
smap_i*oDocumentation*

Release latest

TU Wien

Sep 27, 2022

Contents

1	Citation	3
2	Installation	5
3	Supported Products	7
4	Contribute	9
4.1	Guidelines	9
5	Downloading products	11
6	Reading images	13
6.1	SPL3SMP	13
7	Conversion to time series format	15
7.1	SPL3SMP	16
7.2	Reading converted time series data	16
8	Contents	17
8.1	Downloading products	17
8.2	Reading images	18
8.3	Conversion to time series format	18
8.4	License	19
8.5	Developers	20
8.6	Changelog	20
8.7	smap_io	21
9	Indices and tables	23

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 pyresample_  
↪pandas  
$ 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=3.6 # 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

Additional products will be added when need arises, feel free to open an issue to add a new data product or even better a pull request.

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

Downloading products

SMAP products can be downloaded via HTTPS. You have to register an account with NASA's Earthdata portal: <https://urs.earthdata.nasa.gov/>

After that you can use the command line program `smap_download` and your username and password to download data between 2 dates.

In order for the download to work, the program line command “`wget`” must be installed. On Linux this is usually the case. You test this by running.

```
$ wget -V
```

On Windows, running “`wget -V`” in the command line will usually return an error like this

```
C:\Users>wget -V
'wget' is not recognized as an internal or external command,
operable program or batch file.
```

In that case you need to install `wget` first. There are many tutorials available online. e.g. :

- <https://builtvisible.com/download-your-website-with-wget/>
- https://www.jcchouinard.com/wget/#Download_Wget_on_Windows

In short, you need to download “`wget.exe`” into a location where Windows command line can find it (specified in your “`PATH`” environment variable; on Windows command simply run ‘`path`’ to show all included directories). You can also add another directory to your Windows `PATH` as described for example [here](#). Then the command `wget -V` should also work on windows (the download module of `smap_io` will use this function).

To test it you can run the following example. This would download all available h5 files of the latest SMAP SPL3SMP data into the folder `~/workspace/smap_data`. For more options on other available parameters run `smap_download --help`.

```
mkdir ~/workspace/smap_data
smap_download ~/workspace/smap_data --username *name* --password *password*
```


6.1 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
from datetime import datetime
import os
root_path = os.path.join(os.path.dirname(__file__),
                        'test_data', 'SPL3SMP')
ds = SPL3SMP_Ds(root_path, overpass=None, var_overpass_str=False)
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.

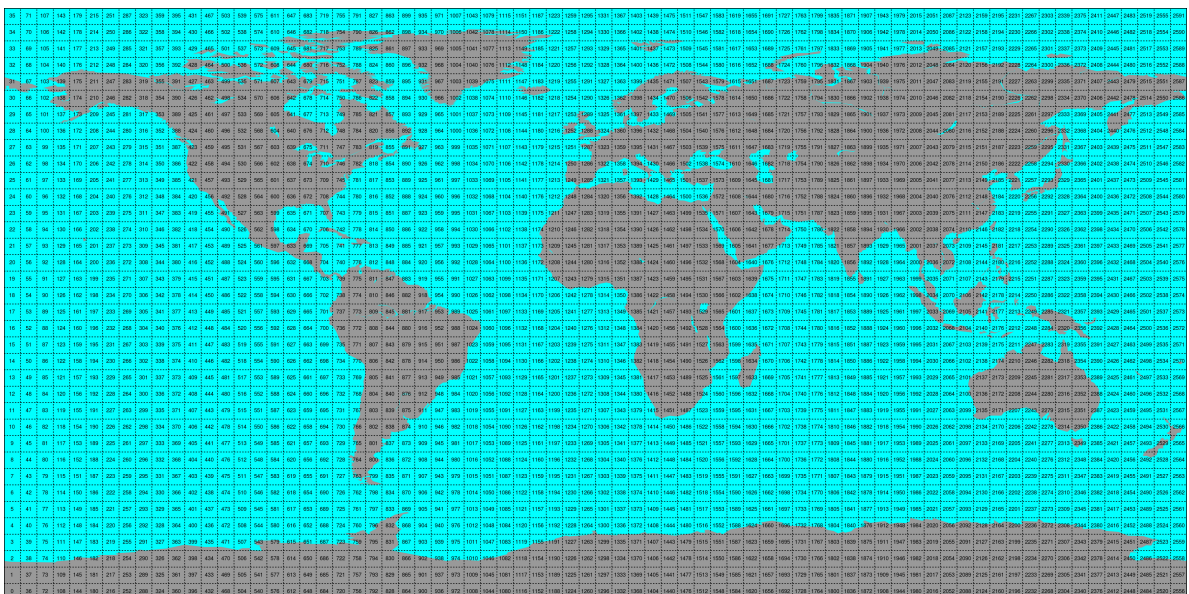
If you only have a single image you can also read the data directly

```
from smap_io import SPL3SMP_Img
import os
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=None, var_overpass_str=False)
image = ds.read()
assert list(image.data.keys()) == ['soil_moisture']
assert image.data['soil_moisture'].shape == (406, 964)
```


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.



7.1 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`.

Note: If a `RuntimeError: NetCDF: Bad chunk sizes.` appears during reshuffling, consider downgrading the `netcdf4` library via:

```
conda install -c conda-forge netcdf4=1.2.2
```

7.2 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.

8.1 Downloading products

SMAP products can be downloaded via HTTPS. You have to register an account with NASA's Earthdata portal: <https://urs.earthdata.nasa.gov/>

After that you can use the command line program `smap_download` and your username and password to download data between 2 dates.

