

BUFR decoding

Dominique Lucas
User Support

Content

- What is BUFR
- BUFR tools
- BUFR format
- BUFR decoding
- Practical examples

What is BUFR

- Binary representation of meteorological data or Binary Universal Form for data Representation.
- Continuous bit stream made of sequence of octets.
- Table driven code.
- Self descriptive code.
- Machine independent.
- Compression available for improved transmission speed.

BUFR tools – data validation

The screenshot shows a Mozilla Firefox browser window displaying the BUFR/CREX format checker page. The browser's address bar shows the URL <http://www.ecmwf.int/products/data/d/check/>. The page header includes the ECMWF logo and navigation links: Home, Your Room, Login, Contact, Feedback, Site Map, and a search box. A menu of categories is visible: About Us, Products, Services, Research, Publications, and News&Events, each with sub-links. The main content area is titled "BUFR/CREX format checker" and explains the tool's purpose: "The purpose of the checker is to verify WMO BUFR/CREX templates used to create data for GTS exchange. The file size is limited to 500 kilobytes." Below this, there is a "Select file to upload" section with a text input field containing the file path `/home/ectrain/trx/bufr_decode/ast145` and a "Browse..." button. A "Validate" button is positioned below the input field. The footer of the page shows the date "17-02-2009" and the ECMWF logo.

<http://old.ecmwf.int/products/data/d/check/>

BUFR tools – data validation

BUFR/CREX format checker - Mozilla Firefox

File Edit View History Bookmarks Tools Help

http://www.ecmwf.int/products/data/d/check/

Most Visited ActivPack Web Help D... Entity Management S... Top Level Display FootPrints

number of messages: 1

File ast145 is valid

Message 1

Section 0

Length of Section 0:	8 byte(s)
Total length of BUFR message:	360 byte(s)
BUFR edition number:	3

Section 1

Length of Section 1:	18 byte(s)
Originating subcentre:	0
Originating centre:	98
Update sequence number:	0
Flag (Presence of section 2):	128
Local table version number:	1
BUFR message type:	4
BUFR message subtype:	145
Year:	3
Month:	3
Day:	2
Hour:	14
Minute:	40

BUFR data examiner - metview4

File Edit View Profiles Help

Key profile: mv System::Default

File: IUSD40_OKLI.bufr

Permissions: -rw-r----- Owner: usl Group: us Size: 8.0KB Modified: 2011-03-07 14:39

Total number of messages: 4

Go to message: 3 Go to subset: 1 (Number of subsets: 1)

Index	△	Typ	Sut	C	Ssc	Date	Time	Lat1
1		2	0	89	1	2007-11-20	18:00	N/A
2		2	0	89	1	2007-11-20	12:00	N/A
3		2	0	89	1	2007-11-20	06:00	N/A
4		2	0	89	1	2007-11-20	00:00	N/A

```
$ metview -e bufr ~trx/bufr_decode/bufr_file
```

Section 0-3 Data Data, bitmaps expanded

Section	Name	Value
Section 0		
	LENGTH OF SECTION 0 (BYTES)	8
	TOTAL LENGTH OF BUFR MESSAGE (BYTES)	1286
	BUFR EDITION NUMBER	3
Section 1		
	LENGTH OF SECTION 1 (BYTES)	18
	BUFR EDITION NUMBER	3
	ORIGINATING SUB-CENTRE	0
	ORIGINATING CENTRE	89
	UPDATE SEQUENCE NUMBER	0
	FLAG (PRESENCE OF SECTION 2)	0
	BUFR MESSAGE TYPE	2
	BUFR MESSAGE SUBTYPE	0
	VERSION NUMBER OF LOCAL TABLE	0
	YEAR	7

