ecCodes: Using BUFR Tools
Part 1

Computer User Training Course 2019

Shahram Najm

Development Section
Forecast Department

ECMWF

Contents

- BUFR Tools basics and getting help
- Information tools
- Inspection tools

ecCodes command line tools – basic concepts

- The ecCodes tools are a set of command line programs for interactive and batch processing of BUFR data
- They provide ready and tested solutions to the most common processing of BUFR data
- Their use will avoid the need to write new code and thus speed up your work
  - Consider using ecCodes tools instead of writing your own program
- The tools are provided with a common set of options so that it is quick to apply the same options to different tools
- Use of the tools is recommended whenever possible!

BUFR Tools – basics

All of the tools use a common syntax

bufr_<tool> [options] bufr_file [bufr_file] … [output_bufr]

- **Tool to count the messages in a BUFR file**
  - **bufr_count**
- **Tools to inspect the content of and compare BUFR files**
  - **bufr_dump, bufr_ls, bufr_get, bufr_compare**
- **Tool to copy some messages**
  - **bufr_copy**
- **Tools to change the content of a BUFR message**
  - **bufr_filter**

Getting help

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

```
> bufr_dump

NAME    bufr_dump

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

USAGE
        bufr_dump [options] bufr_file bufr_file ...

OPTIONS
        -j s/f/a  JSON mode (JavaScript Object Notation).
        -p        Plain dump
        ...
```

Documentation

- **The ecCodes home page is available at**

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

- **The BUFR Tools are documented at**

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

  Includes some examples of how to use the tools

- **The WMO FM 94 BUFR edition 3 and edition 4 Manuals can be obtained from**

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

- **The ecCodes software can be downloaded from**

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

codes_info – information about ecCodes installation

The codes_info tool gives basic information about the ecCodes package being used

- **ecCodes Version**
- **Path to definition files:** ECCODES_DEFINITION_PATH
- **Path to sample files:** ECCODES_SAMPLES_PATH

```
> codes_info
eccodes Version 2.10.0

Default definition files path is used: /usr/local/eccodes/2.10.0/share/eccodes/definitions
Definition files path can be changed by setting ECCODES_DEFINITION_PATH environment variable

Default SAMPLES path is used: /usr/local/eccodes/2.10.0/share/eccodes/samples
SAMPLES path can be changed by setting ECCODES_SAMPLES_PATH environment variable
```

bufr_count – count BUFR messages

- **Counts (very quickly) the number of BUFR messages in a list of files**
- **Syntax**

```
bufr_count [-v] bufr_file1 [bufr_file2 …]
```

(takes wildcards)

Without the ‘-v’ option, it prints the total number of messages. With ‘-v’ (verbose) it prints the number of messages per file as well as the total

```
> bufr_count syn*.bufr
  5

> bufr_count -v syn*.bufr
  1 syno_3.bufr
  1 syno_4.bufr
  3 syno_multi.bufr
  5 total
```

bufr_dump – dump content of BUFR files

- **Use bufr_dump to dump the content of a file containing one or more BUFR messages**
- **Various output formats are supported:**
  - **Plain mode prints ‘key=value’ pairs**
  - **JSON mode prints in JavaScript Object Notation**
  - **Octet mode provides a WMO documentation style dump (no unpacking)**
- **The default format (without any options) is the JSON mode**
  - **JSON is an open standard format that uses human-readable text to transmit data objects consisting of attribute-value pairs**
  - **Easy to visualise/navigate**

bufr_dump – examples

The simplest format is the "Plain" mode (option "-p"): each key printed with its value:

```
> bufr_dump -p ahws_139.bufr
```

```
    edition=3
    masterTableNumber=0
    bufrHeaderSubCentre=0
    bufrHeaderCentre=98
    updateSequenceNumber=0
    dataCategory=12
    ...
    localNumberOfObservations=492
    satelliteID=4
    observedData=1
    compressedData=1
    unexpandedDescriptors=312061
    directionOfMotionOfMovingObservingPlatform={
          294, 294, 294, 294, 294, 294, 294, 294, 294, 294,...}
    #1#backscatter={
          -14.3, -13.06, -12.1, -11.59, -11.4, -11.46, -11.98, ...}
    ...
    #2#backscatter={
          -13.41, -11.91, -10.74, -10.78, -10.75, -10.99, -12.01, -12.26 ... }
    ...
    surfaceSoilMoisture=MISSING
```

bufr_dump – examples

Without any options you get the JSON output:

```
> bufr_dump ahws_139.bufr
```

```
{ "messages" : [[
    { "key" : "edition",
      "value" : 3 },
    { "key" : "masterTableNumber",
      "value" : 0 },
   ...
   [ { "key" : "beamIdentifier",
       "value" : 1,
       "units" : "CODE TABLE" },
       [{ "key" : "radarIncidenceAngle",
          "value" : [47.91, 48.63, 49.34, 50.01, 50.7, ...],
          "units" : "deg" },
          [{
            "key" : "antennaBeamAzimuth",
            "value" : [126.79, 125.16, 123.52, 121.96, ... ],
            "units" : "deg" }
           ],
   ...
   [ { "key" : "beamIdentifier",
       "value" : 2,
       "units" : "CODE TABLE" },
       ...
```

