bmi_wavewatch3

Release 0.1

Eric Hutton

Aug 02, 2022

CONTENTS:

bmi_wavewatch3

CHAPTER

ONE

ABOUT

The *bmi_wavewatch3* Python package provides both a command line interface and a programming interface for downloading and working with WAVEWATCH III data.

bmi_wavewatch3 provides access to the following raster data sources,

- 30 year wave hindcast Phase 1
- 30 year wave hindcast Phase 2
- Production hindcast Singlegrid
- Production hindcast Multigrid

All data sources provide both global and regional grids.

CHAPTER

TWO

INSTALLATION

bmi_wavewatch3 can be installed by running `pip install bmi-wavewatch3`. It requires Python >= 3.8 to run.

If you simply can't wait for the latest release, you can install *bmi_wavewatch3* directly from GitHub,

```
$ pip install git+https://github.com/csdms/bmi-wavewatch3
```

bmi_wavewatch3 is also available through *conda*, `conda install bmi-wavewatch3 -c conda-forge`.

CHAPTER

THREE

USAGE

To get started, you can download *WAVEWATCH III* data by date with the *ww3* command (use *ww3 –help* to print a brief message),

```
$ ww3 fetch "2010-05-22"
```

You can also do this through Python,

```
>>> from bmi_wavewatch3 import WaveWatch3
>>> WaveWatch3.fetch("2010-05-22")
```

The *bmi_wavewatch3* package provides the `WaveWatch3` class for downloading data and presenting it as an *xarray Dataset*.

```
>>> from bmi_wavewatch3 import WaveWatch3
>>> ww3 = WaveWatch3("2010-05-22")
>>> ww3.data
<xarray.Dataset>
...
```

Use the `inc` method to advance in time month-by-month,

```
>>> ww3.date
'2010-05-22'
>>> ww3.inc()
'2010-06-22'
>>> ww3.data.time
<xarray.DataArray 'time' ()>
array('2010-06-01T00:00:00.000000000', dtype='datetime64[ns]')
...
```

This will download new datasets as necessary and load the new data into the `data` attribute.

Note: If the new data are not cached on you computer, you will notice a delay while the new data are download. If the `lazy` flag is set, the download will only occur once you try to access the data (i.e. `ww3.data`), otherwise the data are downloaded as soon as the date is set.