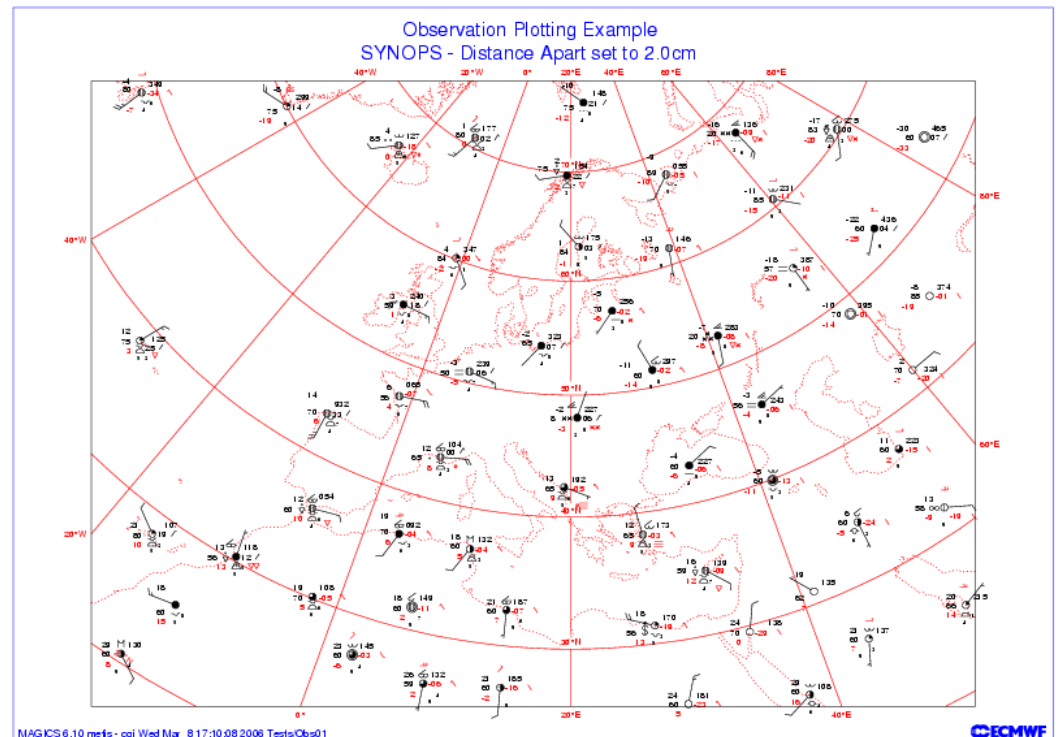
Status: OK

BUFR format

- Indicator section
- Identification section
- Optional section
- Data description section
- Data section
- End section
 - All sections are padded with “0”s if needed to occupy even number of octets.

