# csv2bufr

*Release 0.5.1*

## World Meteorological Organization (WMO)

**2023-03-16**

# TABLE OF CONTENTS:

**Author**
World Meteorological Organization (WMO)

**Contact**
https://github.com/wmo-im/csv2bufr

**Release**
0.5.1

**Date**
2023-03-16

CHAPTER

ONE

# OVERVIEW

The csv2bufr Python module contains both a Command Line Interface (CLI) and an API to convert data stored in a CSV file to the WMO BUFR data format. More information on the BUFR format can be found in the WMO Manual on Codes, Volume I.2

CHAPTER

TWO

# INSTALLATION

## 2.1 Dependencies

The csv2bufr module relies on the ecCodes software library to perform the BUFR encoding. This needs to be installed prior to installing any of the Python packages, instructions can be found on the ecCodes documentation pages: https://confluence.ecmwf.int/display/ECC.

The following Python packages are required by the csv2bufr module:

- eccodes (NOTE: this is separate from the ecCodes library)

Additionally, the command line interface to csv2bufr requires:

- click

All the above packages can be installed by running:

```
pip install -r requirements.txt
```

## 2.2 Installation

### 2.2.1 Docker

The quickest way to install and run the software is via a Docker image containing all the required libraries and Python modules:

```
docker pull wmoim/csv2bufr
```

This installs a Docker image based on Ubuntu and includes the ecCodes software library, dependencies noted above and the csv2bufr module (including the command line interface).

### 2.2.2 Source

Alternatively, csv2bufr can be installed from source. First clone the repository and navigate to the cloned folder / directory:

```
git clone https://github.com/wmo-im/csv2bufr.git -b dev
cd csv2bufr
```

If running in a Docker environment, build the Docker image and run the container:

```
docker build -t csv2bufr .
docker run -it -v ${pwd}:/app csv2bufr
cd /app
```

The above step can be skipped if not using Docker. Now install the module and test:

```
python3 setup.py install
csv2bufr --help
```

The following output should be shown:

```
Usage: csv2bufr [OPTIONS] COMMAND [ARGS]...

  csv2bufr

Options:
  --version  Show the version and exit.
  --help     Show this message and exit.

Commands:
  data      data workflows
  mappings  stored mappings
```

CHAPTER

THREE

# QUICK START

The csv2bufr Python module contains both a command line interface and an API to convert data stored in a CSV file to the WMO BUFR data format. For example, the command line interface reads in data from a CSV file, converts it to BUFR and writes out the data to the specified directory. e.g.:

```
csv2bufr data transform <my-csv-file.csv> \
    --bufr-template <csv-to-bufr-mapping.json> \
    --output <output-directory-path>
```

This command is explained in more detail below.

## 3.1 Command line interface

The following example transforms the data in file `my-csv-file.csv` to BUFR using template `csv-to-bufr-mapping.json` and writes the output to directory `output-directory-path`:

```
csv2bufr data transform <my-csv-file.csv> \
    --bufr-template <csv-to-bufr-mapping.json> \
    --output <output-directory-path>
```

The command is built on the Python Click module and is formed of three components (`csv2bufr data transform`), 1 arguments and 2 mandatory options (specified by –). The argument specifies the file to process of the data being processed. The options specify various configuration files to use.

1. `my-csv-file.csv`: argument specifying the CSV data file to process
2. `--bufr-template csv-to-bufr-mapping.json`: option followed by the bufr mapping template to use
3. `--output-dir output-directory-path`: option followed by output directory to write BUFR file to. The output filename is set using the md5 checksum of the BUFR data to ensure uniqueness, future versions will use the WIGOS ID and timestamp of the data to set the filename.

The output BUFR files can be validated using a tool such as the ECMWF BUFR validator.

### 3.1.1 Input CSV file

Currently, a single station per file is supported with each row treated as a separate record and one BUFR file per record created. The format of the input CSV file has a few requirements:

- A comma (i.e. `,`) must be used as the delimiter.
- Strings must be quoted.
- Missing values must be encoded as "None".
- The final row in the file must contain data and not be a new line.
- The timestamp of the records must be separated into components, i.e. year, month, day etc must each be in a separate column.
- The date/time elements should be in Universal Time Coordinated (UTC).
- The file must contain the WIGOS station identifier

### 3.1.2 WIGOS Station Identifier

Each station must have a WIGOS Station Identifier. More information can be found in the Guide to the WMO Integrated Observing System, section 2 (WMO-No. 1165).

### 3.1.3 BUFR mapping template (`--bufr-template`)

The mapping from CSV to BUFR is specified in a JSON file (see the *BUFR template mapping page*).

## 3.2 API

The command line interface uses the `transform` function from the csv2bufr module. This can be used directly, e.g.:

```
# import modules
import json
from csv2bufr import transform

# load data from file
with open("my-csv-file.csv") as fh:
    data = fh.read()

# load mapping
with open("csv-to-bufr-mapping.json") as fh:
    mapping = json.load(fh)

# call transform function
result = transform(data, mapping)

# iterate over items
for item in result:
    # get id and phenomenon time to use in output filename
    wsid = item["_meta"]["wigos_station_identifier"]  # WIGOS station ID
    geometry = item["_meta"]["geometry"]  # GeoJSON geometry object
```

(continues on next page)

(continued from previous page)

```
    timestamp = item["_meta"]["properties"]["datetime"]  # phenomenonTime as datetime␣
↪object
    timestamp = timestamp.strftime("%Y%m%dT%H%MZ")  # convert to string
    # set filename
    output_file = f"{wsid}_{timestamp}.bufr4"
    # save to file
    with open(output_file, "wb") as fh:  # note binary write mode
        fh.write(item["bufr4"])
```

The `transform` function returns an iterator that can be used to iterate over each line in the data file. Each item returned contains a dictionary with the following elements:

- `item["bufr4"]` binary BUFR data
- `item["_meta"]` GeoJSON dictionary containing metadata elements
- `item["_meta"]["id"]` identifier for result (set combination of `wigos_station_identifier` and `datetime`)
- `item["_meta"]["geometry"]` GeoJSON geometry object of location of data
- `item["_meta"]["properties"]` key/value pairs of properties/attributes
- `item["_meta"]["properties"]["md5"]` the md5 checksum of the encoded BUFR data
- `item["_meta"]["properties"]["wigos_station_identifier"]` WIGOS station identifier
- `item["_meta"]["properties"]["datetime"]` characteristic date of data contained in result (from BUFR)
- `item["_meta"]["properties"]["originating_centre"]` originating centre for data (from BUFR)
- `item["_meta"]["properties"]["data_category"]` data category (from BUFR)

CHAPTER

FOUR

# ANATOMY OF A BUFR4 MESSAGE

In order to understand the mapping between a CSV file and its BUFR encoding it is first helpful to understand the anatomy of a BUFR message. A BUFR message is encoded in binary and contains 6 sections as shown in the diagram below. The ecCodes keys for the different elements are also shown as these are used to set certain elements (highlighted in red) as part of the conversion to BUFR. The non-highlighted elements (including those without an ecCodes key) are either set by the eccodes module based on the data, have a default value or can be omitted / set to missing.

