MONET Documentation

Barry Baker

Apr 11, 2024

GETTING STARTED

MONET is an open-source project and Python package that aims to create a common platform for atmospheric composition data analysis for weather and air quality models.

MONET was developed to evaluate the Community Multiscale Air Quality Model (CMAQ) for the NOAA National Air Quality Forecast Capability (NAQFC) modeling system. MONET is designed to be a modularized Python package for

1. pairing model output to observational
2. leveraging the pandas Python package for easy searching
3. analyzing and visualizing data

This process introduces a convenient method for evaluating model output. MONET processes data that is easily searchable and that can be grouped using meta-data found within the observational datasets. Common statistical metrics (e.g., bias, correlation, and skill scores), plotting routines such as scatter plots, timeseries, spatial plots, and more are included in the package. MONET is well-modularized and can add further observational datasets and different models.

Our goal is to provide easy tools to retrieve, read, and combine datasets in order to speed scientific research. Currently, MONET is able to process several models and observations related to air composition and meteorology.

Please *cite* our work.

CHAPTER

ONE

WHAT'S NEW

MONET v2.2.0 has been released (2020-03-27). MONET has re-engineered the way it deals with multidimensional observations or model output by using an xarray accessor giving MONET a flexible and intuitive way of expanding xarray for multidimensional geospatial information commonly used in meteorology, climate and air quality all while making it easier on the user to use MONET and add to it.

Important: MONET also underwent a major restructure with v2.2.0. All I/O functions have been moved to a sister project: MONETIO.

MONET features include:

- *xarray accessor* for both `xarray.DataArray` and `xarray.Dataset` using the `.monet` attribute
- *pandas accessor* for `pandas.DataFrame` using the `.monet` attribute
- vertical interpolation using python-stratify using the `.monet.stratify` function
- spatial interpolation using `.monet.remap` including:
  - nearest neighbor finder
  - constant latitude interpolation
  - constant longitude interpolation
  - remap DataArray to current grid using pyresample nearest neighbor or xESMF
  - remap entire dataset to current grid using pyresample nearest neighbor or xESMF
  - find nearest i,j or lat,lon
  - interpolate to constant latitude or longitude
- simplified *`combine tool`* to combine point source data with multidimensional xarray objects

CHAPTER

TWO

REFERENCE

Baker, Barry; Pan, Li. 2017. “Overview of the Model and Observation Evaluation Toolkit (MONET) Version 1.0 for Evaluating Atmospheric Transport Models.” Atmosphere 8, no. 11: 210. doi:10.3390/atmos8110210.

CHAPTER

THREE

GET IN TOUCH

Ask questions, suggest features or view source code on GitHub.

CHAPTER

FOUR

INDICES

- genindex
- modindex

4.1 Overview: Why MONET?

4.1.1 Features

Retrieving, loading, and combining data and putting into a common format is the core of MONET. MONET uses the pandas and xarray data formats for data analysis.

- Open point observations in a common format. pandas excels at working with tabular data or point measurements. It is used for time series analysis and statistical measures.
- Open model and satellite data in a common format. xarray is used when N-dimensional arrays are needed.
- Retrieving observational datasets for given time and space.
- Efficiently combine/interpolate model and observational datasets.
- Provide easy plotting using proven tools in Python
- Perform statistics between model runs or observations or models and observations.