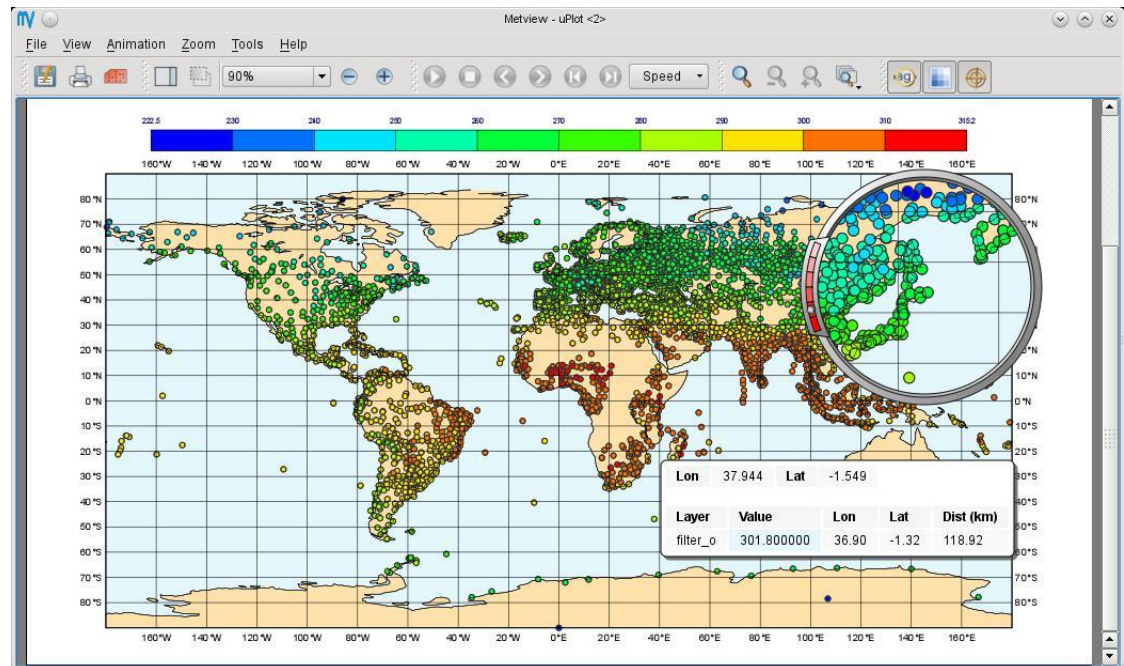
Section 0 - Indication section

- 4 characters 'BUFR'
- Length of message
- Edition Number



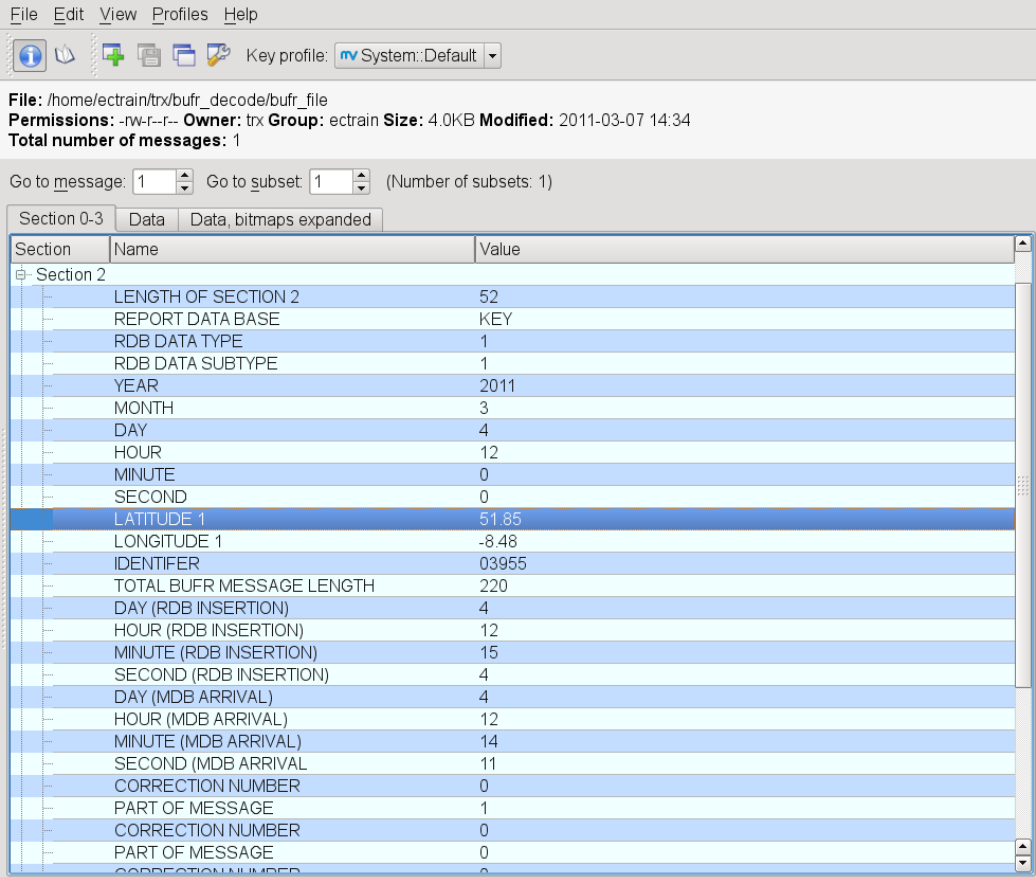
Section 1 - Identification section

- Originating Centre
- Data Category (Table A) and sub-category
- Version number of tables
- Date and time



Section 2 – Optional section

- At ECMWF, section 2 contains data used by MARS.



