GRIB decoding

Computer User Training Course 2017

Paul Dando

User Support

advisory@ecmwf.int

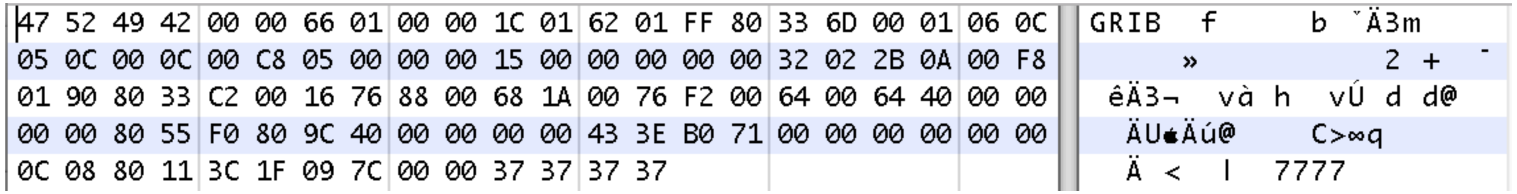
ECMWF

Contents

- GRIB description and overview
  - GRIB edition 1 and GRIB edition 2
  - Major differences between GRIB edition 1 and 2
  - Status of ECMWF migration to GRIB edition 2
- Using GRIB Tools
  - Introduction to ecCodes
  - Inspecting the content of GRIB messages
  - Decoding GRIB messages
  - Manipulation of GRIB messages
- Decoding GRIB messages with Fortran 90 … and Python

GRIB

- GRIB – “General Regularly-distributed Information in Binary form”
- Code defined by the WMO / CBS in 1985 for the exchange of large volumes of gridded data
- Machine independent
- Requires software for encoding and decoding



```
47 52 49 42 00 00 66 01 00 00 1C 01 62 01 FF 80 33 6D 00 01 06 0C   GRIB  f      b ˇÄ3m
05 0C 00 0C 00 C8 05 00 00 00 15 00 00 00 00 00 32 02 2B 0A 00 F8         »           2 +  ˉ
01 90 80 33 C2 00 16 76 88 00 68 1A 00 76 F2 00 64 00 64 40 00 00    êÄ3¬  và h  vÚ d d@
00 00 80 55 F0 80 9C 40 00 00 00 00 43 3E B0 71 00 00 00 00 00 00     ÄU●Äú@     C>∞q
0C 08 80 11 3C 1F 09 7C 00 00 37 37 37 37                             Ä <  |  7777
```

- Currently there are two different coding standards

  GRIB edition 1

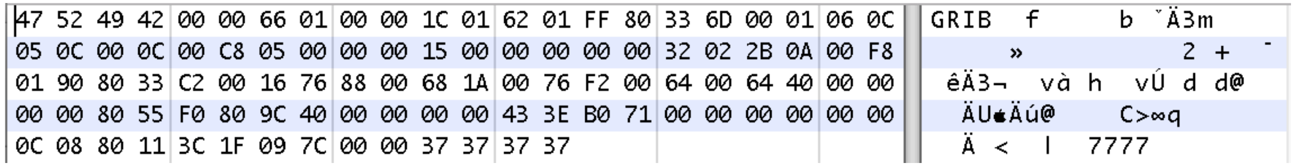
  - Currently used for ECMWF operational surface and pressure level data

  GRIB edition 2

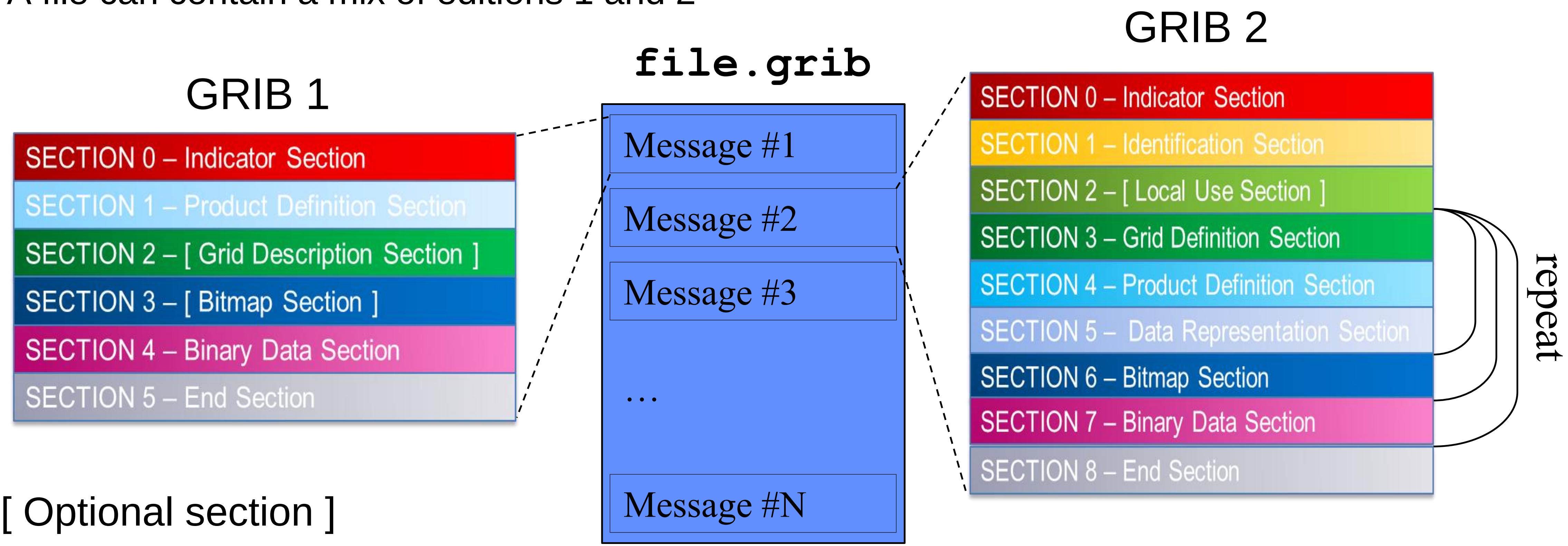
  - Used for ECMWF operational model level data since 18 May 2011
  - Some new surface parameters

GRIB Structure

- A file may contain one or more GRIB messages
- Each message contains several sections
- Data descriptors are self-defining
- A file can contain a mix of editions 1 and 2



```
47 52 49 42 00 00 66 01 00 00 1C 01 62 01 FF 80 33 6D 00 01 06 0C   GRIB  f      b ˇÄ3m
05 0C 00 0C 00 C8 05 00 00 00 15 00 00 00 00 00 32 02 2B 0A 00 F8        »          2 +  ˉ
01 90 80 33 C2 00 16 76 88 00 68 1A 00 76 F2 00 64 00 64 40 00 00    êÄ3¬  và h  vÚ d d@
00 00 80 55 F0 80 9C 40 00 00 00 00 43 3E B0 71 00 00 00 00 00 00     ÄUÄú@     C>∞q
0C 08 80 11 3C 1F 09 7C 00 00 37 37 37 37                             Ä <  |  7777
```



GRIB 1
SECTION 0 – Indicator Section
SECTION 1 – Product Definition Section
SECTION 2 – [Grid Description Section]
SECTION 3 – [Bitmap Section]
SECTION 4 – Binary Data Section
SECTION 5 – End Section
file.grib
Message #1
Message #2
Message #3
…
Message #N
GRIB 2
SECTION 0 – Indicator Section
SECTION 1 – Identification Section
SECTION 2 – [Local Use Section]
SECTION 3 – Grid Definition Section
SECTION 4 – Product Definition Section
SECTION 5 – Data Representation Section
SECTION 6 – Bitmap Section
SECTION 7 – Binary Data Section
SECTION 8 – End Section
repeat

[Optional section]

GRIB 1 & GRIB 2 – Major differences

- The coding principles for GRIB edition 1 and 2 are similar but their implementation is very different
- The structure of GRIB 1 and GRIB 2 messages is different
  - Both have sections but with different meanings
- In GRIB 2 several variables are defined with more precision
  - In GRIB 1 latitudes and longitudes are in milli-degrees
  - In GRIB 2 latitudes and longitudes are in micro-degrees
- In GRIB 2 longitude values must lie between 0˚ and 360˚
- Encoding of the parameter is very different
- In GRIB 2 the description of the data (parameter, time, statistics, grid…) is template / table based
  - More flexible … but also more complex !

Use of GRIB 2 at ECMWF

What is currently affected ?

