smap$_{i}oDocumentation$

Release

TU Wien

Feb 11, 2019

Contents

SMAP (Soil Moisture Active Passive) data readers.

Works great in combination with pytesmo.

CHAPTER 1

Citation

If you use the software in a publication then please cite it using the Zenodo DOI. Be aware that this badge links to the latest package version.

Please select your specific version at https://doi.org/10.5281/zenodo.596391 to get the DOI of that version. You should normally always use the DOI for the specific version of your record in citations. This is to ensure that other researchers can access the exact research artefact you used for reproducibility.

You can find additional information regarding DOI versioning at http://help.zenodo.org/#versioning

CHAPTER 2

Installation

Setup of a complete environment with conda can be performed using the following commands:

```
conda create -q -n smap_io -c conda-forge numpy h5py pyproj netcdf4==1.2.2 pyresample␣
↪scipy pandas matplotlib
source activate smap_io
pip install smap_io
```

You can also install all needed (conda and pip) dependencies at once using the following commands after cloning this repository. This is recommended for developers of the package.

```
git clone https://github.com/TUW-GEO/smap_io.git --recursive
cd smap_io
conda create -n smap_io python=2.7 # or any supported python version
source activate smap_io
conda update -f environment.yml
python setup.py develop
```

CHAPTER 3

Supported Products

- SPL3SMP: SMAP L3 Radiometer Global Daily 36 km EASE-Grid Soil Moisture

CHAPTER 4

Contribute

We are happy if you want to contribute. Please raise an issue explaining what is missing or if you find a bug. We will also gladly accept pull requests against our master branch for new features or bug fixes.

4.1 Guidelines

If you want to contribute please follow these steps:

- Fork the smap_io repository to your account
- make a new feature branch from the smap_io master branch
- Add your feature
- please include tests for your contributions in one of the test directories We use py.test so a simple function called test_my_feature is enough
- submit a pull request to our master branch

CHAPTER 5

Reading images

5.1 SPL3SMP

After downloading the data you will have a path with subpaths of the format `YYYY.MM.DD`. Let's call this path `root_path`. To read 'soil_moisture' data for the descending overpass of a certain date use the following code:

```
from smap_io import SPL3SMP_Ds
root_path = os.path.join(os.path.dirname(__file__),
                         'test_data', 'SPL3SMP')
ds = SPL3SMP_Ds(root_path, overpass='AM')
image = ds.read(datetime(2015, 4, 1))
assert list(image.data.keys()) == ['soil_moisture']
assert image.data['soil_moisture'].shape == (406, 964)
```

The returned image is of the type pygeobase.Image. Which is only a small wrapper around a dictionary of numpy arrays.

If you only have a single image you can also read the data directly

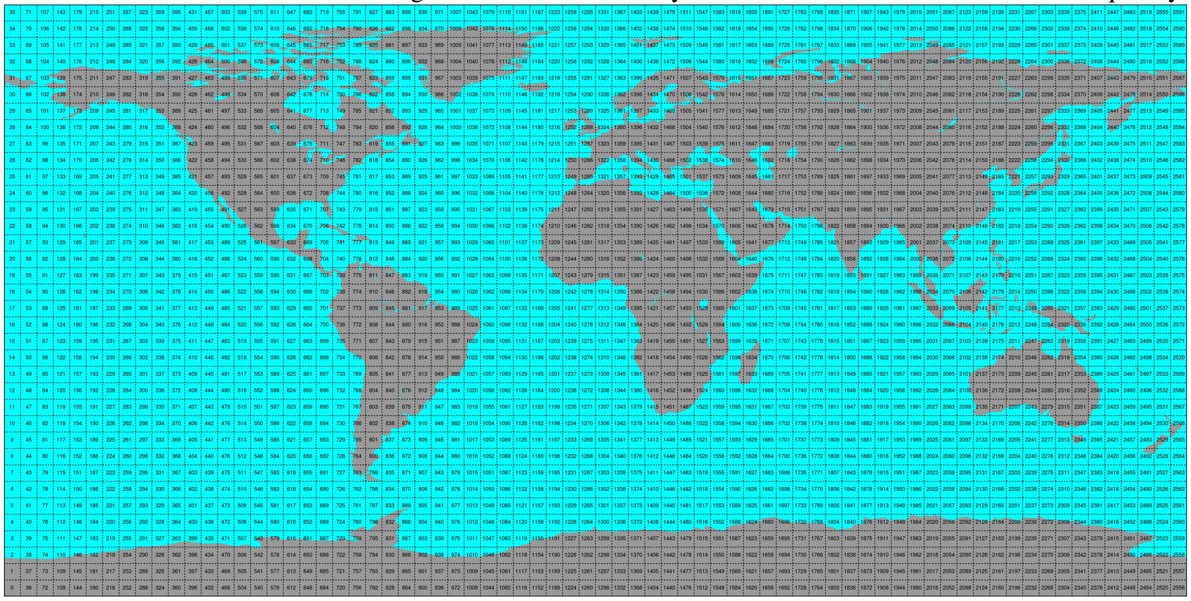
```
from smap_io import SPL3SMP_Img
fname = os.path.join(os.path.dirname(__file__),
                     'test_data', 'SPL3SMP', '2015.04.01',
                     'SMAP_L3_SM_P_20150401_R13080_001.h5')
ds = SPL3SMP_Img(fname, overpass='PM')
image = ds.read()
assert list(image.data.keys()) == ['soil_moisture']
assert image.data['soil_moisture_pm'].shape == (406, 964)
```

