Pyleoclim Documentation

Release 0.3.0

Deborah Khider, Julien Emile-Geay, Feng Zhu

Jul 12, 2017

CONTENTS

Contents:

CHAPTER

ONE

PYLEOCLIM

1.1 What is it?

Pyleoclim is a Python package primarily geared towards the analysis and visualization of paleoclimate data. Such data often come in the form of timeseries with missing values and age uncertainties, and the package includes several low-level methods to deal with these issues, as well as high-level methods that re-use those to perform scientific workflows.

The package assumes that the data are stored in the Linked Paleo Data (LiPD) format and makes extensive use of the LiPD utilities. The package is aware of age ensembles stored via LiPD and uses them for time-uncertain analyses very much like GeoChronR.

Current Capabilities:

- binning
- interpolation
- plotting maps, timeseries, and basic age model information
- paleo-aware correlation analysis (isopersistent, isospectral, and classical t-test)
- weighted wavelet Z transform (WWZ)

Future capabilities:

- paleo-aware singular spectrum analysis (AR(1) null eigenvalue identification, missing data)
- spectral analysis (Multi-Taper Method, Lomb-Scargle)
- cross-wavelet analysis
- index reconstruction
- climate reconstruction
- ensemble methods for most of the above

1.2 Version Information

0.3.0: Compatibility with LiPD 1.3 and Spectral module added
0.2.5: Fix error on loading (Looking for Spectral Module)
0.2.4: Fix load error from init
0.2.3: Freeze LiPD version to 1.2 to avoid conflicts with 1.3
0.2.2: Change progressbar to tqdm and add standardization function
0.2.1: Update package requirements

0.2.0: Restructure the package so that the main functions can be called without the use of a LiPD files and associated timeseries objects.
0.1.4: Rename functions using camel case convention and consistency with LiPD utilities version 0.1.8.5
0.1.3: Compatible with LiPD utilities version 0.1.8.5
Function openLiPD() renamed openLiPDs()
0.1.2: Compatible with LiPD utilities version 0.1.8.3
Uses Basemap instead of cartopy
0.1.1: Freezes the package prior to version 0.1.8.2 of LiPD utilities
0.1.0: First release

1.3 Installation

Python v3.4+ is required. Tested with Python v3.5 Pyleoclim is published through Pypi and easily installed via pip:

```
pip install pyleoclim
```

1.4 Quickstart guide

1. Open your command line application (Terminal or Command Prompt)
2. Install with command:

   pip install pyleoclim

3. Wait for installation to complete, then:

1. Import the package into your favorite Python environment (we recommend the use of Spyder, which comes standard with the Anaconda build)
2. Use Jupyter Notebook to go through the tutorial contained in the PyleolimQuickstart.ipynb

1.5 Requirements

- LiPD v0.2.2+
- pandas v0.19+
- numpy v1.12+
- matplotlib v2.0+
- Basemap v1.0.7+
- scipy v0.19.0+
- statsmodel v0.8.0+
- seaborn v0.7.0+
- scikit-learn v0.17.1+
- tqdm v4.14.0+
- pathos v0.2.0+

The installer will automatically check for the needed updates.

1.6 Further information

GitHub: https://github.com/LinkedEarth/Pyleoclim_util
LinkedEarth: http://linked.earth
Python and Anaconda: http://conda.pydata.org/docs/test-drive.html
Jupyter Notebook: http://jupyter.org/

1.7 Contact

Please report issues to linkedearth@gmail.com

1.8 License

The project is licensed under the GNU Public License .

1.9 Disclaimer

This material is based upon work supported by the U.S. National Science Foundation under Grant Number ICER-1541029. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the investigators and do not necessarily reflect the views of the National Science Foundation.

CHAPTER

TWO

MAIN FUNCTIONS

2.1 Using Pyleoclim with a LiPD file

2.1.1 Getting started

Pyleoclim relies heavily on the concept of timeseries objects introduced in LiPD and implemented in the LiPD utilities.

