Interpolation

Computer User Training Course 2017

Paul Dando

User Support

advisory@ecmwf.int

ECMWF

Contents

- Introduction
- Overview of Interpolation
- Spectral Transformations
- Grid point Transformations
- Interpolation Options
- Future plans
- Practical

Introduction

- Weather data can have different representations
- Interpolation is how we recalculate data in a different representation
- Interpolation is available in
  - MARS
  - Operational dissemination
  - Metview graphics package

Documentation:

https://software.ecmwf.int/emoslib

Introduction - Interpolation “black box”

INPUT FIELD

GRIB Product

Data array

→

INTERPOLATION

- Transformations
  - Spectral → Spectral
  - Spectral → Grid-point
  - Grid-point → Grid-point
- Change resolution
- Sub-area extractions
- Derived fields
  - e.g. U and V from vorticity and divergence
- Rotated grids

OUTPUT FIELD

GRIB Product

Data array

→

Introduction – Interpolation black box (2)

- Input can be a GRIB product or value array
- Output can be a GRIB product or value array
- For GRIB products, characteristics / info read from the GRIB header

- A number of Fortran routines (part of EMOSLIB) perform the interpolation
- MARS (and Metview) calls these for you

- Possible to make calls to these functions yourself
- Example programs on internet pages for EMOSLIB

Spectral Transformations

- Some data (e.g. pressure and some model level) is stored in Spectral format
- These fields cannot be plotted directly
  - Need to be transformed to grid points

$$\zeta(\lambda,\mu,\eta,t) = \sum_{m=-T}^{T} \sum_{n=|m|}^{T} \zeta_n^m(\eta,t)\, Y_n^m(\lambda,\mu)$$

Grid point — Spherical harmonic coefficients

Spectral to grid-point

- Latitude/Longitude
- Regular and Reduced Gaussian
- Automatic truncation based on output grid resolution
- Interpolation coefficient files created (in `$PPDIR`)

Spectral to Spectral

- With truncation

Spectral to grid-point: truncation

- *Automatic* truncation before interpolation reduces resources needed and avoids spurious “aliased” values

Grid increment	Truncation
2.5 ≤ Δx	T63
1.5 ≤ Δx < 2.5	T106
0.6 ≤ Δx < 1.5	T213
0.4 ≤ Δx < 0.6	T319
0.3 ≤ Δx < 0.4	T511
0.15 ≤ Δx < 0.3	T799
0.09 ≤ Δx < 0.15	T1279
0.0 ≤ Δx < 0.09	T2047