CHAPTER 6

Conversion to time series format

For a lot of applications it is favorable to convert the image based format into a format which is optimized for fast time series retrieval. This is what we often need for e.g. validation studies. This can be done by stacking the images into a netCDF file and choosing the correct chunk sizes or a lot of other methods. We have chosen to do it in the following way:

- Store the grid points in a 1D array. This also allows reduction of the data volume by e.g. only saving the points over land.
- Store the time series in netCDF4 in the Climate and Forecast convention Orthogonal multidimensional array representation
- Store the time series in 5x5 degree cells. This means there will be 2566 cell files and a file called `grid.nc` which contains the information about which grid point is stored in which file. This allows us to read a whole 5x5 degree area into memory and iterate over the time series quickly.



6.1 SPL3SMP

This conversion can be performed using the smap_repurpose command line program. An example would be:

```
smap_repurpose /SPL3SMP_data /timeseries/data 2015-04-01 2015-04-02 soil_moisture␣
↪soil_moisture_error --overpass AM
```

Which would take SMAP SPL3SMP data stored in /SPL3SMP_data from April 1st 2015 to April 2nd 2015 and store the parameters soil_moisture and soil_moisture_error for the AM overpass as time series in the folder /timeseries/data. When the PM overpass is selected, time series variables will be renamed with the suffix *_pm*.

Conversion to time series is performed by the repurpose package in the background. For custom settings or other options see the repurpose documentation and the code in smap_io.reshuffle.

6.2 Reading converted time series data

For reading the data the smap_repurpose command produces the class SMAPTs can be used. Optional arguments that are passed to the parent class (OrthoMultiTs, as defined in pynetcf.time_series) can be passed as well:

```
from smap_io.interface import SMAPTs
ds = SMAPTs(ts_path, parameters=['soil_moisture','soil_moisture_error'],
            ioclass_kws={'read_bulk': True})
# read_ts takes either lon, lat coordinates or a grid point indices.
# and returns a pandas.DataFrame
ts = ds.read_ts(45, 15) # (lon, lat)
```

Bulk reading speeds up reading multiple points from a cell file by storing the file in memory for subsequent calls.

CHAPTER 7

Contents

7.1 License

7.2 Developers

- Christoph Paulik <christoph.paulik@geo.tuwien.ac.at>
- Wolfgang Preimesberger <wolfgang.preimesberger@geo.tuwien.ac.at>

7.3 Changelog

7.3.1 Version 0.x

-

7.3.2 Version 0.3

- Add test for download
- Update documentation
- Add kwargs to time series reader
- Add option for download checking
- Add CRID reading
- Name PM variables *_pm in time series
- Add download module
- Add SMAP L3 v4 and v5 support
- Update readme

7.3.3 Version 0.2

- Add metadata from netCDF file to returned image.
- Add option to return data as 1D arrays.
- Add image to time series conversion and time series reading interface.

7.3.4 Version 0.1

- Initial version with one dataset supported.

7.4 smap_io

7.4.1 smap_io package

Submodules

smap_io.download module

smap_io.interface module

smap_io.reshuffle module

Module contents

CHAPTER 8

Indices and tables

- genindex
- modindex
- search