- Since 18 May 2011 all model level fields for HRES and ENS (including the monthly extension) are encoded in GRIB 2
  - GRIB 1 model level data are no longer produced or disseminated
- Most surface and all pressure level fields are encoded in GRIB 1
  - Some recently introduced surface fields are encoded in GRIB 2 (e.g. ptype)
- Staged migration of remaining GRIB 1 fields to GRIB 2 “will follow”

And what’s not ?

- The wave model
- The System-4 seasonal forecast model
- ERA-Interim

Example GRIB edition 1 message

```
=============   MESSAGE 1 ( length=4284072 )              =============
1-4   identifier = GRIB
5-7   totalLength = 4284072
8     editionNumber = 1
====================== SECTION_1 ( length=52, padding=0 ) ======================
1-3   section1Length = 52
4     table2Version = 128
5     centre = 98 [European Center for Medium-Range Weather Forecasts (grib1/0.table) ]
6     generatingProcessIdentifier = 141
7     gridDefinition = 255
8     section1Flags = 128 [10000000]
9     indicatorOfParameter = 129 [Geopotential  (m**2 s**-2) (grib1/2.98.128.table) ]
10    indicatorOfTypeOfLevel = 1 [Surface  (of the Earth, which includes sea surface)  (grib1/3.table) ]
11-12 level = 0
13    yearOfCentury = 16
14    month = 2
15    day = 24
16    hour = 0
17    minute = 0
18    unitOfTimeRange = 1 [Hour (grib1/4.table) ]
…
```

Example GRIB edition 2 message

```
#==============   MESSAGE 1 ( length=4284160 )             ==============
1-4       identifier = GRIB
5-6       reserved = MISSING
7         discipline = 0 [Meteorological products (grib2/tables/5/0.0.table) ]
8         editionNumber = 2
9-16      totalLength = 4284160
======================   SECTION_1 ( length=21, padding=0 )    ======================
1-4       section1Length = 21
5         numberOfSection = 1
6-7       centre = 98 [European Centre for Medium-Range Weather Forecasts (grib1/0.table) ]
8-9       subCentre = 0
10        tablesVersion = 5 [Version implemented on 4 November 2009 (grib2/tables/1.0.table) ]
11        localTablesVersion = 0 [Local tables not used  (grib2/tables/5/1.1.table) ]
12        significanceOfReferenceTime = 1 [Start of forecast (grib2/tables/5/1.2.table) ]
13-14     year = 2016
15        month = 2
16        day = 22
17        hour = 12
18        minute = 0
19        second = 0
20        productionStatusOfProcessedData = 0 [Operational products (grib2/tables/5/1.3.table) ]
21        typeOfProcessedData = 1 [Forecast products (grib2/tables/5/1.4.table) ]
======================   SECTION_2 ( length=17, padding=0 )    ======================
1-4       section2Length = 17
…
```

Introducing ecCodes

- The ecCodes is a package developed by ECMWF for encoding and decoding of GRIB (and BUFR) data
- The library includes:
  - an Application Programming Interface
  - a set of command line tools (the GRIB Tools) to provide a quick and easy way to manipulate data
  - Fortran 90, C and Python interfaces which give access to the main features of the library
- It provides the user with a higher level of access, hiding the binary layer of the message
- It provides an easy and reliable way of encoding and decoding both GRIB 1 and GRIB 2 messages
- It decodes / encodes both GRIB editions with the SAME function calls

ecCodes approach

- ecCodes uses a key / value approach to access the information in a GRIB message

  numberOfPointsAlongAParallel ➔ Number of points along a parallel

  numberOfPointsAlongAMeridan ➔ Number of points along a meridian

  …

- The set of keys available changes from one message to another depending on:
  - the GRIB edition
  - the content of the message

- Changing the values of some keys can cause some other keys to disappear and new keys to become available

- Aliases are available for some of the keys

  numberOfPointsAlongAParallel ➔ Ni or Nx or numberOfColumns

  numberOfPointsAlongAMeridan ➔ Nj or Ny or numberOfRows

Coded and computed keys

- The value of a key is not always coded in the GRIB message
- Some keys are combinations of several other keys and provided through a given algorithm or can be just temporary (transient)
- Coded keys
  - Linked directly to the octets of the GRIB message
  - Values obtained by decoding the octet e.g. indicatorOfParameter
- Computed keys
  - Obtained by combining other keys (coded or computed)
  - Provide a synthesis of information contained in the message giving access to complex attributes
  - Setting the value of a computed key sets all related keys in a cascade
    - e.g. setting typeOfGrid=regular_ll will set all the various keys in the Grid Definition Section for a regular lat-long grid
  - MARS keywords are available as computed keys

ecCodes keys – parameter

- The definition of the parameter is very different in the two editions

GRIB 1 keys	GRIB 2 keys
centre	discipline
table2Version	parameterCategory
indicatorOfParameter	parameterNumber
levelType	typeOfFirstFixedSurface
level	scaleFactorOfFirstFixedSurface
…	scaledValueOfFirstFixedSurface
	typeOfSecondFixedSurface
	scaleFactorOfSecondFixedSurface
	scaledValueOfSecondFixedSurface
	productDefinitionTemplateNumber
	…

ecCodes keys – parameter

- ecCodes provides some edition-independent keys to identify a parameter

Key name	Example value
paramId	151
shortName	msl
centre	ecmf (or 98)
name	Mean sea level pressure
unit	Pa

- This set of keys is the parameter *namespace*
- There are several different namespaces
  - ‘parameter’, ‘time’, ‘geography’, ‘vertical’, ‘statistics’, ‘mars’

ecCodes keys – time

- Start of forecast run

Key name	Example values
dataDate	20160224 (YYYYMMDD)
dataTime	0, 600, 1200, 1800

- Forecast Step

Key name	Example values
stepType	instant, accum, avg, max, min, …
stepUnits	s, m, h, 3h, 6h, 12h, D, M, Y, 10Y, 30Y, C
startStep	0, 3, …
endStep (= step)	0, 3, ..
stepRange	3-6, 6 (“startStep-endStep” , “endStep”)

- Validity of the forecast

Key name	Example values
validityDate	20160224 (YYYYMMDD)
validityTime	0, 300, 1200, 1800

ecCodes keys – vertical and geography

- Vertical namespace

Key name	Example values
typeOfLevel	hybrid, surface, depthBelowLandLayer, isobaricInhPa, …
level	0, 1, 137, 1000, 850, …

- Geography namespace

Key name	Example values
gridType	reduced_gg, regular_ll, sh, …
latitudeOfFirstGridPointInDegrees	90.0, 55.5, …
longitudeOfFirstGridPointInDegrees	0.0, 350.0, …
latitudeOfLastGridPointInDegrees	-90.0, 35.0, …
longitudeOfLastGridPointInDegrees	360.0, 50.0, …
iDirectionIncrementInDegrees	0.5, …
jDirectionIncrementInDegrees	0.5, …
N	640, 320, …
…	…

ecCodes keys – MARS

- There is a namespace consisting of all the MARS keywords

Key name	Example values
date	20160224 (YYYYMMDD)
time	0000, 0600, 1200, 1800
step	3, 6, 9, 12, …
class	od, …
stream	oper, enfo,…
expver	0001
type	an, fc, cf, pf, …
levtype	sfc, pl, ml
levelist	500, 850, …
param	151.128

ecCodes keys and parameters for GRIB – THE Reference

- Parameters in GRIB
  - GRIB Parameters Database - http://apps.ecmwf.int/codes/grib/param-db

- GRIB keys
  - GRIB Edition 1 - http://apps.ecmwf.int/codes/grib/format/grib1/
  - GRIB Edition 2 - http://apps.ecmwf.int/codes/grib/format/grib2/
  - GRIB Edition Independent - http://apps.ecmwf.int/codes/grib/format/edition-independent/

- Disclaimer

  The official copy of the FM-92 GRIB document from which the relevant information contained in the following pages is copied can be obtained from the WMO web site: http://www.wmo.int/pages/prog/www/WMOCodes.html

GRIB Tools – basic concepts

- The easiest way to inspect a GRIB file and to find the keys available is to use the GRIB Tools

  - grib_ls   to get a summary of the content

  - grib_dump   to get a more detailed view

- The GRIB tools are part of the ECMWF ecCodes package

- They are a set of command line programs for interactive and batch decoding and most common processing of both GRIB 1 and GRIB 2 data

- Use of the tools avoids the need to write new code and thus speeds up your work

  - Consider using GRIB Tools instead of writing your own programs

- The tools have many command line options in common

  - the same options can be applied to different tools

- Use of the tools is recommended whenever possible !