- *MARS retrievals can override using resol keyword, e.g. resol=106*

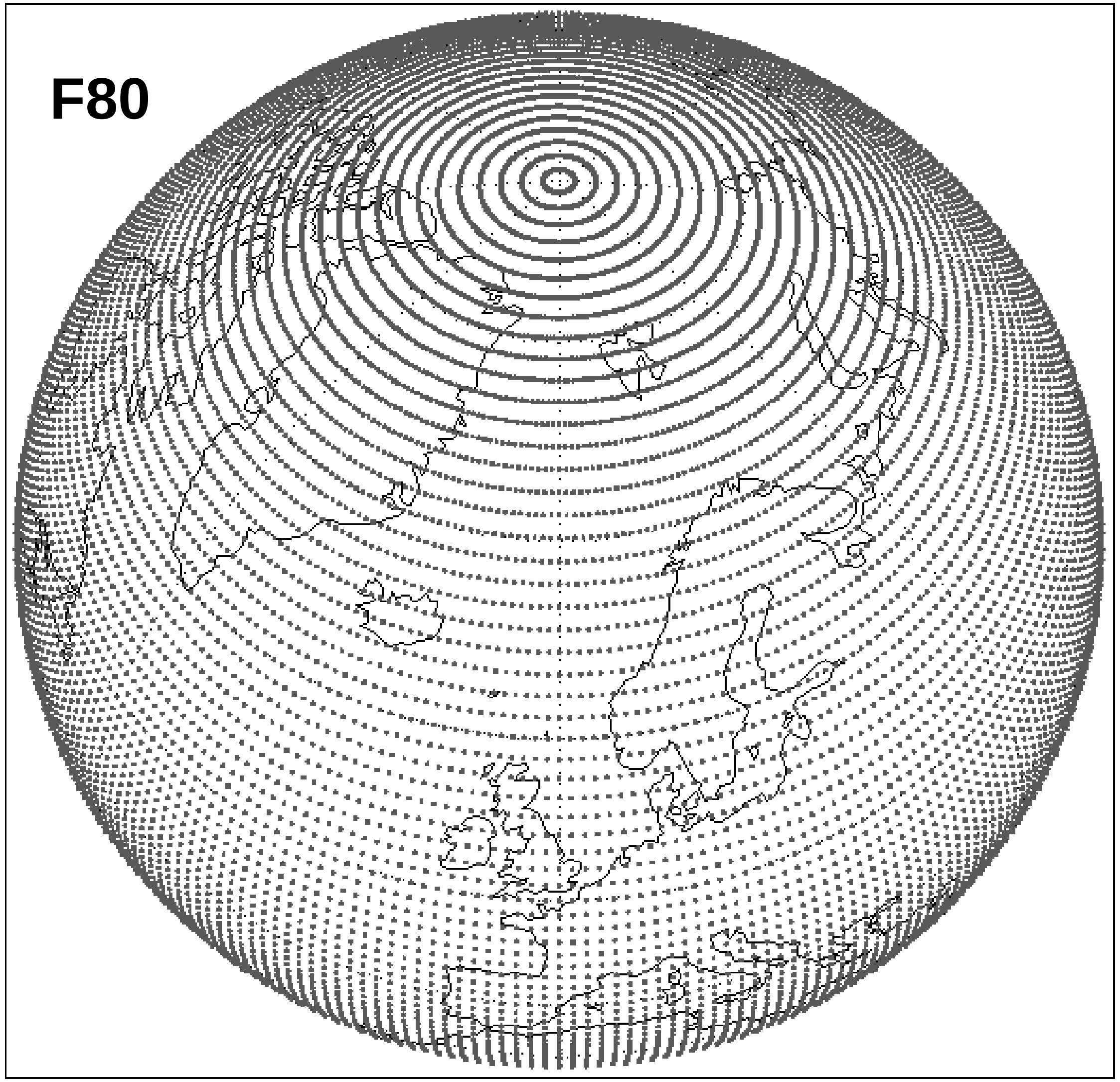
Grid-point Transformations

- Allowed combinations

TO →	Regular Lat /Lon	Regular Gaussian	Reduced Gaussian
Regular Lat /Lon	✓	✓	⊘
Regular Gaussian	✓	✓	⊘
Reduced Gaussian	✓	✓	✓

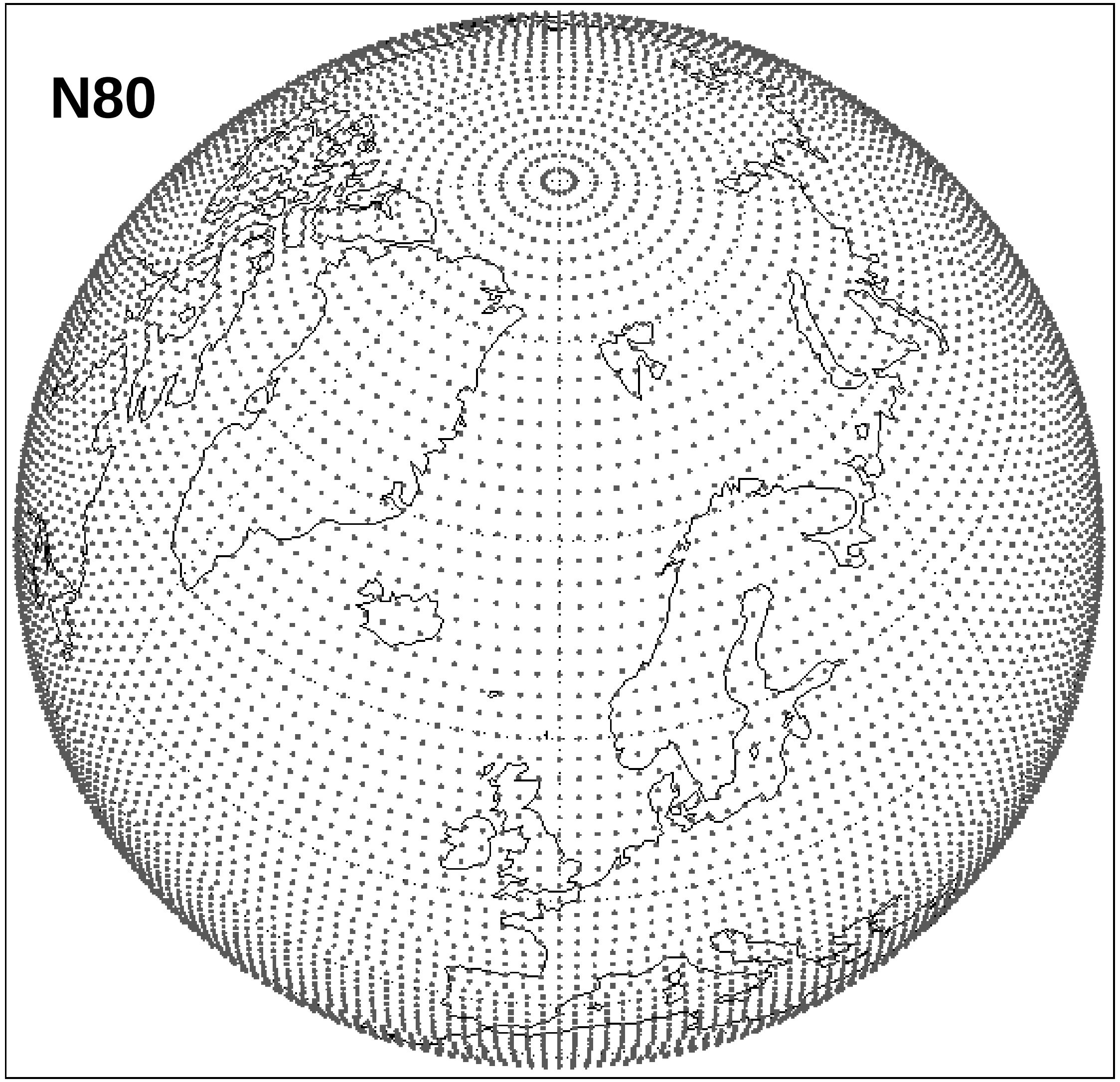
- NB cannot interpolate to a reduced Gaussian grid from a different representation