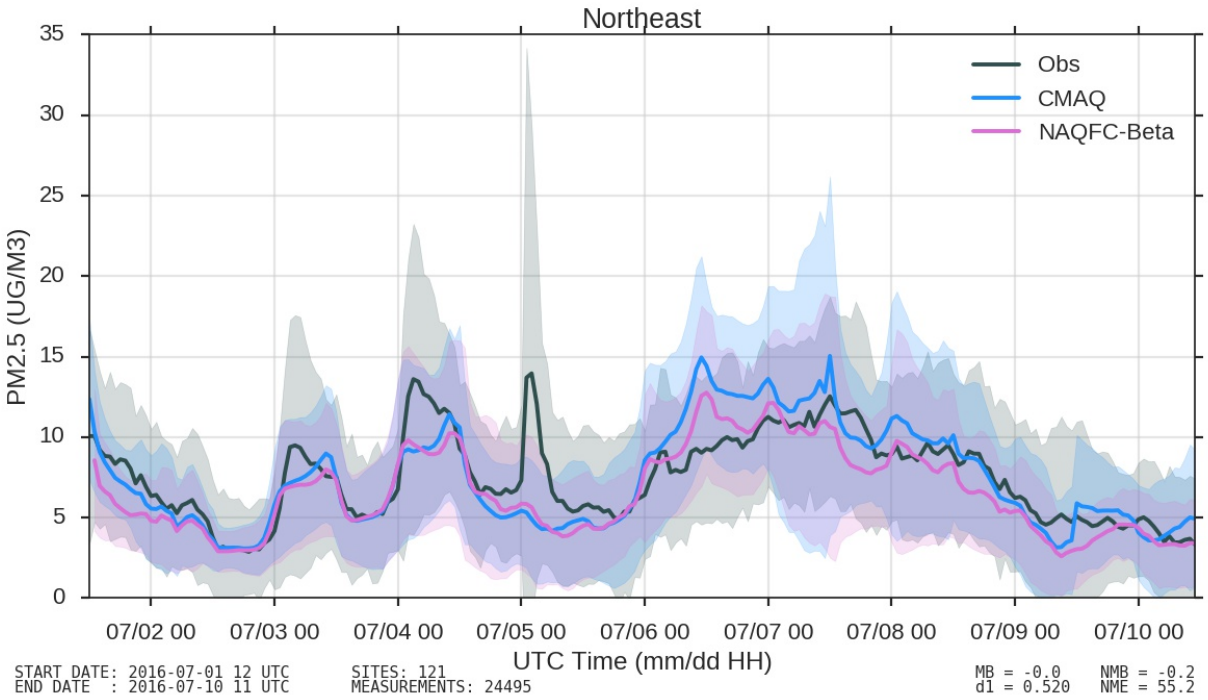


Fig. 1: Time Series

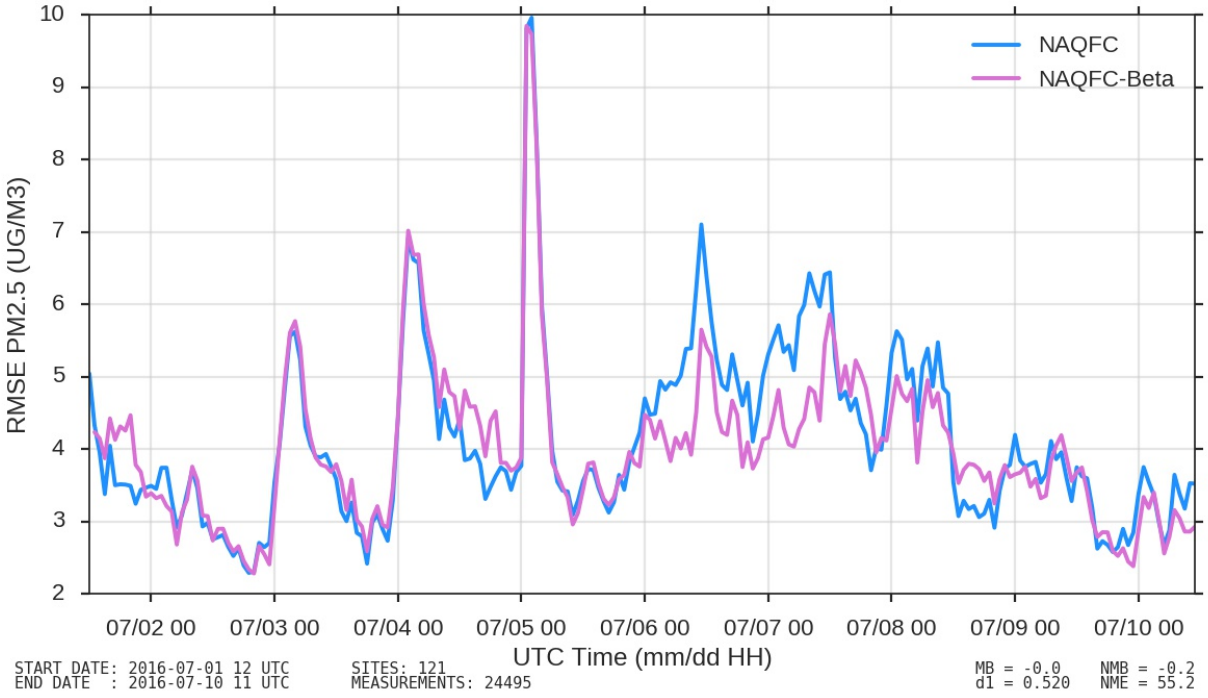


Fig. 2: Time Series of RMSE

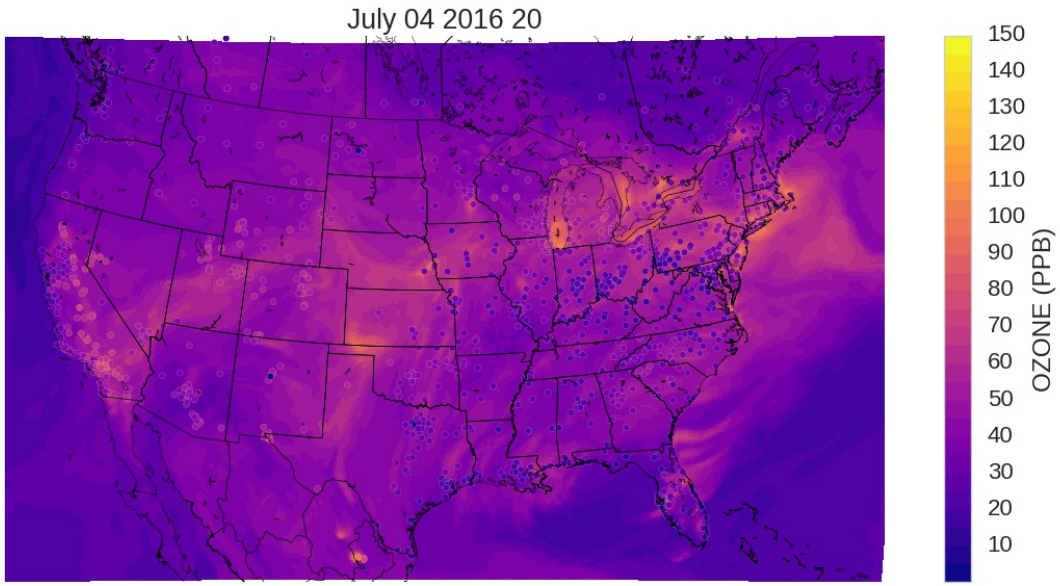
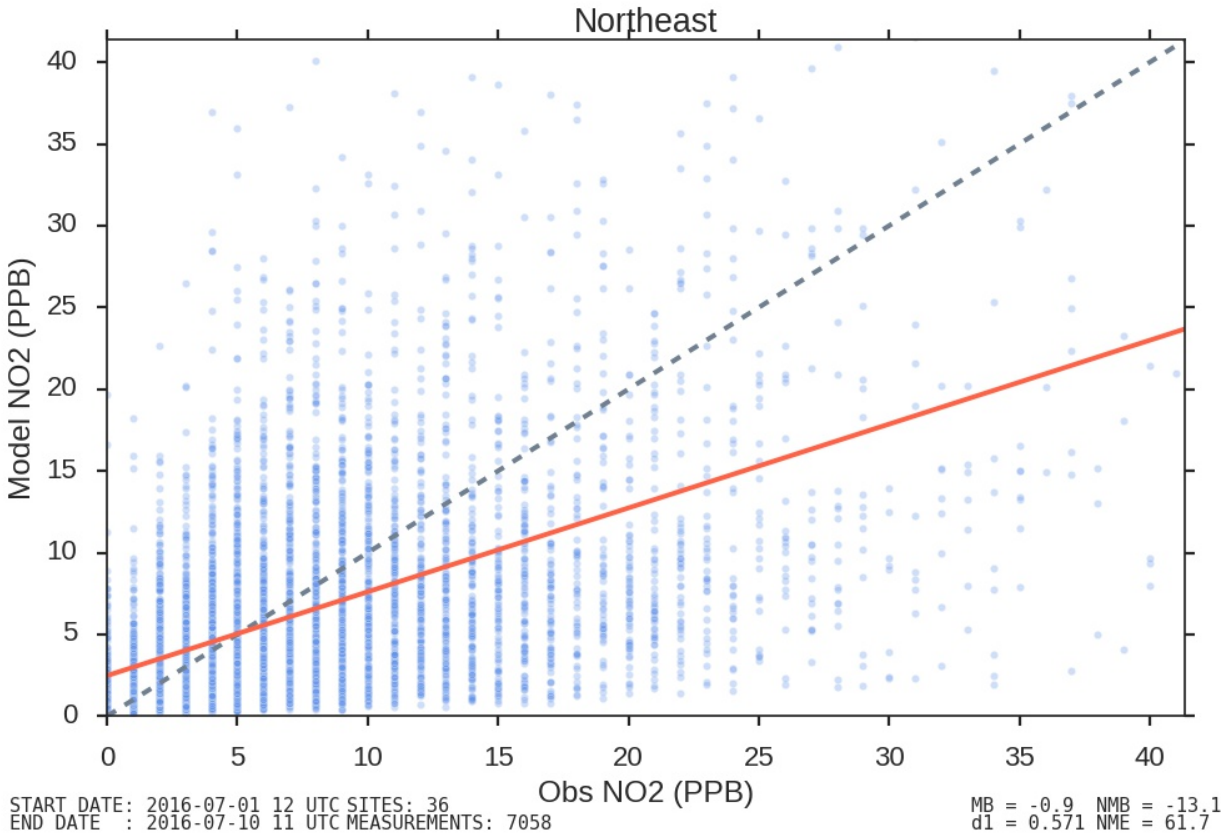
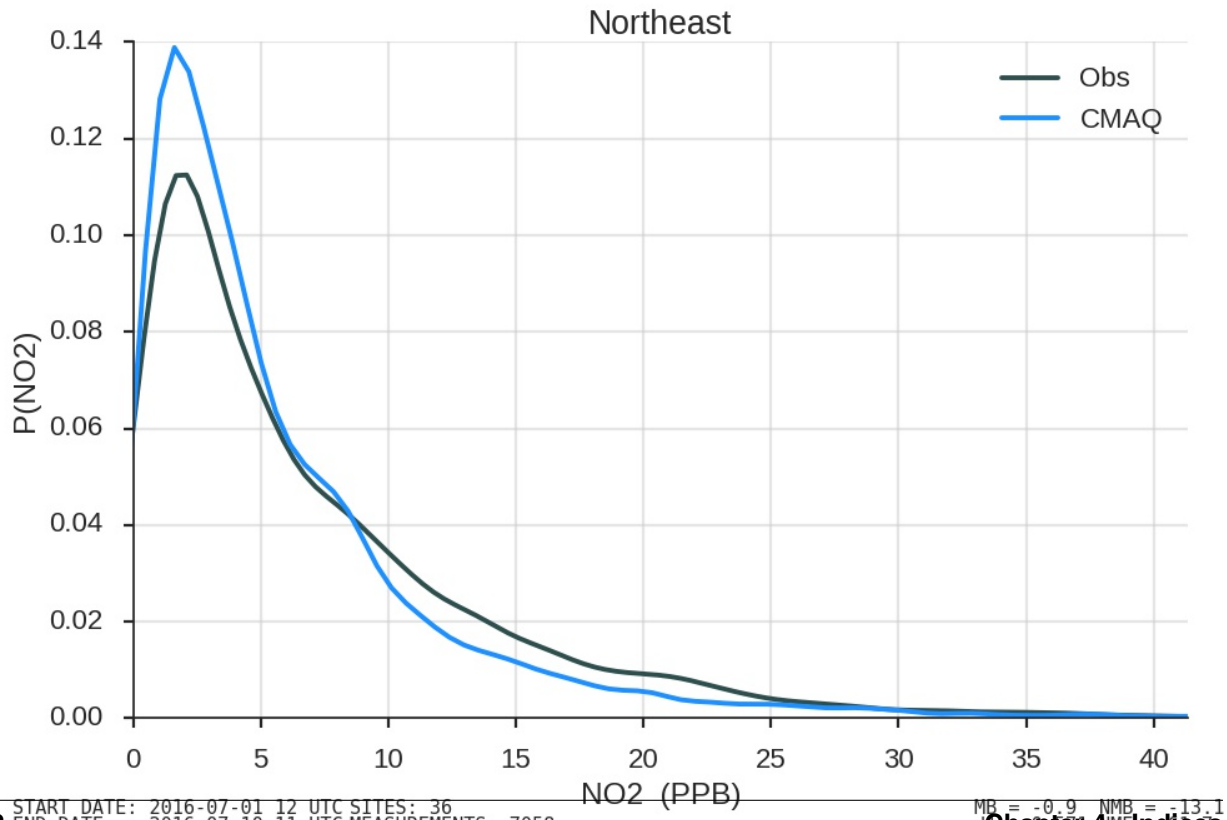
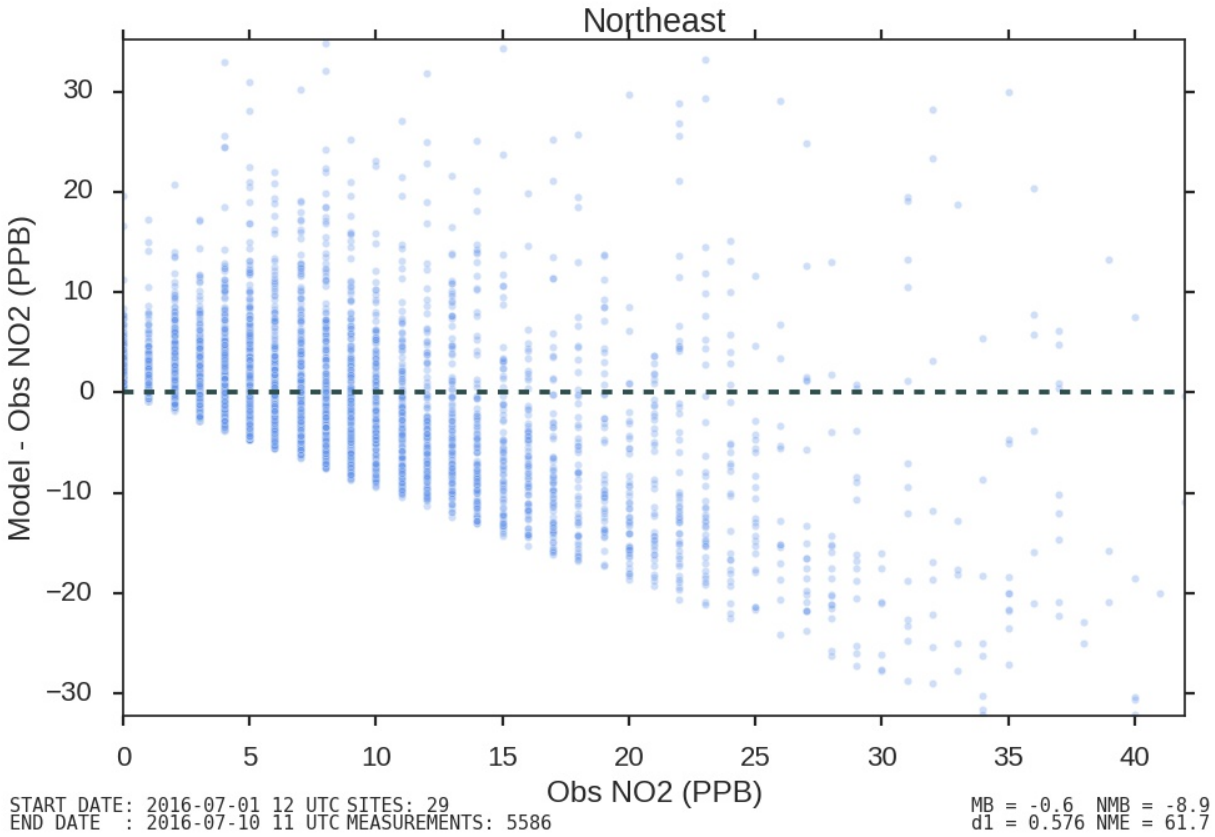


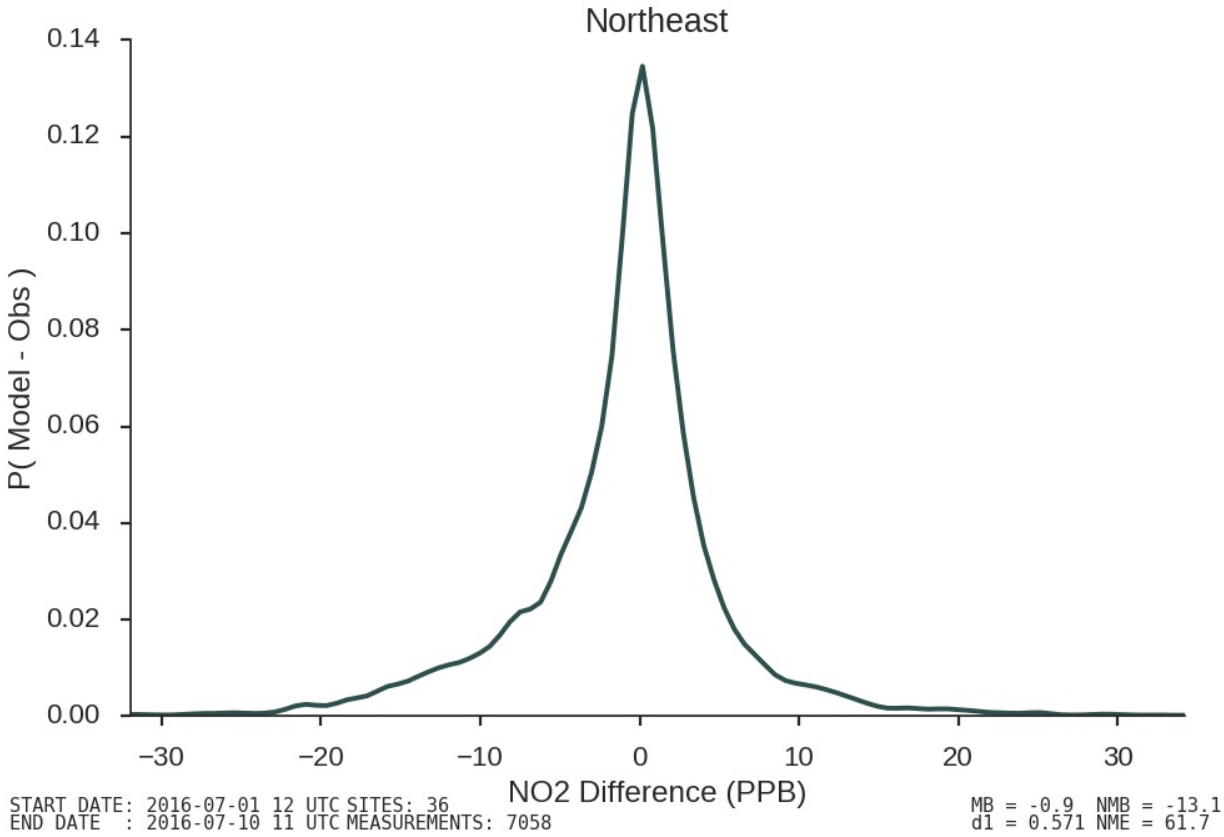
Fig. 3: Spatial Plots

4.1.2 Gallery


Northeast
Model NO2 (PPB)
Obs NO2 (PPB)
0
5
10
15
20
25
30
35
40
START DATE: 2016-07-01 12 UTC SITES: 36
END DATE : 2016-07-10 11 UTC MEASUREMENTS: 7058
MB = -0.9 NMB = -13.1
d1 = 0.571 NME = 61.7


Northeast
Obs
CMAQ
P(NO2)
NO2 (PPB)
0.00
0.02
0.04
0.06
0.08
0.10
0.12
0.14
0
5
10
15
20
25
30
35
40
START DATE: 2016-07-01 12 UTC SITES: 36
END DATE : 2016-07-10 11 UTC MEASUREMENTS: 7058
MB = -0.9 NMB = -13.1


Northeast
Model - Obs NO2 (PPB)
Obs NO2 (PPB)
30
20
10
0
−10
−20
−30
0
5
10
15
20
25
30
35
40
START DATE: 2016-07-01 12 UTC SITES: 29
END DATE : 2016-07-10 11 UTC MEASUREMENTS: 5586
MB = -0.6 NMB = -8.9
d1 = 0.576 NME = 61.7



4.2 Installation

4.2.1 Required dependencies

- Python 2.7[1], 3.4, 3.5, or 3.6
- numpy (1.11 or later)
- pandas (0.18.0 or later)
- xarray (0.10 or later)
- dask
- pseudonetcdf
- xesmf (1.9.0 or later)
- netcdf4
- matplotlib
- seaborn

[1] MONET has dropped support for python 2.7 and requires python 3.6+. For more information see the following references:
- Python 3 Statement
- Tips on porting to Python 3

- cartopy

4.2.1.1 For parallel computing

- dask.array (0.9.0 or later): required for

4.2.1.2 For plotting

- matplotlib: required for plotting
- cartopy: recommended for *plotting maps*
- seaborn: for better color palettes

4.2.2 Instructions

MONET itself is a pure Python package, but some of its dependencies may not be. The simplest way to install MONET is to install it from the conda-forge feedstock:

```
$ conda install -c conda-forge monet
```

This will install all of the dependencies needed by MONET and MONET itself.

xesmf is an optional dependency and can be installed easily from conda-forge, Note xesmf is not available on windows due to the dependency on esmpy and esmf.:

```
$ conda install -c conda-forge monet
```

To install MONET from source code you must install with `pip`. This can be done directly from the GitHub page:

```
$ pip install git+https://github.com/noaa-oar-arl/MONET.git
```

or you can manually download it from GitHub and install it from source:

```
$ git clone https://github.com/noaa-oar-arl/MONET.git
$ cd MONET
$ pip install .
```

4.3 MONET Xarray Accessor

MONET can add georeferencing tools to xarray's data structures through their accessor mechanism. These tools can be accessed via a special `.monet` attribute, *available* for both `xarray.DataArray` and `xarray.Dataset` objects after a simple `import monet` in your code.