The information cotained in Sections 0, 1, and 3 are essentially metadata specifying:

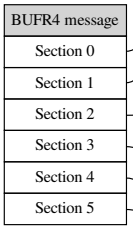
- the version of the BUFR tables used (editionNumber, masterTableNumber, masterTableVersionNumber);
- where the data have come from (bufrHeaderCentre, bufrHeaderSubCentre);
- the typical time of the observation (typicalYear … typicalSecond);
- the type of data and what parameters are included (dataCategory, internationalDataSubCategory and unexpandedDescriptors).

These are all specified in the BUFR template mapping file.

The edition number and master table number should be 4 and 0 respectively. BUFR edition 4 is the latest version and whilst it is possible to define new BUFR tables for different application areas only Master Table 0 (MT0) has been defined (meteorology). The tables in MT0 are updated approximately every 6 months as part of the WMO fast track process and the authoritative source can be found in the WMO Manual on Codes, Volume I.2. Updates to this can be delayed and the latest version, including machine readable tables, can be found at https://github.com/wmo-im/BUFR4/releases and it is recommended to use the the most recent release. The `master` branch contains the current working copy of the tables and is subject to change.

The BUFR header centre and bufr header sub centre are specified in Common Code Tables C-11 and C-12 respectively. The typical time of observation (typicalYear … typicalSecond) should be determined based on the data to be encoded. Within the csv2bufr module and CLI only a single observation / weather report is encoded per file and so these should be set to those columns in the CSV specifying the year, month, day etc. More information is provided in the page on the BUFR template mapping (link to follow).

The data category should be set according to BUFR Table A, i.e. 0 for “Surface data - land” and 1 for “Surface data - sea”. The international data sub category should be set according to Common Code Table C-13. The unexpanded descriptors specifies the data to be encoded in the data section is comprised of a list of the BUFR descriptors. These descriptors are detailed further on the next page (BUFR descriptors).




| Section 0 – Indicator section | | |
|---|---|---|
| ecCodes key | Length (bytes) | Description |
| | 4 | The string "BUFR" in CCITT-IA5 |
| totalLength | 3 | Total length of the BUFR message in bytes |
| editionNumber | 1 | BUFR edition number (4) |

| Section 1 – Identification section | | |
|---|---|---|
| ecCodes key | Length (bytes) | Description |
| section1Length | 3 | Length of section 1 |
| masterTableNumber | 1 | BUFR master table user (MT0) |
| bufrHeaderCentre | 2 | Identification of originating/generating centre (see Common Code Table C–11) |
| bufrHeaderSubCentre | 2 | Identification of originating/generating sub-centre (allocated by originating/generating centre – see Common Code Table C–12) |
| updateSequenceNumber | 1 | Update sequence number (zero for original messages and for messages containing only delayed reports; incremented for the other updates) |
| section1Flags | 1 | Flag indicating presence of optional section (section 2) |
| dataCategory | 1 | Data category (Table A) |
| internationalDataSubCategory | 1 | International data sub-category (see Common Code Table C–13 and Note 3) |
| dataSubCategory | 1 | Local data sub-category (defined locally by automatic data-processing (ADP) centres – see Note 3) |
| masterTablesVersionNumber | 1 | BUFR master table version number (see Common Code Table C–0 and Note 2) |
| localTablesVersionNumber | 1 | Version number of local tables used to augment master table in use – see Note 2 |
| typicalYear | 2 | Typical year of BUFR message contents |
| typicalMonth | 1 | Typical month of BUFR message contents |
| typicalDay | 1 | Typical day of BUFR message contents |
| typicalHour | 1 | Typical hour of BUFR message contents |
| typicalMinute | 1 | Typical minute of BUFR message contents |
| typicalSecond | 1 | Typical second of BUFR message contents |
| | 0+ | Optional data – for local use by ADP centres |

| Section 2 – Optional section | | |
|---|---|---|
| ecCodes key | Length (bytes) | Description |
| | 3 | Length of section if present |
| | 1 | Set to zero |
| | 0+ | Reserved for local use by ADP centres |

| Section 3 – Data description section | | |
|---|---|---|
| ecCodes key | Length (bytes) | Description |
| section3Length | 3 | length of section |
| | 1 | Set to zero |
| numberOfSubsets | 2 | Number of data subsets |
| section3Flags | 1 | Flags used to indicate compressed and observed data |
| unexpandedDescriptors | 2+ | List of data descriptors describing data contained in section 4 e.g. [301150, 307080]) |

| Section 4– Data section | | |
|---|---|---|
| ecCodes key | Length (bytes) | Description |
| ecCodes key | Length (bytes) | Description |
| | 3 | Length of section 4 |
| | 1 | Set to zero |
| | | Binary string of encoded data. |

| Section 5 – End section | | |
|---|---|---|
| ecCodes key | Length (bytes) | Description |
| | 4 | The string "7777" encode in CCITT IA-5 |

CHAPTER

FIVE

# BUFR4 DESCRIPTORS

## 5.1 Introduction

As part of the BUFR format, a list of the included parameters is embedded within the file using the unexpandedDescriptors element of Section 3. This element takes a list of the parameters to be included and the order of those parameters. For example, the unexpanded descriptor list (in text, see format of descriptors below):

```
["unexpandedDescriptors"] = ["wigosID", "year", "month", "day", "hour", "minute", \
                             "latitude", "longitude", "pressure reduced to mean sea␣
↪level"]
```

would specify that the station identifier (wigosID) followed by the date, time, location and then the pressure reduced to mean sea level would be included in the data section. For conciseness, aliases (or sequences in BUFR terminology) exist to group commonly reported parameters together, for example grouping the year, month and day together in the group date. Using sequences, the above example becomes:

```
["unexpandedDescriptors"] = ["wigosID", "date", "time", "location", "pressure reduced to␣
↪mean sea level"]
```

where

```
["date"] = ["year", "month", "day"]
["time"] = ["hour", "minute"]
["location"] = ["latitude", "longitude"]
```

These sequences form building blocks that can be used to create much longer sequences using a small number of descriptors.