Briefly, timeseries objects are dictionaries containing the ChronData values and PaleoData values as well as the metadata associated with the record. If one record has three ProxyObservations (e.g., Mg/Ca, d18O, d13C) then it will have three timeseries objects, one for each of the observations.

The LiPD utilities function lipd.extractTs() returns a list of dictionaries for the selected LiPD files, which need to be passed to Pyleoclim along with the path to the directory containing the LiPD files.

This is done through the functions pyleoclim.readLiPD() and pyleoclim.extracTs(), which are lightweight versions of their counterparts in the LiPD utilities:

pyleoclim.**readLipd**(*usr_path=''*)
Read Lipd files into a dictionary

This function is based on the function of the same name in the LiPD utilities. Sets the dictionary as global variable so that it doesn't have to be provided as an argument for every function.

Parameters **usr_path** (*str*) – The path to a directory or a single file. (Optional argument)

Returns lipd_dict - a dictionary containing the LiPD library

pyleoclim.**extractTs**(*lipds=None*)
Extract timeseries dictionary

This function is based on the function of the same name in the LiPD utilities. Set the dictionary as a global variable so that it doesn't have to be provided as an argument for every function.

Parameters

- **lipds** (*dict*) – A dictionary of LiPD files obtained through the
- **function** (readLipd) –

Returns ts_list - A list of timeseries object

2.1.2 Mapping

pyleoclim.**mapAllArchive**(*lipd_dict=''*, *markersize=50*, *background='shadedrelief'*, *saveFig=False*, *dir=''*, *format='eps'*)
Map all the available records loaded into the workspace by archiveType.

Map of all the records into the workspace by archiveType. Uses the default color palette. Enter pyleoclim.plot_default for detail.

Parameters

- **lipd_dict** (*dict*) – A dictionary of LiPD files. (Optional)
- **markersize** (*int*) – The size of the markers. Default is 50
- **background** (*str*) – Plots one of the following images on the map: bluemarble, etopo, shadedrelief, or none (filled continents). Default is shadedrelief.
- **saveFig** (*bool*) – Default is to not save the figure
- **dir** (*str*) – The absolute path of the directory in which to save the figure. If not provided, creates a default folder called 'figures' in the LiPD working directory (lipd.path).
- **format** (*str*) – One of the file extensions supported by the active backend. Default is "eps". Most backend support png, pdf, ps, eps, and svg.

Returns The figure

pyleoclim.**mapLipd**(*timeseries=''*, *countries=True*, *counties=False*, *rivers=False*, *states=False*, *background='shadedrelief'*, *scale=0.5*, *markersize=50*, *marker='ro'*, *saveFig=False*, *dir=''*, *format='eps'*)

Create a Map for a single record

Orthographic projection map of a single record.

Parameters

- **timeseries** – a LiPD timeseries object. Will prompt for one if not given
- **countries** (*bool*) – Draws the country borders. Default is on (True).
- **counties** (*bool*) – Draws the USA counties. Default is off (False).
- **rivers** (*bool*) – Draws the rivers. Default is off (False).
- **states** (*bool*) – Draws the American and Australian states borders. Default is off (False)
- **background** (*str*) – Plots one of the following images on the map: bluemarble, etopo, shadedrelief, or none (filled continents). Default is shadedrelief
- **scale** (*float*) – useful to downgrade the original image resolution to speed up the process. Default is 0.5.
- **markersize** (*int*) – default is 50
- **marker** (*str*) – a string (or list) containing the color and shape of the marker. Default is by archiveType. Type pyleo.plot_default to see the default palette.
- **saveFig** (*bool*) – default is to not save the figure
- **dir** (*str*) – the full path of the directory in which to save the figure. If not provided, creates a default folder called 'figures' in the LiPD working directory (lipd.path).
- **format** (*str*) – One of the file extensions supported by the active backend. Default is "eps". Most backend support png, pdf, ps, eps, and svg.

Returns The figure

2.1.3 Plotting

pyleoclim.**plotTs**(*timeseries=''*, *x_axis=''*, *markersize=50*, *marker='default'*, *saveFig=False*, *dir=''*, *format='eps'*)

Plot a single time series.

