pybv

Release 0.6.0

pybv developers

Oct 02, 2023

CONTENTS

`pybv` is a lightweight exporter to the BrainVision data format.

The BrainVision data format is a recommended data format for use in the Brain Imaging Data Structure.

The documentation can be found under the following links:

- for the stable release
- for the latest (development) version

CHAPTER

ONE

ABOUT THE BRAINVISION DATA FORMAT

BrainVision is the name of a file format commonly used for storing electrophysiology data. Originally, it was put forward by the company Brain Products, however the simplicity of the format has allowed for a diversity of tools reading from and writing to the format.

The format consists of three separate files:

1. A text header file (`.vhdr`) containing meta data
2. A text marker file (`.vmrk`) containing information about events in the data
3. A binary data file (`.eeg`) containing the voltage values of the EEG

Both text files are based on the Microsoft Windows INI format consisting of:

- sections marked as `[square brackets]`
- comments marked as `; comment`
- key-value pairs marked as `key=value`

The binary `.eeg` data file is written in little-endian format without a Byte Order Mark (BOM), in accordance with the specification by Brain Products. This ensures that the data file is uniformly written irrespective of the native system architecture.

A documentation for the BrainVision file format is provided by Brain Products. You can view the specification as hosted by Brain Products.

CHAPTER

TWO

INSTALLATION

`pybv` runs on Python version 3.7 or higher.

`pybv`'s only dependency is `numpy`. However, we currently recommend that you install MNE-Python for reading BrainVision data. See their installation instructions.

After you have a working installation of MNE-Python (or only `numpy` if you do not want to read data and only write it), you can install `pybv` through the following:

- `pip install --upgrade pybv`

or if you use conda:

- `conda install --channel conda-forge pybv`

CHAPTER

THREE

CONTRIBUTING

The development of `pybv` is taking place on GitHub.

For more information, please see CONTRIBUTING.md

CHAPTER

FOUR

USAGE

4.1 Writing BrainVision files

The primary functionality provided by `pybv` is the `write_brainvision` function. This writes a numpy array of data and provided metadata into a collection of BrainVision files on disk.

```
from pybv import write_brainvision

# for further parameters see our API documentation
write_brainvision(data=data, sfreq=sfreq, ch_names=ch_names,
                  fname_base=fname, folder_out=tmpdir,
                  events=events)
```

4.2 Reading BrainVision files

Currently, `pybv` recommends using MNE-Python for reading BrainVision files.

Here is an example of the MNE-Python code required to read BrainVision data:

```
import mne

# Import the BrainVision data into an MNE Raw object
raw = mne.io.read_raw_brainvision('tmp/test.vhdr', preload=True)

# Reconstruct the original events from our Raw object
events, event_ids = mne.events_from_annotations(raw)
```

CHAPTER

FIVE

ACKNOWLEDGEMENTS

This package was originally adapted from the Philistine package by palday. It copies much of the BrainVision exporting code, but removes the dependence on MNE. Several features have been added, such as support for individual units for each channel.

5.1 API Documentation

Here we list the Application Programming Interface (API) for `pybv`.

5.1.1 pybv (`pybv`)

`write_brainvision`(*, data, sfreq, ch_names)	Write raw data to BrainVision format [1].

pybv.write_brainvision

`pybv.`**`write_brainvision`**`(`*, *data*, *sfreq*, *ch_names*, *ref_ch_names=None*, *fname_base*, *folder_out*, *overwrite=False*, *events=None*, *resolution=0.1*, *unit='µV'*, *fmt='binary_float32'*, *meas_date=None*`)`

Write raw data to BrainVision format [1].

Parameters

data
[`np.ndarray`, `shape` (`n_channels`, `n_times`)] The raw data to export. Voltage data is assumed to be in **Volts** and will be scaled as specified by `unit`. Non-voltage channels (as specified by `unit`) are never scaled (e.g. `'°C'`).

sfreq
[`python:int` | `python:float`] The sampling frequency of the data.

ch_names
[`python:list of` {`python:str` | `python:int`}, `len` (`n_channels`)] The names of the channels.

ref_ch_names
[`python:str` | `python:list of python:str`, `len` (`n_channels`) | `python:None`] The name of the channel used as a reference during the recording. If references differed between channels, you may supply a list of reference channel names

corresponding to each channel in `ch_names`. If `None` (default), assume that all channels are referenced to a common channel that is not further specified (BrainVision default).