## 5.2 Format of BUFR descriptors

Whilst text strings have been used above to represent the parameters to report this has been done for ease of reading and explanation. Within the BUFR format these are specified by 6 digit codes of the form FXXYYY, with each of F, XX and YYY having specific meaning.

- F: Type of BUFR descriptor
  - 0: element descriptor (BUFR Table B);
  - 1: replication (or repetition) descriptor;
  - 2: operator (BUFR Table C);
  - 3: sequence descriptor (BUFR Table D)

- XX: Sub-table (class) within type of descriptor
- YYY: Index within in sub-table.

As an example, the table below shows the first few entries from BUFR Table B 01.

For the above example, using the FXXYYY notation, the list of abbreviated (or unexpanded) descriptors becomes:

```
["unexpandedDescriptors"] = [301150, 301011, 301012, 301023, 010051]
```

where:

```
[301150] = [001125, 001126, 001127, 001128] # (wigosID)
         = ["WIGOS identifer series", "WIGOS identifier issuer", "WIGOS identifier issue␣
↪number", "WIGOS local identifier"]
[301011] = [004001,  004002, 004003] # (date)
         = ["year", "month",  "day"]
[301012] = [004004,   004005] # (time)
         = ["hour", "minute"]
[301023] = [    005002,      006002] # (location)
         = ["latitude", "longitude"]
[010051] = ["pressure reduced to mean sea level"]
```

## 5.3 Replication / repetition

Within the BUFR format, elements can be repeated using the replication descriptors (F=1 in FXXYYY). For example, we may want to repeat temperature and humidity measurements as part of an atmospheric profile or, alternatively, the daily minimum and maximum temperatures within a month. In some cases we may know the number of repetitions before encoding and all data of the same type may have the same number of repetitions. In this case the number of replications can be set before hand and included in the sequence. In other cases, there may be a variable number of repetitions and so the number is set at the time of encoding.

When using the replication descriptor (1XXYYY), the XX component indicates the number of following descriptors to repeat and the YYY component the number of replications. For example, to repeat the day of month, maximum and minimum temperatures 5 times we would use:

$$[\text{unexpandedDescriptors}] = \left[ \overbrace{\underbrace{103005}_{FXXYYY}}^{\text{repeat (1) next 3 (03) descriptors 5 (005) times}}, \overbrace{004003, 012016, 012017}^{\text{descriptors to be repeated}} \right]$$

In expanded form, or without using the replication, this would be equivalent to:

$$\begin{aligned} [\text{expandedDescriptors}] = [ \quad & 004003, 012016, 012017, && (5.1)\\ & 004003, 012016, 012017, && (5.2)\\ & 004003, 012016, 012017, && (5.3)\\ & 004003, 012016, 012017, && (5.4)\\ & 004003, 012016, 012017] && (5.5) \end{aligned}$$

If we do not known the number of replications, or there can be a variable number of replications, for a given sequence of descriptors we can set the YYY element (number of replications) to zero and follow the replication descriptor with a delayed replication factor.

$$[\text{unexpandedDescriptors}] = \left[ \underbrace{\overbrace{103000}^{\text{(repeat (1) next 3 (03) items n time)}}, \overbrace{031001}^{\text{(delayed number (n) of replications)}}}_{\text{replication and delayed replication factor}}, \overbrace{004003, 012016, 012017}^{\text{(descriptors to be repeated)}} \right]$$

This works in the same way as the regular replication except that the number of replications (n) is set at the the time of encoding and included in data. Often, within sequences, delayed descriptors are used to specify optional elements using the short delayed descriptor replication factor (031000) that takes a value of either 0 or 1.

Within the csv2bufr module the number of delayed replications needs to be set within the mapping file using the inputDelayedDescriptorReplicationFactor key. More information is provided on the mappings page.

## 5.4 Scope of descriptors

BUFR Table B descriptors within classes 0 - 8 contain metadata about the observations. For example, the location of an observations, the instrumentation used to make an observation or the time period over which an observation was made or averaged. These descriptors remain in force and apply to all subsequent elements until they are either reused or set to missing. For example the sequence `[007032, 012001, 007032, 012001, 007032]` could be used to record air temperature measurements at two different heights, e.g.:

```
# set height of sensor for following observations
[007032] = ["heightOfSensorAboveLocalGroundOrDeckOfMarinePlatform"] = 2.0
[012001] = ["airTemperature"] = 280.15 # air temperature at 2 m height
# redefine height of sensor to 10 m
[007032] = ["heightOfSensorAboveLocalGroundOrDeckOfMarinePlatform"] = 10.0
[012001] = ["airTemperature"] = 280.07 # air temperature at 10 m height
# cancel height of sensor, following observations will have an undefined height
[007032] = ["heightOfSensorAboveLocalGroundOrDeckOfMarinePlatform"] = None
```

## 5.5 Commonly used sequences

Listed below are some commonly used sequences:

- 307080: Sequence for representation of synoptic reports from a fixed land station suitable for SYNOP data.
- 315008: Sequence for the representation of data from moored buoys.
- 315009: Sequence for the representation of data from drifting buoys.
- As this documentation is developed further additional examples will be added.

## 5.6 Further information

The description of the BUFR operators (F = 2 in the FXXYYY notation) is beyond the scope of this documentation. For users wanting to define new sequences, including the use of the operators, it is recommended to refer to Volume I.2 of the WMO Manual on Codes. However, before defining a new sequence it is recommended to check if any of the existing sequence meet the user requirements. See the *support page* for information on how to get further information and support..