GRIB Tools – more basics

- All of the tools use a common syntax

```
grib_<tool> [options] grib_file grib_file … [output_grib]
```

- There is tools for getting information about the ecCodes installation
  - codes_info
- There are tools to inspect the content of and compare GRIB messages
  - grib_ls, grib_dump, grib_get, grib_get_data, grib_compare
- There are tools for counting and copying some messages
  - grib_count, grib_copy
- There are tools for making changes to the content of a GRIB message and converting from GRIB to NetCDF
  - grib_set, grib_filter, grib_to_netcdf

GRIB Tools – getting help

- UNIX ‘man’-style pages are available for each tool by running the tool without any options or input file

```
> grib_dump
NAME    grib_dump

DESCRIPTION
        Dump the content of a grib file in different formats.

USAGE
        grib_dump [options] grib_file grib_file ...

OPTIONS
        -O      Octet mode. WMO documentation style dump.
        -D      Debug mode.
        -d      Print all data values.
…
```

grib_ls – list the content of GRIB files

- Use grib_ls to get a summary of the content of GRIB files
- Without options grib_ls prints a default list of keys
- Options exist to specify the set of keys to print or to print other keys in addition to the default set
- Output can be ordered
  - e.g. order by ascending or descending step
- grib_ls does not fail if a key is not found
- grib_ls can also be used to find the grid point(s) nearest to a specified latitude-longitude and print the value of the field at that point
  - Modes available to obtain one or four nearest grid points

grib_ls – usage

```
grib_ls [options] grib_file grib_file ...
```

Basic options

`-p key1,key2,…`	Keys to print
`-P key1,key2,…`	Additional keys to print
`-w key1=val1,key2!=val2…`	Where option
`-B "key asc, key desc"`	Order by: "step asc, centre desc"
`-n namespace`	Print keys for `namespace`
`-m`	Print MARS keys
`-i index`	Print data value at given index
`-l lat,lon[,MODE,FILE]`	Value(s) nearest to lat-lon point
`-F format`	Format for floating point values
`-W width`	Minimum column width (default 10)

grib_ls – examples

```
> grib_ls file.grib1
file.grib1
edition   centre   typeOfLevel  level    dataDate ... dataType  shortName packingType  gridType
1         ecmf     isobaricInhPa  1000   20170222 ...  an        t         spectral_complex  sh
1         ecmf     isobaricInhPa  500    20170222 ...  an        t         spectral_complex  sh
1         ecmf     isobaricInhPa  200    20170222 ...  an        t         spectral_complex  sh
1         ecmf     isobaricInhPa  100    20170222 ...  an        t         spectral_complex  sh
4 of 4 grib messages in file.grib1

4 of 4 total grib messages in 1 files
```

- Use **-p** option to specify a list of keys to be printed

```
> grib_ls -p centre,dataDate,shortName,paramId,typeOfLevel,level file.grib1
file.grib1
centre      dataDate    shortName   paramId         typeOfLevel      level
ecmf          20170222    t           130           isobaricInhPa    1000
ecmf          20170222    t           130           isobaricInhPa    500
ecmf          20170222    t           130           isobaricInhPa    200
ecmf          20170222    t           130           isobaricInhPa    100
4 of 4 grib messages in file.grib1

4 of 4 total grib messages in 1 files
```

grib_ls – examples

- When a key is not present in the GRIB file, it returns “not found” for this key

```
> grib_ls -p my_key  file.grib1
file.grib1
my_key
not_found

> echo $?
0
```

exit code returned = 0

- Similar behaviour to grib_get (see later)
  - grib_ls is more for interactive use
  - use grib_get within scripts

Using the ‘where’ option

- The where option **-w** can be used with all the GRIB Tools
- Constraints are of the form key=value or key!=value or key=value1/value2/value2

```
-w key1=value1,key2:i!=value2,key3:s=value3
```

- Messages are processed only if they match ALL the key / value constraints

```
> grib_ls -w level=100 file.grib1                        “IS”
…
> grib_ls -w level!=100 file.grib1                       “NOT”
…
> grib_ls -w level=100,stepRange=3 file.grib1            “AND”
…
> grib_ls -w level=100/200/300/500 file.grib1            “OR”
…
```

Specifying the type of the key

- All ecCodes keys have a default type
  - e.g. string, integer, floating point
- The type of the key can be specified as follows:
  - `key` → native type
  - `key:i` → integer
  - `key:s` → string
  - `key:d` → double

```
> grib_ls -p centre:i,dataDate,shortName,paramId,typeOfLevel,level file.grib1
file.grib1
centre      dataDate    shortName   paramId         typeOfLevel      level
98          20160222    t           130             isobaricInhPa    1000
98          20160222    t           130             isobaricInhPa    500
98          20160222    t           130             isobaricInhPa    200
98          20160222    t           130             isobaricInhPa    100
4 of 4 grib messages in file.grib1

4 of 4 total grib messages in 1 files
```

grib_dump – dump content of GRIB files

- Use grib_dump to get a detailed view of the content of a file containing one or more GRIB messages
- Various output formats are supported
  - Octet mode provides a WMO documentation style dump
  - Debug mode prints all keys available in the GRIB file
  - Octet and Debug modes cannot be used together
  - Octet content can also be printed in hexadecimal format
- Options also exist to print key aliases and key type information

grib_dump – usage

```
grib_dump [options] grib_file grib_file ...
```

Basic options

`-O`	Octet mode (WMO Documentation style)
`-D`	Debug mode
`-a`	Print key alias information
`-t`	Print key type information
`-H`	Print octet content in hexadecimal
`-w key{=/!=}value,…`	Where option
`-d`	Print all data values
`...`	

grib_dump – examples

```
> grib_dump file.grib1
***** FILE: file.grib1
#==============   MESSAGE 1 ( length=4284072 )            ==============
GRIB {
  editionNumber = 1;
  table2Version = 128;
  # European Center for Medium-Range Weather Forecasts (grib1/0.table)
  centre = 98;
  generatingProcessIdentifier = 141;
  # Geopotential  (m**2 s**-2)  (grib1/2.98.128.table)
  indicatorOfParameter = 129;
  # Surface  (of the Earth, which includes sea surface)  (grib1/3.table)
  indicatorOfTypeOfLevel = 1;
  level = 0;
  # Forecast product valid at reference time + P1  (P1>0)  (grib1/5.table)
  timeRangeIndicator = 0;
  # Unknown code table entry (grib1/0.ecmf.table)
  subCentre = 0;
  paramId = 129;
  #-READ ONLY- units = m**2 s**-2;
  #-READ ONLY- nameECMF = Geopotential;
  #-READ ONLY- name = Geopotential;
  decimalScaleFactor = 0;
  dataDate = 20170222;
  dataTime = 0; ...
```

Some keys are read only

keys are case sensitive:

dataDate, dataTime

grib_dump examples: WMO octet mode

```
> grib_dump –O file.grib1
```

```
***** FILE: file.grib1
==============   MESSAGE 1 ( length=4284072 )             ==============
1-4   identifier = GRIB
5-7   totalLength = 4284072
8     editionNumber = 1
====================== SECTION_1 ( length=52, padding=0 ) ======================
1-3   section1Length = 52
4     table2Version = 128
5     centre = 98 [European Center for Medium-Range Weather Forecasts (grib1/0.table) ]
6     generatingProcessIdentifier = 141
7     gridDefinition = 255
8     section1Flags = 128 [10000000]
9     indicatorOfParameter = 129 [Geopotential  (m**2 s**-2) (grib1/2.98.128.table) ]
10    indicatorOfTypeOfLevel = 1 [Surface  (of the Earth, which includes sea surface)  (grib1/3.table) ]
11-12 level = 0
13    yearOfCentury = 17
14    month = 2
15    day = 22
16    hour = 0
17    minute = 0
18    unitOfTimeRange = 1 [Hour (grib1/4.table) ]
…
```

grib_dump examples: Octet mode with types, aliases and Hex

```
> grib_dump -OtaH  file.grib1

***** FILE: file.grib1
==============   MESSAGE 1 ( length=4284072 )            ==============
1-4   ascii identifier = GRIB ( 0x47 0x52 0x49 0x42 )
5-7   g1_message_length totalLength = 4284072 ( 0x41 0x5E 0xA8 )
8     unsigned editionNumber = 1 ( 0x01 ) [ls.edition]
====================== SECTION_1 ( length=52, padding=0 ) ======================
1-3   section_length section1Length = 52 ( 0x00 0x00 0x34 )
4     unsigned table2Version = 128 ( 0x80 ) [gribTablesVersionNo]
5     codetable centre = 98 ( 0x62 ) [European Center for Medium-Range Weather Forecasts(grib1/0.table) ]
                     [identificationOfOriginatingGeneratingCentre,originatingCentre, ls.centre,
centreForTable2]
6     unsigned generatingProcessIdentifier = 141 ( 0x88 ) [generatingProcessIdentificationNumber, process]
7     unsigned gridDefinition = 255 ( 0xFF )
8     codeflag section1Flags = 128 [10000000] ( 0x80 )
9     codetable indicatorOfParameter = 129 ( 0x81 ) [Geopotential (m**2 s**-2)(grib1/2.98.128.table) ]
10    codetable indicatorOfTypeOfLevel = 1 ( 0x01 ) [Surface  (of the Earth,which includes sea surface)
   (grib1/3.table) ] [levelType, mars.levtype]
11-12 unsigned level = 0 ( 0x00 0x00 ) [vertical.topLevel vertical.bottomLevel, ls.level, lev]
13    unsigned yearOfCentury = 17 ( 0x10 )
14    unsigned month = 2 ( 0x02 )
15    unsigned day = 22 ( 0x18 )
16    unsigned hour = 0 ( 0x00 )
…
```

