oceansdb Documentation

Release 0.6

Guilherme Castelao

Jan 02, 2020

Contents

Contents:

CHAPTER 1

OceansDB

Package to subsample, or interpolate, climatologies like WOA to any coordinates.

This package started with functions to obtain climatological values to compare with measured data, allowing a quality control check by comparison. It hence needed to work for any coordinates requested. I split these functionalities from CoTeDe into this standalone package to allow more people to use it for other purposes.

- Free software: 3-clause BSD style license - see LICENSE.rst
- Documentation: https://oceansdb.readthedocs.io.

1.1 Features

- If the database files are not localy available, automatically download it.
- Extract, or interpolate if necessary, climatologic data on requested coordinates;
  - Can request a single point, a profile or a section;
  - Ready to handle -180 to 180 or 0 to 360 coordinate system;
- Ready to use with:
  - World Ocean Atlas (WOA)
  - CSIRO Atlas Regional Seas (CARS)
  - ETOPO (topography)

1.2 Quick howto use

Inside python:

```
>>> import oceansdb
>>> db = oceansdb.WOA()
```

Find out what is available:

```
>>> db.keys()
```

Average temperature at one point:

```
>>> t = db['sea_water_temperature'].extract(var='mean', doy=136.875, depth=0, lat=17.
↪5, lon=-37.5)
```

A profile of salinity:

```
>>> t = db['sea_water_salinity'].extract(var='mean', doy=136.875, depth=[0, 10, 15,␣
↪18], lat=17.5, lon=-37.5)
```

A full depth section of temperature:

```
>>> t = db['sea_water_temperature'].extract(var='mean', doy=136.875, lat=17.48, lon=[-
↪39, -37.5, -35.2])
```

Using CARS instead of WOA:

```
>>> db = oceansdb.CARS()
>>> t = db['sea_water_temperature'].extract(var='mean', doy=136.875, lat=17.48, lon=[-
↪39, -37.5, -35.2], depth=[0,10,120,280])
```

Or to get topography for one point from the 1 min arc resolution:

```
>>> db = oceansdb.ETOPO(resolution='1min')
>>> h = db['topography'].extract(lat=17.5, lon=0)
```

CHAPTER 2

Installation

2.1 Stable release

To install OceansDB, run this command in your terminal:

```
$ pip install oceansdb
```

This is the preferred method to install OceansDB, 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 OceansDB can be downloaded from the Github repo.

You can either clone the public repository:

```
$ git clone git://github.com/castelao/oceansdb
```

Or download the tarball:

```
$ curl  -OL https://github.com/castelao/oceansdb/tarball/master
```

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

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

CHAPTER 3

Usage

To use oceansdb in a project:

```
import oceansdb
```

Now create a World Ocean Atlas database object by:

```
db = oceansdb.WOA()
```

On the first time you run this, it might take sometime since it needs to download the actual database files. You don't need to do anything other than wait.

The propriety extract() of the database object is who takes care of sub-sample, and if necessary interpolate, to give you the variable(s) on the requested coordinates. The returned output is always a dictionary, even if you requested only one variable from the database.

To get temperature at one point:

```
>>> t = db['TEMP'].extract(var='t_mn', doy=136.875, depth=0, lat=17.5, lon=-37.5)
```

The WOA climatologic temperature will be available as t['t_mn'].

```
>>> t.keys()
['t_mn']

>>> t['t_mn'].shape
(1,)

>>> t['t_mn']
masked_array(data = [ 24.60449791],
             mask = False,
       fill_value = 1e+20)
```

If you prefer you can obtain all available variables by not defining var, like:

```
>>> t = db['PSAL'].extract(doy=136.875, depth=[0, 10, 15, 18], lat=17.5, lon=-37.5)

>>> t.keys()
['s_dd', 's_sd', 's_se', 's_mn']
```

To get one profile of salinity:

```
>>> t = db['PSAL'].extract(var='t_mn', doy=136.875, depth=[0, 10, 15, 18], lat=17.5,␣
↪lon=-37.5)
```

To get one section of temperature:

```
>>> t = db['TEMP'].extract(var='t_mn', doy=136.875, lat=17.5, lon=[-39, -37.5, -35])
```

To get a regular 3D grid:

```
>>> t = db['TEMP'].extract(var='t_mn', depth=[0, 10.23], doy=136.875, lat=[15, 17.5,␣
↪23], lon=[-39, -37.5, -35, -32.73])

>>> t['t_mn'].shape
(2, 3, 4)
```

To use bathymetry let's first load ETOPO

```
>>> db = oceansdb.ETOPO()
```

Let's check the variables available in ETOPO

```
>>> db.keys()
['topography']
```