4.3.1 Initializing the Accessor

All you have to do is import monet and xarray.

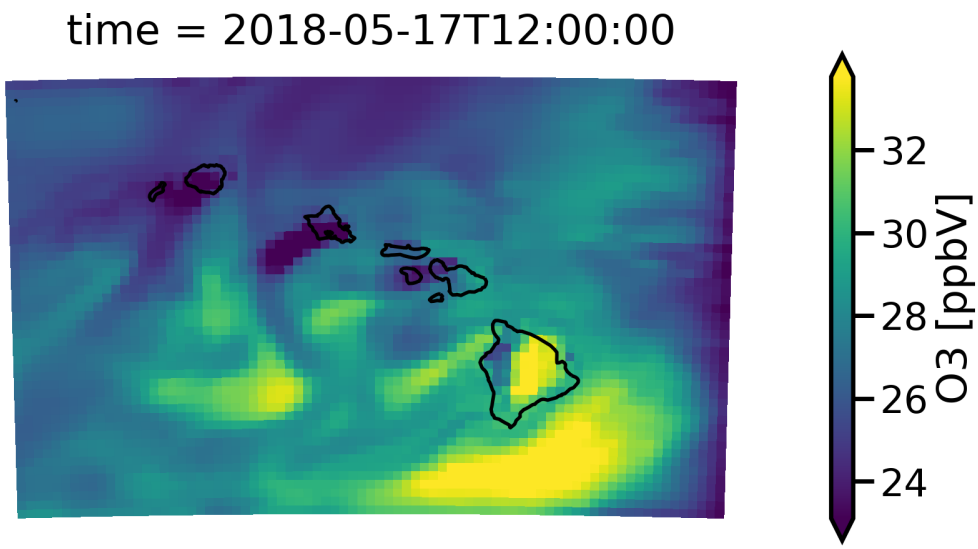
```
import monet
import xarray as xr

cmaqfile = monet.__path__ + '/../data/aqm.t12z.aconc.ncf'

from monet.models import cmaq

c = cmaq.open_dataset(cmaqfile)

c.O3[0,0,:,:].monet.quick_map()
```



4.3.2 Interpolation Methods

The MONET accessor provides several useful interpolation routines including:

- getting the nearest point to a given latitude and longitude
- interpolating to a constant latitude or longitude
- interpolating to a vertical levels
- remapping entire 2D `DataArray` or `DataSet`.

4.3.2.1 Find Nearest Point

To find the nearest latitude/longitude point you just need to use the *nearest_latlon()* method. In this example we will continue to use the CMAQ test file above. We will find the closest grid points to lat=20.5, lon=-157.4.

```
c.monet.nearest_latlon(lat=20.5,lon=-157.4)
```

```
<xarray.Dataset>
Dimensions:    (time: 48, x: 1, y: 1, z: 1)
Coordinates:
* time       (time) datetime64[ns] 2018-05-17T12:00:00 2018-05-17T13:00:00 ...
  latitude   (y, x) float64 dask.array<shape=(1, 1), chunksize=(1, 1)>
  longitude  (y, x) float64 dask.array<shape=(1, 1), chunksize=(1, 1)>
Dimensions without coordinates: x, y, z
Data variables:
  O3         (time, z, y, x) float32 dask.array<shape=(48, 1, 1, 1), chunksize=(48, 1, 1,
↪ 1)>
```

Notice that the length of the dimensions are now (time: 48, x: 1, y: 1, z: 1). If you wanted to only find the nearest location for a single variable you can use the accessor on the `DataArray`.

```
c.O3.monet.nearest_latlon(lat=20.5,lon=-157.4)
```

```
<xarray.DataArray 'O3' (time: 48, z: 1, y: 1, x: 1)>
dask.array<shape=(48, 1, 1, 1), dtype=float32, chunksize=(48, 1, 1, 1)>
Coordinates:
 * time       (time) datetime64[ns] 2018-05-17T12:00:00 2018-05-17T13:00:00 ...
   latitude   (y, x) float64 dask.array<shape=(1, 1), chunksize=(1, 1)>
   longitude  (y, x) float64 dask.array<shape=(1, 1), chunksize=(1, 1)>
Dimensions without coordinates: z, y, x
Attributes:
   long_name:   O3
   units:       ppbV
   var_desc:    Variable O3
   _FillValue:  nan
```

4.4 Installing on WCOSS

If you have access to the NOAA WCOSS machines you can create your own Python environments very easily using Intel Python.:

```
module load ips/19.0.5.281
```

Then it is suggested to create a .condarc file that lives in your `$HOME` folder to point to a location that will house your Conda environments. Below is a sample `.condarc` file.

```
channels:
  - intel
  - conda-forge
  - defaults
envs_dirs:
```

(continues on next page)

(continued from previous page)

```
  - /gpfs/dell2/emc/verification/noscrub/User.Name/conda/envs
pkgs_dirs:
  - /gpfs/dell2/emc/verification/noscrub/User.Name/conda/pkgs
```

Next you should start a new environment by cloning the default environment to a new name. This can be done in a single command.

```
conda create -n myenv --clone="/usrx/local/prod/intel/2019UP05/intelpython3"
```

A prompt should come up and tell you to activate the environment you just created, myenv.

```
source activate myenv
```

From here you can install any package the same way you could on regular Anaconda installations.

```
conda install -c conda-forge monet
```

4.5 Tutorial

Examples have been moved to the MONETIO docs.

4.6 Get in Touch

Ask questions, suggest features or view source code on GitHub.

If an issue arrises please post on the GitHub issues.

4.7 API

4.7.1 Top-level functions

monet.dataset_to_monet(dset[, lat_name, ...])	Rename xarray DataArray or Dataset coordinate variables for use with monet functions, returning a new xarray object.
monet.rename_to_monet_latlon(ds)	Rename latitude/longitude variants to `lat`/`lon`, returning a new xarray object.
monet.rename_latlon(ds)	Rename latitude/longitude variants to `lat`/`lon`, returning a new xarray object.

4.7.1.1 monet.dataset_to_monet

monet.**dataset_to_monet**(*dset*, *lat_name='latitude'*, *lon_name='longitude'*, *latlon2d=False*)

Rename xarray DataArray or Dataset coordinate variables for use with monet functions, returning a new xarray object.

Parameters

- **dset** (*xarray.DataArray or xarray.Dataset*) – A given data obj to be renamed for monet.
- **lat_name** (*str*) – Name of the latitude array.
- **lon_name** (*str*) – Name of the longitude array.
- **latlon2d** (*bool*) – Whether the latitude and longitude data is two-dimensional.

4.7.1.2 monet.rename_to_monet_latlon

monet.**rename_to_monet_latlon**(*ds*)

Rename latitude/longitude variants to `lat`/`lon`, returning a new xarray object.

Parameters **ds** (*xarray.DataArray or xarray.Dataset*)

4.7.1.3 monet.rename_latlon

monet.**rename_latlon**(*ds*)

Rename latitude/longitude variants to `lat`/`lon`, returning a new xarray object.

Parameters **ds** (*xarray.DataArray or xarray.Dataset*)

4.7.2 Modules

monet.met_funcs	This package contains the main routines for estimating variables related to the Monin-Obukhov (MO) Similarity Theory, such as MO length, adiabatic correctors for heat and momentum transport.
monet.plots	
monet.util	

4.7.2.1 monet.met_funcs

This package contains the main routines for estimating variables related to the Monin-Obukhov (MO) Similarity Theory, such as MO length, adiabatic correctors for heat and momentum transport. It requires the following package.

References

Functions

calc_L(ustar, T_A_K, rho, c_p, H, LE)	Calculates the Monin-Obukhov length.
calc_Psi_H(zoL)	Calculates the adiabatic correction factor for heat transport.
calc_Psi_M(zoL)	Adiabatic correction factor for momentum transport.
calc_c_p(p, ea)	Calculates the heat capacity of air at constant pressure.
calc_delta_vapor_pressure(T_K)	Calculate the slope of saturation water vapour pressure.
calc_lambda(T_A_K)	Calculates the latent heat of vaporization.
calc_lapse_rate_moist(T_A_K, ea, p)	Calculate moist-adiabatic lapse rate (K/m)
calc_mixing_ratio(ea, p)	Calculate ratio of mass of water vapour to the mass of dry air (-)
calc_pressure(z)	Calculates the barometric pressure above sea level.
calc_psicr(c_p, p, Lambda)	Calculates the psicrometric constant.
calc_rho(p, ea, T_A_K)	Calculates the density of air.
calc_richardson(u, z_u, d_0, T_R0, T_R1, ...)	Richardson number.
calc_stephan_boltzmann(T_K)	Calculates the total energy radiated by a blackbody.
calc_sun_angles(lat, lon, stdlon, doy, ftime)	Calculates the Sun Zenith and Azimuth Angles (SZA & SAA).
calc_theta_s(xlat, xlong, stdlng, doy, year, ...)	Calculates the Sun Zenith Angle (SZA).
calc_u_star(u, z_u, L, d_0, z_0M)	Friction velocity.
calc_vapor_pressure(T_K)	Calculate the saturation water vapour pressure.
flux_2_evaporation(flux[, T_K, time_domain])	Converts heat flux units (W m-2) to evaporation rates (mm time-1) to a given temporal window

monet.met_funcs.**calc_L**(*ustar*, *T_A_K*, *rho*, *c_p*, *H*, *LE*)

Calculates the Monin-Obukhov length.

Parameters

- **ustar** (*float*) – friction velocity (m s-1).
- **T_A_K** (*float*) – air temperature (Kelvin).
- **rho** (*float*) – air density (kg m-3).
- **c_p** (*float*) – Heat capacity of air at constant pressure (J kg-1 K-1).
- **H** (*float*) – sensible heat flux (W m-2).
- **LE** (*float*) – latent heat flux (W m-2).

Returns **L** – Obukhov stability length (m).

Return type float

References

[Brutsaert2005]

monet.met_funcs.**calc_Psi_H**(*zoL*)

Calculates the adiabatic correction factor for heat transport.

Parameters **zoL** (*float*) – stability coefficient (unitless).

Returns **Psi_H** – adiabatic corrector factor for heat transport (unitless).

Return type float

References

[Brutsaert2005]

monet.met_funcs.**calc_Psi_M**(*zoL*)

Adiabatic correction factor for momentum transport.

Parameters **zoL** (*float*) – stability coefficient (unitless).

Returns **Psi_M** – adiabatic corrector factor for momentum transport (unitless).

Return type float

References

[Brutsaert2005]

monet.met_funcs.**calc_c_p**(*p*, *ea*)

Calculates the heat capacity of air at constant pressure.

Parameters

- **p** (*float*) – total air pressure (dry air + water vapour) (mb).
- **ea** (*float*) – water vapor pressure at reference height above canopy (mb).

Returns **c_p**

Return type heat capacity of (moist) air at constant pressure (J kg-1 K-1).

References

based on equation (6.1) from Maarten Ambaum (2010): Thermal Physics of the Atmosphere (pp 109).

monet.met_funcs.**calc_delta_vapor_pressure**(*T_K*)

Calculate the slope of saturation water vapour pressure.

Parameters **T_K** (*float*) – temperature (K).

Returns **s** – slope of the saturation water vapour pressure (kPa K-1)

Return type float

monet.met_funcs.**calc_lambda**(*T_A_K*)

Calculates the latent heat of vaporization.

Parameters **T_A_K** (*float*) – Air temperature (Kelvin).

Returns **Lambda** – Latent heat of vaporisation (J kg-1).