Regular (or Full) Gaussian grids



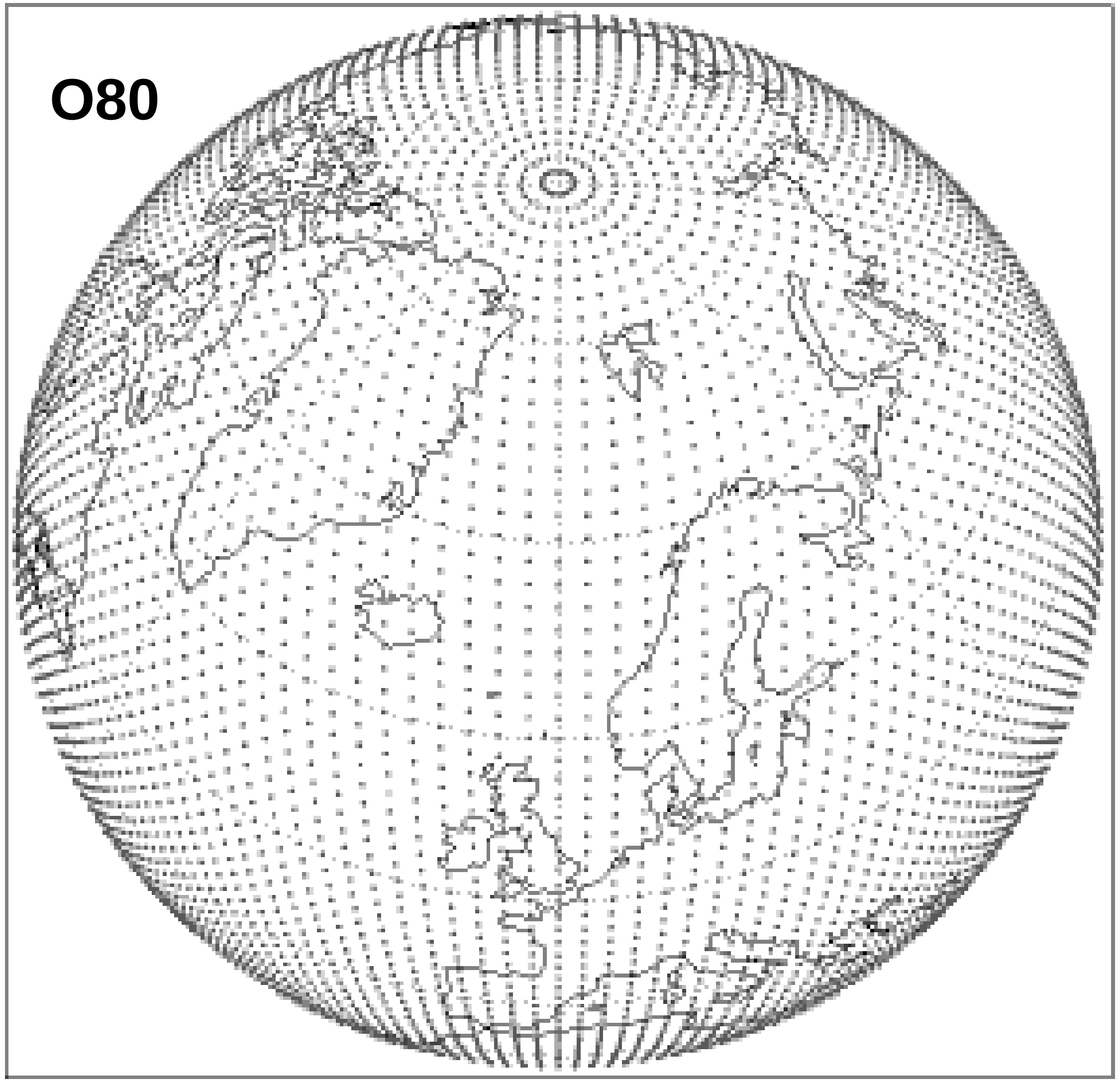
- N lines of latitude between pole and equator
- Latitude spacing not regular but is symmetric about equator
- 4 x N equally spaced points at each latitude
- No latitude points at poles or equator
- Special treatment at poles