Parameters

- **timeseries** (*A*) – By default, will prompt the user for one.
- **x_axis** (*str*) – The representation against which to plot the paleo-data. Options are "age", "year", and "depth". Default is to let the system choose if only one available or prompt the user.
- **markersize** (*int*) – default is 50.
- **marker** (*str*) – a string (or list) containing the color and shape of the marker. Default is by archiveType. Type pyleo.plot_default to see the default palette.
- **saveFig** (*bool*) – default is to not save the figure
- **dir** (*str*) – the full path of the directory in which to save the figure. If not provided, creates a default folder called 'figures' in the LiPD working directory (lipd.path).
- **format** (*str*) – One of the file extensions supported by the active backend. Default is "eps". Most backend support png, pdf, ps, eps, and svg.

Returns The figure.

pyleoclim.**histTs**(*timeseries=''*, *bins=None*, *hist=True*, *kde=True*, *rug=False*, *fit=None*, *hist_kws={'label': 'Histogram'}*, *kde_kws={'label': 'KDE fit'}*, *rug_kws={'label': 'Rug'}*, *fit_kws={'label': 'Fit'}*, *color='default'*, *vertical=False*, *norm_hist=True*, *saveFig=False*, *format='eps'*, *dir=''*)

Plot a univariate distribution of the PaleoData values

This function is based on the seaborn displot function, which is itself a combination of the matplotlib hist function with the seaborn kdeplot() and rugplot() functions. It can also fit scipy.stats distributions and plot the estimated PDF over the data.

Parameters

- **timeseries** – A timeseries. By default, will prompt the user for one.
- **bins** (*int*) – Specification of hist bins following matplotlib(hist), or None to use Freedman-Diaconis rule
- **hist** (*bool*) – Whether to plot a (normed) histogram
- **kde** (*bool*) – Whether to plot a gaussian kernel density estimate
- **rug** (*bool*) – Whether to draw a rugplot on the support axis
- **fit** – Random variable object. An object with fit method, returning a tuple that can be passed to a pdf method of positional arguments following a grid of values to evaluate the pdf on.
- **kde, rug, fit}_kws** (*{hist,*) – Dictionaries. Keyword arguments for underlying plotting functions. If modifying the dictionary, make sure the labels "hist", "kde", "rug" and "fit" are still passed.
- **color** (*str*) – matplotlib color. Color to plot everything but the fitted curve in. Default is to use the default paletter for each archive type.
- **vertical** (*bool*) – if True, oberved values are on y-axis.

- **norm_hist** (*bool*) – If True (default), the histrogram height shows a density rather than a count. This is implied if a KDE or fitted density is plotted
- **saveFig** (*bool*) – default is to not save the figure
- **dir** (*str*) – the full path of the directory in which to save the figure. If not provided, creates a default folder called 'figures' in the LiPD working directory (lipd.path).
- **format** (*str*) – One of the file extensions supported by the active backend. Default is "eps". Most backend support png, pdf, ps, eps, and svg.

Returns fig - The figure

Summary Plots

Summary plots are a special categories of plots enabled by Pyleoclim. They allow to plot specific information about a timeseries but are not customizable.

pyleoclim.**summaryTs**(*timeseries=''*, *x_axis=''*, *saveFig=False*, *dir=''*, *format='eps'*)

Basic summary plot

Plots the following information: the time series, a histogram of the PaleoData_values, location map, age-depth profile if both are available from the paleodata, metadata about the record.

Parameters

- **timeseries** – a timeseries object. By default, will prompt for one
- **x_axis** (*str*) – The representation against which to plot the paleo-data. Options are "age", "year", and "depth". Default is to let the system choose if only one available or prompt the user.
- **saveFig** (*bool*) – default is to not save the figure
- **dir** (*str*) – the full path of the directory in which to save the figure. If not provided, creates a default folder called 'figures' in the LiPD working directory (lipd.path).
- **format** (*str*) – One of the file extensions supported by the active backend. Default is "eps". Most backend support png, pdf, ps, eps, and svg.

