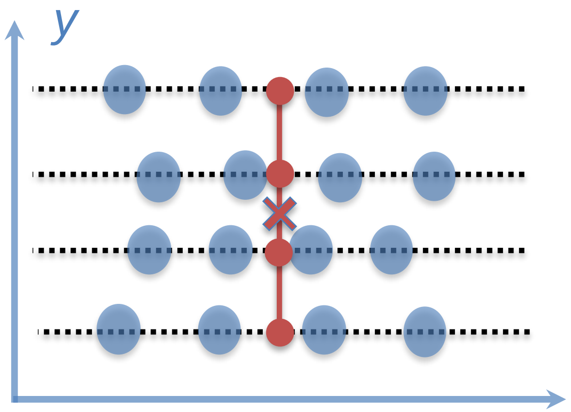
Data files are publicly downloadable on https://get.ecmwf.int/#browse/browse:atlas
(Atlas will download this for you if data is not present)

Recently developed interpolation methods

- UnstructuredBilinearLonLat – by Toby Searle (UK Met Office):
  Bilinear interpolation in lon-lat coordinates for unstructured quadrilateral meshes, as in NEMO
- CubedSphereBilinear – by Oliver Lomax, Marek Wlasak (UK MetOffice):
  Bilinear interpolation specific to cubed sphere source meshes, taking advantage of its structure
- Structured interpolation methods (not based on meshes, as used in IFS semi-Lagrangian method)
  linear, cubic, quasicubic, monotonicity limiter