Return type float

References

based on Eq. 3-1 Allen FAO98

monet.met_funcs.**calc_lapse_rate_moist**(*T_A_K*, *ea*, *p*)

Calculate moist-adiabatic lapse rate (K/m)

Parameters

- **T_A_K** (*float or numpy array*) – air temperature at reference height (K).
- **ea** (*float or numpy array*) – water vapor pressure at reference height (mb).
- **p** (*float or numpy array*) – total air pressure (dry air + water vapour) at reference height (mb).

Returns **Gamma_w** – moist-adiabatic lapse rate (K/m)

Return type float or numpy array

References

https://glossary.ametsoc.org/wiki/Adiabatic_lapse_rate

monet.met_funcs.**calc_mixing_ratio**(*ea*, *p*)

Calculate ratio of mass of water vapour to the mass of dry air (-)

Parameters

- **ea** (*float or numpy array*) – water vapor pressure at reference height (mb).
- **p** (*float or numpy array*) – total air pressure (dry air + water vapour) at reference height (mb).

Returns **r** – mixing ratio (-)

Return type float or numpy array

References

https://glossary.ametsoc.org/wiki/Mixing_ratio

monet.met_funcs.**calc_pressure**(*z*)

Calculates the barometric pressure above sea level.

Parameters **z** (*float*) – height above sea level (m).

Returns **p** – air pressure (mb).

Return type float

monet.met_funcs.**calc_psicr**(*c_p*, *p*, *Lambda*)

Calculates the psicrometric constant.

Parameters

- **c_p** (*float*) – heat capacity of (moist) air at constant pressure (J kg-1 K-1).
- **p** (*float*) – atmopheric pressure (mb).
- **Lambda** (*float*) – latent heat of vaporzation (J kg-1).

Returns **psicr** – Psicrometric constant (mb C-1).

Return type float

monet.met_funcs.**calc_rho**(*p*, *ea*, *T_A_K*)

Calculates the density of air.

Parameters

- **p** (*float*) – total air pressure (dry air + water vapour) (mb).
- **ea** (*float*) – water vapor pressure at reference height above canopy (mb).
- **T_A_K** (*float*) – air temperature at reference height (Kelvin).

Returns **rho** – density of air (kg m-3).

Return type float

References

based on equation (2.6) from Brutsaert (2005): Hydrology - An Introduction (pp 25).

monet.met_funcs.**calc_richardson**(*u*, *z_u*, *d_0*, *T_R0*, *T_R1*, *T_A0*, *T_A1*)

Richardson number.

Estimates the Bulk Richardson number for turbulence using time difference temperatures.

Parameters

- **u** (*float*) – Wind speed (m s-1).
- **z_u** (*float*) – Wind speed measurement height (m).
- **d_0** (*float*) – Zero-plane displacement height (m).
- **T_R0** (*float*) – radiometric surface temperature at time 0 (K).
- **T_R1** (*float*) – radiometric surface temperature at time 1 (K).
- **T_A0** (*float*) – air temperature at time 0 (K).
- **T_A1** (*float*) – air temperature at time 1 (K).

Returns **Ri** – Richardson number.

Return type float

References

[Norman2000]

monet.met_funcs.**calc_stephan_boltzmann**(*T_K*)

Calculates the total energy radiated by a blackbody.

Parameters **T_K** (*float*) – body temperature (Kelvin)

Returns **M** – Emitted radiance (W m-2)

Return type float

monet.met_funcs.**calc_sun_angles**(*lat*, *lon*, *stdlon*, *doy*, *ftime*)

Calculates the Sun Zenith and Azimuth Angles (SZA & SAA).

Parameters

- **lat** (*float*) – latitude of the site (degrees).
- **long** (*float*) – longitude of the site (degrees).
- **stdlng** (*float*) – central longitude of the time zone of the site (degrees).
- **doy** (*float*) – day of year of measurement (1-366).
- **ftime** (*float*) – time of measurement (decimal hours).

Returns

- **sza** (*float*) – Sun Zenith Angle (degrees).
- **saa** (*float*) – Sun Azimuth Angle (degrees).

monet.met_funcs.**calc_theta_s**(*xlat*, *xlong*, *stdlng*, *doy*, *year*, *ftime*)

Calculates the Sun Zenith Angle (SZA).

Parameters

- **xlat** (*float*) – latitude of the site (degrees).
- **xlong** (*float*) – longitude of the site (degrees).
- **stdlng** (*float*) – central longitude of the time zone of the site (degrees).
- **doy** (*float*) – day of year of measurement (1-366).
- **year** (*float*) – year of measurement .
- **ftime** (*float*) – time of measurement (decimal hours).

Returns **theta_s** – Sun Zenith Angle (degrees).

Return type float

References

Adopted from Martha Anderson's fortran code for ALEXI which in turn was based on Cupid.

monet.met_funcs.**calc_u_star**(*u*, *z_u*, *L*, *d_0*, *z_0M*)

Friction velocity.

Parameters

- **u** (*float*) – wind speed above the surface (m s-1).
- **z_u** (*float*) – wind speed measurement height (m).
- **L** (*float*) – Monin Obukhov stability length (m).
- **d_0** (*float*) – zero-plane displacement height (m).
- **z_0M** (*float*) – aerodynamic roughness length for momentum transport (m).

References

[Brutsaert2005]

monet.met_funcs.**calc_vapor_pressure**(*T_K*)

Calculate the saturation water vapour pressure.

Parameters **T_K** (*float*) – temperature (K).

Returns **ea** – saturation water vapour pressure (mb).

Return type float

monet.met_funcs.**flux_2_evaporation**(*flux*, *T_K=293.15*, *time_domain=1*)

Converts heat flux units (W m-2) to evaporation rates (mm time-1) to a given temporal window

Parameters

- **flux** (*float or numpy array*) – heat flux value to be converted, usually refers to latent heat flux LE to be converted to ET
- **T_K** (*float or numpy array*) – environmental temperature in Kelvin. Default=20 Celsius
- **time_domain** (*float*) – Temporal window in hours. Default 1 hour (mm h-1)

Returns **ET** – evaporation rate at the time_domain. Default mm h-1

Return type float or numpy array

4.7.2.2 monet.plots

Functions

savefig(fname, *[, loc, decorate, logo, ...])	Save figure and add logo.
sp_scatter_bias(df[, col1, col2, ax, ...])	

monet.plots.**cmap_discretize**(*cmap*, *N*)

Return a discrete colormap from the continuous colormap cmap.

cmap: colormap instance, eg. cm.jet. N: number of colors.

Example x = resize(arange(100), (5,100)) djet = cmap_discretize(cm.jet, 5) imshow(x, cmap=djet)

monet.plots.**kdeplot**(*df*, *title=None*, *label=None*, *ax=None*, ***kwargs*)

Short summary.

Parameters

- **df** (*type*) – Description of parameter *df*.
- **col** (*type*) – Description of parameter *col* (the default is 'obs').
- **title** (*type*) – Description of parameter *title* (the default is None).
- **label** (*type*) – Description of parameter *label* (the default is None).
- **ax** (*type*) – Description of parameter *ax* (the default is ax).
- ****kwargs** (*type*) – Description of parameter ***kwargs*.

Returns Description of returned object.

Return type type

monet.plots.**savefig**(*fname*, ***, *loc=1*, *decorate=True*, *logo=None*, *logo_height=None*, ***kwargs*)

Save figure and add logo.

Parameters

- **fname** (*str*) – Output file name or path. Passed to `plt.savefig`. Must include desired file extension (`.jpg` or `.png`).
- **loc** (*int*) – The location for the logo.
  - 1 – bottom left (default)
  - 2 – bottom right
  - 3 – top right
  - 4 – top left
- **decorate** (*bool, default: True*) – Whether to add the logo.
- **logo** (*str, optional*) – Path to the logo to be used. If not provided, the MONET logo is used.
- **logo_height** (*float or int, optional*) – Desired logo height in pixels. If not provided, the original logo image dimensions are used. Modify to scale the logo.
- ****kwargs** (*dict*) – Passed to the `plt.savefig` function.

Return type None

monet.plots.**scatter**(*df*, *x=None*, *y=None*, *title=None*, *label=None*, *ax=None*, ***kwargs*)

Short summary.

Parameters

- **df** (*type*) – Description of parameter *df*.
- **x** (*type*) – Description of parameter *x* (the default is 'obs').
- **y** (*type*) – Description of parameter *y* (the default is 'model').

- ****kwargs** (*type*) – Description of parameter ***kwargs*.

Returns Description of returned object.

Return type type

monet.plots.**timeseries**(*df*, *x='time'*, *y='obs'*, *ax=None*, *plotargs={}*, *fillargs={'alpha': 0.2}*, *title=''*, *ylabel=None*, *label=None*)

Short summary.

Parameters

- **df** (*type*) – Description of parameter *df*.
- **col** (*type*) – Description of parameter *col* (the default is 'Obs').
- **ax** (*type*) – Description of parameter *ax* (the default is None).
- **sample** (*type*) – Description of parameter *sample* (the default is 'H').
- **plotargs** (*type*) – Description of parameter *plotargs* (the default is {}).
- **fillargs** (*type*) – Description of parameter *fillargs* (the default is {}).
- **title** (*type*) – Description of parameter *title* (the default is '').
- **label** (*type*) – Description of parameter *label* (the default is None).

Returns Description of returned object.

Return type type

Modules

monet.plots.colorbars	Colorbar helper functions
monet.plots.mapgen	map utilities
monet.plots.plots	plotting routines
monet.plots.taylordiagram(df[, marker, ...])	

monet.plots.colorbars

Colorbar helper functions

Functions

cmap_discretize(cmap, N)	Return a discrete colormap from the continuous colormap cmap.
colorbar_index(ncolors, cmap[, minval, ...])	

monet.plots.colorbars.**cmap_discretize**(*cmap*, *N*)

Return a discrete colormap from the continuous colormap cmap.

cmap: colormap instance, eg. cm.jet. N: number of colors.

Example x = resize(arange(100), (5,100)) djet = cmap_discretize(cm.jet, 5) imshow(x, cmap=djet)

monet.plots.mapgen

map utilities

Functions

draw_map([crs, natural_earth, coastlines, ...])	Short summary.

monet.plots.mapgen.**draw_map**(*crs=None*, *natural_earth=False*, *coastlines=True*, *states=False*, *countries=True*, *resolution='10m'*, *extent=None*, *figsize=(10, 5)*, *linewidth=0.25*, *return_fig=False*, ***kwargs*)

Short summary.

Parameters

- **ax** (*type*) – Description of parameter *ax* (the default is None).
- **natural_earth** (*bool*) – Description of parameter *natural_earth* (the default is True).
- **coastlines** (*bool*) – Description of parameter *coastlines* (the default is True).
- **states** (*bool*) – Description of parameter *states* (the default is True).
- **countries** (*bool*) – Description of parameter *countries* (the default is True).
- **state_resolutions** (*bool*) – Description of parameter *state_resolutions* (the default is '10m').
- **extent** (*[lon_min,lon_max,lat_min,lat_max]*) – Description of parameter *extent* (the default is None).

Returns Description of returned object.

Return type type

monet.plots.plots

plotting routines

Functions

kdeplot(df[, title, label, ax])	Short summary.
make_spatial_contours(modelvar, gridobj, date, m)	
make_spatial_plot(modelvar, m[, dpi, ...])	
normval(vmin, vmax, cmap)	
scatter(df[, x, y, title, label, ax])	Short summary.
spatial(modelvar, **kwargs)	
spatial_bias_scatter(df, m, date[, vmin, ...])	
taylordiagram(df[, marker, col1, col2, ...])	
timeseries(df[, x, y, ax, plotargs, ...])	Short summary.
wind_barbs(ws, wdir, gridobj, m, **kwargs)	
wind_quiver(ws, wdir, gridobj, m, **kwargs)	

monet.plots.plots.**kdeplot**(*df*, *title=None*, *label=None*, *ax=None*, ***kwargs*)

Short summary.

Parameters

- **df** (*type*) – Description of parameter *df*.
- **col** (*type*) – Description of parameter *col* (the default is 'obs').
- **title** (*type*) – Description of parameter *title* (the default is None).
- **label** (*type*) – Description of parameter *label* (the default is None).
- **ax** (*type*) – Description of parameter *ax* (the default is ax).
- ****kwargs** (*type*) – Description of parameter ***kwargs*.

Returns Description of returned object.

Return type type

monet.plots.plots.**scatter**(*df*, *x=None*, *y=None*, *title=None*, *label=None*, *ax=None*, ***kwargs*)

Short summary.