Original reduced Gaussian grids



- Lines of latitude same as a regular Gaussian grid
- 4 x N points at the equator
- Fewer longitude points at latitudes close to poles
- Local east-west grid length similar for all latitudes
- Used up to March 2016

Octahedral reduced Gaussian grids



- Lines of latitude same as a regular Gaussian grid
- **20** longitude points at the latitude nearest the pole
- Increases by 4 points at each latitude line from pole towards the equator
- $4N + 16$ longitude points at latitude lines closest to equator
- Total number of points = $4N(N + 9)$
- Used since March 2016

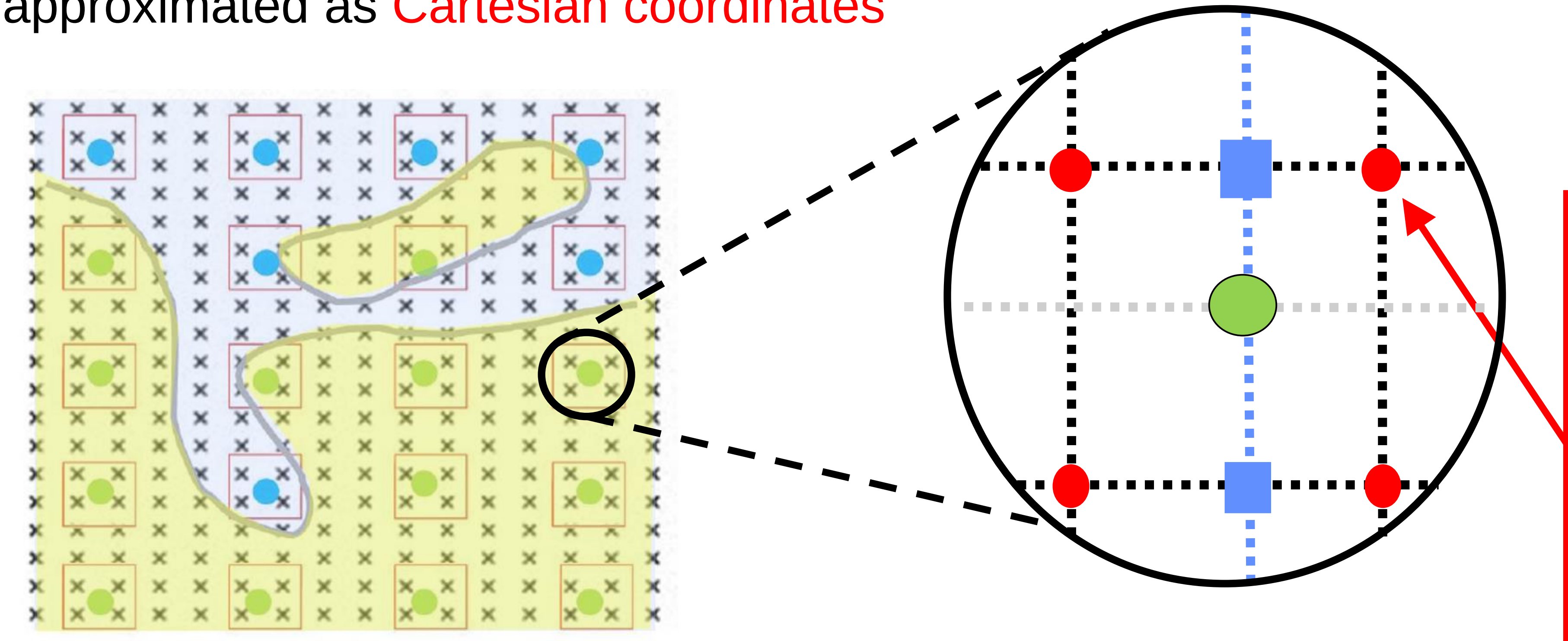
Interpolation Options

These apply only to Grid-point Interpolation

- Interpolation schemes
    - Bilinear
    - Nearest-neighbour
    - 12-point scheme for interpolation to rotated lat-long grids
- Treatment of
    - land-sea masks
    - precipitation
- Geographical sub-areas