StructuredColumns FunctionSpace



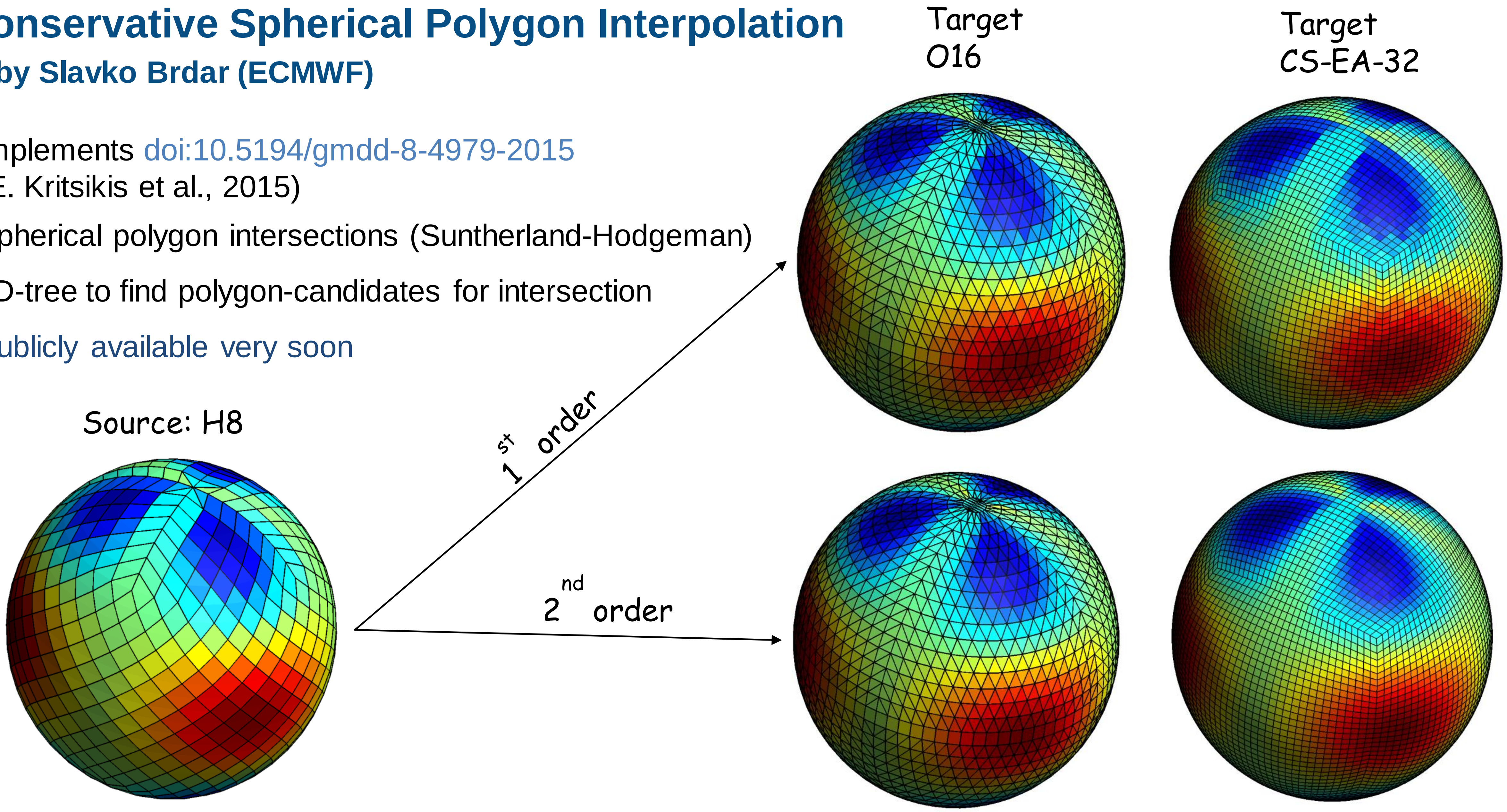
Cubic stencil

- GridBoxAverage – by Pedro Maciel (ECMWF):
  Conservative interpolation method between structured grids in lon-lat coordinates
- ConservativeSphericalPolygon – by Slavko Brdar (ECMWF)
  Conservative interpolation method up to second order for unstructured meshes

Conservative Spherical Polygon Interpolation

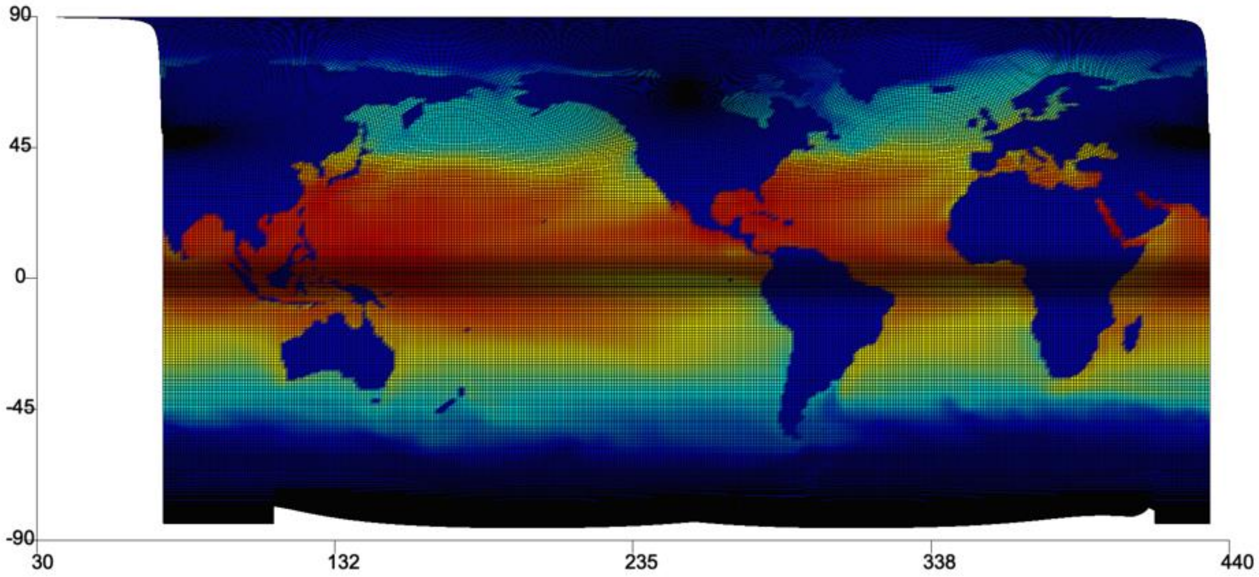
by Slavko Brdar (ECMWF)