In order for the download to work, the program line command “wget” must be installed. On Linux this is usually the case. You test this by running.

```
$ wget -V
```

On Windows, running “wget -V” in the command line will usually return an error like this

```
C:\Users>wget -V
'wget' is not recognized as an internal or external command,
operable program or batch file.
```

In that case you need to install `wget` first. There are many tutorials available online. e.g. :

- <https://builtvisible.com/download-your-website-with-wget/>
- https://www.jcchouinard.com/wget/#Download_Wget_on_Windows

In short, you need to download “wget.exe” into a location where Windows command line can find it (specified in your “PATH” environment variable; on Windows command simply run ‘path’ to show all included directories). You can also add another directory to your Windows PATH as described for example [here](#). Then the command `wget -V` should also work on windows (the download module of *smap_io* will use this function).

To test it you can run the following example. This would download all available h5 files of the latest SMAP SPL3SMP data into the folder `~/workspace/smap_data`. For more options on other available parameters run `smap_download --help`.

```
mkdir ~/workspace/smap_data
smap_download ~/workspace/smap_data --username *name* --password *password*
```

8.2 Reading images

8.2.1 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
from datetime import datetime
import os
root_path = os.path.join(os.path.dirname(__file__),
                        'test_data', 'SPL3SMP')
ds = SPL3SMP_Ds(root_path, overpass=None, var_overpass_str=False)
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.

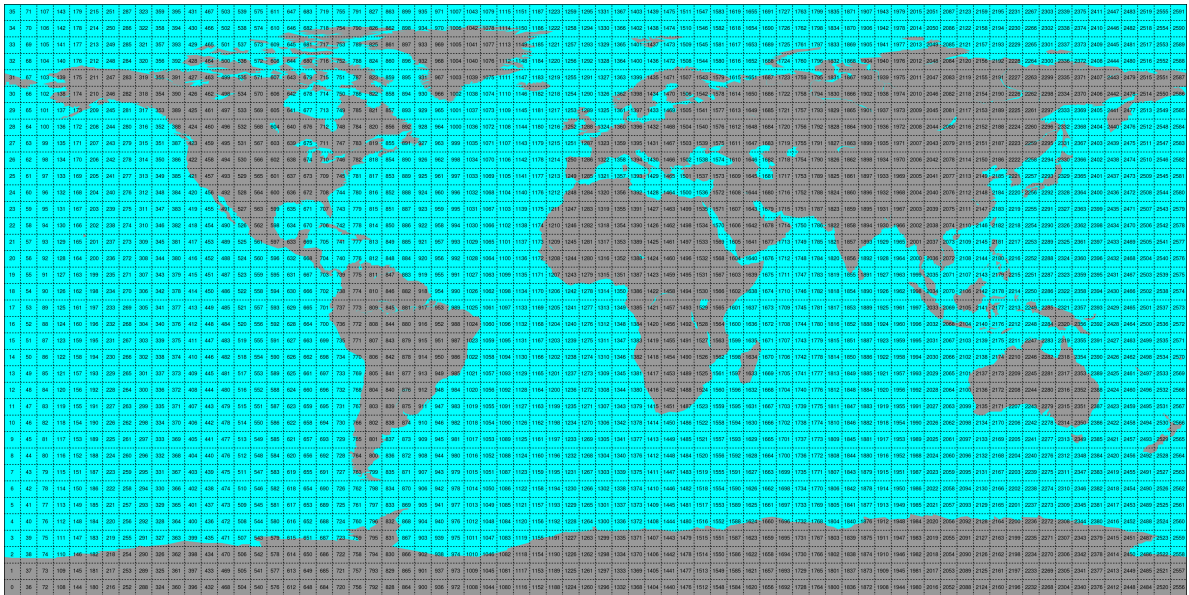
If you only have a single image you can also read the data directly

```
from smap_io import SPL3SMP_Img
import os
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=None, var_overpass_str=False)
image = ds.read()
assert list(image.data.keys()) == ['soil_moisture']
assert image.data['soil_moisture'].shape == (406, 964)
```

8.3 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.



8.3.1 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 variable names 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`.

Note: If a `RuntimeError: NetCDF: Bad chunk sizes.` appears during reshuffling, consider downgrading the `netcdf4` library via:

```
conda install -c conda-forge netcdf4=1.2.2
```

8.4 License

The MIT License (MIT)

Copyright (c) 2016 TU Wien

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

8.5 Developers

- Christoph Paulik <cpaulik@vandersat.com>
- Wolfgang Preimesberger <wolfgang.preimesberger@geo.tuwien.ac.at>

8.6 Changelog

8.6.1 Unreleased

-

8.6.2 Version 0.5

- Add support to download all smap products and test actual download
- Allow reshuffling land points only (SPL3SMP)
- Allow reshuffling points in bounding box only (SPL3SMP)
- Testdata module is now using GitLFS and hosted at TUW
- Meta package follows pyscaffold 4 standards, yapf formatting added

8.6.3 Version 0.4

- Switch to new pyscaffold structure
- Add support for SMAP L3 v6 data,
- Remove download support for v4 and v5 (decommissioned), reading still possible.
- Add option to rename variables with orbit indicator during reading.

8.6.4 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

8.6.5 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.

8.6.6 Version 0.1

- Initial version with one dataset supported.

8.7 smap_{io}

8.7.1 smap_{io} package

Submodules

smap_{io}.download module

smap_{io}.grid module

smap_{io}.interface module

smap_{io}.reshuffle module

Module contents

CHAPTER 9

Indices and tables

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