Note: The reference channel name specified here does not need to appear in `ch_names`. It is permissible to specify a reference channel that is not present in `data`.

fname_base
[`python:str`] The base name for the output files. Three files will be created (.vhdr, .vmrk, .eeg) and all will share this base name.

folder_out
[`python:str`] The folder where output files will be saved. Will be created if it does not exist yet.

overwrite
[bool] Whether or not to overwrite existing files. Defaults to False.

events
[`np.ndarray`, `shape` (`n_events`, 2) `or` (`n_events`, 3) | `python:None`] Events to write in the marker file. This array has either two or three columns. The first column is always the zero-based index of each event (corresponding to the "time" dimension of the data array). The second column is a number associated with the "type" of event. The (optional) third column specifies the length of each event (default 1 sample). Currently all events are written as type "Stimulus" and must be numeric. Defaults to None (not writing any events).

resolution
[`python:float` | `np.ndarray`, `shape` (nchannels,)] The resolution in *unit* in which you'd like the data to be stored. If float, the same resolution is applied to all channels. If ndarray with n_channels elements, each channel is scaled with its own corresponding resolution from the ndarray. Note that *resolution* is applied on top of the default resolution that a data format (see *fmt*) has. For example, the binary_int16 format by design has no floating point support, but when scaling the data in µV for 0.1 resolution (default), accurate writing for all values >= 0.1 µV is guaranteed. In contrast, the binary_float32 format by design already supports floating points up to 1e-6 resolution, and writing data in µV with 0.1 resolution will thus guarantee accurate writing for all values >= 1e-7 µV (`1e-6 * 0.1`).

unit
[`python:str` | `python:list of python:str`] The unit of the exported data. This can be one of 'V', 'mV', 'µV' (or equivalently 'uV') , or 'nV', which will scale the data accordingly. Defaults to 'µV'. Can also be a list of units with one unit per channel. Non-voltage channels are stored as is, for example temperature might be available in `°C`, which `pybv` will not scale.

fmt
[`python:str`] Binary format the data should be written as. Valid choices are 'binary_float32' (default) and 'binary_int16'.

meas_date
[`datetime.datetime` | `python:str` | `python:None`] The measurement date specified as a datetime.datetime object. Alternatively, can be a string in the format 'YYYYMMDDhhmmssuuuuuu' ('u' stands for microseconds). Note that setting a measurement date implies that one additional event is created in the .vmrk file. To prevent this, set this parameter to None (default).

Notes

iEEG/EEG/MEG data is assumed to be in V, and we will scale these data to µV by default. Any unit besides µV is officially unsupported in the BrainVision specification. However, if one specifies other voltage units such as 'mV' or 'nV', we will still scale the signals accordingly in the exported file. We will also write channels with non-voltage units such as `°C` as is (without scaling). For maximum compatibility, all signals should be written as µV.

References

[1]

Examples

```
>>> data = np.random.random((3, 5))
>>> # write data with varying units
... # Note channels A1 and A2 are expected to be in Volt and will get
... # rescaled to µV and mV respectively.
... # TEMP is expected to be in some other unit (i.e., NOT Volt), and
... # will not get scaled (it is "written as is")
... write_brainvision(data=data, sfreq=1, ch_names=['A1', 'A2', 'TEMP'],
...                   folder_out='./',
...                   fname_base='pybv_test_file',
...                   unit=['µV', 'mV', '°C'])
>>> # remove the files
>>> for ext in ['.vhdr', '.vmrk', '.eeg']:
...     os.remove('pybv_test_file' + ext)
```

5.2 Authors

People who contributed to this software across releases (in **alphabetical order**):

- Adam Li
- Aniket Pradhan
- Chris Holdgraf
- Clemens Brunner
- Phillip Alday
- Richard Höchenberger
- Stefan Appelhoff
- Tristan Stenner

5.3 Changelog

Here we list a changelog of pybv.