Bilinear Interpolation

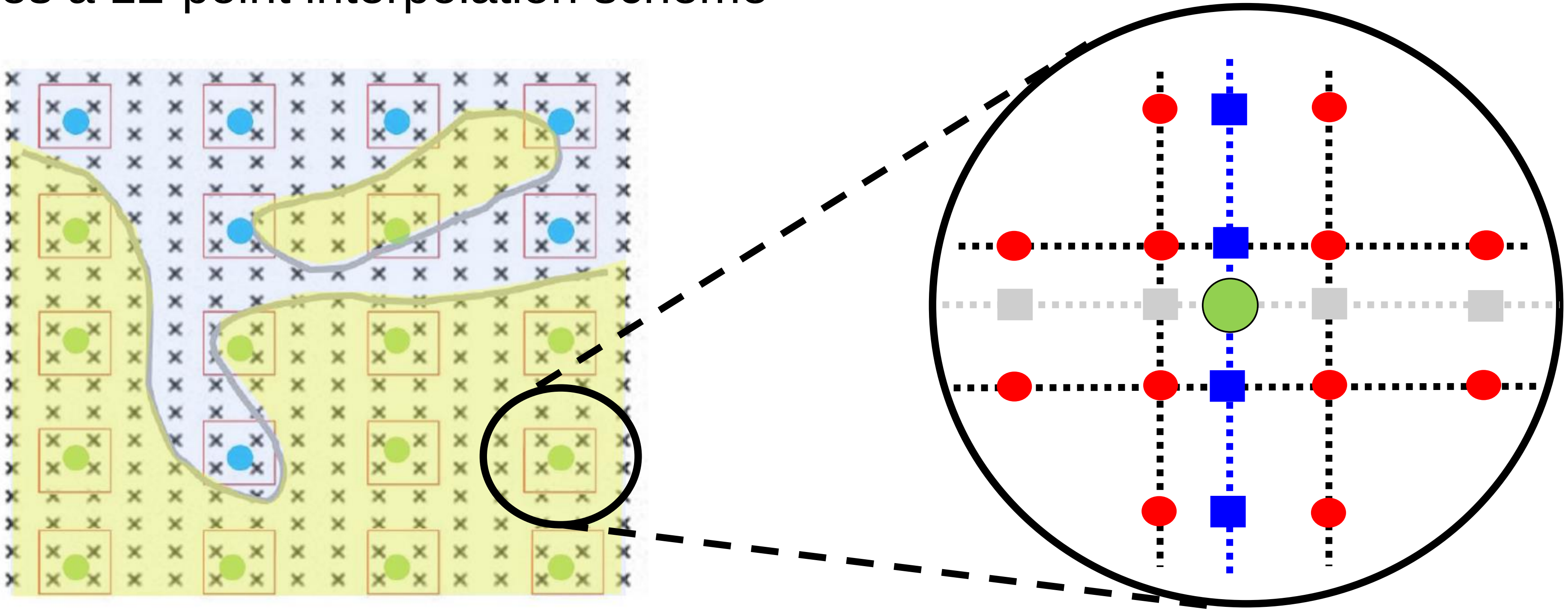
- Default for all parameters except vegetation, precipitation type and soil type fields and Wave 2D spectra
- Each point of output grid generated from 4 neighbouring points of input grid – approximated as Cartesian coordinates



Vegetation, soil type, precipitation type fields and Wave 2D spectra use nearest neighbour

