
climate_indices Documentation

Release 1.0.10

James Adams

Dec 18, 2022

Contents:

1	Getting started	3
1.1	Configure the Python environment	3
1.2	NCO	4
2	Indices Processing	5
2.1	Example Input and Output Datasets	7
2.2	Example Command Line Invocations	7
3	For Developers	11
3.1	Download the code	11
3.2	Testing	11
4	Get involved	13
5	Copyright and licensing	15
6	Citation	17

This project contains Python implementations of various climate index algorithms which provide a geographical and temporal picture of the severity of precipitation and temperature anomalies useful for climate monitoring and research.

The following indices are provided:

- [SPI](#), Standardized Precipitation Index, utilizing both gamma and Pearson Type III distributions
- [SPEI](#), Standardized Precipitation Evapotranspiration Index, utilizing both gamma and Pearson Type III distributions
- [PET](#), Potential Evapotranspiration, utilizing either [Thornthwaite](#) or [Hargreaves](#) equations
- [PNP](#), Percentage of Normal Precipitation

The following are provided as experimental/development versions only, not fully vetted nor suitable for research purposes:

- [PDSI](#), Palmer Drought Severity Index
- [scPDSI](#), Self-calibrated Palmer Drought Severity Index
- [PHDI](#), Palmer Hydrological Drought Index
- [Z-Index](#), Palmer moisture anomaly index (Z-index)
- [PMDI](#), Palmer Modified Drought Index

This Python implementation of the above climate index algorithms is being developed with the following goals in mind:

- to provide an open source software package to compute a suite of climate indices commonly used for climate monitoring, with well documented code that is faithful to the relevant literature and which produces scientifically verifiable results
- to provide a central, open location for participation and collaboration for researchers, developers, and users of climate indices
- to facilitate standardization and consensus on best-of-breed climate index algorithms and corresponding compliant implementations in Python
- to provide transparency into the operational code used for climate monitoring activities at NCEI/NOAA, and consequent reproducibility of published datasets computed from this package
- to incorporate modern software engineering principles and scientific programming best practices

The installation and configuration described below is performed using a bash shell, either on Linux, Windows, or MacOS.

Windows users will need to install and configure a bash shell in order to follow the usage shown below. We recommended either [babun](#) or [Cygwin](#) for this purpose.

1.1 Configure the Python environment

This project's code is written in Python 3. It is recommended to use either the [Miniconda3](#) (minimal Anaconda) or [Anaconda3](#) distribution. The below instructions will be Anaconda specific (although relevant to any Python [virtual environment](#)), and assume the use of a bash shell.

A new Anaconda [environment](#) can be created using the [conda](#) environment management system that comes packaged with Anaconda. In the following examples, we'll use an environment named *indices_env* (any environment name can be used instead of *indices_env*) which will be created and populated with all required dependencies through the use of the provided `setup.py` file.

First, create the Python environment:

```
$ conda create -n indices_env
```

The environment can now be 'activated':

```
$ source activate indices_env
```

Once the environment has been activated then subsequent Python commands will run in this environment where the package dependencies for this project are present.

Now the package can be added to the environment along with all required modules (dependencies) via [pip](#):

```
$ python -m pip install climate-indices
```

For development of the package itself it pays to install the dependencies via the *requirements.txt* file:

```
$ python -m pip install -r requirements.txt
```

When adding dependencies to the package they should be added to the *pyproject.toml* file in the dependencies section, then we can rebuild the requirements.txt file using pip-tools:

```
$ python -m piptools compile -o requirements.txt pyproject.toml
```

1.2 NCO

NetCDF Operators is a requirement and must be installed for utilization of this package. Instructions for installation on various platforms is available [here](#). If using an Anaconda environment as advised above then it's as simple as running the following command within the activated conda environment:

```
$ conda install -c conda-forge nco
```


CHAPTER 2

Indices Processing

The installation will provide an “entry point” script which interacts with the core computational package to compute one or more climate indices. This script is `process_climate_indices` and is used to compute indices corresponding to gridded NetCDF datasets as well as US climate division NetCDF datasets.