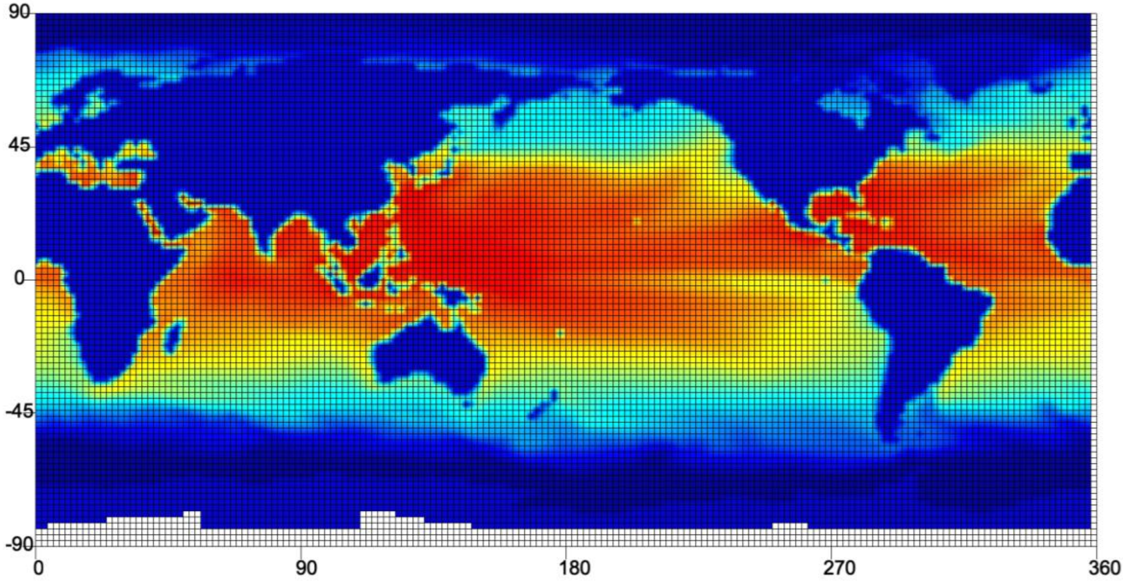
- Implements doi:10.5194/gmdd-8-4979-2015 (E. Kritsikis et al., 2015)
- Spherical polygon intersections (Suntherland-Hodgeman)
- kD-tree to find polygon-candidates for intersection
- Publicly available very soon



More on Interpolation

- Support for missing values – by Pedro Maciel (ECMWF)
- Adjoint interpolation and halo exchange – by Marek Wlasak (UK MetOffice)
- Ignore invalid elements (as present in ORCA grids)





Remapping of sea water potential temperature from the eORCA1_T grid (left) to a 2-degree regular longitude-latitude grid (right) using linear finite-element interpolation methods based on meshes generated by Atlas.

Spectral Transforms in Atlas

C++ example

```
auto grid = Grid{ “O1280” };

auto gp = StructuredColumns{ grid, levels(137) };
auto sp = Spectral{ 1279, levels(137) };

auto gpfield = gp.createField<double>();
auto spfield = sp.createField<double>();

auto trans = Trans{ gp, sp, type(“ifs”) };

trans.dirtrans( gpfield, spfield );
trans.invtrans( spfield, gpfield );
```

Grid (Octahedral Gaussian O1280)

FunctionSpaces gp and sp

Creation of fields
through FunctionSpace

Internally sets up IFS-trans

Transforms

Choice of different implementations (easily extensible, think e.g. GPU specific)

```
Trans::backend( “ifs” );
```

- MPI parallel
- invtrans to global Gaussian grids

```
Trans::backend( “local” )
```

- MPI serial
- invtrans to regional/global unstructured grids

ecTrans

Until very recently (March 2022), a special license for the IFS spectral transforms library was required:
ESCAPE license, RAPS license, member state