CHAPTER

SIX

# BUFR TEMPLATE MAPPING

The mapping between the input CSV data and the output BUFR data is specified in a JSON file. The csv2bufr module validates the mapping file against the schema shown at the bottom of this page prior to attempted the transformation to BUFR. This schema specifies 7 primary properties all of which are mandatory:

- `inputDelayedDescriptorReplicationFactor` - array of integers, values for the delayed descriptor replication factors to use
- `inputShortDelayedDescriptorReplicationFactor` - array of integers, values for the short delayed descriptor replication factors to use
- `inputExtendedDelayedDescriptorReplicationFactor` - array of integers, values for the extended delayed descriptor replication factors to use
- `number_header_rows` - integer, the number of header rows in the file before the first data, including the row with column names.
- `column_names_row` - integer, the row number that gives the column names.
- `wigos_station_identifier` - either constant WIGOS station identifier (e.g. `const:0-20000-0-123`) or column from csv data file containing the WSI (e.g. `data:WSI_column`).
- `header` - array of objects (see below), header section containing metadata
- `data` - array of object (see below) section mapping from the CSV columns to the BUFR elements

The header and data sections contain arrays of `bufr_element` objects mapping to either the different fields in the header sections of the BUFR message or to the data section respectively. More information is provided below. In both cases the field `eccodes_key` from the `bufr_element` object is used to indicate the BUFR element mapped rather than the 6 digit BUFR FXXYYY code. The field `value` specifies where the data to encode comes from. This can be one of the following:

- data: this specifies that the data should come from the data file.
- const: this specifies that a constant value should be used

For example, the code block below shows how the pressure reduced to mean sea level would be mapped from the column "mslp" in the CSV file to the BUFR element indicated by the eccodes key "pressureReducedToMeanSeaLevel" (FXXYYY = 010051).

```
{
    "data":[
        {"eccodes_key": "pressureReducedToMeanSeaLevel", "value": "data:mslp"}
    ]
}
```

The code block below gives examples for both constant values and a value read from the data file:

```
{
    "header":[
        {"eccodes_key": "dataCategory", "value": "const:0"}
    ],
    "data":[
        {"eccodes_key": "latitude", "value": "const:46.2234923"},
        {"eccodes_key": "longitude", "value": "const:6.1475485"},
        {"eccodes_key": "pressureReducedToMeanSeaLevel", "value": "data:mslp"},
    ]
}
```

In this example, the `dataCategory` field in BUFR section 1 (see *Anatomy of a BUFR4 message*) is mapped to the constant value 0; the `latitude` and `longitude` to the value specified; and the `pressureReducedToMeanSeaLevel` to the data from the "mslp" column in the CSV file.

The keys used for the header elements are listed on the *Anatomy of a BUFR4 message* page, with the mandatory keys highlighted in red. The list of keys can also be found at:

- https://confluence.ecmwf.int/display/ECC/BUFR+headers

Similarly the keys for the different data elements can be found at:

- https://confluence.ecmwf.int/display/ECC/WMO%3D37+element+table

## 6.1 input<Short|Extended>DelayedDescriptorReplicationFactor

Due to the way that eccodes works any delayed replication factors need to be specified before encoding and included in the mapping file. This currently limits the use of the delayed replication factors to static values for a given mapping. For example every data file that uses a given mapping file has the same optional elements present or the same number of levels in an atmospheric profile present.

For sequences that do not include delayed replications the inputDelayedDescriptorReplicationFactor etc must still be included but may be set to an empty array. e.g.

```
{
    "inputDelayedDescriptorReplicationFactor": [],
    "inputShortDelayedDescriptorReplicationFactor": []
    "inputExtendedDelayedDescriptorReplicationFactor": []
}
```

## 6.2 bufr_element

Each item in the `header` and `data` arrays of the mapping template must conform with the definition of the `bufr_element` object specified in the schema shown below. This object contains an `eccodes_key` field specifying the BUFR element the data are being mapped to as described above and up to 3 others pieces of information:

- the source of the data (`value`)
- valid range information (`valid_min`, `valid_max`)
- simple scaling and offset parameters (`scale`, `offset`)

Only one source can be mapped, if multiple sources are specified the validation of the mapping file by csv2bufr will fail. As noted at the start of this page. the `value` field maps the data to one of:

- data: this specifies that the data should come from the data file
- const: this specifies that a constant value should be used

and takes the form `"value": "<keyword>:<column|value>"` where `<keyword>` is the string `data` or `const`. `<column|value>` can specify either the column name from the data file or it can specify a constant value to use.

The `valid_min` and `valid_max` are optional and can be used to perform a basic quality control of numeric fields. The values to use are specified in the same way as for the `value` element, with the values coming from either a constant value or from the data file. If these fields are specified the csv2bufr module checks the value extracted from the source to the indicated valid minimum and maximum values. If outside of the range the value is set to missing. This check is applied before any scaling of the data.

The `scale` and `offset` fields are conditionally optional, either both can be omitted or both can included. Including only one will result in a failed validation of the mapping file. These allow simple unit conversions to be performed, for example from degrees Celsius to Kelvin or from hectopascals to Pascals. Again, the values are specified in the same way as for the other fields. The scaled values are calculated as:

$$\text{scaled\_value} = \text{value} \times 10^{\text{scale}} + \text{offset}$$

The scaled value is then used to set the indicated BUFR element. For example:

```
{
    "data":[
        {
            "eccodes_key": "pressureReducedToMeanSeaLevel",
            "value": "data:mslp",
            "scale": "const:2",
            "offset": "const:0"
        }
    ]
}
```

Would convert the value contained in the "mslp" column of the CSV file from hPa to Pa by multiplying by 100 and adding 0.

For each of the above elements (`value, valid_min, valid_max, scale, offset`) null values must be excluded from the mapping file.

An individual BUFR descriptor can occur multiple times within a single BUFR message. To allow the indexing of the descriptors within a particular message, and the inclusions of multiple descriptors or keys with the same name, eccodes prepends an index number to the eccodes_key. For the first occurrence the index number can be omitted but for all other cases it should be included. The index is indicated within the eccodes_key using `#index#eccodes_key`, an example is given below.

```
{
    "data":[
        {
            "#1#eccodes_key": "pressureReducedToMeanSeaLevel",
            "csv_column": "data:mslp",
            "scale": "const:2",
            "offset": "const:0"
        }
    ]
}
```

## 6.3 Units

It should be noted that the units of the data to be encoded into BUFR should match those specified in BUFR table B (e.g. see https://confluence.ecmwf.int/display/ECC/WMO%3D37+element+table), i.e. Kelvin for temperatures, Pascals for pressure etc. Simple conversions between units are possible as specified above using the `scale` and `offset` fields. Some additional examples are given below.

```
{
    "data":[
        {
            "eccodes_key": "airTemperature",
            "value": "data:AT-fahrenheiht",
            "scale": "const:-0.25527",
            "offset": "const:459.67"
        },
        {
            "eccodes_key": "airTemperature",
            "value": "data:AT-celsius",
            "scale": "const:0",
            "offset": "const:273.15"
        },
        {
            "eccodes_key": "pressure",
            "value": "data:pressure-hPa",
            "scale": "const:2",
            "offset": "const:0"
        }
    ]
}
```