Practicals

- Work in your $SCRATCH

```
cd $SCRATCH
```

- Make a copy of the practicals directory in your $SCRATCH

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

- This will create a directory in your $SCRATCH containing the GRIB data files for all today's practicals
- There are sub-directories for each practical:

```
ls $SCRATCH/grib_practicals
practical1 practical2 practical3
practical4 practical5 practical6
```

Practical 1: using grib_ls and grib_dump

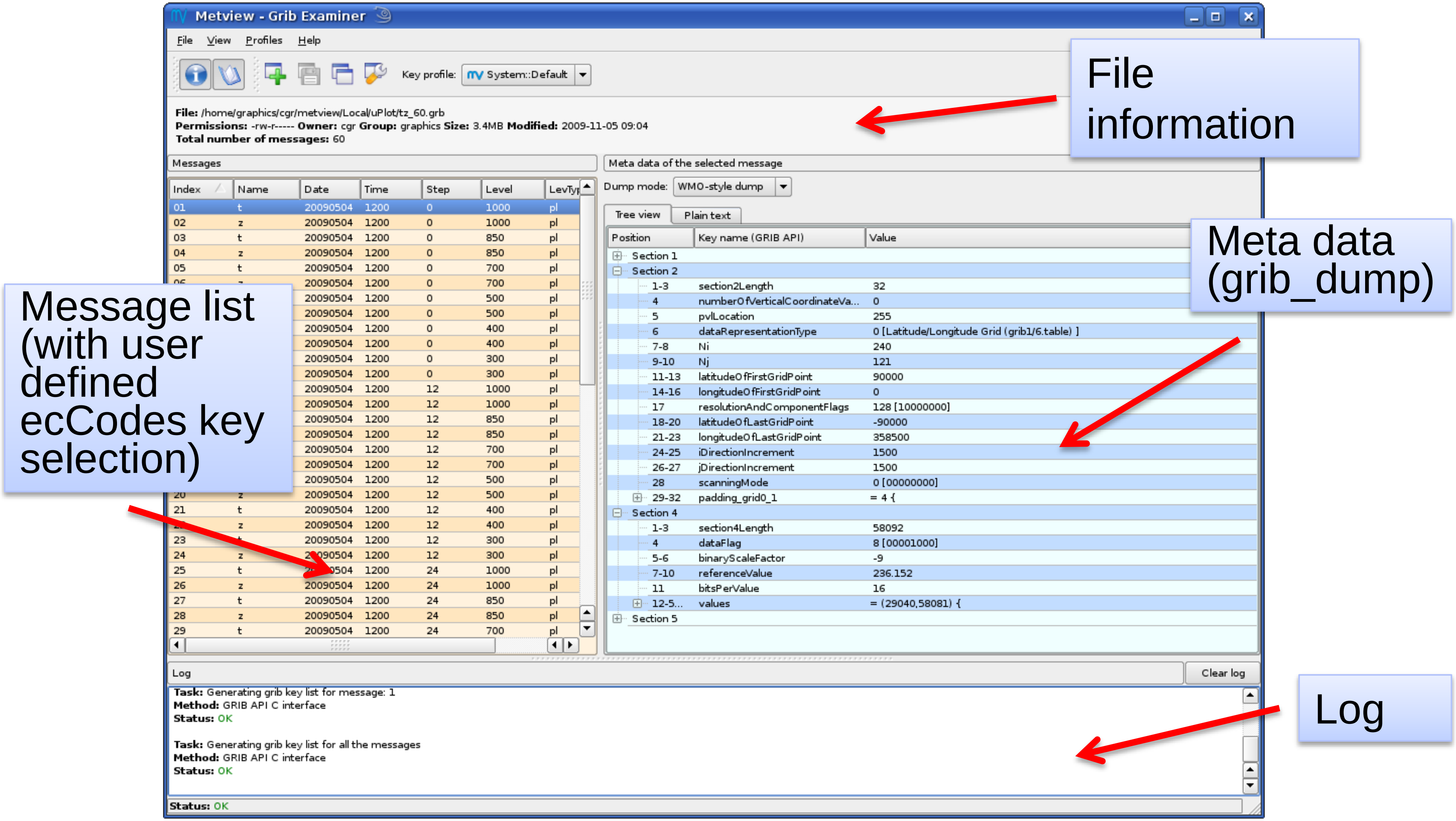
1. Use grib_ls to inspect the content of the files t2m.grib1 and t2m.grib2
   - Which keys does grib_ls show by default ?
   - What fields do the GRIB messages contain ?
   - Print the MARS keys. Add the shortName to the output
   - Order the output in descending step order [Hint: think about strings and integers…]
2. Use grib_ls to print the centre, dataDate, stepRange, typeOfLevel and shortName for forecast step 6 only
   - Output the centre as both a string and an integer
3. Use grib_dump to inspect the fourth (count=4) GRIB message in both files
   - Experiment with the different grib_dump options: `-O`, `-a` and `-t`
   - Identify the parameter, date, time, forecast step and the grid geometry

GRIB Examiner (Metview 4)

- Interactive examiner using ecCodes
- Actively developed and maintained by the Metview team
- Can be started up from the command line. E.g. on ecgate use

```
metview -e grib your_grib_file
```

GRIB Examiner: The user interface



Key profile: System::Default

File: /home/graphics/cgr/metview/Local/uPlot/tz_60.grb
Permissions: -rw-r----- **Owner:** cgr **Group:** graphics **Size:** 3.4MB **Modified:** 2009-11-05 09:04
Total number of messages: 60

Messages

Index	Name	Date	Time	Step	Level	LevTyp
01	t	20090504	1200	0	1000	pl
02	z	20090504	1200	0	1000	pl
03	t	20090504	1200	0	850	pl
04	z	20090504	1200	0	850	pl
05	t	20090504	1200	0	700	pl
06	z	20090504	1200	0	700	pl
[illegible]	[illegible]	20090504	1200	0	500	pl
[illegible]	[illegible]	20090504	1200	0	500	pl
[illegible]	[illegible]	20090504	1200	0	400	pl
[illegible]	[illegible]	20090504	1200	0	400	pl
[illegible]	[illegible]	20090504	1200	0	300	pl
[illegible]	[illegible]	20090504	1200	0	300	pl
[illegible]	[illegible]	20090504	1200	12	1000	pl
[illegible]	[illegible]	20090504	1200	12	1000	pl
[illegible]	[illegible]	20090504	1200	12	850	pl
[illegible]	[illegible]	20090504	1200	12	850	pl
[illegible]	[illegible]	20090504	1200	12	700	pl
[illegible]	[illegible]	20090504	1200	12	700	pl
[illegible]	[illegible]	20090504	1200	12	500	pl
20	z	20090504	1200	12	500	pl
21	t	20090504	1200	12	400	pl
22	z	20090504	1200	12	400	pl
23	t	20090504	1200	12	300	pl
24	z	20090504	1200	12	300	pl
25	t	20090504	1200	24	1000	pl
26	z	20090504	1200	24	1000	pl
27	t	20090504	1200	24	850	pl
28	z	20090504	1200	24	850	pl
29	t	20090504	1200	24	700	pl