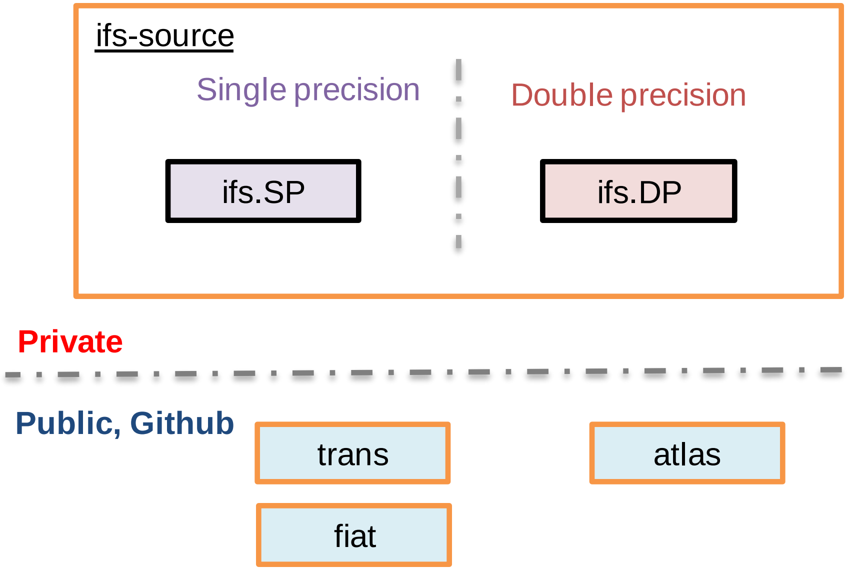
Happy to announce: **the IFS trans library is now open-source!!!**

https://github.com/ecmwf-ifs/ectrans

- Latest version available as will be used in upcoming IFS version CY48R1

- IFS will use ecTrans as a dependency
  - Branches / Implementations can be switched more easily

- GPU adaptation in branch under publicly visible development (Andreas Mueller, Nils Wedi, Sam Hatfield, Olivier Marsden)
- Atlas adapted to use ecTrans



- Foster new developments based on this very efficient spherical harmonics transforms library

Extending Atlas with custom plugins see https://sites.ecmwf.int/docs/atlas/design/plugin_architecture

Plugin architecture: example atlas-orca plugin library, dynamically loaded at runtime.

Self registration of OrcaGrid and OrcaMeshGenerator at runtime

→ It will show up in atlas tools that had no prior notion of ORCA, e.g. *atlas-grids* or *atlas-meshgen*

Automatic discovery and loading of plugins when you

- explicitly link plugin library into executable
- install plugin in same install prefix as atlas
- specify plugin names and search paths at run time

```
export ATLAS_PLUGINS=atlas-orca
export ATLAS_PLUGINS_SEARCH_PATHS=…
```

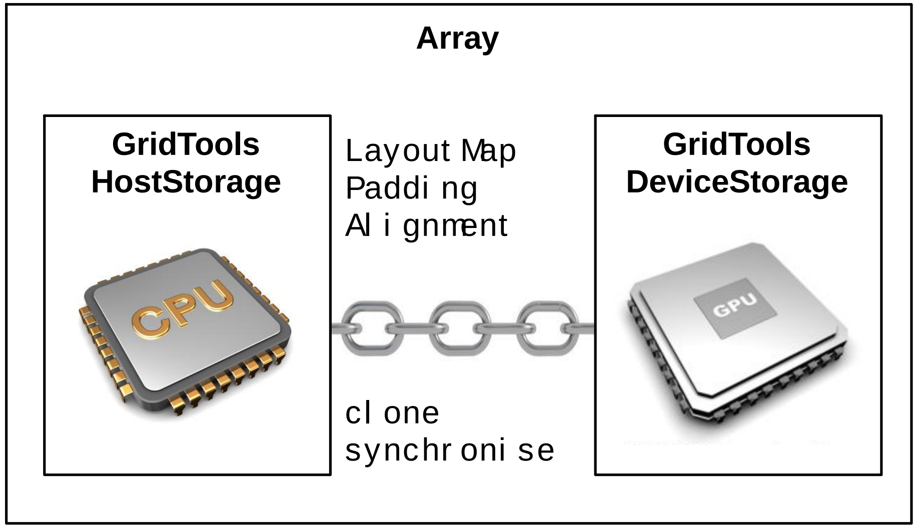
Software using Atlas abstractions for Grid / MeshGenerator / Partitioner / FunctionSpace / Interpolation / … has now automatically access to more concrete implementations

When to use a plugin?