## 6.4 Schema

```
{
    "$id": "csv2bufr.wis2.0.node.wis",
    "$schema": "https://json-schema.org/draft/2020-12/schema",
    "type": "object",
    "properties": {
        "inputShortDelayedDescriptorReplicationFactor": {
            "type": "array",
            "items": {"type": "integer"}
        },
        "inputDelayedDescriptorReplicationFactor": {
            "type": "array",
            "items": {"type": "integer"}
        },
        "inputExtendedDelayedDescriptorReplicationFactor": {
            "type": "array",
            "items": {"type": "integer"}
        },
        "number_header_rows": {
```

(continues on next page)

(continued from previous page)

```
            "type": "integer",
            "description": "Number of header rows in file before the data"
        },
        "column_names_row": {
            "type": "integer",
            "description": "Which header line the column names is given on"

        },
        "wigos_station_identifier": {
            "type": "string",
            "description": "Either the WIGOS station identifier for the data or the␣
↪column in the CSV file containing the identifier"
        },
        "delimiter": {
            "type": "string",
            "description": "The delimiter used to separate fields in the input csv file,␣
↪must be one of ',', ';'. '|' or [tab]"
        },
        "quoting": {
            "type": "string",
            "description": "CSV quoting method to use, must be one of QUOTE_NONNUMERIC,␣
↪QUOTE_ALL, QUOTE_MINIMAL or QUOTE_NONE"
        },
        "quotechar": {
            "type": "string",
            "description": "quote character to use, e.g. \", ' etc"
        },
        "header":{
            "type": "array",
            "items": {"$ref": "#/$defs/bufr_element"},
            "description": "Contents of header sections of BUFR message"
        },
        "data": {
            "type": "array",
            "items": {"$ref": "#/$defs/bufr_element"},
            "description": "mapping from CSV file (or metadata json file) to BUFR"
        }
    },
    "required" : [
        "inputShortDelayedDescriptorReplicationFactor",
        "inputDelayedDescriptorReplicationFactor",
        "inputExtendedDelayedDescriptorReplicationFactor",
        "column_names_row","number_header_rows","header","data"],

    "$defs":{
        "bufr_element": {
            "type": "object",
            "properties": {
                "eccodes_key": {
                    "type": "string",
                    "descripition": "eccodes key used to set the value in the BUFR data"
                },
```

(continues on next page)

(continued from previous page)

```
                "value": {
                    "type": [
                        "string"
                    ],
                    "description": "where to extract the value from, can be one off 'data
↪','metadata','const','array' followed by the value or column header"
                },
                "valid_min": {
                    "type": "string",
                    "description": "Minimum valid value for parameter if set"
                },
                "valid_max": {
                    "type": "string",
                    "description": "Maximum value for for the parameter if set"
                },
                "scale": {
                    "type": "string",
                    "description": "Value used to scale the data by before encoding␣
↪using the same conventions as in BUFR"
                },
                "offset": {
                    "type": "string",
                    "description": "Value added to the data before encoding to BUFR␣
↪following the same conventions as BUFR"
                }
            },
            "required": ["eccodes_key", "value"],
            "allOf": [
                {
                    "dependentRequired": {"scale": ["offset"]}
                },
                {
                    "dependentRequired": {"offset": ["scale"]}
                }
            ]
        }
    }
}
```

CHAPTER

SEVEN

# EXAMPLES

This page follows through a worked example:

1. Example data file
2. Creating a new mapping file
3. Editing the mapping file
4. Running the transformation

All the example files used are downloadable at the end of this page.

## 7.1 Data file (example-data.csv)

Table 1: example-data.csv

| wsi | year | month | day | hour | minute | latitude | longitude | slp | mslp | ppp | a | brmh |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0-20000-0-06700 | 2022 | 02 | 10 | 6 | 0 | 46.2475 | 6.12774 | 978.3 | 1029.90 | -0.4 | 8 | 412.3 |

## 7.2 Creating a new mapping file

A command line tool to create an empty BUFR mapping template has been included as part of the csv2bufr module. This can be invoked using the `csv2bufr mappings create <BUFR descriptors>` command. E.g.:

```
csv2bufr mappings create 301150 301011 301012 301021 007031 302001 --output bufr-
↪mappings.json
```

generates the following file:

```
{
    "inputDelayedDescriptorReplicationFactor": [],
    "inputShortDelayedDescriptorReplicationFactor": [],
    "inputExtendedDelayedDescriptorReplicationFactor": [],
    "wigos_station_identifier": "data:wsi",
    "number_header_rows": 1,
    "column_names_row": 1,
    "header": [
        {
```

(continues on next page)

(continued from previous page)


```
            "eccodes_key": "edition",
            "value": ""
        },
        {
            "eccodes_key": "masterTableNumber",
            "value": ""
        },
        {
            "eccodes_key": "bufrHeaderCentre",
            "value": ""
        },
        {
            "eccodes_key": "bufrHeaderSubCentre",
            "value": ""
        },
        {
            "eccodes_key": "updateSequenceNumber",
            "value": ""
        },
        {
            "eccodes_key": "dataCategory",
            "value": ""
        },
        {
            "eccodes_key": "internationalDataSubCategory",
            "value": ""
        },
        {
            "eccodes_key": "dataSubCategory",
            "value": ""
        },
        {
            "eccodes_key": "masterTablesVersionNumber",
            "value": ""
        },
        {
            "eccodes_key": "localTablesVersionNumber",
            "value": ""
        },
        {
            "eccodes_key": "typicalYear",
            "value": ""
        },
        {
            "eccodes_key": "typicalMonth",
            "value": ""
        },
        {
            "eccodes_key": "typicalDay",
            "value": ""
        },
        {
```


(continues on next page)