To get topography for one point:

```
>>> db['topography'].extract(lat=15, lon=38)
{'height': masked_array(data=[1012],
             mask=[False],
       fill_value=999999,
            dtype=int32)}
```

To get topography along a latitude:

```
>>> db['topography'].extract(lat=15, lon=[-25, -30, -38, -40, -45])
{'height': masked_array(data=[-4150, -5451, -5588, -5217, -3840],
             mask=[False, False, False, False, False],
       fill_value=999999,
            dtype=int32)}
```

To get topography along a longitude:

```
>>> db['topography'].extract(lat=[10, 15, 20, 25], lon=38)
{'height': masked_array(data=[1486, 1012, -759, 797],
             mask=[False, False, False, False],
       fill_value=999999,
            dtype=int32)}
```

To get topography along a area:

```
>>> db['topography'].extract(lat=[10, 15, 20, 25], lon=[30, 38, 40])
{'height': masked_array(
data=[[413, 1486, 1227],
      [504, 1012, 210],
      [294, -759, -217],
      [241, 797, 1050]],
mask=[[False, False, False],
      [False, False, False],
      [False, False, False],
      [False, False, False]],
fill_value=999999,
dtype=int32)}
```

To use ETOPO with 5min resolution instead of the 1min:

```
>>> db = oceansdb.ETOPO(resolution='5min')

>>> db['topography'].extract(lat=15, lon=38)
{'height': masked_array(data=[1372.0],
          mask=[False],
    fill_value=1e+20,
         dtype=float32)}
```

CHAPTER 4

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:

4.1 Types of Contributions

4.1.1 Report Bugs

Report bugs at https://github.com/castelao/oceansdb/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.

4.1.2 Fix Bugs

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

4.1.3 Implement Features

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

4.1.4 Write Documentation

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

4.1.5 Submit Feedback

The best way to send feedback is to file an issue at https://github.com/castelao/oceansdb/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.
- Remember that this is a volunteer-driven project, and that contributions are welcome :)

4.2 Get Started!

Ready to contribute? Here's how to set up *oceansdb* for local development.

1. Fork the *oceansdb* repo on GitHub.
2. Clone your fork locally:

```
$ git clone git@github.com:your_name_here/oceansdb.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 oceansdb
$ cd oceansdb/
$ 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're done making changes, check that your changes pass flake8 and the tests, including testing other Python versions with tox:

```
$ flake8 WOA tests
$ python setup.py test
$ tox
```

   To get flake8 and tox, just pip install them into your virtualenv.

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.

4.3 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 2.6, 2.7, 3.3, and 3.4, and for PyPy. Check https://travis-ci.org/castelao/oceansdb/pull_requests and make sure that the tests pass for all supported Python versions.

4.4 Tips

To run a subset of tests:

```
$ py.test tests/test_WOA_from_nc.py
```

CHAPTER 5

Credits

5.1 Development Lead

- Guilherme Castelao <guilherme@castelao.net>

5.2 Contributors

Erdem M Karakoylu: Reported bad checksum on CARS' salinity file (issue #7). @sgartzman: Reported bug #8, truncated files on Python-2. @maryjacketti: Reported bug #12, outdated ETOPO documentation. Uday Bhaskar T V S: Reported improper path to DB files for Windows machines. Carin Anderson & Ozan Göktürk: Reported bug #14 - coordinates partially coincident.

CHAPTER 6

History

CHAPTER 7

0.8.0

- Allowing to choose resolution and temporal scale.

CHAPTER 8

0.7.0 (2017-05-11)

- New generic crop functionality.
- Interpolate in steps, in order: time, lat x lon, and finally z. Gives better results for profiles.

CHAPTER 9

0.6.0 (2016-04-14)

- Adding CSIRO Atlas of Regional Seas (CARS), another climatology for temperature and salinity.

CHAPTER 10

0.4.0 (2016-03-29)

- Added etopo5, a world wide bathymetry. Some of this code has its origins on pyAVISO and CoTeDe.

CHAPTER 11

0.2.0 (2016-03)

- Renamed package to OceansDB, planning to include other climatologies and references.

CHAPTER 12

0.1.0 (2016-02-09)

- Generalized interpolator. Extract climatology for point, profile or section.

CHAPTER 13

0.0.1 (2015-12-13)

- First release on PyPI. Extracted from CoTeDe to be a standalone package.

CHAPTER 14

pre-released (~2006)

- The first version was developed in support to quality control thermosalinographs at NOAA. It was later injected inside CoTeDe.

CHAPTER 15

Indices and tables

- genindex
- modindex
- search