Meta data of the selected message

Dump mode: WMO-style dump

Tree view | Plain text

Position	Key name (GRIB API)	Value
Section 1		
Section 2		
1-3	section2Length	32
4	numberOfVerticalCoordinateVa...	0
5	pvlLocation	255
6	dataRepresentationType	0 [Latitude/Longitude Grid (grib1/6.table)]
7-8	Ni	240
9-10	Nj	121
11-13	latitudeOfFirstGridPoint	90000
14-16	longitudeOfFirstGridPoint	0
17	resolutionAndComponentFlags	128 [10000000]
18-20	latitudeOfLastGridPoint	-90000
21-23	longitudeOfLastGridPoint	358500
24-25	iDirectionIncrement	1500
26-27	jDirectionIncrement	1500
28	scanningMode	0 [00000000]
29-32	padding_grid0_1	= 4 {
Section 4		
1-3	section4Length	58092
4	dataFlag	8 [00001000]
5-6	binaryScaleFactor	-9
7-10	referenceValue	236.152
11	bitsPerValue	16
12-5...	values	= (29040,58081) {
Section 5		

Log

Clear log

```
Task: Generating grib key list for message: 1
Method: GRIB API C interface
Status: OK

Task: Generating grib key list for all the messages
Method: GRIB API C interface
Status: OK
```

Status: OK

File information

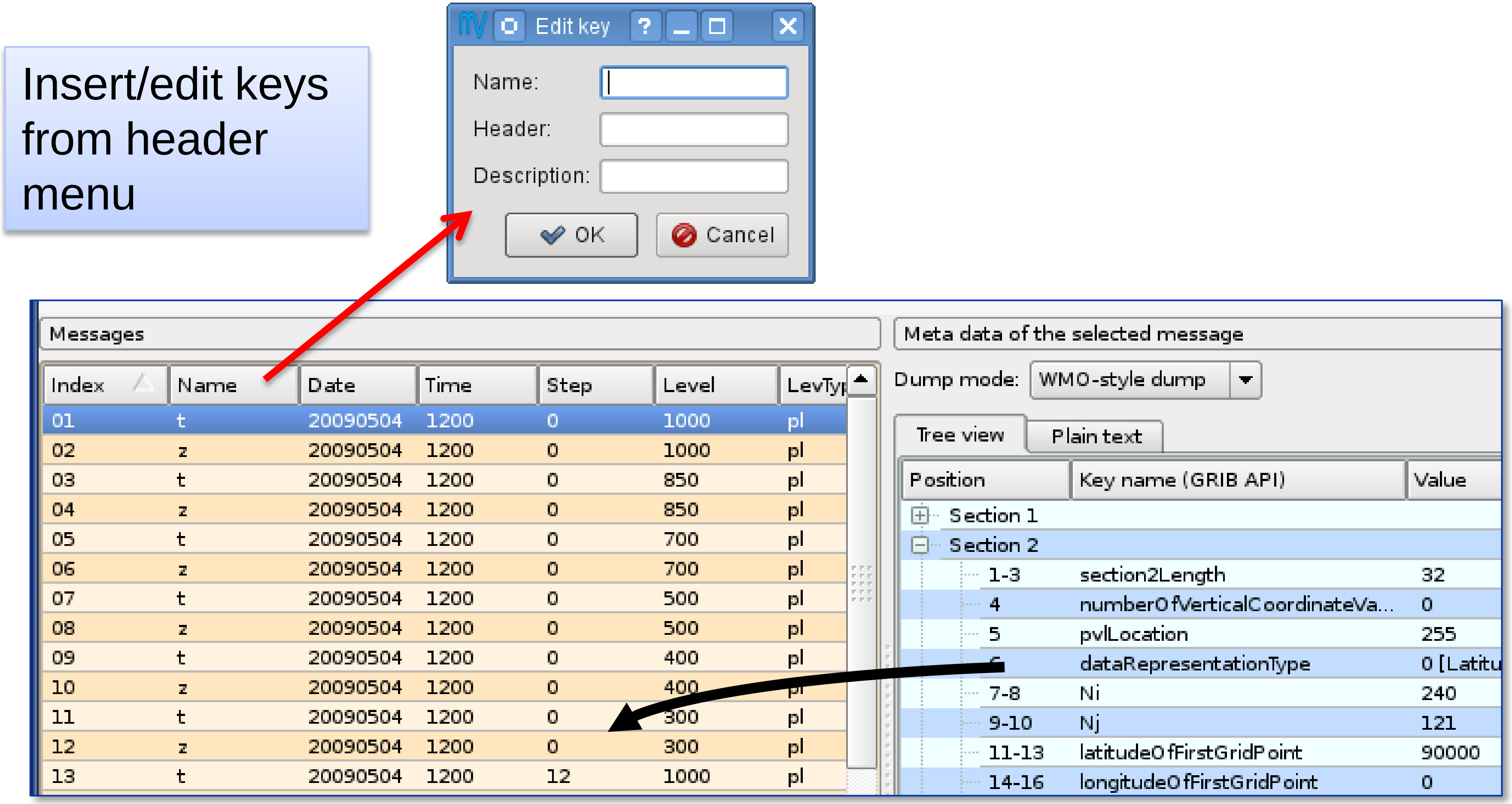
Meta data (grib_dump)

Message list (with user defined ecCodes key selection)

Log

GRIB Examiner: managing keys

Insert/edit keys from header menu



Edit key

Name:

Header:

Description:

Messages

Index	Name	Date	Time	Step	Level	LevTyp
01	t	20090504	1200	0	1000	pl
02	z	20090504	1200	0	1000	pl
03	t	20090504	1200	0	850	pl
04	z	20090504	1200	0	850	pl
05	t	20090504	1200	0	700	pl
06	z	20090504	1200	0	700	pl
07	t	20090504	1200	0	500	pl
08	z	20090504	1200	0	500	pl
09	t	20090504	1200	0	400	pl
10	z	20090504	1200	0	400	pl
11	t	20090504	1200	0	300	pl
12	z	20090504	1200	0	300	pl
13	t	20090504	1200	12	1000	pl