- Extend existing software based on Atlas.
- Develop atlas functionality under separate version control. No need to have atlas branch that needs regular maintenance to stay up to date with latest release. Just update atlas.
- Model specific extensions not useful for community

Atlas on GPUs

- **Two linked memory spaces: host (CPU) and device (GPU)**
- **Built on GridTools storage layer**



Array

GridTools HostStorage

Layout Map
Padding
Alignment

clone
synchronise

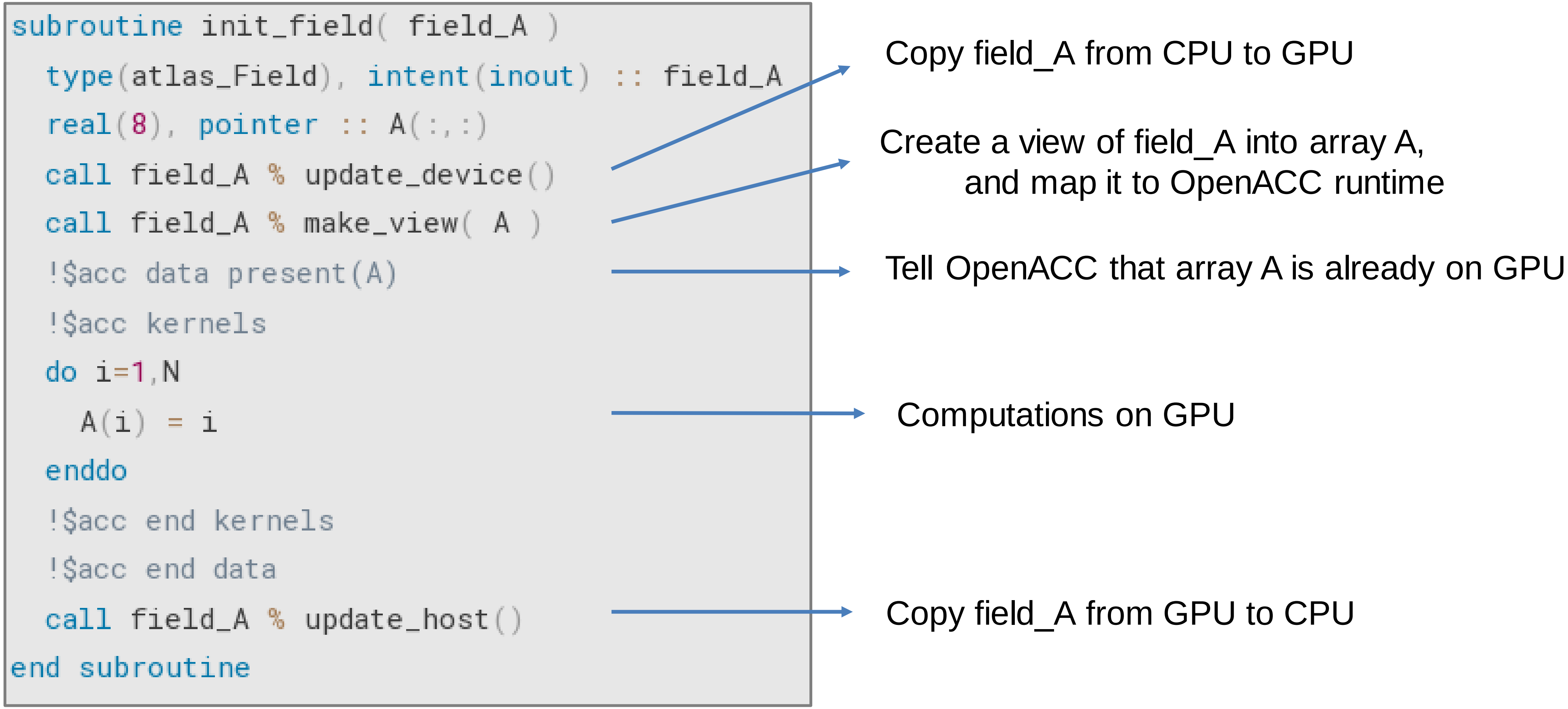
GridTools DeviceStorage

Thanks to Carlos Osuna (MeteoSwiss)