Parameters

- **df** (*type*) – Description of parameter *df*.
- **x** (*type*) – Description of parameter *x* (the default is 'obs').
- **y** (*type*) – Description of parameter *y* (the default is 'model').
- ****kwargs** (*type*) – Description of parameter ***kwargs*.

Returns Description of returned object.

Return type type

monet.plots.plots.**timeseries**(*df*, *x='time'*, *y='obs'*, *ax=None*, *plotargs={}*, *fillargs={'alpha': 0.2}*, *title=''*, *ylabel=None*, *label=None*)

Short summary.

Parameters

- **df** (*type*) – Description of parameter *df*.
- **col** (*type*) – Description of parameter *col* (the default is 'Obs').
- **ax** (*type*) – Description of parameter *ax* (the default is None).
- **sample** (*type*) – Description of parameter *sample* (the default is 'H').
- **plotargs** (*type*) – Description of parameter *plotargs* (the default is {}).
- **fillargs** (*type*) – Description of parameter *fillargs* (the default is {}).
- **title** (*type*) – Description of parameter *title* (the default is '').
- **label** (*type*) – Description of parameter *label* (the default is None).

Returns Description of returned object.

Return type type

monet.plots.taylordiagram

monet.plots.**taylordiagram**(*df*, *marker='o'*, *col1='obs'*, *col2='model'*, *label1='OBS'*, *label2='MODEL'*, *scale=1.5*, *addon=False*, *dia=None*)

4.7.2.3 monet.util

Functions

calc_13_category_usda_soil_type(clay, sand, silt)	Calculate the 13 category usda soil type from the clay sand and silt
calc_24hr_ave(df[, col])	
calc_3hr_ave(df[, col])	
calc_8hr_rolling_max(df[, col, window])	
calc_annual_ave(df[, col])	
findclosest(list, value)	
get_giorgi_region_bounds([index, acronym])	
get_giorgi_region_df(df)	
kolmogorov_zurbenko_filter(df, window, ...)	
linregress(x, y)	
long_to_wide(df)	
nearest(items, pivot)	
search_listinlist(array1, array2)	
wsdir2uv(ws, wdir)	

Modules

monet.util.combinetool	
monet.util.interp_util	Interpolation functions
monet.util.resample	
monet.util.stats	
monet.util.tools	

monet.util.combinetool

Functions

combine_da_to_da(source, target, *[, merge, ...])	Combine xarray data array *source* with with point observations in second data array *target*, returning a new xarray object.
combine_da_to_df(da, df, *[, merge])	Combine xarray data array *da* with spatial information point observations in dataframe *df*, returning a new dataframe.
combine_da_to_df_xesmf(da, df, *[, suffix])	Combine xarray data array *da* with spatial information point observations in dataframe *df*, returning a new dataframe.
combine_da_to_df_xesmf_strat(da, daz, df, ...)	This function will combine an xarray data array with spatial information point observations in *df*.
combine_da_to_height_profile(da, dset, *[, ...])	This function will combine an xarray.DataArray to a 2d dataset with dimensions (time,z)

monet.util.combinetool.**combine_da_to_da**(*source*, *target*, *, *merge=True*, *interp_time=False*, ***kwargs*)

Combine xarray data array *source* with with point observations in second data array *target*, returning a new xarray object.

Uses pyresample nearest-neighbor via `.monet.remap_nearest`.

Parameters

- **source** (*xarray.DataArray or xarray.Dataset*) – Gridded data.
- **target** (*xarray.DataArray*) – Point observations.
- **merge** (*bool*) – If false, only return the interpolated source data. If true, merge with the target data.
- **interp_time** (*bool*) – Linearly interpolate to `target.time`.
- **kwargs** (*dict*) – Passed on to `remap_nearest()`.

Return type xarray.Dataset

monet.util.combinetool.**combine_da_to_df**(*da*, *df*, *, *merge=True*, ***kwargs*)

Combine xarray data array *da* with spatial information point observations in dataframe *df*, returning a new dataframe.

Uses pyresample via `.monet.remap_nearest`.

Parameters

- **da** (*xarray.DataArray or xarray.Dataset*) – Data to be interpolated to target grid points. Can be unstructured-grid data (detected by checking `'mio_has_unstructured_grid'` attribute).
- **df** (*pandas.DataFrame*) – Data on target points.
- **merge** (*bool*) – Merge interpolated *df* data with *da* data. Otherwise, return interpolated *da* data only.
- **kwargs** (*dict*) – Passed to `remap_nearest()` (if *da* is not unstructured-grid data).

Return type pandas.DataFrame

monet.util.combinetool.**combine_da_to_df_xesmf**(*da*, *df*, *, *suffix=None*, ***kwargs*)

Combine xarray data array *da* with spatial information point observations in dataframe *df*, returning a new dataframe.

Uses `resample_xesmf()`.

Parameters

- **da** (*xarray.DataArray or xarray.Dataset*) – Data to be interpolated to target grid points.
- **df** (*pandas.DataFrame*) – Data on target points.
- **suffix** (*str, optional*) – Added to the `name` of the new variable, defaults to `'_new'`.
- **kwargs** (*dict*) – Passed on to `resample_xesmf()` (and then to `xesmf.Regridder`).

Return type pandas.DataFrame

monet.util.combinetool.**combine_da_to_df_xesmf_strat**(*da*, *daz*, *df*, ***kwargs*)

This function will combine an xarray data array with spatial information point observations in *df*.

Parameters

- **da** (*xarray.DataArray*) – Data to interpolate.
- **daz** (*xarray.DataArray*) – Vertical coordinate data variable. Must have same shape as *da*.
- **df** (*pandas.DataFrame*) – Point data.
- **kwargs** (*dict*) – Passed on to `resample_xesmf()` (and then to `xesmf.Regridder`).

Return type pandas.DataFrame

monet.util.combinetool.**combine_da_to_height_profile**(*da*, *dset*, *, *radius_of_influence=12000.0*)

This function will combine an xarray.DataArray to a 2d dataset with dimensions (time,z)

Parameters

- **da** (*xarray.DataArray*)
- **dset** (*xarray.Dataset*)

Returns Returns the xarray.Dataset with the *da* added as an additional variable.

Return type xarray.Dataset

monet.util.interp_util

Interpolation functions

Functions

constant_1d_xesmf([longitude, latitude])	Creates a pyreample.geometry.SwathDefinition with a constant latitude along the longitude array.
constant_lat_swathdefition([longitude, latitude])	Creates a pyreample.geometry.SwathDefinition with a constant latitude along the longitude array.
constant_lon_swathdefition([longitude, latitude])	Creates a pyreample.geometry.SwathDefinition with a constant longitude along the latitude array.
latlon_xarray_to_CoordinateDefinition([...])	Create pyresample SwathDefinition from xarray object.
lonlat_to_swathdefinition([longitude, latitude])	Short summary.
lonlat_to_xesmf([longitude, latitude])	Creates an empty xarray.Dataset with the coordinate (longitude, latitude).
nearest_point_swathdefinition([longitude, ...])	Creates a pyreample.geometry.SwathDefinition for a single point.

monet.util.interp_util.**constant_1d_xesmf**(*longitude=None*, *latitude=None*)

Creates a pyreample.geometry.SwathDefinition with a constant latitude along the longitude array. Longitude can be a 1d or 2d np.array or xr.DataArray

Parameters

- **longitude** (*numpy.array or xarray.DataArray*) – Array of longitude values
- **latitude** (*float*) – latitude for constant

Return type pyreample.geometry.SwathDefinition

monet.util.interp_util.**constant_lat_swathdefition**(*longitude=None*, *latitude=None*)

Creates a pyreample.geometry.SwathDefinition with a constant latitude along the longitude array. Longitude can be a 1d or 2d np.array or xr.DataArray

Parameters

- **longitude** (*numpy.array or xarray.DataArray*) – Array of longitude values
- **latitude** (*float*) – latitude for constant

Return type pyreample.geometry.SwathDefinition

monet.util.interp_util.**constant_lon_swathdefition**(*longitude=None*, *latitude=None*)

Creates a pyreample.geometry.SwathDefinition with a constant longitude along the latitude array. latitude can be a 1d or 2d np.array or xr.DataArray

Parameters

- **longitude** – latitude for constant
- **latitude** (*numpy.array or xarray.DataArray*) – Array of longitude values

Return type pyreample.geometry.SwathDefinition

monet.util.interp_util.**latlon_xarray_to_CoordinateDefinition**(*longitude=None*, *latitude=None*)

Create pyresample SwathDefinition from xarray object.

Parameters

- **longitude** (*2d xarray.DataArray*) – Longitude -> must be from -180 -> 180 and monotonically increasing
- **latitude** (*2d xarray.DataArray*) – Latitude -> must be from -90 -> 90 and monotonically increasing

Return type pyresample.CoordinateDefinition

monet.util.interp_util.**lonlat_to_swathdefinition**(*longitude=None*, *latitude=None*)

Short summary.

Parameters

- **longitude** (*type*) – Description of parameter *longitude*.
- **latitude** (*type*) – Description of parameter *latitude*.

Returns Description of returned object.

Return type type

monet.util.interp_util.**lonlat_to_xesmf**(*longitude=None*, *latitude=None*)

Creates an empty xarray.Dataset with the coordinate (longitude, latitude).

Parameters

- **longitude** (*type*) – Description of parameter *longitude*.
- **latitude** (*type*) – Description of parameter *latitude*.

Returns Description of returned object.

Return type type

monet.util.interp_util.**nearest_point_swathdefinition**(*longitude=None*, *latitude=None*)

Creates a pyreample.geometry.SwathDefinition for a single point.

Parameters

- **longitude** (*float*) – longitude.
- **latitude** (*float*) – latitude.

Return type pyreample.geometry.SwathDefinition

monet.util.resample

Functions

resample_stratify(da, levels, vertical[, axis])	
resample_xesmf(source_da, target_da[, cleanup])	

monet.util.stats

Functions

AC(obs, mod[, axis])	Anomaly Correlation
CSI(obs, mod, minval, maxval)	Critical Success Index (1 is perfect - Range 0 -> 1)
E1(obs, mod[, axis])	Modified Coefficient of Efficiency, E1
ETS(obs, mod, minval, maxval)	Equitable Threat Score (1 is perfect - Range -1/3 -> 1)
FB(obs, mod[, axis])	Fractional Bias (%)

continues on next page

Table 1 – continued from previous page

FE(obs, mod[, axis])	Fractional Error (%)
HSS(obs, mod, minval, maxval)	Heidke Skill Score (1 is perfect - below zero means no confidence)
IOA(obs, mod[, axis])	Index of Agreement, IOA
IOA_m(obs, mod[, axis])	Index of Agreement, IOA (avoid single block error in np.ma)
MB(obs, mod[, axis])	Mean Bias
ME(obs, mod[, axis])	Mean Gross Error (model and obs unit)
MNB(obs, mod[, axis])	Mean Normalized Bias (%)
MNE(obs, mod[, axis])	Mean Normalized Gross Error (%)
MNPB(obs, mod, paxis[, axis])	Mean Normalized Peak Bias (%)
MNPE(obs, mod, paxis[, axis])	Mean Normalized Peak Error (%)
MO(obs, mod[, axis])	Mean Observations (obs unit)
MP(obs, mod[, axis])	Mean Predictions (model unit)
MdnB(obs, mod[, axis])	Median Bias
MdnE(obs, mod[, axis])	Median Gross Error (model and obs unit)
MdnNB(obs, mod[, axis])	Median Normalized Bias (%)
MdnNE(obs, mod[, axis])	Median Normalized Gross Error (%)
MdnNPB(obs, mod, paxis[, axis])	Median Normalized Peak Bias (%)
MdnNPE(obs, mod, paxis[, axis])	Median Normalized Peak Bias (%)
MdnO(obs, mod[, axis])	Median Observations (obs unit)
MdnP(obs, mod[, axis])	Median Predictions (model unit)
NMB(obs, mod[, axis])	Normalized Mean Bias (%)
NMB_ABS(obs, mod[, axis])	Normalized Mean Bias - Absolute of the denominator (%)
NME(obs, mod[, axis])	Normalized Mean Error (%)
NME_m(obs, mod[, axis])	Normalized Mean Error (%) (avoid single block error in np.ma)
NME_m_ABS(obs, mod[, axis])	Normalized Mean Error (%) - Absolute of the denominator (avoid single block error in np.ma)
NMPB(obs, mod, paxis[, axis])	Normalized Mean Peak Bias (%)
NMPE(obs, mod, paxis[, axis])	Normalized Mean Peak Error (%)
NMdnB(obs, mod[, axis])	Normalized Median Bias (%)
NMdnE(obs, mod[, axis])	Normalized Median Error (%)
NMdnGE(obs, mod[, axis])	Normalized Median Gross Error (%)
NMdnPB(obs, mod, paxis[, axis])	Normalized Median Peak Bias (%)
NMdnPE(obs, mod, paxis[, axis])	Normalized Median Peak Bias (%)
NO(obs, mod[, axis])	N Observations (#)
NOP(obs, mod[, axis])	N Observations/Prediction Pairs (#)
NP(obs, mod[, axis])	N Predictions (#)
PSUTMNPB(obs, mod[, axis])	Paired Space/Unpaired Time Mean Normalized Peak Bias (%)
PSUTMNPE(obs, mod[, axis])	Paired Space/Unpaired Time Mean Normalized Peak Error (%)
PSUTMdnNPB(obs, mod[, axis])	Paired Space/Unpaired Time Median Normalized Peak Bias (%)
PSUTMdnNPE(obs, mod[, axis])	Paired Space/Unpaired Time Median Normalized Peak Error (%)
PSUTNMPB(obs, mod[, axis])	Paired Space/Unpaired Time Normalized Mean Peak Bias (%)