Meta data of the selected message

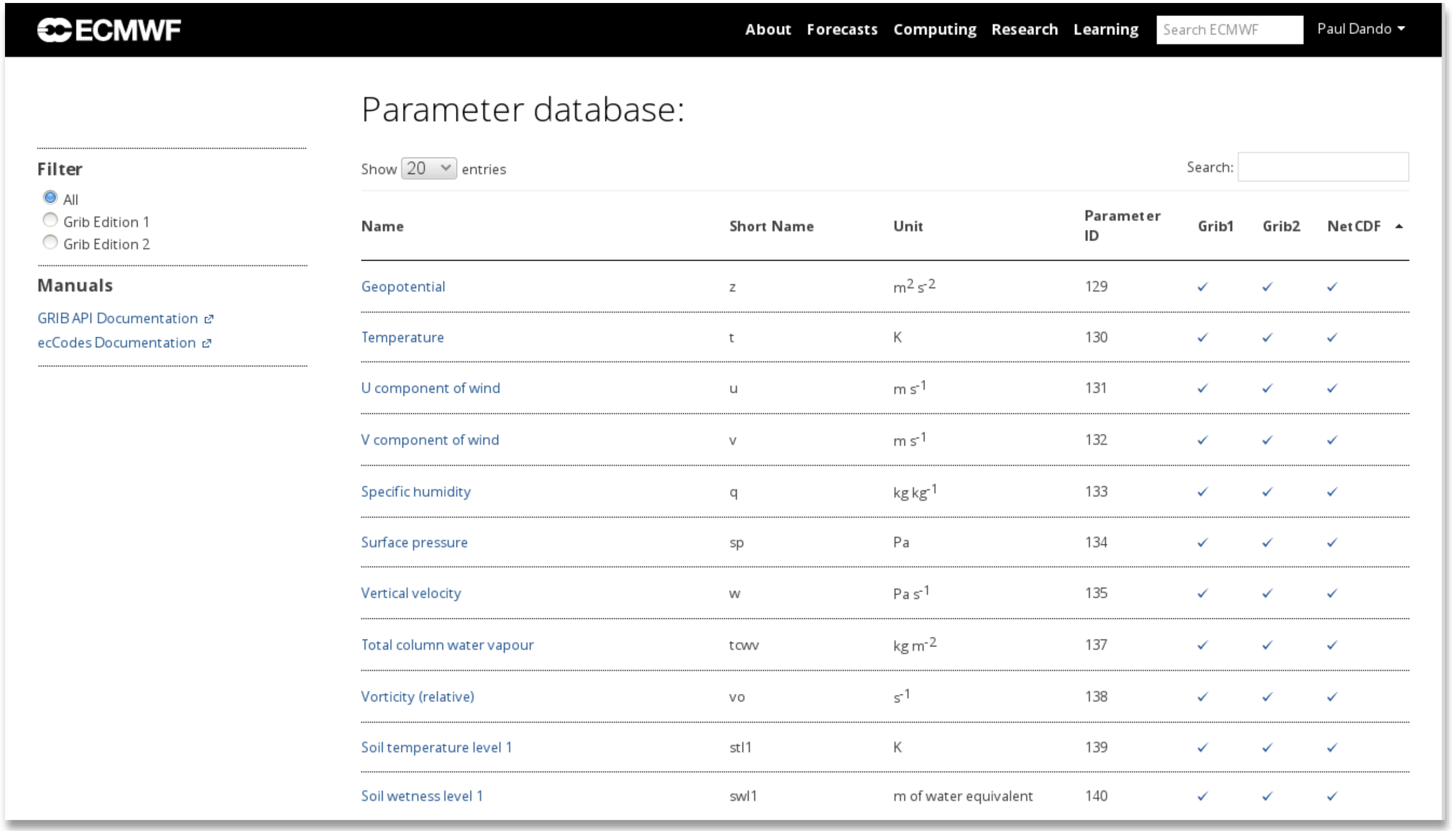
Dump mode: WMO-style dump

Position	Key name (GRIB API)	Value
Section 1		
Section 2		
1-3	section2Length	32
4	numberOfVerticalCoordinateVa...	0
5	pvlLocation	255
6	dataRepresentationType	0 [Latitu
7-8	Ni	240
9-10	Nj	121
11-13	latitudeOfFirstGridPoint	90000
14-16	longitudeOfFirstGridPoint	0

Drag and drop a new key

The parameter database

- The parameter database stores information about the GRIB 1, GRIB 2 and, for some parameters, NetCDF encoding of all parameters recognised by ecCodes



Parameter database:

Filter

● All
○ Grib Edition 1
○ Grib Edition 2

Manuals

GRIB API Documentation
ecCodes Documentation

Show 20 entries

Search:

Name	Short Name	Unit	Parameter ID	Grib1	Grib2	NetCDF
Geopotential	z	$m^2 s^{-2}$	129	✓	✓	✓
Temperature	t	K	130	✓	✓	✓
U component of wind	u	$m s^{-1}$	131	✓	✓	✓
V component of wind	v	$m s^{-1}$	132	✓	✓	✓
Specific humidity	q	$kg kg^{-1}$	133	✓	✓	✓
Surface pressure	sp	Pa	134	✓	✓	✓
Vertical velocity	w	$Pa s^{-1}$	135	✓	✓	✓
Total column water vapour	tcwv	$kg m^{-2}$	137	✓	✓	✓
Vorticity (relative)	vo	s^{-1}	138	✓	✓	✓
Soil temperature level 1	stl1	K	139	✓	✓	✓
Soil wetness level 1	swl1	m of water equivalent	140	✓	✓	✓

- The database is accessible via a web interface at:
  - http://apps.ecmwf.int/codes/grib/param-db

Finding nearest grid points with grib_ls

- The value of a GRIB field close to a specified Latitude/Longitude point can be found with grib_ls

optional

```
grib_ls -l Latitude,Longitude,MODE,file grib_file
```

MODE Can take the values

4 Print values at the 4 nearest grid points (default)

1 Print value at the closest grid point

file Specifies a GRIB file to use as a mask

The closest *land* point (with mask ≥ 0.5) is printed

- GRIB files specified must contain grid point data

Practical 2: using grib_ls –l

- The file t2m.grib1 contains the 2m temperature from the ENS control forecast at 6-hourly time steps for the first 24 hours on the O640 octahedral reduced Gaussian grid

1. Find the value of the 2m temperature at the grid point nearest to ECMWF (Lat 51.42°N, Lon 0.95°W) at each forecast step
   - Be careful to specify the longitude correctly !
   - What is the lat-lon value of the grid point nearest to ECMWF ?
   - How far is the chosen grid point from ECMWF ?
2. Change the command used to output only the forecast step and the 2t value at the nearest grid point
3. Change the command to output the 2t values at the four grid points nearest to ECMWF
4. Use the file lsm.grib1 to provide a land-sea mask
   - Are all four nearest grid points land points (mask ≥ 0.5) ?

grib_get – get key / value pairs

- Use grib_get to get the values of one or more keys from one or more GRIB files – very similar to grib_ls
- By default grib_get fails if an error occurs (e.g. key not found) returning a non-zero exit code
  - Suitable for use in scripts to obtain key values from GRIB messages
  - Can force grib_get not to fail on error
- Options available to get all MARS keys or all keys for a particular namespace
  - Can get other keys in addition to the default set
- Format of floating point values can be controlled with a C-style format statement
- grib_get can also be used to find the grid point(s) nearest to a specified latitude-longitude and print the value of the field at that point
  - Works in the same way as grib_ls

grib_get – examples

- To get the centre of the first (count=1) GRIB message in a file (both as a ‘string’ and a ‘integer’)

```
> grib_get -w count=1 -p centre f1.grib1
ecmf

> grib_get -w count=1 -p centre:i f1.grib1
98
```

- grib_get fails if there is an error

```
> grib_get -p mykey f1.grib1
ECCODES ERROR   :  Key/value not found

> echo $?
246
```

returns the exit code from the previous command

grib_get – examples

- To get all the MARS keys, optionally printing the shortName

```
> grib_get -m f1.grib1
g sfc 20170221 1200 0 167.128 od an oper 0001

> grib_get -m -P shortName f1.grib1
2t g sfc 20170221 1200 0 167.128 od an oper 0001
```

- To get all keys belonging to the statistics namespace

```
 > grib_get -n statistics f1.grib1
314.24 214.613 277.111 21.0494 41379.8 2.48314e-05 0
```

- grib_get –m is the same as grib_get –n mars

grib_get – controlling output format

- The format of floating point values can be controlled by using a C-style format statement with the **–F** option
  - **–F “%.4f”** - Decimal format with 4 decimal places (1.2345)
  - **–F “%.4e”** - Exponent format with 4 decimal places (1.2345E-03)

```
> grib_get –F “%.6f” –p maximum f1.grib1

314.240280

> grib_get –F “%.4e” –p maximum f1.grib1

3.1424e+02
```

- Default format is **–F “%.10e”**

grib_get – stepRange and stepUnits

- The step is always printed as an integer value
- By default the units of the step are printed in hours
- To obtain the step in other units set the stepUnits appropriately with the `-s` option

```
> grib_get -p stepRange f1.grib1
6
12

> grib_get -s stepUnits=m -p stepRange f1.grib1
360
720
```

stepUnits can be s, m, h, 3h, 6h, 12h, D, M, Y, 10Y, 30Y, C

Finding nearest grid points with grib_get

- The value of a GRIB field close to a specified Latitude/Longitude point can be found with grib_get
  - Works in the same way as grib_ls

```
> grib_get -l 52.0,-1.43 f1.grib1

273.58 272.375 273.17 273.531


> grib_get -F “%.5f” -P stepRange -l 52.0,-1.43,1 f1.grib1

0 272.37505
```

- GRIB files specified must contain grid point data

Getting data values at an index point

- The value of a GRIB field at a particular index point can be printed using grib_get with the `-i` option
- For example, find the index of a nearest grid point with grib_ls and then use this with grib_get to build a list of values at that point:

```
> grib_get -F “%.2f” -i 2159 -p step,dummy:s f1.grib1

6 99429.31
12 99360.25
18 99232.31
24 99325.56
```

Forces a space between step and value

- Also returns a value for non-grid point data !

grib_get_data – print data values

- Use grib_get_data to print a list of latitude, longitude (for grid point data) and data values from one or more GRIB files
- The format of the output can be controlled by using a C-style format statement with the **–F** option

  –F “%.4f” – Decimal format with 4 decimal places (1.2345)

  –F “%.4e” – Exponent format with 4 decimal places (1.2345E-03)

  The default format is **–F “%.10e”**
- By default missing values are not printed
  - A user-provided string can be printed in place of any missing values
- By default grib_get_data fails if there is an error
  - Use the **–f** option to force grib_get_data not to fail on error

grib_get_data – usage

```
grib_get_data [options] grib_file grib_file ...
```

Options

`-p key1,key2,…`	Keys to print
`-w key1=val1,key2!=val2,…`	Where option
`-m missingValue`	Specify missing value string
`-F format`	C-style format for output values
`-f`	Do *not* fail on error
`-V`	Print ecCodes Version

```
...
```

grib_get_data – example

```
> grib_get_data -F "%.4f" f1.grib1
Latitude, Longitude, Value
   81.000     0.000 22.5957
   81.000     1.500 22.9009
   81.000     3.000 22.8359
   81.000     4.500 22.3379
   81.000     6.000 21.5547
   81.000     7.500 20.7344
   81.000     9.000 19.8916
   81.000    10.500 18.5747
   81.000    12.000 17.2578
   81.000    13.500 16.1343
   81.000    15.000 14.9785
   81.000    16.500 13.8296
…
```