This Python entry point script is written to be run via bash shell command, i.e.

```
$ process_climate_indices <options>
```

The options for the entry point script are described below:

Option	Description
index	Which of the climate indices to compute. Valid values are ‘spi’, ‘spei’, ‘pnp’, ‘scaled’, ‘pet’, and ‘palmers’. ‘scaled’ indicates all three scaled indices (SPI, SPEI, and PNP) and ‘palmers’ indicates all Palmer indices (PDSI, PHDI, PMDI, SCPDSI, and Z-Index).
periodicity	The periodicity of the input dataset files. Valid values are ‘monthly’ and ‘daily’. NOTE: Only SPI and PNP support daily inputs.
netcdf_precip	Input NetCDF file containing a precipitation dataset, required for all indices except for PET. Requires the use of var_name_precip in conjunction so as to identify the NetCDF’s precipitation variable.
var_name_precip	Name of the precipitation variable within the input precipitation NetCDF.
netcdf_temp	Input NetCDF file containing a temperature dataset, required for PET. If specified in conjunction with an index specification of SPEI or Palmer’s then PET will be computed and written as a side effect, since these indices require PET. This option is mutually exclusive with netcdf_pet/var_name_pet , as either temperature or PET is required as an input (but not both) when computing SPEI and/or Palmer’s. Requires the use of var_name_temp in conjunction so as to identify the NetCDF’s temperature variable.
var_name_temp	Name of the temperature variable within the input temperature NetCDF.
netcdf_pet	Input NetCDF file containing a PET dataset, required for SPEI and Palmer’s. This option is mutually exclusive with netcdf_temp/var_name_temp , as either temperature or PET is required as an input (but not both) when computing SPEI and/or Palmer’s. Requires the use of var_name_pet in conjunction so as to identify the NetCDF’s PET variable.
var_name_pet	Name of the PET variable within the input PET NetCDF.
netcdf_awc	Input NetCDF file containing available water capacity, required for Palmer’s. Requires the use of var_name_awc in conjunction so as to identify the NetCDF’s AWC variable.
awc_var_name	Name of the available water capacity variable within the input AWC NetCDF.
output_file_base	Base file name for all output files. Each computed index will have a corresponding output file whose name will begin with this base name plus the index’s abbreviation plus a month scale (if applicable), connected with underscores, plus the ‘.nc’ extension. For example for SPI at 3-month scale the resulting output files will be named <output_file_base>_spi_gamma_03.nc and <output_file_base>_spi_pearson_03.nc .
scales	Time step scales over which the PNP, SPI, and SPEI values are to be computed. Required when the index argument is ‘spi’, ‘spei’, ‘pnp’, or ‘scaled’. The periodicity option will infer whether the scales used are month or day scales. NOTE: When used for US climate divisions processing this option specifies month scales
calibration_start_year	Initial year of the calibration period.
calibration_end_year	Final year of the calibration period (inclusive).
multi-processing	Valid values are ‘all’ (uses all available CPUs), ‘single’ (uses a single CPU), or ‘all_but_one’ (uses all CPUs minus one). Default value is ‘all_but_one’.
save_params	Saves distribution fitting variables to this file path. The fittings NetCDF is to be used as input when using the <i>load_params</i> option. [NOTE: only for use with the <i>spi</i> entrypoint for SPI.]
load_params	Loads distribution fitting variables from this filepath. The fittings NetCDF file is one that was created by the <i>save_params</i> option. [NOTE: only for use with the <i>spi</i> entrypoint for SPI.]

2.1 Example Input and Output Datasets

Example NetCDF datasets that are valid input to the indices processing scripts described above are available from the associated project [example_climate_indices](#). The input NetCDF files used in the examples below (*nclimdiv.nc*, *nclimgrid_lowres_prcp.nc*, etc.) can be fetched from this repository, as well as associated output NetCDF datasets that can be used to validate result of the below examples.

2.2 Example Command Line Invocations

2.2.1 US Climate Divisions (all indices)

```
$ process_climate_indices --index all --periodicity monthly --scales 3 6
--netcdf_precip /data/nclimdiv.nc --netcdf_temp /data/nclimdiv.nc --netcdf_awc
/data/nclimdiv.nc --output_file_base /data/nclimdiv --var_name_precip
prcp --var_name_temp tavg --var_name_awc awc --calibration_start_year 1951
--calibration_end_year 2010
```

The above command will compute all indices from an input NetCDF dataset containing precipitation, temperature, and available water capacity variables (in this case, the US Climate Divisions NetCDF dataset provided in the example inputs directory). The input dataset is monthly data and the calibration period used will be Jan. 1951 through Dec. 2010. The indices will be computed at 3-month and 6-month scales. Upon completion the individual NetCDF files will contain variables for all computed indices: */data/nclimdiv_pet.nc*, */data/nclimdiv_pnp_03.nc*, */data/nclimdiv_pnp_06.nc*, */data/nclimdiv_spi_gamma_03.nc*, */data/nclimdiv_spi_gamma_06.nc*, */data/nclimdiv_spi_pearson_03.nc*, */data/nclimdiv_spi_pearson_06.nc*, */data/nclimdiv_spei_gamma_03.nc*, */data/nclimdiv_spei_gamma_06.nc*, */data/nclimdiv_spei_pearson_03.nc*, */data/nclimdiv_spei_pearson_06.nc*, */data/nclimdiv_pdsi.nc*, */data/nclimdiv_phdi.nc*, */data/nclimdiv_pmdi.nc*, */data/nclimdiv_scpsi.nc*, and */data/nclimdiv_zindex.nc*. Parallelization will occur utilizing all but one of the available CPUs (default since the *--multiprocessing* option is omitted).