CHAPTER

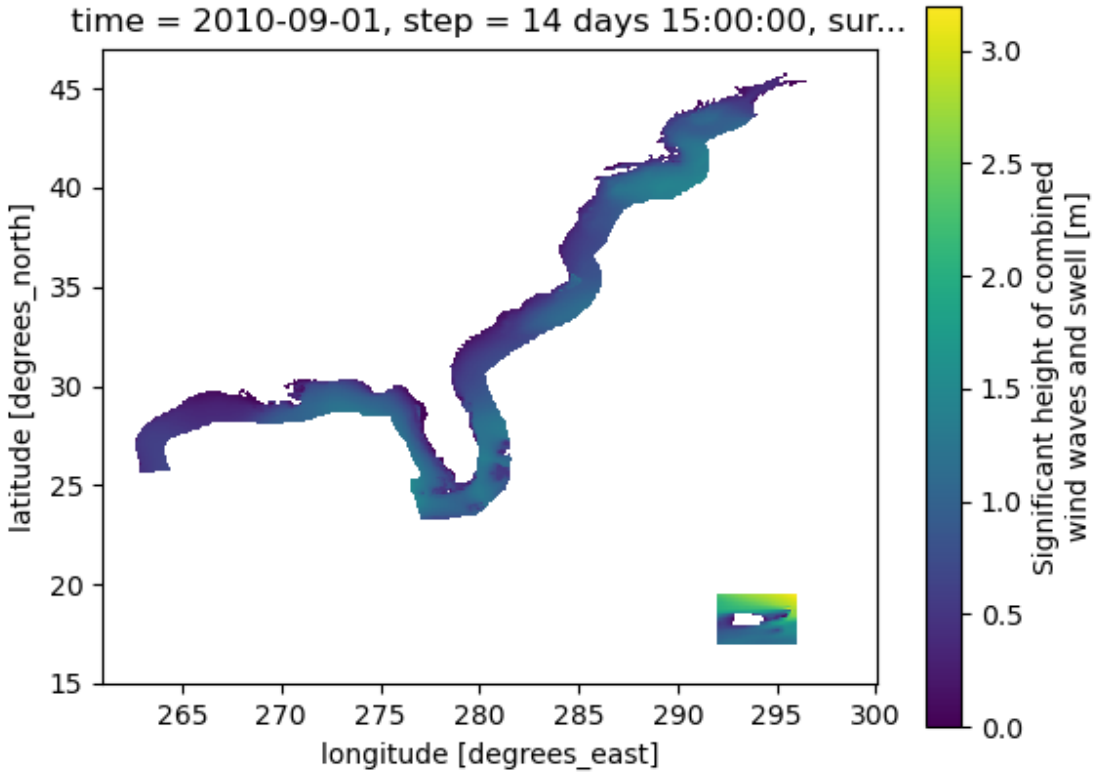
FOUR

EXAMPLE

4.1 Plot data from the command line

Running the following from the command line will plot the variable *significant wave height* from the WAVEWATCH III *at_4m* grid. Note that the time of day (in this case, 15:00) is separated from the date with a `T` (i.e. times can be given as `YYYY-MM-DDTHH`)

```
$ ww3 plot --grid=at_4m --data-var=swh "2010-09-15T15"
```



4.2 Plot data from Python

This example is similar to the previous but uses the *bmi_wavewatch3* Python interface.

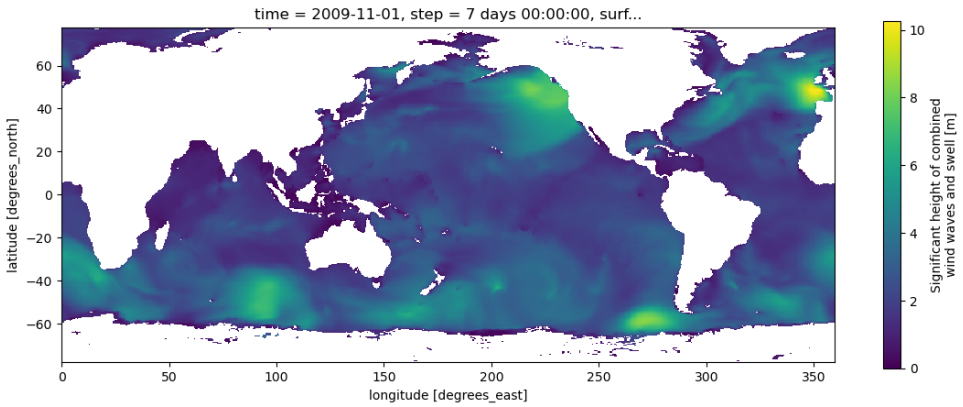
```
>>> from bmi_wavewatch3 import WaveWatch3
>>> ww3 = WaveWatch3("2009-11-08")
```

The data can be accessed as an *xarray Dataset* through the `data` attribute.

```
>>> ww3.data
<xarray.Dataset>
Dimensions:     (step: 241, latitude: 311, longitude: 720)
Coordinates:
    time        datetime64[ns] 2009-11-01
  * step        (step) timedelta64[ns] 0 days 00:00:00 ... 30 days 00:00:00
    surface     float64 1.0
  * latitude    (latitude) float64 77.5 77.0 76.5 76.0 ... -76.5 -77.0 -77.5
  * longitude   (longitude) float64 0.0 0.5 1.0 1.5 ... 358.0 358.5 359.0 359.5
    valid_time  (step) datetime64[ns] dask.array<chunksize=(241,), meta=np.ndarray>
Data variables:
    dirpw       (step, latitude, longitude) float32 dask.array<chunksize=(241, 311, 720),
↪ meta=np.ndarray>
    perpw       (step, latitude, longitude) float32 dask.array<chunksize=(241, 311, 720),
↪ meta=np.ndarray>
    swh         (step, latitude, longitude) float32 dask.array<chunksize=(241, 311, 720),
↪ meta=np.ndarray>
    u           (step, latitude, longitude) float32 dask.array<chunksize=(241, 311, 720),
↪ meta=np.ndarray>
    v           (step, latitude, longitude) float32 dask.array<chunksize=(241, 311, 720),
↪ meta=np.ndarray>
Attributes:
    GRIB_edition:            2
    GRIB_centre:             kwbc
    GRIB_centreDescription:  US National Weather Service - NCEP
    GRIB_subCentre:          0
    Conventions:             CF-1.7
    institution:             US National Weather Service - NCEP
    history:                 2022-06-08T16:08 GRIB to CDM+CF via cfgrib-0.9.1...
```