Contents

5.3.1 0.6.0 (2021-09-29)

Changelog

- `pybv.write_brainvision()` gained a new parameter, `ref_ch_names`, to specify the reference channels used during recording, by Richard Höchenberger and Stefan Appelhoff (#75)

API

- `pybv.write_brainvision()` now has an `overwrite` parameter that defaults to `False`, by Stefan Appelhoff (#78).

Bug

- Fix bug where `pybv.write_brainvision()` would write the binary file in big-endian on a big-endian system, by Aniket Pradhan, Clemens Brunner, and Stefan Appelhoff (#80)

5.3.2 0.5.0 (2021-01-03)

Changelog

- `pybv.write_brainvision()` adds support for channels with non-Volt units, by Adam Li (#66).
- `pybv.write_brainvision()` automatically converts `uV` and `μV` (Greek μ) to `µV` (micro sign µ), by Adam Li (#66).

API

- The `unit` parameter in *`pybv.write_brainvision()`* now accepts a list of units (one unit per channel), by Adam Li (#66).

5.3.3 0.4.0 (2020-11-08)

Changelog

- Passing a “greek small letter mu” to the `unit` parameter in *`pybv.write_brainvision()`* instead of a “micro sign” is now permitted, because the former will be automatically convert to the latter, by Stefan Appelhoff (#47)

Bug

- Fix bug where *`pybv.write_brainvision()`* did not properly deal with commas in channel names and non-numeric events, by Stefan Appelhoff (#53)
- *`pybv.write_brainvision()`* now properly handles sampling frequencies that are not multiples of 10 (even floats), by Clemens Brunner (#59)
- Fix bug where *`pybv.write_brainvision()`* would write a different resolution to the `vhdr` file than specified with the `resolution` parameter. Note that this did *not* affect the roundtrip accuracy of the written data, because of internal scaling of the data, by Stefan Appelhoff (#58)
- Fix bug where values for the `resolution` parameter like `0.5`, `0.123`, `3.143` were not written with adequate decimal precision in *`pybv.write_brainvision()`*, by Stefan Appelhoff (#58)
- Fix bug where *`pybv.write_brainvision()`* did not warn users that a particular combination of `fmt`, `unit`, and `resolution` can lead to broken data. For example high resolution µV data in int16 format. In such cases, an error is raised now, by Stefan Appelhoff (#62)

API

- *`pybv.write_brainvision()`* now accepts keyword arguments only. Positional arguments are no longer allowed, by Stefan Appelhoff (#57)
- In *`pybv.write_brainvision()`*, the `scale_data` parameter was removed from *`pybv.write_brainvision()`*, by Stefan Appelhoff (#58)
- In *`pybv.write_brainvision()`*, the `unit` parameter no longer accepts an argument `None` to automatically determine a unit based on the `resolution`, by Stefan Appelhoff (#58)

5.3.4 0.3.0 (2020-04-02)

Changelog

- Add `unit` parameter for exporting signals in a specific unit (V, mV, µV or uV, nV), by Clemens Brunner (#39)

API

- The order of parameters in `pybv.write_brainvision()` has changed, by Clemens Brunner (#39)

5.3.5 0.2.0 (2019-08-26)

Changelog

- Add option to disable writing a meas_date event (which is also the new default), by Clemens Brunner (#32)
- Support event durations by passing an (N, 3) array to the events parameter (the third column contains the event durations), by Clemens Brunner (#33)

5.3.6 0.1.0 (2019-06-23)

Changelog

- Add measurement date parameter to public API, by Stefan Appelhoff (#29)
- Add binary format parameter to public API, by Tristan Stenner (#22)

Bug

- fix bug with events indexing. VMRK events are now correctly written with 1-based indexing, by Stefan Appelhoff (#29)
- fix bug with events that only have integer codes of length less than 3, by Stefan Appelhoff (#26)

5.3.7 0.0.2 (2019-04-28)

Changelog

- Support channel-specific scaling factors, by Tristan Stenner (#17)

5.3.8 0.0.1 (2018-12-10)

Changelog

- Initial import from philistine package by Phillip Alday and removing dependency on MNE-Python, by Chris Holdgraf, and Stefan Appelhoff

BIBLIOGRAPHY

[1] https://www.brainproducts.com/productdetails.php?id=21&tab=5

INDEX