2.2.2 PET monthly

```
$ process_climate_indices --index pet --periodicity monthly --netcdf_temp
/data/nclimgrid_lowres_tavg.nc --var_name_temp tavg --output_file_base
<out_dir>/nclimgrid_lowres --multiprocessing all_but_one
```

The above command will compute PET (potential evapotranspiration) using the Thornthwaite method from an input temperature dataset (in this case, the reduced resolution nClimGrid temperature dataset provided in the example inputs directory). The input dataset is monthly data and the calibration period used will be Jan. 1951 through Dec. 2010. The output file will be *<out_dir>/nclimgrid_lowres_pet.nc*. Parallelization will occur utilizing all but one of the available CPUs.

2.2.3 SPI daily

```
$ process_climate_indices --index spi --periodicity daily --netcdf_precip
/data/cmorph_lowres_daily_conus_prdp.nc --var_name_precip prdp
--output_file_base <out_dir>/cmorph_lowres_daily_conus --scales 30 90
--calibration_start_year 1998 --calibration_end_year 2016 --multiprocessing
all
```

The above command will compute SPI (standardized precipitation index, both gamma and Pearson Type III distributions) from an input precipitation dataset (in this case, the reduced resolution CMORPH precipitation dataset provided in the example inputs directory). The input dataset is daily data and the calibration period used will be Jan. 1st, 1998 through Dec. 31st, 2016. The index will be computed at 30-day and 90-day timescales. The output files will be `<out_dir>/cmorph_lowres_daily_conus_spi_gamma_30.nc`, `<out_dir>/cmorph_lowres_daily_conus_spi_gamma_90.nc`, `<out_dir>/cmorph_lowres_daily_conus_spi_pearson_30.nc`, and `<out_dir>/cmorph_lowres_daily_conus_spi_pearson_90.nc`. Parallelization will occur utilizing all CPUs.

2.2.4 SPI monthly

```
$ process_climate_indices --index spi --periodicity monthly --netcdf_precip
/data/nclimgrid_lowres_prdp.nc --var_name_precip prdp --output_file_base
<out_dir>/nclimgrid_lowres --scales 6 12 --calibration_start_year 1951
--calibration_end_year 2010 --multiprocessing all
```

The above command will compute SPI (standardized precipitation index, both gamma and Pearson Type III distributions) from an input precipitation dataset (in this case, the reduced resolution nClimGrid precipitation dataset provided in the example inputs directory). The input dataset is monthly data and the calibration period used will be Jan. 1951 through Dec. 2010. The index will be computed at 6-month and 12-month timescales. The output files will be `<out_dir>/nclimgrid_lowres_spi_gamma_06.nc`, `<out_dir>/nclimgrid_lowres_spi_gamma_12.nc`, `<out_dir>/nclimgrid_lowres_spi_pearson_06.nc`, and `<out_dir>/nclimgrid_lowres_spi_pearson_12.nc`. Parallelization will occur utilizing all CPUs.

2.2.5 SPEI monthly

```
$ process_climate_indices --index spei --periodicity monthly --netcdf_precip
/data/nclimgrid_lowres_prdp.nc --var_name_precip prdp --netcdf_pet /
data/nclimgrid_lowres_pet.nc --var_name_pet pet --output_file_base
<out_dir>/nclimgrid_lowres --scales 9 18 --calibration_start_year 1951
--calibration_end_year 2010 --multiprocessing all
```

The above command will compute SPEI (standardized precipitation evapotranspiration index, both gamma and Pearson Type III distributions) from input precipitation and potential evapotranspiration datasets (in this case, the reduced resolution nClimGrid precipitation and PET datasets provided in the example inputs directory). The input datasets are monthly data and the calibration period used will be Jan. 1951 through Dec. 2010. The index datasets will be computed at 9-month and 18-month timescales. The output files will be `<out_dir>/nclimgrid_lowres_spei_gamma_09.nc`, `<out_dir>/nclimgrid_lowres_spei_gamma_18.nc`, `<out_dir>/nclimgrid_lowres_spei_pearson_09.nc`, and `<out_dir>/nclimgrid_lowres_spei_pearson_18.nc`. Parallelization will occur utilizing all CPUs.