continues on next page

Table 1 – continued from previous page

PSUTNMPE(obs, mod[, axis])	Paired Space/Unpaired Time Normalized Mean Peak Error (%)
PSUTNMdnPB(obs, mod[, axis])	Paired Space/Unpaired Time Normalized Median Peak Bias (%)
PSUTNMdnPE(obs, mod[, axis])	Paired Space/Unpaired Time Normalized Median Peak Error (%)
R2(obs, mod[, axis])	Coefficient of Determination (unit squared)
RM(obs, mod[, axis])	Mean Ratio Observations/Predictions (none)
RMSE(obs, mod[, axis])	Root Mean Square Error (model unit)
RMSEs(obs, mod[, axis])	Root Mean Squared Error (obs, mod_hat)
RMSEu(obs, mod[, axis])	Root Mean Squared Error (mod_hat, mod)
RMdn(obs, mod[, axis])	Median Ratio Observations/Predictions (none)
STDO(obs, mod[, axis])	Standard deviation of Observations
STDP(obs, mod[, axis])	Standard deviation of Predictions
USUTPB(obs, mod[, axis])	Unpaired Space/Unpaired Time Peak Bias (%)
USUTPE(obs, mod[, axis])	Unpaired Space/Unpaired Time Peak Error (%)
WDAC(obs, mod[, axis])	Wind Direction Anomaly Correlation
WDIOA(obs, mod[, axis])	Wind Direction Index of Agreement, IOA
WDIOA_m(obs, mod[, axis])	Wind Direction Index of Agreement, IOA (avoid single block error in np.ma)
WDMB(obs, mod[, axis])	Wind Direction Mean Bias
WDMB_m(obs, mod[, axis])	Wind Direction Mean Bias (avoid single block error in np.ma)
WDME(obs, mod[, axis])	Wind Direction Mean Gross Error (model and obs unit)
WDME_m(obs, mod[, axis])	Wind Direction Mean Gross Error (model and obs unit) (avoid single block error in np.ma)
WDMdnB(obs, mod[, axis])	Wind Direction Median Bias
WDMdnE(obs, mod[, axis])	Wind Direction Median Gross Error (model and obs unit)
WDNMB_m(obs, mod[, axis])	Wind Direction Normalized Mean Bias (%) (avoid single block error in np.ma)
WDRMSE(obs, mod[, axis])	Wind Direction Root Mean Square Error (model unit)
WDRMSE_m(obs, mod[, axis])	Wind Direction Root Mean Square Error (model unit) (avoid single block error in np.ma)
circlebias(b)	Short summary.
circlebias_m(b)	avoid single block error in np.ma
d1(obs, mod[, axis])	Modified Index of Agreement, d1
matchedcompressed(a1, a2)	Short summary.
matchmasks(a1, a2)	Short summary.
scores(obs, mod, minval[, maxval])	Calculate scores.
stats(df, minval, maxval)	Short summary.

monet.util.stats.**AC**(*obs*, *mod*, *axis=None*)

Anomaly Correlation

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**CSI**(*obs*, *mod*, *minval*, *maxval*)

Critical Success Index (1 is perfect - Range 0 -> 1)

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **minval** (*type*) – Description of parameter *minval*.
- **maxval** (*type*) – Description of parameter *maxval*.

Returns Description of returned object.

Return type type

monet.util.stats.**E1**(*obs*, *mod*, *axis=None*)

Modified Coefficient of Efficiency, E1

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**ETS**(*obs*, *mod*, *minval*, *maxval*)

Equitable Threat Score (1 is perfect - Range -1/3 -> 1)

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **minval** (*type*) – Description of parameter *minval*.
- **maxval** (*type*) – Description of parameter *maxval*.

Returns Description of returned object.

Return type type

monet.util.stats.**FB**(*obs*, *mod*, *axis=None*)

Fractional Bias (%)

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**FE**(*obs*, *mod*, *axis=None*)

Fractional Error (%)

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**HSS**(*obs*, *mod*, *minval*, *maxval*)

Heidke Skill Score (1 is perfect - below zero means no confidence)

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **minval** (*type*) – Description of parameter *minval*.
- **maxval** (*type*) – Description of parameter *maxval*.

Returns Description of returned object.

Return type type

monet.util.stats.**IOA**(*obs*, *mod*, *axis=None*)

Index of Agreement, IOA

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**IOA_m**(*obs*, *mod*, *axis=None*)

Index of Agreement, IOA (avoid single block error in np.ma)

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**MB**(*obs*, *mod*, *axis=None*)

Mean Bias

Parameters

- **obs** (*type*) – Description of parameter *obs*.

- **mod** (*type*) – Description of parameter *mod*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**ME**(*obs*, *mod*, *axis=None*)

Mean Gross Error (model and obs unit)

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**MNB**(*obs*, *mod*, *axis=None*)

Mean Normalized Bias (%)

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**MNE**(*obs*, *mod*, *axis=None*)

Mean Normalized Gross Error (%)

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**MNPB**(*obs*, *mod*, *paxis*, *axis=None*)

Mean Normalized Peak Bias (%)

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **paxis** (*type*) – Description of parameter *paxis*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**MNPE**(*obs*, *mod*, *paxis*, *axis=None*)

Mean Normalized Peak Error (%)

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **paxis** (*type*) – Description of parameter *paxis*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**MO**(*obs*, *mod*, *axis=None*)

Mean Observations (obs unit)

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**MP**(*obs*, *mod*, *axis=None*)

Mean Predictions (model unit)

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**MdnB**(*obs*, *mod*, *axis=None*)

Median Bias

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**MdnE**(*obs*, *mod*, *axis=None*)

Median Gross Error (model and obs unit)

Parameters

- **obs** (*type*) – Description of parameter *obs*.

- **mod** (*type*) – Description of parameter *mod*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**MdnNB**(*obs*, *mod*, *axis=None*)

Median Normalized Bias (%)

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**MdnNE**(*obs*, *mod*, *axis=None*)

Median Normalized Gross Error (%)

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**MdnNPB**(*obs*, *mod*, *paxis*, *axis=None*)

Median Normalized Peak Bias (%)

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **paxis** (*type*) – Description of parameter *paxis*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**MdnNPE**(*obs*, *mod*, *paxis*, *axis=None*)

Median Normalized Peak Bias (%)

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **paxis** (*type*) – Description of parameter *paxis*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**MdnO**(*obs*, *mod*, *axis=None*)

Median Observations (obs unit)

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**MdnP**(*obs*, *mod*, *axis=None*)

Median Predictions (model unit)

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**NMB**(*obs*, *mod*, *axis=None*)

Normalized Mean Bias (%)

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**NMB_ABS**(*obs*, *mod*, *axis=None*)

Normalized Mean Bias - Absolute of the denominator (%)

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**NME**(*obs*, *mod*, *axis=None*)

Normalized Mean Error (%)

Parameters

- **obs** (*type*) – Description of parameter *obs*.

- **mod** (*type*) – Description of parameter *mod*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**NME_m**(*obs*, *mod*, *axis=None*)

Normalized Mean Error (%) (avoid single block error in np.ma)

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**NME_m_ABS**(*obs*, *mod*, *axis=None*)

Normalized Mean Error (%) - Absolute of the denominator (avoid single block error in np.ma)

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**NMPB**(*obs*, *mod*, *paxis*, *axis=None*)

Normalized Mean Peak Bias (%)

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **paxis** (*type*) – Description of parameter *paxis*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**NMPE**(*obs*, *mod*, *paxis*, *axis=None*)

Normalized Mean Peak Error (%)

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **paxis** (*type*) – Description of parameter *paxis*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**NMdnB**(*obs*, *mod*, *axis=None*)

Normalized Median Bias (%)

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**NMdnE**(*obs*, *mod*, *axis=None*)

Normalized Median Error (%)

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**NMdnGE**(*obs*, *mod*, *axis=None*)

Normalized Median Gross Error (%)

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**NMdnPB**(*obs*, *mod*, *paxis*, *axis=None*)

Normalized Median Peak Bias (%)

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **paxis** (*type*) – Description of parameter *paxis*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**NMdnPE**(*obs*, *mod*, *paxis*, *axis=None*)

Normalized Median Peak Bias (%)

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **paxis** (*type*) – Description of parameter *paxis*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**NO**(*obs*, *mod*, *axis=None*)

N Observations (#)

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**NOP**(*obs*, *mod*, *axis=None*)

N Observations/Prediction Pairs (#)

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**NP**(*obs*, *mod*, *axis=None*)

N Predictions (#)

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**PSUTMNPB**(*obs*, *mod*, *axis=None*)

Paired Space/Unpaired Time Mean Normalized Peak Bias (%)

Parameters

- **obs** (*type*) – Description of parameter *obs*.

- **mod** (*type*) – Description of parameter *mod*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**PSUTMNPE**(*obs*, *mod*, *axis=None*)

Paired Space/Unpaired Time Mean Normalized Peak Error (%)

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**PSUTMdnNPB**(*obs*, *mod*, *axis=None*)

Paired Space/Unpaired Time Median Normalized Peak Bias (%)

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**PSUTMdnNPE**(*obs*, *mod*, *axis=None*)

Paired Space/Unpaired Time Median Normalized Peak Error (%)

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**PSUTNMPB**(*obs*, *mod*, *axis=None*)

Paired Space/Unpaired Time Normalized Mean Peak Bias (%)

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**PSUTNMPE**(*obs*, *mod*, *axis=None*)

Paired Space/Unpaired Time Normalized Mean Peak Error (%)

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**PSUTNMdnPB**(*obs*, *mod*, *axis=None*)

Paired Space/Unpaired Time Normalized Median Peak Bias (%)

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**PSUTNMdnPE**(*obs*, *mod*, *axis=None*)

Paired Space/Unpaired Time Normalized Median Peak Error (%)

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**R2**(*obs*, *mod*, *axis=None*)

Coefficient of Determination (unit squared)

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**RM**(*obs*, *mod*, *axis=None*)

Mean Ratio Observations/Predictions (none)

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.

- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**RMSE**(*obs*, *mod*, *axis=None*)

Root Mean Square Error (model unit)

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**RMSEs**(*obs*, *mod*, *axis=None*)

Root Mean Squared Error (obs, mod_hat)

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**RMSEu**(*obs*, *mod*, *axis=None*)

Root Mean Squared Error (mod_hat, mod)

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**RMdn**(*obs*, *mod*, *axis=None*)

Median Ratio Observations/Predictions (none)

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**STDO**(*obs*, *mod*, *axis=None*)

Standard deviation of Observations

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**STDP**(*obs*, *mod*, *axis=None*)

Standard deviation of Predictions

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**USUTPB**(*obs*, *mod*, *axis=None*)

Unpaired Space/Unpaired Time Peak Bias (%)

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**USUTPE**(*obs*, *mod*, *axis=None*)

Unpaired Space/Unpaired Time Peak Error (%)

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**WDAC**(*obs*, *mod*, *axis=None*)

Wind Direction Anomaly Correlation

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.

- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**WDIOA**(*obs*, *mod*, *axis=None*)

Wind Direction Index of Agreement, IOA

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**WDIOA_m**(*obs*, *mod*, *axis=None*)

Wind Direction Index of Agreement, IOA (avoid single block error in np.ma)

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**WDMB**(*obs*, *mod*, *axis=None*)

Wind Direction Mean Bias

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**WDMB_m**(*obs*, *mod*, *axis=None*)

Wind Direction Mean Bias (avoid single block error in np.ma)

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**WDME**(*obs*, *mod*, *axis=None*)

Wind Direction Mean Gross Error (model and obs unit)

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**WDME_m**(*obs*, *mod*, *axis=None*)

Wind Direction Mean Gross Error (model and obs unit) (avoid single block error in np.ma)

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**WDMdnB**(*obs*, *mod*, *axis=None*)

Wind Direction Median Bias

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**WDMdnE**(*obs*, *mod*, *axis=None*)

Wind Direction Median Gross Error (model and obs unit)

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**WDNMB_m**(*obs*, *mod*, *axis=None*)

Wind Direction Normalized Mean Bias (%) (avoid single block error in np.ma)

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.

- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**WDRMSE**(*obs*, *mod*, *axis=None*)

Wind Direction Root Mean Square Error (model unit)

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**WDRMSE_m**(*obs*, *mod*, *axis=None*)

Wind Direction Root Mean Square Error (model unit) (avoid single block error in np.ma)

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**circlebias**(*b*)

Short summary.

Parameters **b** (*type*) – Description of parameter *b*.

Returns Description of returned object.

Return type type

monet.util.stats.**circlebias_m**(*b*)

avoid single block error in np.ma

Parameters **b** (*type*) – Description of parameter *b*.

Returns Description of returned object.

Return type type

monet.util.stats.**d1**(*obs*, *mod*, *axis=None*)

Modified Index of Agreement, d1

Parameters

- **obs** (*type*) – Description of parameter *obs*.
- **mod** (*type*) – Description of parameter *mod*.
- **axis** (*type*) – Description of parameter *axis*.

Returns Description of returned object.

Return type type

monet.util.stats.**matchedcompressed**(*a1*, *a2*)

Short summary.

Parameters

- **a1** (*type*) – Description of parameter *a1*.
- **a2** (*type*) – Description of parameter *a2*.

Returns Description of returned object.

Return type type

monet.util.stats.**matchmasks**(*a1*, *a2*)

Short summary.

Parameters

- **a1** (*type*) – Description of parameter *a1*.
- **a2** (*type*) – Description of parameter *a2*.

Returns Description of returned object.

Return type type

monet.util.stats.**scores**(*obs*, *mod*, *minval*, *maxval=100000.0*)

Calculate scores.

Parameters

- **obs** (*array-like*) – Observation values ("truth").
- **mod** (*array-like*) – Model values ("prediction"). Should be the same size as *obs*.
- **minval, minval** (*float*) – Interval to test (exclusive on both sides).

Returns **a, b, c, d** – Counts of hits, misses, false alarms, and correct negatives.

Return type float

monet.util.stats.**stats**(*df*, *minval*, *maxval*)

Short summary.

Parameters

- **df** (*type*) – Description of parameter *df*.
- **minval** (*type*) – Description of parameter *minval*.
- **maxval** (*type*) – Description of parameter *maxval*.

Returns Description of returned object.

Return type type

monet.util.tools

Functions

calc_24hr_ave(df[, col])	
calc_3hr_ave(df[, col])	
calc_8hr_rolling_max(df[, col, window])	
calc_annual_ave(df[, col])	
findclosest(list, value)	
get_epa_region_bounds([index, acronym])	
get_epa_region_df(df)	
get_giorgi_region_bounds([index, acronym])	
get_giorgi_region_df(df)	
get_relhum(temp, press, vap)	
kolmogorov_zurbenko_filter(df, col, window, ...)	KZ filter implementation series is a pandas series window is the filter window m in the units of the data (m = 2q+1) iterations is the number of times the moving average is evaluated
linregress(x, y)	
long_to_wide(df)	
search_listinlist(array1, array2)	
wsdir2uv(ws, wdir)	

monet.util.tools.**kolmogorov_zurbenko_filter**(*df*, *col*, *window*, *iterations*)

KZ filter implementation series is a pandas series window is the filter window m in the units of the data (m = 2q+1) iterations is the number of times the moving average is evaluated

4.7.3 DataArray Accessor

DataArray.monet.wrap_longitudes([lon_name])	Ensures longitudes are from -180 -> 180.
DataArray.monet.tidy([lon_name])	Tidy's DataArray–wraps longitudes and sorts lats and lons.
DataArray.monet.is_land([return_xarray])	Check the mask of land and ocean, returning corresponding boolean mask.
DataArray.monet.is_ocean([return_xarray])	Check the mask of land and ocean, returning corresponding boolean mask.
DataArray.monet.cftime_to_datetime64([name])	Convert to datetime64.
DataArray.monet.structure_for_monet([...])	This will attempt to restructure a given DataArray for use within MONET.
DataArray.monet.stratify(levels, vertical[, ...])	Resample in the vertical with stratify.
DataArray.monet.window([lat_min, lon_min, ...])	Function to window, ie select a specific region, given the lower left latitude and longitude and the upper right latitude and longitude
DataArray.monet.interp_constant_lat([lat, ...])	Interpolates the data array to constant longitude.
DataArray.monet.interp_constant_lon([lon])	Interpolates the data array to constant longitude.
DataArray.monet.nearest_ij([lat, lon])	Uses pyresample to interpolate to find the i, j index of grid with respect to the given lat lon.
DataArray.monet.nearest_latlon([lat, lon, ...])	Uses xesmf to interpolate to a given latitude and longitude.
DataArray.monet.quick_imshow([map_kws, ...])	This function takes an xarray DataArray and quickly creates a figure using cartopy and matplotlib `imshow`.
DataArray.monet.quick_map([map_kws, ...])	This function takes an xarray DataArray and quickly creates a figure using cartopy and matplotlib `pcolormesh`.
DataArray.monet.quick_contourf([map_kws, ...])	This function takes an xarray DataArray and quickly creates a figure using cartopy and matplotlib `contourf`.
DataArray.monet.remap_nearest(data, **kwargs)	Remap *data* from another grid to the current self grid using pyresample nearest-neighbor interpolation.
DataArray.monet.remap_xesmf(data, **kwargs)	Remap *data* from another grid to the current grid of self using xESMF.
DataArray.monet.combine_point(data[, ...])	Combine self data with point data in dataframe *data*.

4.7.3.1 xarray.DataArray.monet.wrap_longitudes

DataArray.monet.**wrap_longitudes**(*lon_name='longitude'*)

Ensures longitudes are from -180 -> 180.

Modifies longitude variable in place.

Parameters **lon_name** (*str*)

Returns self

Return type xarray.DataArray

4.7.3.2 xarray.DataArray.monet.tidy

DataArray.monet.**tidy**(*lon_name='longitude'*)

Tidy's DataArray–wraps longitudes and sorts lats and lons.

lon wrapping applied in-place but not sorting.

Return type xarray.DataArray

4.7.3.3 xarray.DataArray.monet.is_land

DataArray.monet.**is_land**(*return_xarray=False*)

Check the mask of land and ocean, returning corresponding boolean mask.

Parameters **return_xarray** (*bool*) – If True, return the data array with the ocean values set to NaN. If False, return a numpy boolean array of the land (True).

Return type xarray.DataArray or numpy.array

4.7.3.4 xarray.DataArray.monet.is_ocean

DataArray.monet.**is_ocean**(*return_xarray=False*)

Check the mask of land and ocean, returning corresponding boolean mask.

Parameters **return_xarray** (*bool*) – If True, return the data array with the land values set to NaN. If False, return a numpy boolean array of the ocean (True).

Return type xarray.DataArray or numpy.array

4.7.3.5 xarray.DataArray.monet.cftime_to_datetime64

DataArray.monet.**cftime_to_datetime64**(*name=None*)

Convert to datetime64.

Parameters **name** (*str, optional*) – Variable to apply the transformation to (in place). `'time'` assumed by default.

Returns self

Return type xarray.DataArray

4.7.3.6 xarray.DataArray.monet.structure_for_monet

DataArray.monet.**structure_for_monet**(*lat_name='lat'*, *lon_name='lon'*, *return_obj=True*)

This will attempt to restructure a given DataArray for use within MONET.

Parameters

- **lat_name** (*str*)
- **lon_name** (*str*)
- **return_obj** (*bool*) – If true, return modified object, else modify in place.

4.7.3.7 xarray.DataArray.monet.stratify

DataArray.monet.**stratify**(*levels*, *vertical*, *axis=1*)

Resample in the vertical with stratify.

Parameters

- **levels** – Values to interpolate to.
- **vertical** – Vertical dimension coordinate variable.
- **axis** (*int*)

Return type xarray.DataArray

4.7.3.8 xarray.DataArray.monet.window

DataArray.monet.**window**(*lat_min=None*, *lon_min=None*, *lat_max=None*, *lon_max=None*, *rectilinear=False*)

Function to window, ie select a specific region, given the lower left latitude and longitude and the upper right latitude and longitude

Parameters

- **lat_min** (*float*) – lower left latitude .
- **lon_min** (*float*) – lower left longitude.
- **lat_max** (*float*) – upper right latitude.
- **lon_max** (*float*) – upper right longitude.
- **rectilinear** (*bool*) – flag if this is a rectilinear lat lon grid

Returns returns the windowed object.

Return type xr.DataArray

4.7.3.9 xarray.DataArray.monet.interp_constant_lat

DataArray.monet.**interp_constant_lat**(*lat=None*, *lat_name='latitude'*, *lon_name='longitude'*, ***kwargs*)

Interpolates the data array to constant longitude.

Parameters **lon** (*float*) – Latitude on which to interpolate to

Returns DataArray of at constant longitude

Return type DataArray

4.7.3.10 xarray.DataArray.monet.interp_constant_lon

DataArray.monet.**interp_constant_lon**(*lon=None*, ***kwargs*)

Interpolates the data array to constant longitude.

Parameters **lon** (*float*) – Latitude on which to interpolate to

Returns DataArray of at constant longitude

Return type DataArray

4.7.3.11 xarray.DataArray.monet.nearest_ij

DataArray.monet.**nearest_ij**(*lat=None*, *lon=None*, ***kwargs*)

Uses pyresample to interpolate to find the i, j index of grid with respect to the given lat lon.

Parameters

- **lat** (*float*) – latitude in question
- **lon** (*float*) – longitude in question
- ****kwargs** (*dict*) – pyresample kwargs for nearest neighbor interpolation

Returns Returns the i (x index) and j (y index) of the given latitude longitude value

Return type i,j

4.7.3.12 xarray.DataArray.monet.nearest_latlon

DataArray.monet.**nearest_latlon**(*lat=None*, *lon=None*, *cleanup=True*, *esmf=False*, ***kwargs*)

Uses xesmf to interpolate to a given latitude and longitude. Note that the conservative method is not available.

Parameters

- **lat** (*float*)
- **lon** (*float*)
- **kwargs** (*dict*) – Passed on to resampling routines.

Return type xarray.Dataset

4.7.3.13 xarray.DataArray.monet.quick_imshow

DataArray.monet.**quick_imshow**(*map_kws=None*, *roll_dateline=False*, ***kwargs*)

This function takes an xarray DataArray and quickly creates a figure using cartopy and matplotlib `imshow`.

Note that this should only be used for regular grids.

Parameters

- **map_kws** (*dictionary*) – kwargs for monet.plots.mapgen.draw_map
- **roll_dateline** (*bool*) – roll_dateline is meant to help with global datasets that the longitudes range from 0 to 360 instead of -180 to 180. Otherwise a white line appears at 0 degrees.
- ****kwargs** – kwargs for the xarray.DataArray.plot.imshow function

Return type matplotlib.axes.Axes

4.7.3.14 xarray.DataArray.monet.quick_map

DataArray.monet.**quick_map**(*map_kws=None*, *roll_dateline=False*, ***kwargs*)

This function takes an xarray DataArray and quickly creates a figure using cartopy and matplotlib `pcolormesh`.

Parameters

- **map_kws** (*dictionary*) – kwargs for monet.plots.mapgen.draw_map
- **roll_dateline** (*bool*) – roll_dateline is meant to help with global datasets that the longitudes range from 0 to 360 instead of -180 to 180. Otherwise a white line appears at 0 degrees.
- ****kwargs** – kwargs for the xarray.DataArray.plot.pcolormesh function

Return type matplotlib.axes.Axes

4.7.3.15 xarray.DataArray.monet.quick_contourf

DataArray.monet.**quick_contourf**(*map_kws=None*, *roll_dateline=False*, ***kwargs*)

This function takes an xarray DataArray and quickly creates a figure using cartopy and matplotlib `contourf`.

Parameters

- **map_kws** (*dictionary*) – kwargs for monet.plots.mapgen.draw_map
- **roll_dateline** (*bool*) – roll_dateline is meant to help with global datasets that the longitudes range from 0 to 360 instead of -180 to 180. Otherwise a white line appears at 0 degrees.
- ****kwargs** – kwargs for the xarray.DataArray.plot.contourf function

Return type matplotlib.axes.Axes

4.7.3.16 xarray.DataArray.monet.remap_nearest

DataArray.monet.**remap_nearest**(*data*, ***kwargs*)

Remap *data* from another grid to the current self grid using pyresample nearest-neighbor interpolation.