Rotation from reduced Gaussian grids

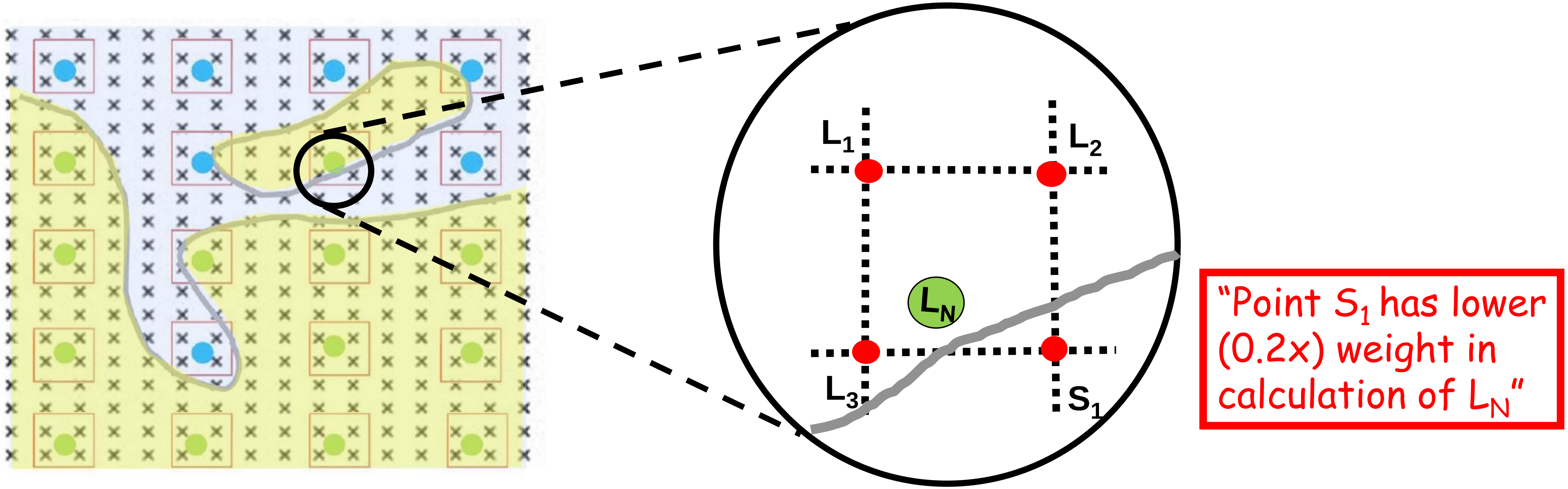
- Uses a 12-point interpolation scheme



- Spline fitting can produce non-physical values for some fields, e.g., cloud cover
  - Consider using bilinear interpolation for such fields
  - i.e. with MARS keyword interpolation = bilinear

Land-Sea Masks

- Land-sea masks represented as values 0 and 1 (or fractional)
- If land-sea mask of neighbouring point differs from grid-point being generated, weight of input point(s) is modified to reduce effect



- Land-sea masks are applied by default to surface fields (except MSL and LSM or interpolations to reduced Gaussian grids)

Precipitation – an "accumulated field"

- Rules are applied to prevent spreading of 'trace' amounts:
- Interpolated value for precipitation at a point is set to zero if:
  - the calculated value is less than a defined threshold
  - its neighbour with the highest weight had no precipitation
- Polar values for precipitation are always the average of nearest Gaussian line with no threshold check applied
- For ENS fields, accumulated fields can use "double" interpolation
  - E.g. Interpolate from O640 to O320 and then to lat-lon

Geographical Sub-areas

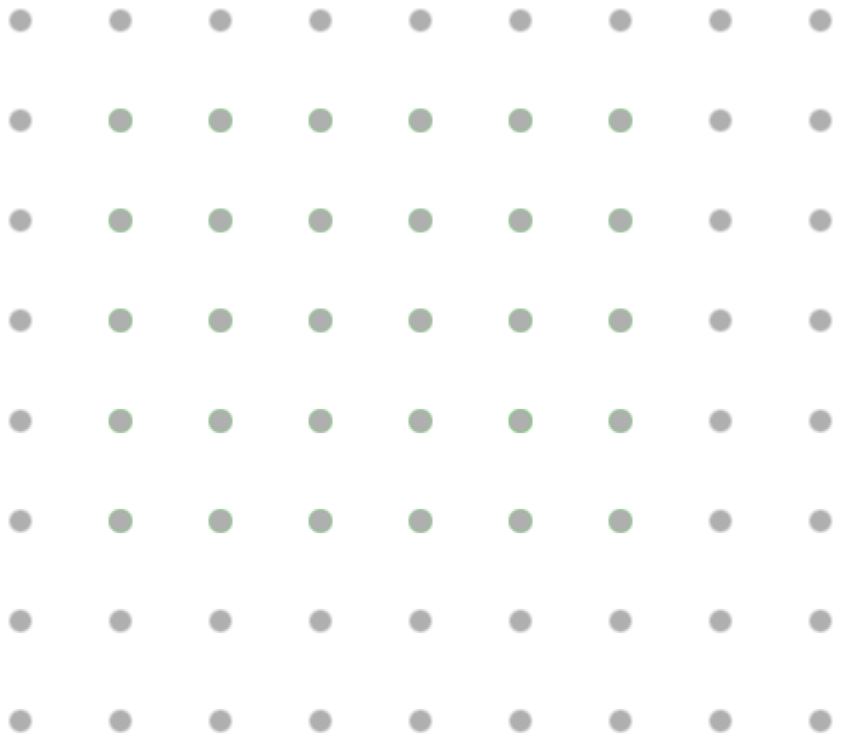
- Sub-areas can be created for new fields by specifying latitude / longitude boundaries (north / west / south / east)
- Sub-areas are based on the full global grid
  - Global regular grids have a line of longitude at the 0° meridian
  - Regular latitude-longitude grids have a line of latitude at the equator
  - Gaussian grids are symmetrical about the equator
- Boundaries of sub-areas are expanded outwards towards global grid (for rotations, boundaries are preserved)
  - Can change behaviour in MARS by setting the environment variable