2.2.6 Palmers monthly

```
$ process_climate_indices --index palmers --periodicity monthly
--netcdf_precip /data/nclimgrid_lowres_prdp.nc --var_name_precip prdp
--netcdf_pet /data/nclimgrid_lowres_pet.nc --var_name_pet pet --netcdf_awc /
data/nclimgrid_lowres_soil.nc --var_name_awc awc --output_file_base <out_dir>/
nclimgrid_lowres --calibration_start_year 1951 --calibration_end_year 2010
--multiprocessing all
```

The above command will compute the Palmer drought indices: PDSI (original Palmer Drought Severity Index), PHDI (Palmer Hydrological Drought Index), PMDI (Palmer Modified Drought Index), Z-Index (Palmer Z-Index), and SCPDSI (Self-calibrated Palmer Drought Severity Index) from input precipitation, potential evapotranspiration, and available water capacity datasets (in this case, the reduced resolution nClimGrid precipi-

tation, PET, and AWC datasets provided in the example inputs directory). The input datasets are monthly data and the calibration period used will be Jan. 1951 through Dec. 2010. The output files will be `<out_dir>/nclimgrid_lowres_pdsi.nc`, `<out_dir>/nclimgrid_lowres_phdi.nc`, `<out_dir>/nclimgrid_lowres_pmdi.nc`, `<out_dir>/nclimgrid_lowres_scpdsi.nc`, and `<out_dir>/nclimgrid_lowres_zindex.nc`. Parallelization will occur utilizing all CPUs.

2.2.7 Pre-compute SPI distribution fitting variables

In order to pre-compute fitting parameters for later use as inputs to subsequent SPI calculations we can save both gamma and Pearson distribution fitting parameters to NetCDF, and later use this file as input for SPI calculations over the same calibration period.

```
$ spi --periodicity monthly --scales 1 2 3 6 9 12 24 36 48 60 72
--calibration_start_year 1998 --calibration_end_year 2016 --netcdf_precip /
data/nclimgrid/nclimgrid_prcp.nc --var_name_precip prcp --output_file_base
/data/nclimgrid/nclimgrid --multiprocessing all --save_params /data/nclimgrid/
nclimgrid_fitting.nc --overwrite
```

```
$ spi --periodicity monthly --scales 1 2 3 6 9 12 24 36 48 60 72
--calibration_start_year 1998 --calibration_end_year 2016 --netcdf_precip /
data/nclimgrid/nclimgrid_prcp.nc --var_name_precip prcp --output_file_base
/data/nclimgrid/nclimgrid --multiprocessing all --load_params /data/nclimgrid/
nclimgrid_fitting.nc
```

In the above example we demonstrate how distribution fitting parameters can be saved as NetCDF. This fittings NetCDF can then be used as pre-computed variables in subsequent SPI computations. Initial command computes both distribution fitting values and SPI for various month scales. The distribution fitting variables are written to the file specified by the `--save_params` option. The second command also computes SPI but instead of computing the distribution fitting values it loads the pre-computed fitting values from the NetCDF file specified by the `--load_params` option.

3.1 Download the code

Clone this repository:

```
$ git clone https://github.com/monocongo/climate_indices.git
```

Move into the source directory:

```
$ cd climate_indices
```

Within this directory, there are four subdirectories:

- `climate_indices`: main computational package
- `tests`: unit tests for the main package
- `notebooks`: Jupyter Notebooks describing the internals of the computational modules
- `docs`: documentation files

3.2 Testing

Initially, all tests should be run for validation:

```
$ tox
```

If you run the above from the main branch and get an error then please send a report and/or add an issue, as all tests should pass.

CHAPTER 4

Get involved

Please use, make suggestions, and contribute to this code. Without diverse participation and community adoption this project will not reach its potential.

Are you aware of other indices that would be a good addition here? Can you identify bottlenecks and help optimize performance? Can you suggest new ways of comparing these implementations against others (or other criteria) in order to determine best-of-breed? Please fork the code and have at it, and/or contact us to see if we can help.

- Read our [contributing guidelines](#)
- File an [issue](#), or submit a [pull request](#)
- Send us an [email](#)

CHAPTER 5

Copyright and licensing

This is a developmental version of code that is originally developed at NCEI/NOAA, official release version available on drought.gov. This software is under BSD 3-Clause license, copyright James Adams, 2017. Please read more on our [license](#) page.

You can cite *climate_indices* in your projects and research papers via the BibTeX entry below.

```
@misc {climate_indices,  
  author = "James Adams",  
  title  = "climate_indices, an open source Python library providing reference_  
↪implementations of commonly used climate indices",  
  url    = "https://github.com/monocongo/climate_indices",  
  month  = "may",  
  year   = "2017--"  
}
```