Returns The figure

2.1.4 Statistics

pyleoclim.**statsTs**(*timeseries=''*)

Calculate simple statistics of a timeseries

Parameters **timeseries** – sytem will prompt for one if not given

Returns the mean, median, min, max, standard deviation and the inter-quartile range (IQR) of a timeseries.

Examples

```
>>> mean, median, min_, max_, std, IQR = pyleo.statsTs(timeseries)
```

pyleoclim.**corrSigTs**(*timeseries1=''*, *timeseries2=''*, *x_axis=''*, *interp_step=''*, *start=''*, *end=''*, *nsim=1000*, *method='isospectral'*, *alpha=0.5*)

Estimates the significance of correlations between non IID timeseries.

Function written by. F. Zhu.

Parameters

- **timeseries2** (*timeseries1,*) – timeseries object. Default is blank.
- **x-axis** (*str*) – The representation against which to express the paleo-data. Options are "age", "year", and "depth". Default is to let the system choose if only one available or prompt the user.
- **interp_step** (*float*) – the step size. By default, will prompt the user.
- **start** (*float*) – Start time/age/depth. Default is the maximum of the minima of the two timeseries
- **end** (*float*) – End time/age/depth. Default is the minimum of the maxima of the two timeseries
- **nsim** (*int*) – the number of simulations. Default is 1000
- **method** (*str*) – method use to estimate the correlation and significance. Available methods include:
  - 'ttest': T-test where the degrees of freedom are corrected for

    the effect of serial correlation
  - 'isopersistant': AR(1) modeling of the two timeseries
  - 'isospectral' (default): phase randomization of original

    inputs.

  The T-test is parametric test, hence cheap but usually wrong except in idyllic circumstances. The others are non-parametric, but their computational requirements scales with nsim.
- **alpha** (*float*) – significance level for critical value estimation. Default is 0.05

Returns

r (float) - correlation between the two timeseries

sig (bool) - Returns True if significant, False otherwise

p (real) - the p-value

2.1.5 Timeseries

Basic manipulations of the timeseries objects.

pyleoclim.**binTs**(*timeseries=''*, *x_axis=''*, *bin_size=''*, *start=''*, *end=''*)

Bin the paleoData values of the timeseries

Parameters

- **By default, will prompt the user for one.** (*timeseries.*) –
- **x-axis** (*str*) – The representation against which to plot the paleo-data. Options are "age", "year", and "depth". Default is to let the system choose if only one available or prompt the user.
- **bin_size** (*float*) – the size of the bins to be used. By default, will prompt for one

- **start** (*float*) – Start time/age/depth. Default is the minimum
- **end** (*float*) – End time/age/depth. Default is the maximum

Returns

binned_values- the binned output,

bins- the bins (centered on the median, i.e. the 100-200 bin is 150),

n- number of data points in each bin,

error- the standard error on the mean in each bin

pyleoclim.**interpTs**(*timeseries=''*, *x_axis=''*, *interp_step=''*, *start=''*, *end=''*)

Simple linear interpolation

Simple linear interpolation of the data using the numpy.interp method

Parameters

- **Default is blank, will prompt for it** (*timeseries.*) –
- **x-axis** (*str*) – The representation against which to plot the paleo-data. Options are "age", "year", and "depth". Default is to let the system choose if only one available or prompt the user.
- **interp_step** (*float*) – the step size. By default, will prompt the user.
- **start** (*float*) – Start year/age/depth. Default is the minimum
- **end** (*float*) – End year/age/depth. Default is the maximum

Returns

interp_age - the interpolated age/year/depth according to the end/start and time step,

interp_values - the interpolated values

pyleoclim.**standardizeTs**(*timeseries=''*, *scale=1*, *ddof=0*, *eps=0.001*)

Centers and normalizes the paleoData values of a given time series.

Constant or nearly constant time series not rescaled.

