# ecCodes

#### GRIB Fortran 90 - Python APIs Practicals 2

Dominique Lucas and Xavi Abellan

Dominique.Lucas@ecmwf.int Xavier.Abellan@ecmwf.int



# Practical 2: ecCodes indexing

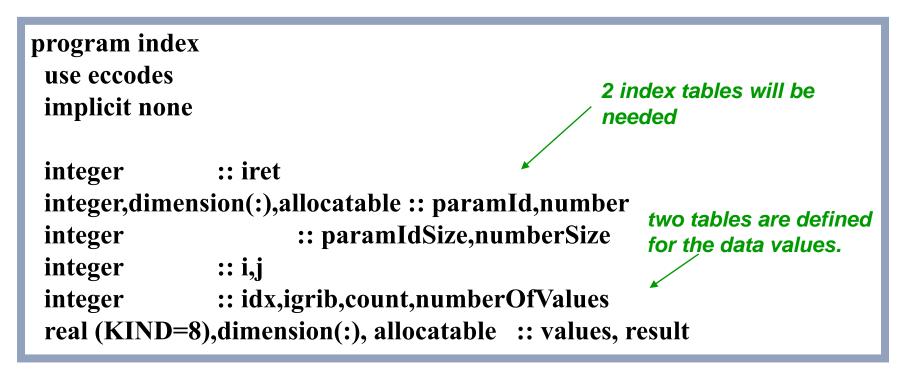
ecgate\$ cd \$SCRATCH/eccodes\_api\_practicals/exercise2 ecgate\$ cd F90 # or 'cd Python' ecgate\$ ls Makefile eccodes\_index.f90 eps grib\_api\_index.f90

• The file 'eps' contains ENS fields for all (50) (perturbed) ensemble members for some (4) parameters, e.g. see output of 'grib\_ls eps'.

• The objective of this exercise is to write a Fortran or a Python program using ecCodes and the indexed access method to compute the ensemble mean for one parameter.

• The files eccodes\_index.f90 and grib\_api\_index.f90 (or .py) contain a skeleton of the code or script. Please complete one of them. If you are confident, you can start from 'scratch'.

# Practical 2: Main program



#### Compile/link with:

Fortran: gfortran –o eccodes\_index eccodes\_index.f90 \$ECCODES\_INCLUDE \$ECCODES\_LIB or use make.

#### Practical 2: ecCodes indexing

• Run the resulting code with:

\$ eccodes\_index # (or grib\_api\_index) for Fortran

\$ python eccodes\_index.py # (or grib\_api\_index.py) for Python

• Now change the link for the input file 'eps' to the grib2 file (also available from ~trx/ecCodes/data) and run the program again.

\$ make grib2

# Practical 3: ecCodes timings

```
$ cd $SCRATCH
$ cd eccodes_api_practicals/exercise3
$ ls
Makefile ensmean_indexed.f90 ensmean_indexed_read.f90 run dirs
ensmean_reduced.f90 ensmean.f90 input run.out
$ make
$ run
```

• The 4 Fortran codes do the same thing. They all compute ensemble means and standard deviations with ecCodes for 12 fields (4 parameters – 3 levels).

## Practical 3: ecCodes timings

• The code in ensmean.f90 reads the complete grib file for each computation of a mean and std. It also decodes the data values even if a field is not used.

- The code in ensmean\_reduced.f90 is like the first code, but the data values for a field are decoded only when they are needed.
- The code in ensmean\_indexed.f90 builds an index based on the keys parameter, ensemble number and level. The grib messages are then accessed through this index. The index is then saved into a file.
- The code in ensmean\_indexed\_read.f90 is exactly the same as the previous except that the index is read from a file, not built.
- Note the different run times! Beware of best access method for different access pattern:
  - Sequential i/o (codes\_grib\_new\_from\_file) suitable for sequential access.
  - Indexed i/o (codes\_new\_from\_index) suitable for random access.

## Practical 4: ecCodes encoding

```
$ cd $SCRATCH
$ cd eccodes_api_practicals/exercise4
$ cd F90 # or Python
$ ls
Makefile eccodes_create.f90 eps grib_api_create.f90
```

• The objective of this exercise is to extend the code used in practical 2 to create a new grib message containing the ensemble mean, using ecCodes.

- Different options are available to create a grib message:
  - Clone the new field to be produced from one of the input grib fields.
  - Use a sample (or template) from the default samples directory. See 'codes\_info'.
  - Use a sample from a private samples directory.

# Practical 4: ecCodes encoding

- The first option is the easiest to implement. For simplicity, we suggest you to use this option.
- The file codes\_create.f90 (or grib\_api\_create.py) contain a skeleton of the code to create a grib message. Please can you try to add the code needed to create a grib message. Follow the instructions in the code. Use 'make' to compile the codes, then run the program or run the Python script.
- To change the GRIB headers, see local definitions under <u>http://apps.ecmwf.int/codes/grib/format/grib1/local/</u> and <u>http://apps.ecmwf.int/codes/grib/format/mars/type/</u> for the datatype.
- Now change the link for the input file 'eps' to the grib2 file (also available from ~trx/ecCodes/data) and run the program again.

#### \$ make grib2

# Practical 5: ecCodes grid packing

\$ cd \$SCRATCH \$ cd eccodes\_api\_practicals/exercise5 \$ ls eps.grib1 eps.grib2 ls.out pack\_data.cmd pack\_data.out \$ ./pack\_data.cmd

- The objective of this exercise is to see the impact of different types of packing.
- For simplicity, we only look at some packing types for grid point data.
- Note that the timings may vary.
- Note that ecCodes may not do the packing requested. Check the packingType of output files with grib\_ls.

**ECCMUF** ECCODES 2018 - GRIB FORTRAN 90 AND PYTHON APIS - PRACTICALS

# Practical 5: ecCodes API grid packing

- Which packing is the fastest, the slowest?
- Which packing does achieve the best compression'?
- Which packing types are not available for GRIB1?