The screenshot shows a software interface with a menu bar (File, Edit, View, Profiles, Help) and a toolbar. Below the toolbar, the file path is `/home/ectrain/trx/bufr_decode/bufr_file`. The permissions are `-rw-r--r--`, owner is `trx`, group is `ectrain`, size is `4.0KB`, and modified date is `2011-03-07 14:34`. The total number of messages is `1`. The interface also has a 'Go to message' field set to `1` and a 'Go to subset' field set to `1`. The main window displays a table with the following data:

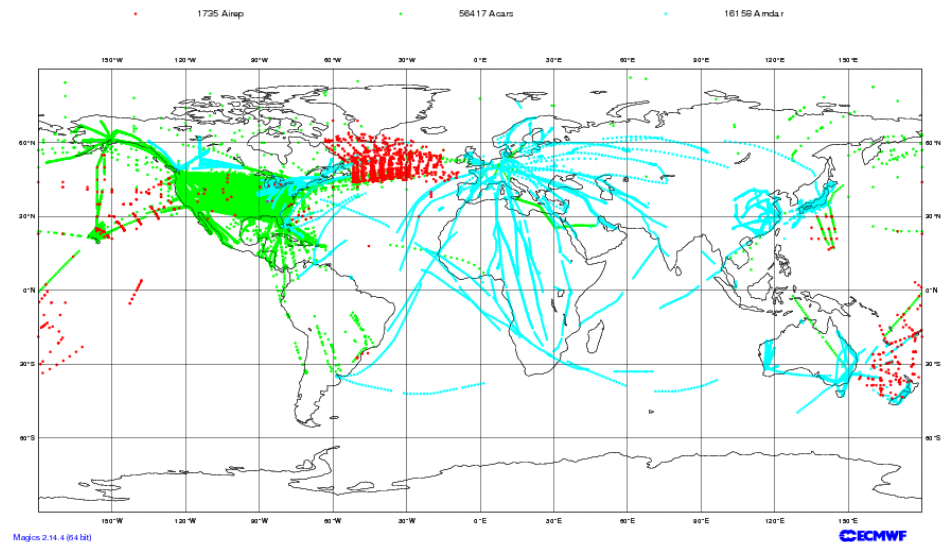
Section	Name	Value
Section 2		
	LENGTH OF SECTION 2	52
	REPORT DATA BASE	KEY
	RDB DATA TYPE	1
	RDB DATA SUBTYPE	1
	YEAR	2011
	MONTH	3
	DAY	4
	HOUR	12
	MINUTE	0
	SECOND	0
	LATITUDE 1	51.85
	LONGITUDE 1	-8.48
	IDENTIFER	03955
	TOTAL BUFR MESSAGE LENGTH	220
	DAY (RDB INSERTION)	4
	HOUR (RDB INSERTION)	12
	MINUTE (RDB INSERTION)	15
	SECOND (RDB INSERTION)	4
	DAY (MDB ARRIVAL)	4
	HOUR (MDB ARRIVAL)	12
	MINUTE (MDB ARRIVAL)	14
	SECOND (MDB ARRIVAL)	11
	CORRECTION NUMBER	0
	PART OF MESSAGE	1
	CORRECTION NUMBER	0
	PART OF MESSAGE	0
	CORRECTION NUMBER	0

Status: OK

Section 3 – Data description section

- Number of data subsets
- Flag for compression
- Data descriptors

ECMWF Data Coverage (All obs DA) - Aircraft
04/Mar/2013; 00 UTC
Total number of obs = 74310



Section 3 – Data descriptors

F type	X category	Y entry
2 bits	6 bits	8 bits

- F = 0 Element Descriptor – Bufr table B
- F = 1 Replication descriptor
 - X = number of descriptors to repeat
 - Y = number of times the descriptors are repeated
- F = 2 Operator Descriptor – “Internal table C”
- F = 3 Sequence Descriptor – Bufr table D

“Table C” - Data Descriptor operators

- 201yyy - Change data width
- 202yyy - Change scale
- 203yyy - Change reference value