C++ example

```
// Create field (double precision, with 2 dimensions)
auto field = Field( datatype(“real64”), shape(Ni,Nj) );

// Create a host view to interpret as 2D Array of doubles
auto host_view = make_host_view<double,2>(field);

// Modify data on host
for ( int i=0; i<Ni; ++i ) {
  for ( int j=0; j<Nj; ++j ) {
    host_view(i,j) = ...
}}

// Synchronise memory-spaces
field.syncHostDevice();

// Create a device view to interpret as 2D Array of doubles
auto device_view = make_device_view<double,2>(field);

// Use e.g. CUDA to process the device view…
some_cuda_kernel<<<1,1>>>(device_view);

// ... or GridTools or Kokkos
```

Atlas on GPUS with OpenACC for Fortran



```
subroutine init_field( field_A )
  type(atlas_Field), intent(inout) :: field_A
  real(8), pointer :: A(:,:)
  call field_A % update_device()
  call field_A % make_view( A )
  !$acc data present(A)
  !$acc kernels
  do i=1,N
    A(i) = i
  enddo
  !$acc end kernels
  !$acc end data
  call field_A % update_host()
end subroutine
```

Copy field_A from CPU to GPU

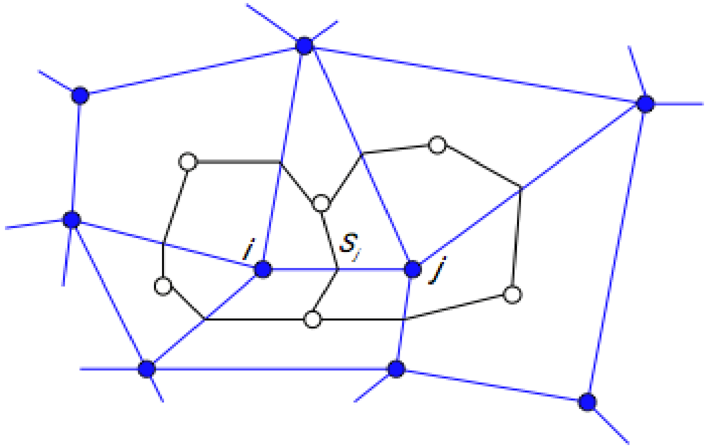
Create a view of field_A into array A,
and map it to OpenACC runtime