    $MARS_INTERPOLATION_INWARDS
- Sub-areas not currently supported for reduced Gaussian grids – full global grid is produced for these

```
retrieve,
  type     = fc,
  param    = t,
  levelist = 1000/500,
  grid     = 1.5/1.5,
  area     = 75/-20/10/60,
  target   = "t_ll_eu.grb"
```

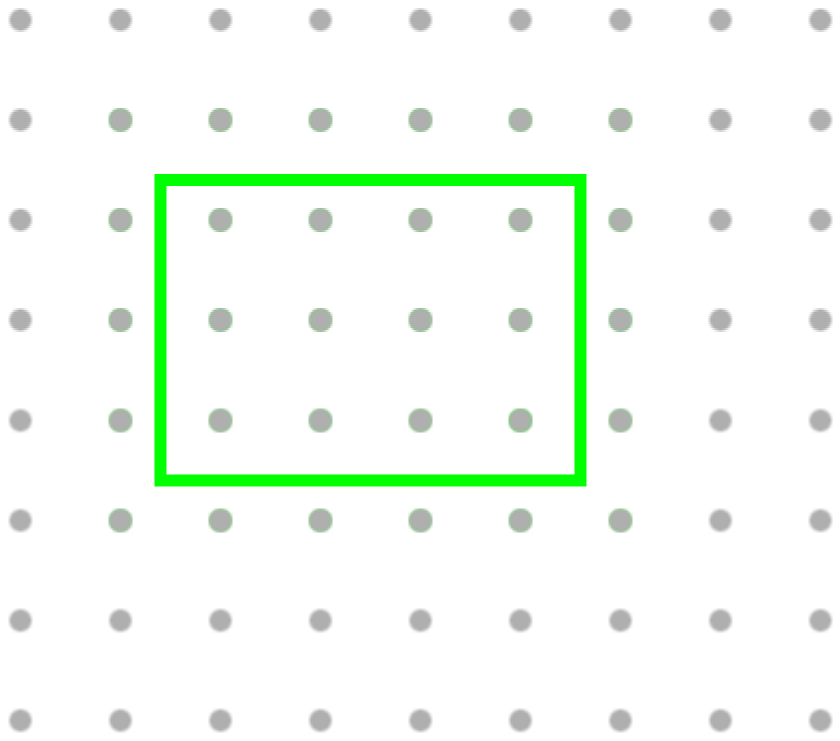
Geographical sub-areas – an example

- Adjustment of Sub areas
- Original (regular Lat / Lon) grid



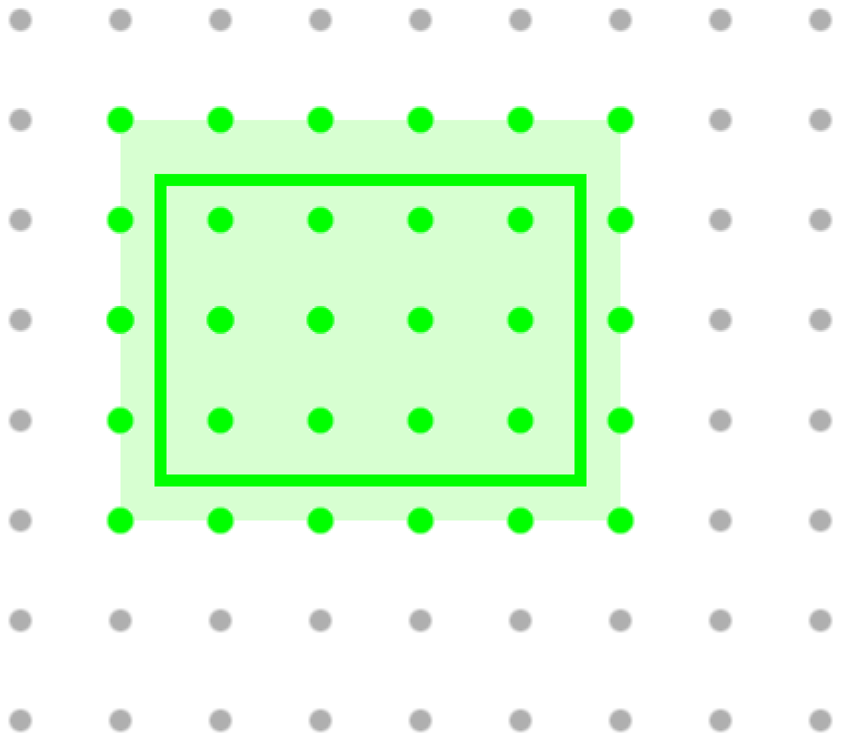
Geographical sub-areas – an example

- User requests a subarea
- In this case, their subarea falls between grid points



Geographical sub-areas – an example

- The subarea is widened
  - to encompass all points within and around the specified subarea
  - e.g. for 1x1 grid, NWSE (10.5, 2.5, -20.3, 84.2) becomes (11, 2, -21, 85)



Interfaces to the interpolation

- Fortran interface
  - Low level interface
  - Code needs to be complied and linked with Emoslib library
  - Special functions for GRIB2 (*intf2* & *intuvp2*)

  https://software.ecmwf.int/emoslib/Field+interpolation+software

- MARS/Metview interface
  - Recommended high level interface
  - Interpolation during data retrieval from archive
  - Options are described in MARS user guide
  - Same interface even if underlying interpolation package will change
  - This is what we use for the practical exercises…

  https://software.ecmwf.int/wiki/display/UDOC/Post-processing+keywords

```
retrieve,
  type     = fc,
  param    = t,
  levelist = 1000/500,
  grid     = 1.5/1.5,
  area     = 75/-20/10/60,
  target   = "t_ll_eu.grb"
```

Interpolation with MARS: the recipe book

- To a regular 1.5˚x1.5˚ lat-lon grid

```
retrieve,
   type     = fc,
   param    = t,
   levelist = 1000/500,
   grid     = 1.5/1.5,
   target   = "t_ll.grb"
```

- To an F640 regular Gaussian grid

```
retrieve,
   type     = fc,
   param    = t,
   levelist = 1000/500,
   grid     = F640,
   target   = "t_regular_gg.grb"
```

- To an O320 octahedral reduced Gaussian grid

```
retrieve,
   type     = fc,
   param    = t,
   levelist = 1000/500,
   grid     = O320,
   target   = "t_reduced_gg.grb"