- 222000 - Quality information

This table is internal to the BUFR software.
See:

http://www.wmo.int/pages/prog/www/WMOCodes/WMO306_v12/LatestVERSION/LatestVERSION.html

Section 4 - Data section

- Binary data

Section 5 - End section

- 4 digits '7777'

BUFR Tables

- Table A - Data category
- Table B - Classification of elements
- Table C - Code and flag table (*)
- Table D - List of common sequences

http://www.wmo.int/pages/prog/www/WMOCodes/WMO306_vI2/LatestVERSION/LatestVERSION.html

(*) Not to mix with the table C with 'data descriptor operators' mentioned before.

Table A - Data category

- Used in the Section 1 (element 9) of the BUFR message
- Example:

<u>Code figure</u>	<u>Meaning</u>
0	Surface data - land
1	Surface data – sea
2	Vertical soundings (not satellite) ...
31	Oceanographic data

Table B - Classification of elements

Element Name	Unit	Scale	Reference	#bits
• 005001 Latitude (high accuracy)	Degree	5	-9000000	25
• 007003 Geopotential	m**2/s**2	-1	-400	17
• 002019 Satellite instruments	Code table	0	0	11
• 008001 Vertical sounding signifi	Flag Table	0	0	7
• 001006 Aircraft flight number	CCITTIA5	0	0	64
• 011012 Wind speed at 10m	m/s	1	0	12

- (obs. * 10**scale – Reference) is encoded into #bits bits
- For coded or flagged values, the element descriptor indicates the number of the table describing the codes/flags.

- 0 - Table B entry
 - 05 - Location (horizontal 1) class
 - 01 - Identification
 - 08 - Significance qualifiers

Table C – Code and flag tables

- 0 20 003 – Present Weather

<u>Code figure</u>	<u>Meaning</u>
0	Cloud development not observed or not observable
1	Clouds generally dissolving or becoming less developed
...	
10	Mist
11	Patches of shallow fog or ice fog
...	
61	Rain, not freezing, continuous; slight at time of obs.
...	
171	Snow, slight
172	Snow, moderate
173	Snow, heavy
...	
511	Missing

Table D - List of common sequence

- Table D can contain sequences of table B entries, Table D entries and Operators. It is not needed but saves a lot of space.

– 301027	301001	WMO block and station
	002011	Radiosonde type
	002012	Radiosonde computational method
	301011	Date
	301012	Time
	301022	Lat/Long and station height

Bufr software

- PBIO routines
 - PBOPEN - open bufr file for read/write
 - PBBUFR - read bufr message
 - PBWRITE - write bufr message

```
!
!   Open input file for reading.
!
!   CALL PBOPEN (KUNIT, INPUT_FILE, OPEN_MODE, ISTATUS)
!
!   Check return code.
!
!   WRITE ( * , * ) ' '
!   WRITE ( * , * ) 'BFDEMO: After PBOPEN, status code = ', ISTATUS
!   WRITE ( * , * ) ' '
!   IF (ISTATUS .NE. 0) THEN
!       CALL PBCLOSE (KUNIT, ISTATUS)
!       STOP 'BFDEMO: PBOPEN failed.'
!   END IF
!
!   This is the beginning of a loop through BUFR records
!   reading one field at a time from the input datafile.
!
!   LOOP: DO WHILE (.TRUE.)
!
!       WRITE ( * , * ) '*****'
!       WRITE ( * , * ) ' '
!       CALL PBBUFR (KUNIT, KBUFF, JBUFL * NBYTES_SP, KBUFL, ISTATUS)
!       WRITE ( * , * ) 'BFDEMO: After PBBUFR, status code = ', ISTATUS
!
!   END DO
!
```

BUFR decoding - BUFREX

- CALL BUFREX (kbuf1, kbuff, ksup, ksec0, ksec1, ksec2, ksec3, ksec4, kelem, cnames, cunits, kval, values, cvals, kerr)
- Input arguments
 - kbuf1 - length of bufr message in words
 - kbuff - array containing bufr message
 - kelem - expected number of expanded elements
 - kval - size of values array