bufr_dump – examples

With option "-ja" you get the JSON output plus key attributes:

```
> bufr_dump -ja ahws_139.bufr
```

```
...
 { "key" : "beamIdentifier",
   "value" : 1,
   "index" : 21,
   "code" : "008085",
   "units" : "CODE TABLE",
   "scale" : 0,
   "reference" : 0,
   "width" : 3},
   [ { "key" : "radarIncidenceAngle",
       "value" : [ ... ],
       "index" : 22,
       "code" : "002111",
       "units" : "deg",
       "scale" : 2,
       "reference" : 0,
       "width" : 13
   },
...
```

bufr_dump – examples

With option “-jf” you get the FLAT JSON output plus key attributes:

```
> bufr_dump -jf ahws_139.bufr
```

```
{ "key" : "centre",
  "value" : 99,
  "index" : 1,
  "code" : "001033",
  "units" : "CODE TABLE",
  "scale" : 0,
  "reference" : 0,
  "width" : 8 },
...
{ "key" : "beamIdentifier",
  "value" : 1,
  "index" : 21,
  "code" : "008085",
  "units" : "CODE TABLE",
  "scale" : 0,
  "reference" : 0,
  "width" : 3 },
{
  "key" : "radarIncidenceAngle",
  "value" : [63.96, 63.5, 63.04, ...]
  ...
}
...
```

bufr_dump – missing values

bufr_dump with JSON shows MISSING values as "null":

```
> bufr_dump -jf ahws_139.bufr
```

```
{
 "key" : "surfaceSoilMoisture",
  "value" : null,
  "index" : 65,
  "code" : "040001",
  "units" : "%",
  "scale" : 1,
  "reference" : 0,
  "width" : 10 },
...
{
  "key" : "backscatter",
  "value" : null,
  "index" : 72,
  "code" : "021062",
  "units" : "dB",
  "scale" : 2,
  "reference" : -5000,
  "width" : 13 },

...
```

bufr_dump – examples

Octet mode: WMO documentation style (low-level, no unpacking the data section):

```
> bufr_dump -O ahws_139.bufr
```

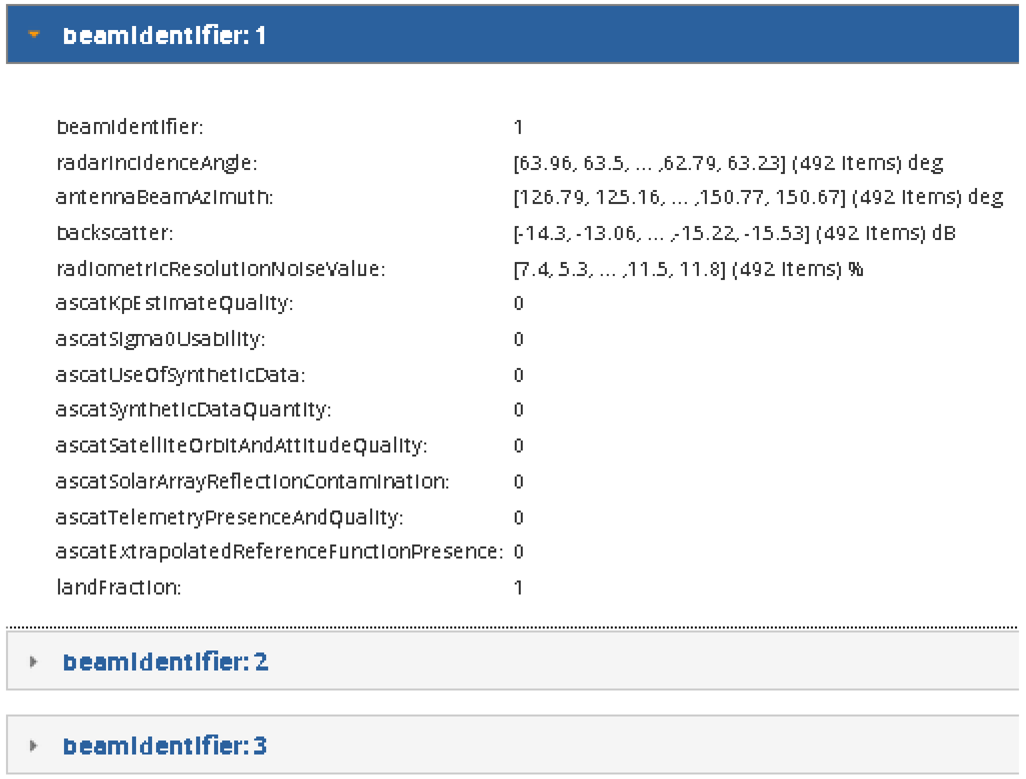
```
***** FILE: ahws_139.bufr
#==============   MESSAGE 1 ( length=13854 )              ==============
1-4        identifier = BUFR
5-7        totalLength = 13854
8          edition = 3
======================   SECTION_1 ( length=18, padding=0 )    ======================
1-3        section1Length = 18
4          masterTableNumber = 0
5          bufrHeaderSubCentre = 0 [Absent (common/c-1.table) ]
6          bufrHeaderCentre = 98 [European Centre for Medium-Range Weather Forecasts (common/c-1.table) ]
7          updateSequenceNumber = 0
8          section1Flags = 128 [10000000]
9          dataCategory = 12
10         dataSubCategory = 139
11         masterTablesVersionNumber = 13
12         localTablesVersionNumber = 1
13         typicalYearOfCentury = 12
14         typicalMonth = 11
15         typicalDay = 2
...
======================   SECTION_2 ( length=52, padding=0 )    ======================
1-3        section2Length = 52
4          reservedSection2 = 0
5          rdbType = 12
18         ...
```