Parameters

- **x** (*array*) – vector of (real) numbers as a time series, NaNs allowed
- **scale** (*real*) – a scale factor used to scale a record to a match a given variance
- **axis** (*int or None*) – axis along which to operate, if None, compute over the whole array
- **ddof** (*int*) – degress of freedom correction in the calculation of the standard deviation
- **eps** (*real*) – a threshold to determine if the standard deviation is too close to zero

Returns

the standardized time series (z-score), Z = (X - E[X])/std(X)*scale, NaNs allowed

- mu (real): the mean of the original time series, E[X]
- sig (real): the standard deviation of the original time series, std[X]

Return type

- z (array)

References

1.Tapio Schneider's MATLAB code: http://www.clidyn.ethz.ch/imputation/standardize.m

2.The zscore function in SciPy: https://github.com/scipy/scipy/blob/master/scipy/stats/stats.py

@author: fzhu

2.1.6 Analysis in the frequency domain

pyleoclim.**wwzTs**(*timeseries=''*, *wwz=False*, *psd=True*, *wwz_default=True*, *psd_default=True*, *wwaplot_default=True*, *psdplot_default=True*, *fig=True*, *saveFig=False*, *dir=''*, *format='eps'*)

Weigthed wavelet Z-transform analysis

Wavelet analysis for unevenly spaced data adapted from Foster et al. (1996)

Parameters

- **timeseries** (*dict*) – A LiPD timeseries object (Optional, will prompt for one.)
- **wwz** (*bool*) – If True, will perform wavelet analysis
- **psd** (*bool*) – If True, will inform the power spectral density of the timeseries
- **wwz_default** – If True, will use the following default parameters:

  wwz_default = {'tau':None,'freqs':None,'c':1/(8*np.pi2),'Neff':3,'nMC':200,** 'nproc':8,'detrend':'no','method':'Kirchner_f2py'}.

  Modify the values for specific keys to change the default behavior.

- **psd_default** – If True, will use the following default parameters:

  psd_default = {'tau':None, 'freqs': None, 'c':1e-3, 'nproc':8, 'nMC':200, 'detrend':'no', 'Neff':3, 'anti_alias':False, 'avgs':2, 'method':'Kirchner_f2py'}

  Modify the values for specific keys to change the default behavior.

- **wwaplot_default** – If True, will use the following default parameters:

  wwaplot_default={'Neff':3, 'AR1_q':AR1_q, 'coi':coi, 'levels':None, 'tick_range':None, 'yticks':None, 'ylim':None, 'xticks':None, 'xlabels':None, 'figsize':[20,8], 'clr_map':'OrRd', 'cbar_drawedges':False, 'cone_alpha':0.5, 'plot_signif':True, 'signif_style':'contour', 'plot_cone':True}

  Modify the values for specific keys to change the default behavior.

- **psdplot_default** – If True, will use the following default parameters:

  psdplot_default={'lmstyle':None, 'linewidth':None, 'xticks':None, 'xlim':None, 'ylim':None, 'figsize':[20,8], 'label':'PSD', 'plot_ar1':True, 'psd_ar1_q95':psd_ar1_q95, 'psd_ar1_color':sns.xkcd_rgb["pale red"]}

  Modify the values for specific keys to change the default behavior.

- **fig** (*bool*) – If True, plots the figure
- **saveFig** (*bool*) – default is to not save the figure
- **dir** (*str*) – the full path of the directory in which to save the figure. If not provided, creates a default folder called 'figures' in the LiPD working directory (lipd.path).

- **format** (*str*) – One of the file extensions supported by the active backend. Default is "eps". Most backend support png, pdf, ps, eps, and svg.

Returns

A dictionary of outputs.

For wwz:

- wwa (array): The weights wavelet amplitude
- AR1_q (array): AR1 simulations
- coi (array): cone of influence
- freqs (array): vector for frequencies
- tau (array): the evenly-spaced time points, namely the time

shift for wavelet analysis.

- Neffs (array): The matrix of effective number of points in the

time-scale coordinates.

- coeff (array): The wavelet transform coefficients

For psd:

- psd (array): power spectral density
- freqs (array): vector of frequency
- psd_ar1_q95 (array): the 95% quantile of the psds of AR1 processes

fig: The figure

References

Foster, G. (1996). Wavelets for period analysis of unevenly sampled time series. The Astronomical Journal, 112(4), 1709-1729.