Tell OpenACC that array A is already on GPU

Computations on GPU

Copy field_A from GPU to CPU

Atlas is to be used to bridge gap between Application and DSL toolchain

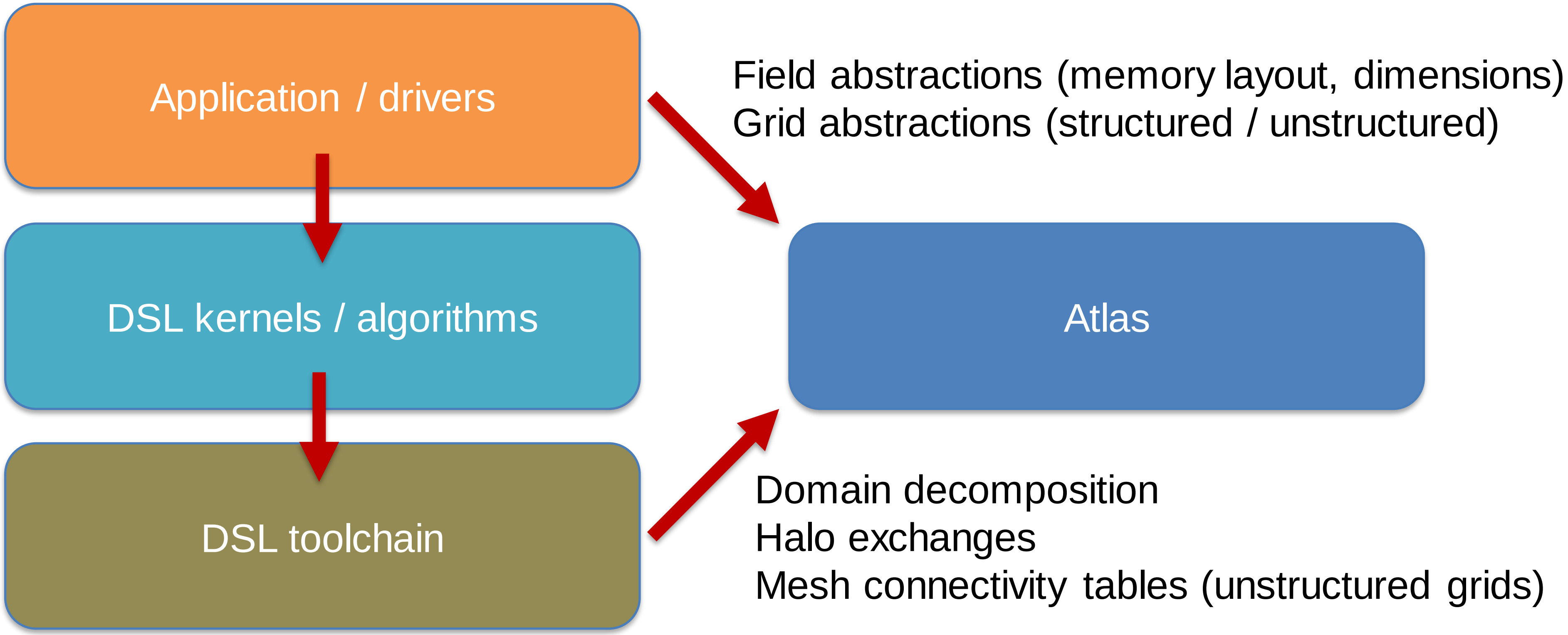


$$\nabla_i \cdot \mathbf{A} = \frac{1}{\mathcal{V}_i}\sum_{j=1}^{l(i)} A_j^{\perp} S_j$$

Divergence operator

Driver (Fortran)

```
TYPE( atlas_Field ) :: A_field, divA_field
...
CALL DIVERGENCE_SETUP( atlas_Grid( “O1280” ) )
...
CALL DIVERGENCE_OPERATOR( A_field, divA_field )
```



Application / drivers

DSL kernels / algorithms

DSL toolchain

Atlas

Field abstractions (memory layout, dimensions)
Grid abstractions (structured / unstructured)

Domain decomposition
Halo exchanges
Mesh connectivity tables (unstructured grids)

Future developments

Using Atlas as a backend in the IFS Field API (Michael Lange)

- MultiField concept: a field wraps a non-contiguous slice of a larger multidimensional array (1:NPROMA,1:NLEV, FIELD_INDEX, 1:NBLK)
- Provide GPU synchronization support for this:
  GridTools storage may not be able to handle the non-contiguous slice

Using Atlas remapping to couple ESM components

- Ocean model, wave model

Cornerstone for many of ECMWF Destination Earth developments to come.

- Vehicle to couple IFS with external plugins

Recent Atlas developments for Earth System Modelling

7th ENES HPC Workshop
Monday, May 9 2022 - Wednesday, May 11 2022
Barcelona Supercomputing Center (BSC), Spain

Willem Deconinck
with presented contributions by Slavko Brdar, Pedro Maciel (ECMWF),
Oliver Lomax, Marek Wlasak, Toby Searle (UK Met Office)

ECMWF
willem.Deconinck@ecmwf.int

Thank you!
Questions?

ECMWF

esiwace
CENTRE OF EXCELLENCE IN SIMULATION OF WEATHER AND CLIMATE IN EUROPE

ESiWACE2 is funded through the European Union's Horizon 2020 research and innovation programme under the grant agreement No. - 823988.