(continued from previous page)


```
            "eccodes_key": "typicalHour",
            "value": ""
        },
        {
            "eccodes_key": "typicalMinute",
            "value": ""
        },
        {
            "eccodes_key": "typicalSecond",
            "value": ""
        },
        {
            "eccodes_key": "typicalDate",
            "value": ""
        },
        {
            "eccodes_key": "typicalTime",
            "value": ""
        },
        {
            "eccodes_key": "numberOfSubsets",
            "value": ""
        },
        {
            "eccodes_key": "observedData",
            "value": ""
        },
        {
            "eccodes_key": "compressedData",
            "value": ""
        },
        {
            "eccodes_key": "unexpandedDescriptors",
            "value": "array:301150,301011,301012,301021,7031,302001"
        },
        {
            "eccodes_key": "subsetNumber",
            "value": ""
        }
    ],
    "data": [
        {
            "eccodes_key": "#1#wigosIdentifierSeries",
            "value": "",
            "valid_min": "const:0",
            "valid_max": "const:14",
            "scale": "const:0",
            "offset": "const:0"
        },
        {
            "eccodes_key": "#1#wigosIssuerOfIdentifier",
            "value": "",
```


(continues on next page)

(continued from previous page)

```
            "valid_min": "const:0",
            "valid_max": "const:65534",
            "scale": "const:0",
            "offset": "const:0"
        },
        {
            "eccodes_key": "#1#wigosIssueNumber",
            "value": "",
            "valid_min": "const:0",
            "valid_max": "const:65534",
            "scale": "const:0",
            "offset": "const:0"
        },
        {
            "eccodes_key": "#1#wigosLocalIdentifierCharacter",
            "value": ""
        },
        {
            "eccodes_key": "#1#year",
            "value": "",
            "valid_min": "const:0",
            "valid_max": "const:4094",
            "scale": "const:0",
            "offset": "const:0"
        },
        {
            "eccodes_key": "#1#month",
            "value": "",
            "valid_min": "const:0",
            "valid_max": "const:14",
            "scale": "const:0",
            "offset": "const:0"
        },
        {
            "eccodes_key": "#1#day",
            "value": "",
            "valid_min": "const:0",
            "valid_max": "const:62",
            "scale": "const:0",
            "offset": "const:0"
        },
        {
            "eccodes_key": "#1#hour",
            "value": "",
            "valid_min": "const:0",
            "valid_max": "const:30",
            "scale": "const:0",
            "offset": "const:0"
        },
        {
            "eccodes_key": "#1#minute",
            "value": "",
```

(continues on next page)

(continued from previous page)

```
            "valid_min": "const:0",
            "valid_max": "const:62",
            "scale": "const:0",
            "offset": "const:0"
        },
        {
            "eccodes_key": "#1#latitude",
            "value": "",
            "valid_min": "const:-90.0",
            "valid_max": "const:245.5443",
            "scale": "const:0",
            "offset": "const:0"
        },
        {
            "eccodes_key": "#1#longitude",
            "value": "",
            "valid_min": "const:-180.0",
            "valid_max": "const:491.08862",
            "scale": "const:0",
            "offset": "const:0"
        },
        {
            "eccodes_key": "#1#heightOfBarometerAboveMeanSeaLevel",
            "value": "",
            "valid_min": "const:-400.0",
            "valid_max": "const:12707.0",
            "scale": "const:0",
            "offset": "const:0"
        },
        {
            "eccodes_key": "#1#nonCoordinatePressure",
            "value": "",
            "valid_min": "const:0",
            "valid_max": "const:163820",
            "scale": "const:0",
            "offset": "const:0"
        },
        {
            "eccodes_key": "#1#pressureReducedToMeanSeaLevel",
            "value": "",
            "valid_min": "const:0",
            "valid_max": "const:163820",
            "scale": "const:0",
            "offset": "const:0"
        },
        {
            "eccodes_key": "#1#3HourPressureChange",
            "value": "",
            "valid_min": "const:-5000",
            "valid_max": "const:5220",
            "scale": "const:0",
            "offset": "const:0"
```

(continues on next page)

(continued from previous page)

```
        },
        {
            "eccodes_key": "#1#characteristicOfPressureTendency",
            "value": "",
            "valid_min": "const:0",
            "valid_max": "const:14",
            "scale": "const:0",
            "offset": "const:0"
        }
    ]
}
```

This file includes the representable range (`valid_min` and `valid_max`) for the different BUFR elements. These should be set to the physical range where applicable.

## 7.3 Customising the mapping file (bufr-mappings-edited.json)

Editing the bufr mappings file to map to the above example CSV data we have:

```
{
    "inputDelayedDescriptorReplicationFactor": [],
    "inputShortDelayedDescriptorReplicationFactor": [],
    "inputExtendedDelayedDescriptorReplicationFactor": [],
    "wigos_station_identifier": "data:wsi",
    "number_header_rows": 1,
    "column_names_row": 1,
    "header": [
        {
            "eccodes_key": "edition",
            "value": "const:4"
        },
        {
            "eccodes_key": "masterTableNumber",
            "value": "const:0"
        },
        {
            "eccodes_key": "updateSequenceNumber",
            "value": "const:0"
        },
        {
            "eccodes_key": "dataCategory",
            "value": "const:0"
        },
        {
            "eccodes_key": "internationalDataSubCategory",
            "value": "const:6"
        },
        {
            "eccodes_key": "masterTablesVersionNumber",
            "value": "const:36"
```

(continues on next page)

(continued from previous page)


```
        },
        {
            "eccodes_key": "typicalYear",
            "value": "data:year"
        },
        {
            "eccodes_key": "typicalMonth",
            "value": "data:month"
        },
        {
            "eccodes_key": "typicalDay",
            "value": "data:day"
        },
        {
            "eccodes_key": "typicalHour",
            "value": "data:hour"
        },
        {
            "eccodes_key": "typicalMinute",
            "value": "data:minute"
        },
        {
            "eccodes_key": "numberOfSubsets",
            "value": "const:1"
        },
        {
            "eccodes_key": "observedData",
            "value": "const:1"
        },
        {
            "eccodes_key": "compressedData",
            "value": "const:0"
        },
        {
            "eccodes_key": "unexpandedDescriptors",
            "value": "array: 301150, 301011, 301012, 301021, 7031, 302001"
        }
    ],
    "data": [
        {
            "eccodes_key": "#1#wigosIdentifierSeries",
            "value": "metadata:wsi_series",
            "valid_min": "const:0",
            "valid_max": "const:0"
        },
        {
            "eccodes_key": "#1#wigosIssuerOfIdentifier",
            "value": "metadata:wsi_issuer",
            "valid_min": "const:0",
            "valid_max": "const:65534"
        },
        {
```


