# ecCodes: GRIB Keys

**Shahram Najm**
**Development Section**
**Forecast Department**

# GRIB keys

- Each key has a native type (real, integer, string) and conversions are provided from one type to another when possible
- The set of keys available changes from one message to another as it depends on the content of the message
- Changing the value of some keys can cause some other keys to disappear and new keys to be available
- The value of a key is not always coded in the GRIB message because it can be the result of the combination of several other keys through a given algorithm or just temporary (transient). Therefore we talk about
  - CODED keys ( coded in the message as they are )
  - COMPUTED keys ( temporary or computed from other keys )

# GRIB keys: Aliases

- A key can have several names (aliases)
- For example the key "Ni" has the alias "numberOfPointsAlongAParallel" which can be used in its place
- To see the aliases for a key, use grib_dump command with the "-a" switch:

```
> grib_dump -Oa my.grib
…
31-34   Ni = 16 [numberOfPointsAlongAParallel, Nx, numberOfColumns]
35-38   Nj = 31 [numberOfPointsAlongAMeridian, Ny, numberOfRows]
…
```

# GRIB keys: Namespaces

- A namespace is a name for a set of keys
- A key belonging to a namespace can be get/set by prefixing it with the namespace or simply without any prefix:

  time.step == step

  parameter.paramId == paramId

- Several namespaces are available e.g.
  - parameter
  - time
  - geography
  - vertical
  - statistics

# GRIB keys: Reference

- **GRIB1** http://apps.ecmwf.int/codes/grib/format/grib1/
- **GRIB2** http://apps.ecmwf.int/codes/grib/format/grib2/
- **Edition independent**
  http://apps.ecmwf.int/codes/grib/format/edition-independent/

# GRIB keys

- The easiest way to inspect a GRIB file is by using the tools
  - `grib_ls` to get a summary of the content
  - `grib_dump` to get a more detailed view
  - `grib_filter` to get a custom output format

- We will cover the tools later on in the course
- Most of what will follow will make more sense once you've had more hands-on experience! Bear with me ☺

# GRIB keys: file related

- **count**
  - Message number in a file
- **countTotal**
  - Message number in a set of files
- **offset**
  - Position in bytes of the start of a message in a file

# GRIB keys: data values

- **values**
  - array of all the data values and missing values
- **numberOfCodedValues**
  - number of values in the data section (missing values excluded)
- **numberOfPoints**
  - number of grid points = size of the values array
- **numberOfMissing**
  - number of missing values
- **max, min, average**
  - maximum, minimum and average of the field

**(We will cover missing data values later)**

# GRIB keys: time

- Start of the forecast run
  - dataDate YYYYMMDD (20070212)
  - dataTime (0000, 0600, 1200,…)
- Forecast step
  - stepType (instant, accum, avg, max, min, diff, rms, sd, cov, …)
  - stepUnits (s, m, h, 3h, 6h, 12h, D, M, Y, 10Y, 30Y, C)
  - startStep
  - endStep
  - stepRange ( “startStep-endStep” “endStep” )
  - step
- Validity of the forecast
  - validityDate
  - validityTime

# GRIB keys: geography

- **latitudes, longitudes**
  - array with all the latitudes/longitudes for each point of the grid
- **latLonValues**
  - array with all the latitudes/longitudes/values for each point of the grid
    - (lat1,lon1,value1,lat2,lon2,value2,…,latN,lonN,valueN)

# GRIB keys: geography

- Number of points:
  - Ni (Nx) along a parallel or x axis
  - Nj (Ny) along a meridian or y axis
- All latitude and longitude parameters are provided in a InDegrees version
  - longitudeOfFirstGridPoint -> longitudeOfFirstGridPointInDegrees
  - latitudeOfFirstGridPoint -> latitudeOfFirstGridPointInDegrees
  - longitudeOfLastGridPoint -> longitudeOfLastGridPointInDegrees
  - latitudeOfFirstGridPoint -> latitudeOfLastGridPointInDegrees
  - iDirectionIncrement -> iDirectionIncrementInDegrees
  - jDirectionIncrement -> jDirectionIncrementInDegrees

# GRIB keys: geography (gridType)

- For both editions:
  - -regular_ll
  - -reduced_ll
  - -mercator
  - -lambert
  - -polar_stereographic
  - -UTM
  - -simple_polyconic
  - -albers
  - -miller
  - -rotated_ll
  - -stretched_ll
  - -stretched_rotated_ll
  - -regular_gg
  - -rotated_gg
  - -stretched_gg
  - -stretched_rotated_gg
  - -reduced_gg
  - -sh
  - -rotated_sh
  - -stretched_sh
  - -stretched_rotated_sh
  - -space_view
- For edition 2 only:
  - -triangular_grid
  - -equatorial_azimuthal_equidistant
  - -azimuth_range
  - -cross_section
  - -Hovmoller
  - -time_section
  - -lambert_azimuthal_equal_area

# GRIB keys: geography

- For the following gridTypes a list of the latitudes and longitudes of the grid points can be obtained with grib_get_data (tools) and through a grib_iterator in C/F90/Python.
  - regular_ll (regular lat lon)
  - reduced_ll (reduced lat lon)
  - regular_gg (regular gaussian)
  - reduced_gg (reduced gaussian)
  - lambert (lambert conformal)
- The list of latitudes/longitudes can be obtained also if a bitmap is present.

# GRIB keys: parameter

- The definition of the parameter is very different in the two GRIB editions
- ecCodes provides some edition independent GRIB keys to identify a parameter :
  - paramId
  - shortName
  - name
  - units
  - centre

(Parameters will be covered in more depth later)

# GRIB keys: packingType

- For GRIB edition 1:
  - -grid_simple
  - -grid_simple_matrix
  - -grid_second_order
  - -spectral_complex
  - -spectral_simple
- For GRIB edition 2:
  - -grid_simple
  - -grid_simple_matrix
  - -grid_second_order
  - -spectral_simple
  - -spectral_complex
  - -grid_simple_log_preprocessing
  - -grid_jpeg
  - -grid_png
  - -grid_ieee

# Questions ?