bufr_dump (online) – BUFR Validator

- **You can also view the JSON output with additional functionality via the BUFR validator web page: http://apps.ecmwf.int/codes/bufr/validator/**
- **The array sizes are shown**
- **Tooltips display the key attributes as well as array entries**
- **Here the MISSING value is shown as “missing”**

Note:

- **Only the first message is displayed**
- **There is a 2MB size limit**


beamIdentifier: 1

```
beamIdentifier:                              1
radarIncidenceAngle:                         [63.96, 63.5, … ,62.79, 63.23] (492 items) deg
antennaBeamAzimuth:                          [126.79, 125.16, … ,150.77, 150.67] (492 items) deg
backscatter:                                 [-14.3, -13.06, … ,-15.22, -15.53] (492 items) dB
radiometricResolutionNoiseValue:             [7.4, 5.3, … ,11.5, 11.8] (492 items) %
ascatKpEstimateQuality:                      0
ascatSigma0Usability:                        0
ascatUseOfSyntheticData:                     0
ascatSyntheticDataQuantity:                  0
ascatSatelliteOrbitAndAttitudeQuality:       0
ascatSolarArrayReflectionContamination:      0
ascatTelemetryPresenceAndQuality:            0
ascatExtrapolatedReferenceFunctionPresence:  0
landFraction:                                1
```

beamIdentifier: 2

beamIdentifier: 3

Practical

- Copy the BUFR data files to your $SCRATCH

```
cd $SCRATCH
cp -r ~trx/ecCodes/2019/BufrFiles ./
cd BufrFiles
```

- Experiment with the bufr_dump tool. Store the output JSON file e.g. bufr_dump ahws_139.bufr > ahws_139.json
- View the generated JSON files (e.g. with the editor “kate” which understands JSON and can expand/collapse the nodes)
- Try JSON options “-ja”, “-jf” and also the plain format “-p”
- Look out for keys with MISSING values
- View the dump on http://apps.ecmwf.int/codes/bufr/validator/

bufr_ls – list the content of BUFR files

- **Use bufr_ls to list the high-level content (header) of BUFR files**
- **Without options bufr_ls prints a default list of keys**
  - **The default list printed can vary depending on the type of BUFR**
- **Options exist to specify the set of header keys to print**
- **bufr_ls does not fail if a key is not found**
- **Not suitable for viewing array information (data section). Later on we will cover a more powerful tool for this purpose (bufr_filter)**

bufr_ls – usage

```
bufr_ls [options] bufr_file bufr_file …
```

Options

`-p key[:{s|i|d}],…`	**Keys to print**
`-w key[:{s|i|d}]{=|!=}value,…`	**Where clause**

```
…
```

bufr_ls – examples

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

```
> bufr_ls tropical_cyclone.bufr
tropical_cyclone.bufr
centre    masterTablesVersionNumber  localTablesVersionNumber ...  numberOfSubsets     satelliteID
98        16                         0                        ...  52                  0
98        16                         0                        ...  52                  0
98        16                         0                        ...  37                  0
3 of 3 messages in tropical_cyclone.bufr
```

```
> bufr_ls -p centre:s,numberOfSubsets,satelliteID tropical_cyclone.bufr
tropical_cyclone.bufr
centre            numberOfSubsets  satelliteID
ecmf              52               0
ecmf              52               0
ecmf              37               0
3 of 3 messages in tropical_cyclone.bufr
```

bufr_ls – examples

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

```
> bufr_ls -p my_key  file.bufr

file.bufr
my_key
not found

> echo $?

0
```

exit code returned = 0

bufr_ls – using the 'where' option

- **The 'where option' `-w` can be used with several other BUFR tools**
- **Constraints are of the form key=value or key!=value**

```
-w key[:{s|i|d}]=value, key[:{s|i|d}]!=value
```

- **Messages are processed only if they match ALL key/value constraints**
- **Values separated by '/' (forward slash) represent "OR" condition**

```
> bufr_ls -w numberOfSubsets=52 file.bufr
...
> bufr_ls -w typicalDate!=20090124,centre=80/98 file.bufr
...
> bufr_ls -w count=3 file.bufr
```

bufr_get – get key / value pairs

- Use bufr_get to get the values of one or more header keys from one or more BUFR files – very similar to bufr_ls
- By default bufr_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 messages
  - Can force bufr_get not to fail on error
- Format of floating point values can be controlled with a C-style format statement