(continues on next page)

(continued from previous page)

```
            "eccodes_key": "#1#wigosIssueNumber",
            "value": "metadata:wsi_issue_number",
            "valid_min": "const:0",
            "valid_max": "const:65534"
        },
        {
            "eccodes_key": "#1#wigosLocalIdentifierCharacter",
            "value": "metadata:wsi_local"
        },
        {
            "eccodes_key": "#1#year",
            "value": "data:year",
            "valid_min": "const:2022",
            "valid_max": "const:2025"
        },
        {
            "eccodes_key": "#1#month",
            "value": "data:month",
            "valid_min": "const:1",
            "valid_max": "const:12"
        },
        {
            "eccodes_key": "#1#day",
            "value": "data:day",
            "valid_min": "const:1",
            "valid_max": "const:31"
        },
        {
            "eccodes_key": "#1#hour",
            "value": "data:hour",
            "valid_min": "const:0",
            "valid_max": "const:23"
        },
        {
            "eccodes_key": "#1#minute",
            "value": "data:minute",
            "valid_min": "const:0",
            "valid_max": "const:59"
        },
        {
            "eccodes_key": "#1#latitude",
            "value": "data:latitude",
            "valid_min": "const:-90.0",
            "valid_max": "const:90.0"
        },
        {
            "eccodes_key": "#1#longitude",
            "value": "data:longitude",
            "valid_min": "const:-180.0",
            "valid_max": "const:180.0"
        },
        {
```

(continues on next page)

(continued from previous page)

```
            "eccodes_key": "#1#heightOfBarometerAboveMeanSeaLevel",
            "value": "data:brmh",
            "valid_min": "const:-400.0",
            "valid_max": "const:12707.0"
        },
        {
            "eccodes_key": "#1#nonCoordinatePressure",
            "value": "data:slp",
            "valid_min": "const:0",
            "valid_max": "const:163820",
            "scale": "const:2",
            "offset": "const:0"
        },
        {
            "eccodes_key": "#1#pressureReducedToMeanSeaLevel",
            "value": "data:mslp",
            "valid_min": "const:0",
            "valid_max": "const:163820",
            "scale": "const:2",
            "offset": "const:0"
        },
        {
            "eccodes_key": "#1#3HourPressureChange",
            "value": "data:ppp",
            "valid_min": "const:-5000",
            "valid_max": "const:5220",
            "scale": "const:2",
            "offset": "const:0"
        },
        {
            "eccodes_key": "#1#characteristicOfPressureTendency",
            "value": "data:a",
            "valid_min": "const:0",
            "valid_max": "const:14"
        }
    ]
}
```

Note that the sequence includes no delayed replications and so the `inputDelayedDescriptorReplicationFactor` etc can be left as empty arrays. Elements that would be set to null have been removed.

## 7.4 Transformation

```
csv2bufr data transform \
                ./example-data.csv \
                --bufr-template ./bufr-mappings-edited.json \
                --output-dir ./
```

The links below can be used to download the example files:

- `example-data.csv`

- bufr-mappings-edited.json
- example output (d0464c97a88ea99f119e87629844c5dd.bufr4)

CHAPTER

EIGHT

# PYGEOAPI PLUGIN

## 8.1 Overview

csv2bufr also provides a custom [pygeoapi](https://pygeoapi.io) processing plugin, providing csv2bufr functionality via OGC API - Processes.

## 8.2 Installation

To integrate this plugin in pygeoapi:

- ensure csv2bufr and its dependencies are installed into the pygeoapi deployment environment
- add the processes to the pygeoapi configuration as follows:

```
csv2bufr-transform:
    type: process
    processor:
        name: csv2bufr.pygeoapi_plugin.csv2bufrProcessor
```

- regenerate the pygeoapi OpenAPI configuration

```
pygeoapi openapi generate $PYGEOAPI_CONFIG --output-file $PYGEOAPI_OPENAPI
```

- restart pygeoapi

## 8.3 Usage

The resulting processes will be available at the following endpoints:

- `/processes/csv2bufr-transform`

Note that pygeoapi's OpenAPI/Swagger interface (at `/openapi`) also provides a developer-friendly interface to test and run requests

CHAPTER

NINE

# DEVELOPMENT

See discussion board and issues on GitHub:

- Discussion board
- Issues

CHAPTER

TEN

# SUPPORT

See discussion board and issues on GitHub:

- Discussion board
- Issues

CHAPTER

# ELEVEN

# LICENSE

## 11.1 Software

```
                                 Apache License
                           Version 2.0, January 2004
                        http://www.apache.org/licenses/

   TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

   1. Definitions.

      "License" shall mean the terms and conditions for use, reproduction,
      and distribution as defined by Sections 1 through 9 of this document.

      "Licensor" shall mean the copyright owner or entity authorized by
      the copyright owner that is granting the License.

      "Legal Entity" shall mean the union of the acting entity and all
      other entities that control, are controlled by, or are under common
      control with that entity. For the purposes of this definition,
      "control" means (i) the power, direct or indirect, to cause the
      direction or management of such entity, whether by contract or
      otherwise, or (ii) ownership of fifty percent (50%) or more of the
      outstanding shares, or (iii) beneficial ownership of such entity.

      "You" (or "Your") shall mean an individual or Legal Entity
      exercising permissions granted by this License.

      "Source" form shall mean the preferred form for making modifications,
      including but not limited to software source code, documentation
      source, and configuration files.

      "Object" form shall mean any form resulting from mechanical
      transformation or translation of a Source form, including but
      not limited to compiled object code, generated documentation,
      and conversions to other media types.

      "Work" shall mean the work of authorship, whether in Source or
      Object form, made available under the License, as indicated by a
```

(continues on next page)

