Online Computing Training Week 2021

Working with GRIB and BUFR made easier with ecCodes

Webinar by Paul Dando

Q&As

Answered in recording

What is the major advantage of using GRIB instead of more common formats like netCDF?

- First, GRIB is the format approved by the WMO for distribution of NWP data. In addition, the design for transfer over slow and unreliable connections means that if you lose some bytes from a file containing a number of GRIB or BUFR messages then it's likely that only one or two message will be lost and need to be re-transferred. If the same happens with a netCDF file then it's possible that all of the information contained in the file will be unreadable and the whole file will need to be retransmitted.

Why is it necessary to encode even the data in such a cryptic manner?

- GRIB and BUFR were designed for transferring data over slow TELEX lines and so it was important to make the messages as small as possible. Hence the binary format and the compression of the information.

Do you recommend writing our new developments for dealing with GRIB files with cfgrib, or should we still use ecCodes?

- cfgrib was developed by a third party for, and with oversight of, ECMWF. We try to respond to general queries if we can, but cannot guarantee quick fixes or improvements. cfgrib uses ecCodes internally for GRIB handling and is great if your GRIB data fits a hypercube model and you want to use xarray, but not all GRIB files fit this structure. It only works with regular lat/lon data if you want to slice the data geographically. Another option is to use the Python interface to ECMWF's Metview software, which handles many different grid types and does not require the GRIB fields to form a nice hypercube. It also uses ecCodes internally for GRIB handling. So the advice really depends on what your data looks like, and what you need to do with it. You should try each one to see if they can handle your data.

Something I wanted to ask for a long time - why in some of ECMWF products the surface data is provided in GRIB1 and model-level data in GRIB2? Why not make all of them in GRIB2?

- This is a good question! ECMWF originally introduced GRIB 2 for model level data when increasing the vertical resolution to 137 levels as there is a limitation of 127 levels in GRIB 1. But to avoid unnecessary changes for the majority of users who use only surface level parameters and parameters on constant pressure levels, it was decided to leave these in GRIB 1. In addition, some parameters – especially for the Ensemble Forecast - are specific to ECMWF and there was no WMO standard way available at the time to encode these in GRIB 2. However, the plan was that any new parameters would be encoded in GRIB 2 and there would be a gradual migration of other parameters to GRIB 2. Now the WMO is introducing GRIB 3 and ECMWF's Technical Advisory Committee is looking at whether we should move directly to that. Hopefully, this doesn't mean we end up with a mix of GRIB 1, GRIB 2 and GRIB 3 !

Of course, all old data in our MARS archive will remain encoded in the GRIB edition in which it was originally stored so there will always be a need for ecCodes to be able to decode GRIB 1.

Which is faster - using CLI or python API – and are there any benchmark results ?

- Since the CLI tools are written in C, they will be slightly faster as we do not need to go from Python to C and back (type/data conversions etc). We don't have any benchmarks results that demonstrate this. But don't use the CLI tools to extract the data for reading by the Python interface - that would be very inefficient !

Not answered during webinar

I've had plenty of issues installing ecCodes on Windows and the kernel keeps dying, any tips?!

- I think we need a proper bug report for that one. Did you try with miniconda and install from the conda-forge channel ? At what point does the kernel die - when you use the tools or the Python interface ?

Is it planned to include interpolation / regridding routines in eccodes?

- There are no plans to add this to ecCodes, but Metview (also ECMWF software, using ecCodes for GRIB handling) has a powerful GRIB regridding engine which will be described in Thursday afternoon's webinar!

About the parallel capabilities of ecCodes - how complicated is this in practice? Are the ecCodes tools 'smart' enough to handle the individual fragments (e.g. will command like 'grib_filter [something] input1 input2 ...' work?), or is that the user responsibility to code the needed preprocessing manually?

- The CL tools do not process in parallel. The client has to do this with the API e.g. use POSIX threads etc. Alternatively, the processing itself has to be run in parallel, e.g., grib_filter input1 &; grib_filter input2 & ; ….

I am using Xarray to open a grib2 file. It doesn't show the coordinates (Lcc) with the message: No latitudes/longitudes provided by ecCodes for gridType. Is there any plan to add the ability to read the Lon/lat to encodes? (I could open it using Pynio engine.)

- ecCodes should be able to provide the lat-lon coordinates of data on the lcc projection, both for the Earth as a sphere and an oblate spheroid. Maybe you are using an old version of cfgrib/ecCodes ?

How can we create BUFR or GRIB from plain text formatted somehow?

- Please see an example here: https://confluence.ecmwf.int/display/ECC/bufr_encode_flight. Here we read a CSV file and encode it into BUFR. Metview also has a module for creating regular lat/lon GRIB messages from ASCII input: https://confluence.ecmwf.int/display/METV/LatLong+Matrix. It's a bit basic, but it does the job as long as your matrix coordinates are regular.

I would like to write a GRIB message from a 2D array, do I have to copy a GRIB template file and change some keys and the values

- Yes, the way to create the GRIB is to use the samples (or you can provide your own if you have a GRIB similar to the one you want to create). Typically, you then just need to set the analysis (base) date, time, shortName, the forecast step and the data values

What is the more efficient way to extract a grid-point time series from a series of 2D field grib messages series using ecCodes?

- The most efficient way is to first find the element of the values array that corresponds to the grid point of interest (you do this just once - assuming all time-steps are o the same grid) and then to just extract that element from all of the GRIBs. The C interface will do this for you automatically provided the grids are the same and we will enable this in the Python3 interface too.