BUFR decoding - BUFREX

- Output arguments
 - ksup - array containing supplementary information
 - ksec[0-4] - array containing section [0-4] information
 - cnames - character array containing element names
 - cunits - character array containing element units
 - values - real array containing element values
 - cvals - character array containing char. elem. values
 - kerr - return code

BUFR decoding – BUSEL (expanded descriptors)

- CALL BUSEL(KTDLEN,KTDLST,KTDEXL,KTDEXP,KERR)

Output arguments

- KTDLEN - An INTEGER variable containing number of data descriptors in KTDLST array
- KTDLST - An INTEGER array containing the list of KTDLEN data descriptors
- KTDEXL - An INTEGER variable containing number of expanded data descriptors
- KTDEXP - An INTEGER array containing the list of KTDEXL data descriptors
- KERR - An INTEGER containing error

BUFR decoding

- To access character strings from cvals array, say, for element i.

- $\text{index} = \text{values}(i)/1000$

- $\text{string} = \text{cvals}(\text{index})$

- $\text{values}(i) = \text{index} * 1000 + \text{length}$

For example, if values of a character descriptor is 2008, one will have to look at cvals(2) and the string will be 8 characters long.

- To access i-th element in multi subset message from values array

- $\text{index} = i + (\text{nsub}-1) * \text{kelem}$

BUFR Tables

- BUFR Edition 4 (and 3) naming convention
 - VSSSWWWWWWXXXXYYZZZ
 - v - Bufr table (B, C, D)
 - sss - Master table number (000)
 - wwwww - Originating subcentre
 - xxxxx - Originating centre
 - yyy - Version number of master table used
 - zzz - Version number of local table used

e.g. B0000000000098013001.TXT
C0000000000098013001.TXT
D0000000000098013001.TXT

Where to find more about BUFR

- WMO manual on Codes, Volume I, International Codes, Part B - Binary Codes, WMO No.306, FM-94-IX Ext BUFR
- BUFR User Guide and Reference Manual, ECMWF 2008
<https://software.ecmwf.int/wiki/display/BUFR/BUFRDC+Home>
- Guides to WMO Table Driven Code Forms
<http://www.wmo.int/pages/prog/www/WMOCodes.html>
http://www.wmo.int/pages/prog/www/WMOCodes/WMO306_vI2/LatestVERSION/LatestVERSION.html
- Python interface to BUFR available from KNMI
<https://code.google.com/p/pybufr-ecmwf/>

BUFR decoding - the future ...

- ecCodes: rewrite of “BUFREX”
 - More similar to the GRIB API
 - Fortran 90, C and python interface
 - BUFR tools
 - Conversion to json
- To be beta-released shortly

Practical examples – to be run on ecgate

- Copy files across ...

```
$ cd $SCRATCH
```

```
$ tar xvf ~trx/bufr_decode/practicals.tar
```

```
$ cd bufr_decode
```

- Familiarise yourself with the bufr examiner in metview, e.g.

```
$ metview -e BUFR bufr_file
```

- The script “Retrieve_decode_bufr.cmd” decodes a BUFR Synop message. The source code file ‘bfdemo.f90’ is also available together with the Makefile. The file ‘bfdemo.c’ contains C code.

```
$ make bfdemo
```

```
$ ./bfdemo
```

Practical examples – to be run on ecgate

- Can you adapt the above script/code to write part of the observation data out to a file, e.g.

```
! 001001 WMO BLOCK NUMBER
! 001002 WMO STATION NUMBER
! 004001 YEAR
! 004002 MONTH
! 004003 DAY
! 004004 HOUR
! 004005 MINUTE
! 005001 LATITUDE (HIGH ACCURACY)
! 006001 LONGITUDE (HIGH ACCURACY)
! 010004 PRESSURE
! 011011 WIND DIRECTION AT 10 M
! 011012 WIND SPEED AT 10 M
```

An example job is available in Retrieve_extract_bufc.cmd.
The source code file is bfextract.f90, with the Makefile:

```
$ make bfextract
```

```
$ ./bfextract
```

```
$ cat fort.2
```