Format option applies to values only - not to the Latitudes and Longitudes

grib_get_data – missing values example

```
> grib_get_data –m XXXXX –F “%.4f” f1.grib1
Latitude, Longitude, Value
…
   81.000   90.000 9.4189
   81.000   91.500 8.6782
   81.000   93.000 XXXXX
   81.000   94.500 XXXXX
   81.000   96.000 XXXXX
   81.000   97.500 XXXXX
   81.000   99.000 6.7627
   81.000  100.500 7.4097
   81.000  102.000 7.9307
…
```

Missing values are printed with XXXXX

Practical 3: using grib_get & grib_get_data

1. Use grib_get to obtain a list of all the pressure levels available for parameter T in the file tz_an_pl.grib1
2. Use grib_get to print the stepRange for the field in the file surface.grib1 in (a) hours (b) minutes and (c) seconds
3. Repeat 2. for surface2.grib1 – what happens ?
4. Use grib_get_data to print the latitude, longitude and values for the field in surface.grib1
   - Output values in decimal format with 5 decimal places
   - Output values in exponential format with 10 decimal places
   - Are there any missing values ?
5. Use grib_get_data to print the data values for the temperature at 500 hPa only from the file tz_an_pl.grib1 ?
   - Make sure you print only the data for T500 ! What is printed ?

grib_copy – copy contents of GRIB files

- Use grib_copy to copy selected contents of GRIB files optionally printing some key values
- Without options grib_copy prints no key information
- Options exist to specify the set of keys to print
  - Use verbose option (**`–v`**) to print keys
- Output can be ordered
  - E.g. order by ascending or descending step
- Key values can be used to specify the output file names
- grib_copy fails if a key is not found
  - Use the **`–f`** option to force grib_copy not to fail on error

grib_copy – examples

- To copy only fields at 100 hPa from a file

```
> grib_copy -w level=100 in.grib1 out.grib1
```

- To copy only those fields that are not at 100 hPa

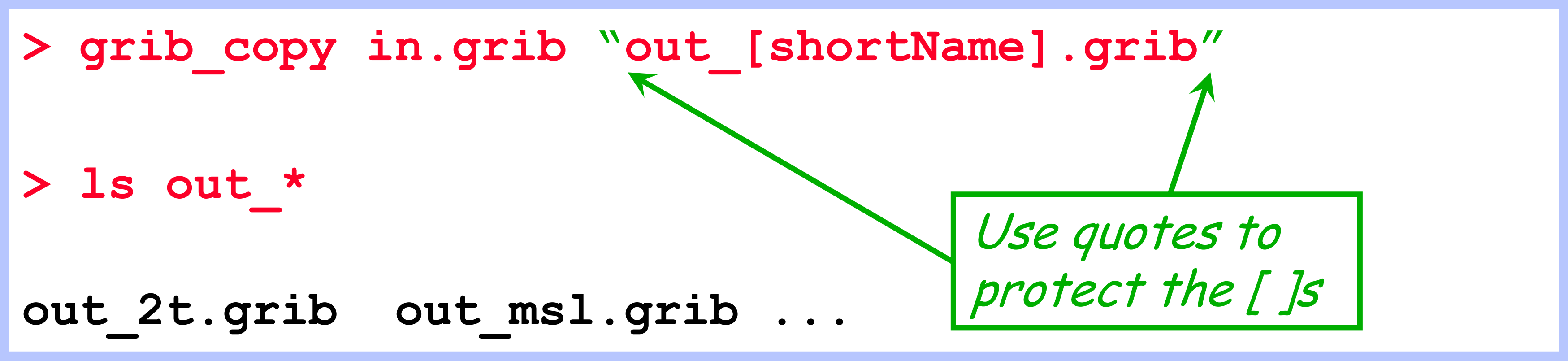
```
> grib_copy -w level!=100 in.grib1 out.grib1
```

- Information can be output using the –v and –p options

```
> grib_copy -v -p shortName in.grib1 out.grib1
in.grib1
shortName
t
1 of 1 grib messages in in.grib1
1 of 1 total grib messages in 1 files
```

grib_copy – using key values in output file

- Key values can be used to specify the output file name



```
> grib_copy in.grib "out_[shortName].grib"

> ls out_*

out_2t.grib  out_msl.grib ...
```

Use quotes to protect the [ ]s

- This provides a convenient way to filter GRIB messages into separate files

grib_set – set key / value pairs

- Use grib_set to
  - Set key / value pairs in the input GRIB file
  - Make simple changes to key / value pairs in the input GRIB file
- Each GRIB message is written to the output file
  - By default this includes messages for which no keys are changed
  - With `–S` (strict) option only messages matching all constraints in the where clause are copied
- An option exists to repack data
  - Sometimes after setting some keys involving properties of the packing algorithm the data needs to be repacked
- grib_set fails when an error occurs
  - e.g. when a key is not found

grib_set – usage

```
grib_set [options] grib_file grib_file … out_grib_file
```

Options

`-s key1=val1,key2=val2,…`	List of key / values to set
`-p key1,key2,…`	Keys to print (only with `–v`)
`-w key1=val1,key2!=val2…`	Where option
`-d value`	Set all data values to `value`
`-f`	Do *not* fail on error
`-v`	Verbose
`-S`	Strict
`-r`	Repack data
`...`	

grib_set – examples

- To set the parameter value of a field to 10m wind speed (10si)

```
> grib_set -s shortName=10si in.grib1 out.grib1
```

- This changes e.g.
  - shortName to 10si
  - paramId to 207
  - name / parameterName to ‘10 metre wind speed’
  - units / parameterUnits to ‘m s ** -1’

  - indicatorOfParameter to 207
  - marsParam to 207.128

grib_set – examples

- Some keys are read-only and cannot be changed directly

```
> grib_set -s name="10 metre wind speed" in.grib1 out.grib1

ECCODES ERROR   :  grib_set_values[0] name (3) failed: Value is read only
```

- The read-only keys can only be set by setting one of the other keys, e.g.
  - shortName=10si
  - paramId=207
  - indicatorOfParameter=207   GRIB edition dependent !

grib_set – modify data values

- An offset can be added to all data values in a GRIB message by setting the key offsetValuesBy

```
> grib_get -F “%.5f” -p max,min,average TK.grib
315.44727 216.96680 286.34257

> grib_set -s offsetValuesBy=-273.15 TK.grib TC.grib

> grib_get -F “%.5f” -p max,min,average TC.grib
42.29726 -56.18321 13.19257
```

grib_set – modify data values

- The data values in a GRIB message can be multiplied by a factor by setting the key scaleValuesBy

```
> grib_get -F “%.2f” -p max,min,average Z.grib
65035.92 -3626.08 2286.30

> grib_set -s scaleValuesBy=0.102 Z.grib1 orog.grib1

> grib_get -F “%.2f” -p max,min,average orog.grib1
6633.64 -369.86 233.20
```

grib_set – using key values in output file

- Key values can be used to specify the output file

```
> grib_set -s time=0000 in.grib “out_[shortName].grib”


> ls out_*
out_2t.grib  out_msl.grib ...
```

- Remember: Use quotes to protect the [ ]s !