(continued from previous page)

```
    copyright notice that is included in or attached to the work
    (an example is provided in the Appendix below).

    "Derivative Works" shall mean any work, whether in Source or Object
    form, that is based on (or derived from) the Work and for which the
    editorial revisions, annotations, elaborations, or other modifications
    represent, as a whole, an original work of authorship. For the purposes
    of this License, Derivative Works shall not include works that remain
    separable from, or merely link (or bind by name) to the interfaces of,
    the Work and Derivative Works thereof.

    "Contribution" shall mean any work of authorship, including
    the original version of the Work and any modifications or additions
    to that Work or Derivative Works thereof, that is intentionally
    submitted to Licensor for inclusion in the Work by the copyright owner
    or by an individual or Legal Entity authorized to submit on behalf of
    the copyright owner. For the purposes of this definition, "submitted"
    means any form of electronic, verbal, or written communication sent
    to the Licensor or its representatives, including but not limited to
    communication on electronic mailing lists, source code control systems,
    and issue tracking systems that are managed by, or on behalf of, the
    Licensor for the purpose of discussing and improving the Work, but
    excluding communication that is conspicuously marked or otherwise
    designated in writing by the copyright owner as "Not a Contribution."

    "Contributor" shall mean Licensor and any individual or Legal Entity
    on behalf of whom a Contribution has been received by Licensor and
    subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of
    this License, each Contributor hereby grants to You a perpetual,
    worldwide, non-exclusive, no-charge, royalty-free, irrevocable
    copyright license to reproduce, prepare Derivative Works of,
    publicly display, publicly perform, sublicense, and distribute the
    Work and such Derivative Works in Source or Object form.

3. Grant of Patent License. Subject to the terms and conditions of
    this License, each Contributor hereby grants to You a perpetual,
    worldwide, non-exclusive, no-charge, royalty-free, irrevocable
    (except as stated in this section) patent license to make, have made,
    use, offer to sell, sell, import, and otherwise transfer the Work,
    where such license applies only to those patent claims licensable
    by such Contributor that are necessarily infringed by their
    Contribution(s) alone or by combination of their Contribution(s)
    with the Work to which such Contribution(s) was submitted. If You
    institute patent litigation against any entity (including a
    cross-claim or counterclaim in a lawsuit) alleging that the Work
    or a Contribution incorporated within the Work constitutes direct
    or contributory patent infringement, then any patent licenses
    granted to You under this License for that Work shall terminate
    as of the date such litigation is filed.
```

(continues on next page)

(continued from previous page)

```
   4. Redistribution. You may reproduce and distribute copies of the
      Work or Derivative Works thereof in any medium, with or without
      modifications, and in Source or Object form, provided that You
      meet the following conditions:

      (a) You must give any other recipients of the Work or
          Derivative Works a copy of this License; and

      (b) You must cause any modified files to carry prominent notices
          stating that You changed the files; and

      (c) You must retain, in the Source form of any Derivative Works
          that You distribute, all copyright, patent, trademark, and
          attribution notices from the Source form of the Work,
          excluding those notices that do not pertain to any part of
          the Derivative Works; and

      (d) If the Work includes a "NOTICE" text file as part of its
          distribution, then any Derivative Works that You distribute must
          include a readable copy of the attribution notices contained
          within such NOTICE file, excluding those notices that do not
          pertain to any part of the Derivative Works, in at least one
          of the following places: within a NOTICE text file distributed
          as part of the Derivative Works; within the Source form or
          documentation, if provided along with the Derivative Works; or,
          within a display generated by the Derivative Works, if and
          wherever such third-party notices normally appear. The contents
          of the NOTICE file are for informational purposes only and
          do not modify the License. You may add Your own attribution
          notices within Derivative Works that You distribute, alongside
          or as an addendum to the NOTICE text from the Work, provided
          that such additional attribution notices cannot be construed
          as modifying the License.

      You may add Your own copyright statement to Your modifications and
      may provide additional or different license terms and conditions
      for use, reproduction, or distribution of Your modifications, or
      for any such Derivative Works as a whole, provided Your use,
      reproduction, and distribution of the Work otherwise complies with
      the conditions stated in this License.

   5. Submission of Contributions. Unless You explicitly state otherwise,
      any Contribution intentionally submitted for inclusion in the Work
      by You to the Licensor shall be under the terms and conditions of
      this License, without any additional terms or conditions.
      Notwithstanding the above, nothing herein shall supersede or modify
      the terms of any separate license agreement you may have executed
      with Licensor regarding such Contributions.

   6. Trademarks. This License does not grant permission to use the trade
      names, trademarks, service marks, or product names of the Licensor,
      except as required for reasonable and customary use in describing the
```

(continues on next page)

(continued from previous page)

```
   origin of the Work and reproducing the content of the NOTICE file.

7. Disclaimer of Warranty. Unless required by applicable law or
   agreed to in writing, Licensor provides the Work (and each
   Contributor provides its Contributions) on an "AS IS" BASIS,
   WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or
   implied, including, without limitation, any warranties or conditions
   of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A
   PARTICULAR PURPOSE. You are solely responsible for determining the
   appropriateness of using or redistributing the Work and assume any
   risks associated with Your exercise of permissions under this License.

8. Limitation of Liability. In no event and under no legal theory,
   whether in tort (including negligence), contract, or otherwise,
   unless required by applicable law (such as deliberate and grossly
   negligent acts) or agreed to in writing, shall any Contributor be
   liable to You for damages, including any direct, indirect, special,
   incidental, or consequential damages of any character arising as a
   result of this License or out of the use or inability to use the
   Work (including but not limited to damages for loss of goodwill,
   work stoppage, computer failure or malfunction, or any and all
   other commercial damages or losses), even if such Contributor
   has been advised of the possibility of such damages.

9. Accepting Warranty or Additional Liability. While redistributing
   the Work or Derivative Works thereof, You may choose to offer,
   and charge a fee for, acceptance of support, warranty, indemnity,
   or other liability obligations and/or rights consistent with this
   License. However, in accepting such obligations, You may act only
   on Your own behalf and on Your sole responsibility, not on behalf
   of any other Contributor, and only if You agree to indemnify,
   defend, and hold each Contributor harmless for any liability
   incurred by, or claims asserted against, such Contributor by reason
   of your accepting any such warranty or additional liability.

END OF TERMS AND CONDITIONS

APPENDIX: How to apply the Apache License to your work.

   To apply the Apache License to your work, attach the following
   boilerplate notice, with the fields enclosed by brackets "[]"
   replaced with your own identifying information. (Don't include
   the brackets!)  The text should be enclosed in the appropriate
   comment syntax for the file format. We also recommend that a
   file or class name and description of purpose be included on the
   same "printed page" as the copyright notice for easier
   identification within third-party archives.

Copyright 2021-2022, World Meteorological Organization (WMO)

Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
```

(continues on next page)

(continued from previous page)

```
You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License.
```

## 11.2 Documentation

CHAPTER

TWELVE

# INDICES AND TABLES

- genindex
- modindex
- search