
xarray-pickler Documentation

Release 0.1.2

Elle Smith

Aug 24, 2021

CONTENTS:

- 1 xarray-pickler** **1**
 - 1.1 Credits 1
- 2 Installation** **3**
 - 2.1 Stable release 3
 - 2.2 From sources 3
- 3 Usage** **5**
 - 3.1 Specifying types 5
 - 3.2 Settings 6
- 4 API** **7**
- 5 Contributing** **9**
 - 5.1 Types of Contributions 9
 - 5.2 Get Started! 10
 - 5.3 Pull Request Guidelines 11
 - 5.4 Tips 11
 - 5.5 Deploying 11
- 6 Credits** **13**
 - 6.1 Developers 13
 - 6.2 Contributors 13
- 7 History** **15**
 - 7.1 0.1.2 (2021-08-24) 15
 - 7.2 0.1.1 (2021-08-24) 15
 - 7.3 0.1.0 (2021-04-22) 15
- 8 Indices and tables** **17**

XARRAY-PICKLER

Simple package to speed up ‘multi-file open’ operations for xarray datasets. Uses a cache of pickle files to store the metadata in the *xarray.Dataset* object.

- Free software: BSD - see LICENSE file in top-level package directory
- Documentation: <https://xarray-pickler.readthedocs.io>.

1.1 Credits

This package was created with [Cookiecutter](#) and the [audreyr/cookiecutter-pypackage](#) project template.

INSTALLATION

2.1 Stable release

To install xarray-pickler, run this command in your terminal:

```
$ pip install xarray_pickler
```

This is the preferred method to install xarray-pickler, as it will always install the most recent stable release.

If you don't have [pip](#) installed, this [Python installation guide](#) can guide you through the process.

2.2 From sources

The sources for xarray-pickler can be downloaded from the [Github repo](#).

You can either clone the public repository:

```
$ git clone git://github.com/cedadev/xarray_pickler
```

Once you have a copy of the source, you can install it with:

```
$ python setup.py install
```


To use xarray-pickler in a project:

```
import xarray_pickler
```

To use the open_dset function

```
from xarray_pickler import open_dset

dpath = "/badc/cmip6/data/CMIP6/CMIP/IPSL/IPSL-CM6A-LR/historical/r1i1p1f1/Amon/rlds/gr/
↪v20180803"

# specify extra kwargs to use with xarray.open_mfdataset()
kwargs = {'parallel': True}

ds = open_dset(dpath, **kwargs)
return ds
```

In etc/config.ini there are settings that can be configured.

Any section of the configuration file can be overwritten by creating a new INI file with the desired sections and values and then setting the environment variable PICKLE_CONFIG as the file path to the new INI file. e.g. PICKLE_CONFIG="path/to/config.ini"

The configuration settings used are listed and explained below. Explanations will be provided as comments in the code blocks if needed. Examples are provided so these settings will not necessarily match up with what is used.

3.1 Specifying types

It is possible to specify the type of the entries in the configuration file, for example if you want a value to be a list when the file is parsed.

This is managed through a [config_data_types] section at the top of the INI file which has the following options:

```
[config_data_types]
# use only in xarray-pickler
lists = pickle_dirs
dicts =
ints = dir_grouping_level
floats =
boolean = use_cftime remove_archive_dir_in_path
# use the below if using the xarray-pickler config settings in other packages
```

(continues on next page)

(continued from previous page)

```
extra_lists =  
extra_dicts =  
extra_ints =  
extra_floats =  
extra_booleans =
```

Simply adding the name of the value you want to format after = will render the correct format. e.g. `boolean = use_cftime` `remove_archive_dir_in_path` will set both `use_cftime` and `remove_archive_dir_in_path` as booleans.

3.2 Settings

The settings that can be configured are:

```
# the default settings that will be used everytime an xarray.Dataset object is opened.  
[open_mfdataset_kwargs]  
use_cftime = True  
combine = by_coords  
  
[paths]  
# how many directory levels to join together to create the name of the pickle file ->  
↪ this reduces the length of the file path  
dir_grouping_level = 4  
# the path to the pickle file stores, to be listed in the order they should be checked ->  
↪ for existing pickles.  
pickle_dirs = /badc/cmip6/metadata/xarray-pickles /gws/nopw/j04/cp4cds1_vol1/metadata/  
↪ xarray-pickles  
# the directory to write new pickle files to  
writeable_pickle_dir = /gws/nopw/j04/cp4cds1_vol1/metadata/xarray-pickles  
# directories where the archive data is stored  
archive_dir = /badc/cmip6/data/  
# whether to remove the archive dir from the full pickle file path  
remove_archive_dir_in_path = True
```

`xarray_pickler.xarray_pickler.open_dset(dpath, force_repickle=False, **kwargs)`

Open `xarray.Dataset` object. If previously pickled, it will be opened from the pickle file stored in the cache. Otherwise, it will be pickled and stored in the cache, if a cache is specified, after it is opened using `xarray.open_mfdataset()` with any extra keyword arguments specified. If there is no writeable pickle directory specified in the config, and the pickle does not already exist, the dataset will just be opened using `xarray` and returned.