What cannot be done with grib_set

- grib_set cannot be used for making transformations to the data representation
  - It cannot be used to transform data from spectral to grid-point representation (and vice-versa)
- grib_set cannot be used transform data from one grid representation to another
  - It cannot be used to transform data from regular or reduced Gaussian grids to regular latitude-longitude grids
- grib_set cannot be used to select sub-areas of data
  - It will change the value of, e.g. latitudeOfFirstGridPointInDegrees etc, but the data will still be defined on the original grid

- The GRIB tools cannot be used to interpolate the data

grib_to_netcdf – convert to NetCDF

- Use grib_to_netcdf to convert GRIB messages to NetCDF
- Input GRIB fields must be on a regular grid
  - typeOfGrid=regular_ll or regular_gg
- Options allow user to specify
  - the NetCDF data type:
    - NC_BYTE, NC_SHORT, NC_INT, NC_FLOAT or NC_DOUBLE
    - NC_SHORT is the default
  - either classic (NetCDF 3) or NetCDF 4 file format
  - the reference date
    - default is 19000101
- Used in the MARS web interface and the public Data Servers to provide data in NetCDF files

grib_to_netcdf – usage

```
grib_to_netcdf [options] grib_file grib_file …
```

Options

`-o output_file`	Output netCDF file
`-R YYYYMMDD`	Use `YYYYMMDD` as reference date
`-D NC_DATATYPE`	NetCDF data type
`-k kind`	Kind of file to be created: 1 → netCDF classic file format 2 → netCDF 64 bit classic file format (Default) 3 → netCDF-4 file format 4 → netCDF-4 classic model file format
`-T`	Do not use time of validity.
`-u dimension`	Set `dimension` to be an unlimited dimension
`-f`	Do *not* fail on error
`...`	

grib_to_netcdf – examples

- To convert the fields in file.grib1 to NetCDF

```
> grib_to_netcdf -o out.nc file.grib1
grib_to_netcdf: Version 2.2.0
grib_to_netcdf: Processing input file 'file1.grib1'.
grib_to_netcdf: Found 1 GRIB fields in 1 file.
grib_to_netcdf: Ignoring key(s): method, type, stream, refdate, hdate
grib_to_netcdf: Creating netCDF file 'out1.nc'
grib_to_netcdf: NetCDF library version: 4.3.2 of May 10 2016 11:12:41 $
grib_to_netcdf: Creating large (64 bit) file format.
grib_to_netcdf: Defining variable 't2m'.
grib_to_netcdf: Done.

> ls -s out.nc
160 out.nc
```

grib_to_netcdf – examples

- To convert the fields in file.grib1 to NetCDF with data type set to NC_FLOAT

```
> grib_to_netcdf -D NC_FLOAT -o out.nc file.grib1
grib_to_netcdf: Version 2.2.0
grib_to_netcdf: Processing input file 'file1.grib1'.
grib_to_netcdf: Found 1 GRIB fields in 1 file.
grib_to_netcdf: Ignoring key(s): method, type, stream, refdate, hdate
grib_to_netcdf: Creating netCDF file 'out1.nc'
grib_to_netcdf: NetCDF library version: 4.3.2 of May 10 2016 11:12:41 $
grib_to_netcdf: Creating large (64 bit) file format.
grib_to_netcdf: Defining variable 't2m'.
grib_to_netcdf: Done.

> ls -s out.nc
316 out.nc
```

Output NetCDF file is about twice the size

Practical 4: modifying GRIB messages

1. The file tz_an_pl.grib1 contains parameters T and Z on five pressure levels.
   - Use grib_copy to create two files, one containing all the pressure levels for parameter T, the other for Z. Check the content of the new files with grib_ls
2. Use grib_ls to inspect the contents of tp.grib
   - What is the parameter set to ?
   - Use grib_set to change the parameter for the message in the file tp.grib to total precipitation (shortName=tp – paramId=228). Check the new message with grib_ls.
3. Use grib_to_netcdf to convert the GRIB messages in file1.grib to NetCDF.
   - Try with both the default data type (NC_SHORT) and NC_FLOAT.
   - Check the data values in each case with ncdump.
4. Use grib_to_netcdf to convert the GRIB messages in file2.grib to NetCDF.
   - What happens … and why ?

ecCodes user interfaces

- For some processing it is more convenient – or even necessary – to write a program
- The ecCodes library supports three user interfaces:
  - C: `#include <eccodes.h>`
  - Fortran 90 interface: `use eccodes`
  - Python interface: `import eccodes`
- At ECMWF two environment variables ECCODES_INCLUDE and ECCODES_LIB are defined to aid compilation and linking of Fortran 90 and C programs
- On ecgate:

```
gcc  myprog.c $ECCODES_INCLUDE $ECCODES_LIB -lm
gfortran myprog.f90 $ECCODES_INCLUDE $ECCODES_LIB
```

General framework for decoding

- Open one or more GRIB files (for read or write)
  - Standard Fortran calls cannot be used to open a GRIB file – you must use codes_open_file
- Calls to load one or more GRIB messages into memory
  - Two main subroutines: codes_grib_new_from_file / codes_new_from_index
  - These return a unique identifier used to manipulate the loaded GRIB messages
- Calls to decode the loaded GRIB messages – only loaded GRIB messages can be decoded
  - codes_get
  - Decode only what you need (not the full message !)
- Release the loaded GRIB messages:
  - codes_release
- Close the opened GRIB files
  - Standard Fortran calls cannot be used to close a GRIB file – you must use codes_close_file

Input arguments
Output arguments

Fortran example – codes_get

```
! Load all the GRIB messages contained in file.grib1
call codes_open_file(ifile, 'file.grib1', 'r')

call  codes_grib_new_from_file(ifile, igrib, iret)
LOOP: do while (iret /= CODES_END_OF_FILE)
! Decode/encode data from the loaded message
      call codes_get(igrib , “dataDate”, date)
      call codes_get(igrib, “typeOfLevel”, levtype)
      call codes_get(igrib, “level”, level)
      call codes_get_size(igrib, “values”, nb_values)
      allocate(values(nb_values))
      call codes_get(igrib, “values”, values)
      print*, date, levtype, level, values(1), values(nb_values)
! Release memory
      deallocate(values)
      call codes_release(igrib)
! Next message
     call codes_grib_new_from_file(ifile,igrib, iret)
end do LOOP
call codes_close_file(ifile)
```

Loop on all the messages in a file. A new grib message is loaded from file. igrib is the grib id to be used in subsequent calls

values is declared as
real, dimension(:), allocatable:: values

Release the memory !

Python example – codes_get

```
#!/usr/bin/env python
import sys
from eccodes import *

# Load all the GRIB messages contained in file.grib1
ifile = open('file.grib1‘)
while 1:
    igrib = codes_grib_new_from_file(ifile)
    if igrib is None: break

    # Decode/encode data from the loaded message
    date = codes_get( igrib , “dataDate”)
    levtype = codes_get(igrib, “typeOfLevel”)
    level = codes_get(igrib, “level”)
    values = codes_get_values(igrib)
    print  date, levtype, level, values[0], values[len(values)-1]


    # Release
    codes_release(igrib)
ifile.close()
```

Loop on all the messages in a file. A new grib message is loaded from file. igrib is the grib id to be used in subsequent calls

Values returned as an array

Release the memory !

Practical 5: GRIB decoding with Fortran 90

- The practical5/Fortran directory contains the program eccodes_demo.f90, a Makefile and some data in grib_file.grib
  - If you prefer there is also a practical5/Python and a practical5/C – see the README file
- Build an executable and run with (for Fortran or C)

```
> make
> ./eccodes_demo > output
```

- Look at the text information written to the output file.
- Use grib_ls and grib_dump to examine the file grib_file.grib
- Change the program, replacing the call to codes_dump with several calls to codes_get to decode the values for the edition, date, time, paramId (or shortName) and level
- Add your own ‘WRITE’ or ‘PRINT ’statements to output this information

ecCodes can do more…

- The idea is to provide a set of high-level keys or subroutines to derive / compute extra information from a loaded GRIB message
- For example:
  - keys (READ-ONLY) to return average, min, max of values, distinct latitudes or longitudes, etc …
  - Subroutines to compute the latitude, longitude and values
    - codes_grib_get_data
  - Subroutines to extract values
    - codes_grib_find_nearest: extract values closest to given geographical points
    - codes_get_element: extract values from a list of indexes
  - Subroutines for indexed access
    - Usually much faster than sequential access for “random” access

For lat/lon, Gaussian, reduced Gaussian grids. It is similar to the grib_get_data GRIB tool

Similar to "grib_ls –l "or "grib_get –l"

GRIB decoding – summary

- Use GRIB Tools where possible
  - It is not always necessary to write a program !
- Use edition-independent keys
  - Provides transparent access to GRIB 1 and GRIB 2 messages
- ECMWF introduced GRIB 2 encoding for all its model level fields in May 2011
- If you do need to write a program think carefully about how the fields are accessed
  - Indexed access can be much faster than sequential access
- If you want to learn more about ecCodes then we hold specialist courses each year

  ecCodes:GRIB and ecCodes:BUFR

Documentation

- The WMO FM 92 GRIB Manuals can be obtained from

  www.wmo.int/pages/prog/www/WMOCodes.html

- The ECMWF ecCodes manual is available at

  https://software.ecmwf.int/wiki/display/ECC

- The GRIB Tools are documented at

  https://software.ecmwf.int/wiki/display/ECC/GRIB+tools

- ecCodes Fortran 90 interface:

  https://software.ecmwf.int/wiki/display/ECC/ecCodes+API+Reference

- ecCodes GRIB examples

  https://software.ecmwf.int/wiki/display/ECC/GRIB+examples