Examples

To run both wwz and psd:

```
>>> dict_out, fig = pyleoclim.wwzTs(wwz=True)
```

Note: This will return a single figure with wwa and psd

To change a default behavior:

```
>>> dict_out, fig = pyleoclim.wwzTs(psd_default = {'nMC':1000})
```

Return type dict_out (dict)

2.2 Using Pyleoclim without a LiPD file

The Pyleoclim modules can be called separately so the main functions can be used without a timeseries objects. The following modules are available:

- pyleoclim.Map: mapping functions
- pyleoclim.Plot: plotting functions
- pyleoclim.Stats: statistics (including correlation)
- pyleoclim.Timeseries: binning, interpolating
- pyleoclim.Spectral: analysis in the frequency domain

In addition, the pyleoclim.LipdUtils module allows the basic manipulation of LiPD files and the pyleoclim.SummaryPlots module contains some functions to extract information about a timeseries object.

CHAPTER

THREE

MAPPING FUNCTIONS

This module uses the Basemap package for mapping.

pyleoclim.Map.**mapAll**(*lat*, *lon*, *criteria*, *projection='robin'*, *lat_0=''*, *lon_0=''*, *llcrnrlat=-90*, *urcrnrlat=90*, *llcrnrlon=-180*, *urcrnrlon=180*, *countries=False*, *counties=False*, *rivers=False*, *states=False*, *background='none'*, *scale=0.5*, *palette=''*, *markersize=50*)

Map the location of all lat/lon according to some criteria

Map the location of all lat/lon according to some criteria. The choice of plotting color/marker is passed through palette according to unique criteria (e.g., record name, archive type, proxy observation type).

Parameters

- **lat** (*list*) – a list of latitude.
- **lon** (*list*) – a list of longitude.
- **criteria** (*list*) – a list of criteria for plotting purposes. For instance, a map by the types of archive present in the dataset or proxy observations.
- **projection** (*string*) – the map projection. Refers to the Basemap documentation for a list of available projections. Only projections supporting setting the map center with a single lat/lon or with the coordinates of the rectangle are currently supported. Default is to use a Robinson projection.
- **lon_0** (*lat_0,*) – the center coordinates for the map. Default is mean latitude/longitude in the list. If the chosen projection doesn't support it, Basemap will ignore the given values.
- **urcrnrlat, llcrnrlon, urcrnrlon** (*llcrnrlat,*) – The coordinates of the two opposite corners of the rectangle.
- **countries** (*bool*) – Draws the countries border. Defaults is off (False).
- **counties** (*bool*) – Draws the USA counties. Default is off (False).
- **rivers** (*bool*) – Draws the rivers. Default is off (False).
- **states** (*bool*) – Draws the American and Australian states borders. Default is off (False).
- **background** (*string*) – Plots one of the following images on the map: bluemarble, etopo, shadedrelief, or none (filled continents). Default is none.
- **scale** (*float*) – Useful to downgrade the original image resolution to speed up the process. Default is 0.5.
- **palette** (*dict*) – A dictionary of plotting color/marker by criteria. The keys should correspond to ***unique*** criteria with a list of associated values. The list should be in the format ['color', 'marker'].
- **markersize** (*int*) – The size of the marker.

Returns The figure

pyleoclim.Map.**mapOne**(*lat*, *lon*, *projection='ortho'*, *lat_0=''*, *lon_0=''*, *llcrnrlat=-90*, *urcrnrlat=90*, *llcrnrlon=-180*, *urcrnrlon=180*, *countries=True*, *counties=False*, *rivers=False*, *states=False*, *background='shadedrelief'*, *scale=0.5*, *markersize=50*, *marker='ro'*)