Parameters

- **(str) (dpath)** – Directory path to netCDF files to generate dataset from e.g. `“/badc/cmip6/data/CMIP6/CMIP/IPSL/IPSL-CM6A-LR/historical/r1i1p1f1/Amon/rlds/gr/v20180803”`
- **force_repickle** – If True, the `xarray.Dataset` object will be repickled, if a writeable pickle directory is specified in the config. Default is False.
- ****kwargs** – Other keyword arguments that can be used in `xarray.open_mfdataset()`. Used only the first time a dataset is pickled or if `force_repickle=True`.

Returns `xarray.Dataset` object

CONTRIBUTING

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given. You can contribute in many ways:

5.1 Types of Contributions

5.1.1 Report Bugs

Report bugs at <https://github.com/cedadev/xarray-pickler/issues>.

If you are reporting a bug, please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

5.1.2 Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with “bug” and “help wanted” is open to whoever wants to implement it.

5.1.3 Implement Features

Look through the GitHub issues for features. Anything tagged with “enhancement” and “help wanted” is open to whoever wants to implement it.

5.1.4 Write Documentation

xarray-pickler could always use more documentation, whether as part of the official xarray-pickler docs, in docstrings, or even on the web in blog posts, articles, and such.

5.1.5 Submit Feedback

The best way to send feedback is to file an issue at <https://github.com/cedadev/xarray-pickler/issues>.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.

5.2 Get Started!

Ready to contribute? Here's how to set up *xarray-pickler* for local development.

1. Fork the *xarray-pickler* repo on GitHub.
2. Clone your fork locally:

```
$ git clone git@github.com:your_name_here/xarray-pickler.git
```

3. Install your local copy into a virtualenv. Assuming you have virtualenvwrapper installed, this is how you set up your fork for local development:

```
$ mkvirtualenv xarray_pickler
$ cd xarray-pickler/
$ python setup.py develop
```

4. Create a branch for local development:

```
$ git checkout -b name-of-your-bugfix-or-feature
```

Now you can make your changes locally.

5. When you are done making changes, check that your changes pass flake8 and the tests, including testing other Python versions with tox:

```
$ flake8 clisops tests
$ black --target-version py36 clisops tests
$ pytest tests
```

To get flake8, black, and tox, just pip install them into your virtualenv. Alternatively, you can use *pre-commit* to perform these checks at the git commit stage:

```
$ pip install pre-commit
$ pre-commit install
$ pre-commit run --all-files
```

6. Commit your changes and push your branch to GitHub:

```
$ git add .
$ git commit -m "Your detailed description of your changes."
$ git push origin name-of-your-bugfix-or-feature
```

7. Submit a pull request through the GitHub website.

5.3 Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

1. The pull request should include tests.
2. If the pull request adds functionality, the docs should be updated. Put your new functionality into a function with a docstring, and add the feature to the list in README.rst.
3. The pull request should work for Python 3.6, 3.7 and 3.8. Check https://travis-ci.com/cedadev/xarray-pickler/pull_requests and make sure that the tests pass for all supported Python versions.

5.4 Tips

To run a subset of tests:

```
$ pytest tests/test_xarray_pickler.py::test_get_pickle_path
```

5.5 Deploying

A reminder for the maintainers on how to deploy. Make sure all your changes are committed (including an entry in HISTORY.rst). Then run:

```
$ git tag <tagname>
$ git push origin <tagname>
```

Where the tag name is the raw version e.g. 0.1.0 GitHub Actions will then deploy to PyPI if tests pass.

CREDITS

6.1 Developers

- Elle Smith <eleonor.smith@stfc.ac.uk>

6.2 Contributors

None yet. Why not be the first?

HISTORY

7.1 0.1.2 (2021-08-24)

Changes: - Fix to typos and URLs in the docs.

7.2 0.1.1 (2021-08-24)

- Second release

Changes: - modified naming of pickle files to avoid “*” and “?” characters in file names.

7.3 0.1.0 (2021-04-22)

- First release on PyPI.

INDICES AND TABLES

- `genindex`
- `modindex`
- `search`