```

- To a subarea of a 0.5˚x0.5˚ lat-lon grid

```
retrieve,
   type     = fc,
   param    = t,
   levelist = 1000/500,
   area     = 75/-20/10/60,
   grid     = 0.5/0.5,
   target   = "t_ll_eu.grb"
```

Interpolation with MARS: the recipe book

- To a subarea of a 0.5°x0.5° lat-lon grid with rotation

```
retrieve,
   type     = fc,
   param    = t,
   levelist = 1000/500,
   area     = 1/-17/-21/8,
   grid     = 0.5/0.5,
   rotation = -32.5/10.0,
   target   = "t_ll_rotated.grb"
```

- To a 0.125°x0.125° lat-lon grid using nearest-neighbour method

```
retrieve,
   type     = fc,
   param    = t,
   levelist = 1000/500,
   grid     = 0.125/0.125,
   interpolation = nearest neig,
   target   = "t_ll_nearest.grb
```

Future plans

- EMOSLIB is not easy to maintain
- A new interpolation package is being written in C++
  - MIR – Meteorological Interpolation and Regridding
  - Improve code, efficiency, maintainability and portability
- The new package will provide a Library and API
  - It will be callable from C, C++, Fortran 90, Python
  - It will include some Unix-style command line tools
- All current EMOSLIB features will be supported
- Some new features will be added
  - Include routines for ‘single-point’ interpolation
  - Handle different grid types
  - Parallelisation / multiple-threaded
- Undergoing extensive testing at ECMWF before release

Practical: Interpolation with MARS

- Work in your $SCRATCH

  `cd $SCRATCH`

- Copy the practicals directory to your $SCRATCH

  `tar -xvf /scratch/ectrain/trx/interpolation_practical.tar`

  - This will create a directory called `interpolation_practical` containing a number of scripts

- First, "`cd interpolation_practical`" and run interp1.ksh:

  `./interp1.ksh`

  - This will retrieve some data from MARS to a file out1.grib

- Next run the other scripts in turn.
  - Each will create a new file called out2.grib, … , out8.grib

- Inspect each output file with grib_ls and grib_dump
  - Note how the grid description in Section 2 of the header differs
  - Look at the MARS requests that create each of the files