This assumes that the dimensions are ordered in y,x,z per pyresample docs.

Parameters

- **data** (*xarray.DataArray or xarray.Dataset*) – Data to be interpolated to nearest points in self.
- **radius_of_influence** (*float*) – Radius of influence (meters), used by `pyresample.kd_tree`.

Returns Data on current (self) grid.

Return type xarray.DataArray or xarray.Dataset

4.7.3.17 xarray.DataArray.monet.remap_xesmf

DataArray.monet.**remap_xesmf**(*data*, ***kwargs*)

Remap *data* from another grid to the current grid of self using xESMF.

Parameters

- **data** (*numpy.ndarray or xarray.DataArray*)
- **kwargs** (*dict*) – Passed on to `resample_xesmf()` and then to `xesmf.Regridder`. `method` defaults to `'bilinear'`.

Returns Resampled object on current grid.

Return type xarray.DataArray

4.7.3.18 xarray.DataArray.monet.combine_point

DataArray.monet.**combine_point**(*data*, *suffix=None*, *pyresample=True*, ***kwargs*)

Combine self data with point data in dataframe *data*.

Parameters

- **data** (*pandas.DataFrame*)
- **suffix** (*str, optional*) – Used to rename new data array(s) if using xemsf.
- **pyresample** (*bool*) – Use pyresample (*combine_da_to_df()*).
- **kwargs** (*dict*) – Passed on to *combine_da_to_df()* or *combine_da_to_df_xesmf()*

Return type pandas.DataFrame

4.7.4 Dataset Accessor

Dataset.monet.wrap_longitudes([lon_name])	Ensures longitudes are from -180 -> 180.
Dataset.monet.tidy([lon_name])	Tidy's Dataset–wraps longitudes and sorts lats and lons.
Dataset.monet.is_land([return_xarray])	checks the mask of land and ocean if the global_land_mask libra.
Dataset.monet.is_ocean([return_xarray])	checks the mask of land and ocean.
Dataset.monet.cftime_to_datetime64([name])	Convert to datetime64.
Dataset.monet.stratify(levels, vertical[, axis])	Resample in the vertical with stratify.
Dataset.monet.window(lat_min, lon_min, ...)	Function to window, ie select a specific region, given the lower left latitude and longitude and the upper right latitude and longitude
Dataset.monet.interp_constant_lat([lat, ...])	Interpolates the data array to constant longitude.
Dataset.monet.interp_constant_lon([lon])	Interpolates the data array to constant longitude.
Dataset.monet.nearest_ij([lat, lon])	Uses pyresample to interpolate to find the i, j index of grid with respect to the given lat lon.
Dataset.monet.nearest_latlon([lat, lon, ...])	Uses xesmf to interpolate to a given latitude and longitude.
Dataset.monet.remap_nearest(data[, ...])	Remap *data* from another grid to the current self grid using pyresample nearest-neighbor interpolation.
Dataset.monet.remap_nearest_unstructured(data)	Find the closest model data (*data*) to the observation (self) for unstructured grid model data.
Dataset.monet.remap_xesmf(data, **kwargs)	Remap *data* from another grid to the current grid of self using xESMF.
Dataset.monet.combine_point(data[, suffix, ...])	Combine self data with point data in dataframe *data*.

4.7.4.1 xarray.Dataset.monet.wrap_longitudes

Dataset.monet.**wrap_longitudes**(*lon_name='longitude'*)

Ensures longitudes are from -180 -> 180.

Modifies longitude variable in place.

4.7.4.2 xarray.Dataset.monet.tidy

Dataset.monet.**tidy**(*lon_name='longitude'*)

Tidy's Dataset–wraps longitudes and sorts lats and lons.

lon wrapping applied in-place but not sorting.

Return type xarray.Dataset

4.7.4.3 xarray.Dataset.monet.is_land

`Dataset.monet.is_land(return_xarray=False)`

checks the mask of land and ocean if the global_land_mask libra.

Parameters **return_xarray** (*bool*) – If True, return the data array with the ocean values set to NaN. If False, return a numpy boolean array of the land (True).

Return type xarray.DataArray or numpy.array

4.7.4.4 xarray.Dataset.monet.is_ocean

`Dataset.monet.is_ocean(return_xarray=False)`

checks the mask of land and ocean.

Parameters **return_xarray** (*bool*) – If True, return the data array with the land values set to NaN. If False, return a numpy boolean array of the ocean (True).

Return type xarray.DataArray or numpy.array

4.7.4.5 xarray.Dataset.monet.cftime_to_datetime64

`Dataset.monet.cftime_to_datetime64(name=None)`

Convert to datetime64.

Parameters **name** (*str, optional*) – Variable to apply the transformation to (in place). `'time'` assumed by default.

Returns self

Return type xarray.Dataset

4.7.4.6 xarray.Dataset.monet.stratify

`Dataset.monet.stratify(levels, vertical, axis=1)`

Resample in the vertical with stratify.

Parameters

- **levels** – Values to interpolate to.
- **vertical** – Vertical dimension coordinate variable.
- **axis** (*int*)

Return type xarray.Dataset

4.7.4.7 xarray.Dataset.monet.window

Dataset.monet.**window**(*lat_min*, *lon_min*, *lat_max*, *lon_max*)

Function to window, ie select a specific region, given the lower left latitude and longitude and the upper right latitude and longitude

Parameters

- **lat_min** (*float*) – lower left latitude .
- **lon_min** (*float*) – lower left longitude.
- **lat_max** (*float*) – upper right latitude.
- **lon_max** (*float*) – upper right longitude.

Returns returns the windowed object.

Return type xr.DataSet

4.7.4.8 xarray.Dataset.monet.interp_constant_lat

Dataset.monet.**interp_constant_lat**(*lat=None*, *lat_name='latitude'*, *lon_name='longitude'*, ***kwargs*)

Interpolates the data array to constant longitude.

Parameters **lon** (*float*) – Latitude on which to interpolate to

Returns DataArray of at constant longitude

Return type DataArray

4.7.4.9 xarray.Dataset.monet.interp_constant_lon

Dataset.monet.**interp_constant_lon**(*lon=None*, ***kwargs*)

Interpolates the data array to constant longitude.

Parameters **lon** (*float*) – Latitude on which to interpolate to

Returns DataArray of at constant longitude

Return type xr.Dataset or xr.DataArray

4.7.4.10 xarray.Dataset.monet.nearest_ij

Dataset.monet.**nearest_ij**(*lat=None*, *lon=None*, ***kwargs*)

Uses pyresample to interpolate to find the i, j index of grid with respect to the given lat lon.

Parameters

- **lat** (*float*) – latitude in question
- **lon** (*float*) – longitude in question
- ****kwargs** (*dict*) – pyresample kwargs for nearest neighbor interpolation

Returns Returns the i (x index) and j (y index) of the given latitude longitude value

Return type i,j

4.7.4.11 xarray.Dataset.monet.nearest_latlon

Dataset.monet.**nearest_latlon**(*lat=None*, *lon=None*, *cleanup=True*, *esmf=False*, ***kwargs*)

Uses xesmf to interpolate to a given latitude and longitude. Note that the conservative method is not available.

Parameters

- **lat** (*float*)
- **lon** (*float*)
- **kwargs** (*dict*) – Passed on to resampling routine.

Return type xarray.Dataset

4.7.4.12 xarray.Dataset.monet.remap_nearest

Dataset.monet.**remap_nearest**(*data*, *radius_of_influence=1000000.0*)

Remap *data* from another grid to the current self grid using pyresample nearest-neighbor interpolation.

Parameters

- **data** (*xarray.DataArray or xarray.Dataset*) – Data to be interpolated to nearest points in self. Must include lat/lon coordinates.
- **radius_of_influence** (*float*) – Radius of influence (meters), used by `pyresample.kd_tree`.

Returns Data on current (self) grid.

Return type xarray.Dataset or xarray.DataArray

4.7.4.13 xarray.Dataset.monet.remap_nearest_unstructured

Dataset.monet.**remap_nearest_unstructured**(*data*)

Find the closest model data (*data*) to the observation (self) for unstructured grid model data.

Based on model grid cell center locations.

Parameters **data** (*xarray.Dataset*) – Data to be interpolated, including lat/lon coordinates.

Returns Data on self grid.

Return type xarray.Dataset

4.7.4.14 xarray.Dataset.monet.remap_xesmf

Dataset.monet.**remap_xesmf**(*data*, ***kwargs*)

Remap *data* from another grid to the current grid of self using xESMF.

Parameters

- **data** (*xarray.DataArray or xarray.Dataset*) – Data to be remapped.
- **kwargs** (*dict*) – Passed on to `resample_xesmf()` and then to `xesmf.Regridder`.

4.7.4.15 xarray.Dataset.monet.combine_point

Dataset.monet.**combine_point**(*data*, *suffix=None*, *pyresample=True*, ***kwargs*)

Combine self data with point data in dataframe *data*.

Parameters

- **data** (*pandas.DataFrame*)
- **suffix** (*str, optional*) – Used to rename new data array(s) if using xemsf.
- **pyresample** (*bool*) – Use pyresample (*combine_da_to_df()*).
- **kwargs** (*dict*) – Passed on to *combine_da_to_df()* or *combine_da_to_df_xesmf()*

Return type pandas.DataFrame

4.7.5 DataFrame Accessor

DataFrame.monet.to_ascii2nc_df([grib_code, ...])	
DataFrame.monet.to_ascii2nc_list(**kwargs)	

DataFrame.monet.rename_for_monet([df])	Rename latitude and longitude columns in the DataFrame.
DataFrame.monet.get_sparse_SwathDefinition()	Creates a `pyreample.geometry.SwathDefinition` for a single point.
DataFrame.monet.remap_nearest(df[, ...])	Remap data in *df* to nearest points in self.
DataFrame.monet.cftime_to_datetime64([col])	Convert to datetime64.

4.7.5.1 pandas.DataFrame.monet.to_ascii2nc_df

DataFrame.monet.**to_ascii2nc_df**(*grib_code=126*, *height_msl=0.0*, *column='aod_550nm'*, *message_type='ADPUPA'*, *pressure=1000.0*, *qc=None*, *height_agl=None*)

4.7.5.2 pandas.DataFrame.monet.to_ascii2nc_list

DataFrame.monet.**to_ascii2nc_list**(***kwargs*)

4.7.5.3 pandas.DataFrame.monet.rename_for_monet

DataFrame.monet.**rename_for_monet**(*df=None*)

Rename latitude and longitude columns in the DataFrame.

Parameters **df** (*pandas.DataFrame, optional*) – To use instead of self.

Return type pandas.DataFrame

4.7.5.4 pandas.DataFrame.monet.get_sparse_SwathDefinition

DataFrame.monet.**get_sparse_SwathDefinition**()

Creates a `pyreample.geometry.SwathDefinition` for a single point.

Return type pyreample.geometry.SwathDefinition

4.7.5.5 pandas.DataFrame.monet.remap_nearest

DataFrame.monet.**remap_nearest**(*df*, *radius_of_influence=100000.0*, *combine=False*)

Remap data in *df* to nearest points in self.

Parameters

- **df** (*pandas.DataFrame*) – Data to be interpolated to nearest points in self.
- **radius_of_influence** (*float*) – Radius of influence (meters), used by `pyresample.kd_tree`.
- **combine** (*bool*) – Merge with original self data.

Returns Interpolated dataframe.

Return type pandas.DataFrame

4.7.5.6 pandas.DataFrame.monet.cftime_to_datetime64

DataFrame.monet.**cftime_to_datetime64**(*col=None*)

Convert to datetime64.

Parameters **col** (*str, optional*) – Column to apply the transformation to (in place). `'time'` assumed by default.

Returns self

Return type pandas.DataFrame

DataFrame.monet.center	The geographic center point of this DataFrame.

4.7.5.7 pandas.DataFrame.monet.center

DataFrame.monet.**center**

The geographic center point of this DataFrame.

Note: Currently just the lon and lat mean values, not necessarily representative of the geographic center.

Returns (lon, lat)

Return type tuple

BIBLIOGRAPHY

[Brutsaert2005] Brutsaert, W. (2005). Hydrology: an introduction (Vol. 61, No. 8). Cambridge: Cambridge University Press.

[Norman2000] Norman, J. M., W. P. Kustas, J. H. Prueger, and G. R. Diak (2000), Surface flux estimation using radiometric temperature: A dual-temperature-difference method to minimize measurement errors, Water Resour. Res., 36(8), 2263-2274, https://doi.org/10.1029/2000WR900033.

PYTHON MODULE INDEX

m

INDEX

K

L

M

N

P

Q

R

S

T

U

W