The `step` attribute points to the current time slice into the data (i.e number of three hour increments since the start of the month),

```
>>> ww3.step
56
>>> ww3.data.swh[ww3.step, :, :].plot()
```



4.2.1 Release Notes

0.2.0 (2022-06-17)

New Features

- Added a new subcommand, *plot*, to the *ww3* command-line program. `ww3 plot` with download (if the data files are not already cached) and create a plot of the requested data. (#13)

Bug Fixes

- Fixed a bug in the reporting of an error caused by an invalide datatime string. (#13)

0.1.1 (2022-06-10)

Other Changes and Additions

- Set up GitHub Action to create a source distribution and push it to *TestPyPI*. This action is only run if the version tag is a prerelease version (i.e. the version string ends with `[ab][0-9]+`). (#10)
- Set up GitHub Action to create a source distribution and push it to *PyPI*. This action is only run if the version tag is a release version (i.e. the version string doesn't end with `[ab][0-9]+`). (#11)

0.1.1b1 (2022-06-09)

New Features

- Added `ww3` command line interface to download WaveWatch III data by date, region and quantity (significant wave height, wind speed, etc.). (#1)
- Added `WaveWatch3` class, which is the main access point for users of this package. This class downloads WaveWatch III data files (if not already cached) and provides a view of the data as an xarray Dataset. Users can then advance through the data month-by-month, downloading additional data as necessary. (#3)
- Added the `ww3 clean` subcommand that removes cached data files. (#4)

- Added `BMIWaveWatch3` class to provide a Basic Model Interface for the *wavewatch3* package. (#5)
- Added additional WaveWatch III data sources from which users can fraw data from. (#6)
- Added `fetch` method to WaveWatch3 to mimic the command line program `ww3 fetch`. (#7)
- Added additional data sources from which to retreive data from. Available data sources now include: Phase 1, Phase 2, Multigrid, Multigrid-extended, and Multigrid-thredds. (#7)
- Added `ww3 info` command to print information (e.g. available grids, quantities, etc.) about data sources. (#7)
- Added a `step` property to `WaveWatch3` to track the current time slice of the data cube. This property is also settable so that a user can use it to advance throught the data (additional data are downloaded in the background as needed). (#8)
- Dates can now be specified as iso-formatted date/time strings. For example, "1944-06-06T06:30". (#8)
- Rename package to `bmi_wavewatch3`. This follows the convention used by other CSDMS data components. (#9)

Documentation Enhancements

- Added package description, installation, usage, and an example to the documentation. (#8)

Other Changes and Additions

- Set up continuous integration using GitHub actions. This includes tests to ensure that the code is styled according to *black*, is free of lint, and passes all unit tests. (#2)
- Added more unit tests, particularly for data sources. (#7)
- Added a GitHub action to build the sphinx-based documentation as part of the continuous integration. (#8)
- Better error reporting for the command line interface for HTTP errors when retreiving data as well as input validation. (#8)
- Set up GitHub Action to create a source distribution and push it to *TestPyPI*. This action is only run if the version tag is a prerelease version (i.e. the version string ends with `[ab][0-9]+`). (#10)

4.2.2 Credits

Development Lead

- Eric Hutton (@mcflugen)

Contributors

None yet. Why not be the first?

4.2.3 The MIT License (MIT)

CHAPTER

FIVE

INDICES AND TABLES

- genindex
- modindex
- search