Map one location on the globe

Parameters

- **lat** (*float*) – a float number representing latitude
- **lon** (*float*) – a float number representing longitude
- **projection** (*string*) – the map projection. Refers to the Basemap documentation for a list of available projections. Only projections supporting setting the map center with a single lat/lon or with the coordinates of the rectangle are currently supported. Default is to use a Robinson projection.
- **lon_0** (*lat_0,*) – the center coordinates for the map. Default is mean latitude/longitude in the list. If the chosen projection doesn't support it, Basemap will ignore the given values.
- **urcrnrlat, llcrnrlon, urcrnrlon** (*llcrnrlat,*) – The coordinates of the two opposite corners of the rectangle.
- **countries** (*bool*) – Draws the countries border. Defaults is off (False).
- **counties** (*bool*) – Draws the USA counties. Default is off (False).
- **rivers** (*bool*) – Draws the rivers. Default is off (False).
- **states** (*bool*) – Draws the American and Australian states borders. Default is off (False).
- **background** (*string*) – Plots one of the following images on the map: bluemarble, etopo, shadedrelief, or none (filled continents). Default is none.
- **scale** (*float*) – Useful to downgrade the original image resolution to speed up the process. Default is 0.5.
- **markersize** (*int*) – The size of the marker.
- **marker** (*str or list*) – color and type of marker.

CHAPTER

FOUR

PLOTTING FUNCTIONS

pyleoclim.Plot.**plot**(*x*, *y*, *markersize=50*, *marker='ro'*, *x_label=''*, *y_label=''*, *title=''*)

Make a 2-D plot

Parameters

- **x** (*numpy array*) – a 1xn numpy array of values for the x-axis
- **y** (*numpy array*) – a 1xn numpy array for the y-axis
- **markersize** (*int*) – the size of the marker
- **marker** (*string or list*) – color and shape of the marker
- **x_axis_label** (*str*) – the label for the x-axis
- **y_axis_label** (*str*) – the label for the y-axis
- **title** (*str*) – the title for the plot

Returns The figure

pyleoclim.Plot.**plot_hist**(*y*, *bins=None*, *hist=True*, *label=''*, *kde=True*, *rug=False*, *fit=None*, *hist_kws={'label': 'Histogram'}*, *kde_kws={'label': 'KDE fit'}*, *rug_kws={'label': 'rug'}*, *fit_kws={'label': 'fit'}*, *color='0.7'*, *vertical=False*, *norm_hist=True*)

Plot a univariate distribution of the PaleoData values

This function is based on the seaborn displot function, which is itself a combination of the matplotlib hist function with the seaborn kdeplot() and rugplot() functions. It can also fit scipy.stats distributions and plot the estimated PDF over the data.

Parameters

- **y** (*array*) – nx1 numpy array. No missing values allowed
- **bins** (*int*) – Specification of hist bins following matplotlib(hist), or None to use Freedman-Diaconis rule
- **hist** (*bool*) – Whether to plot a (normed) histogram
- **label** (*str*) – The label for the axis
- **kde** (*bool*) – Whether to plot a gaussian kernel density estimate
- **rug** (*bool*) – Whether to draw a rugplot on the support axis
- **fit** – Random variable object. An object with fit method, returning a tuple that can be passed to a pdf method of positional arguments following a grid of values to evaluate the pdf on.

- **kde, rug, fit}_kws** (*{hist,*) – Dictionaries. Keyword arguments for underlying plotting functions. If modifying the dictionary, make sure the labels "hist", "kde", "rug" and "fit" are still passed.
- **color** (*str*) – matplotlib color. Color to plot everything but the fitted curve in.
- **vertical** (*bool*) – if True, oberved values are on y-axis.
- **norm_hist** (*bool*) – If True (default), the histrogram height shows a density rather than a count. This is implied if a KDE or fitted density is plotted

Returns fig - The figure

CHAPTER

FIVE

STATISTICS FUNCTIONS

pyleoclim.Stats.**simpleStats**(*y*, *axis=None*)

Computes simple statistics

Computes the mean, median, min, max, standard deviation, and interquartile range of a numpy array y.

Parameters

- **y** (*array*) – A Numpy array
- **axis** (*int, typle of ints*) – Optional. Axis or Axes along which the means are computed, the default is to compute the mean of the flattened array. If a tuple of ints, performed over multiple axes

Returns The mean, median, min, max, standard deviation and IQR by columns

pyleoclim.Stats.**corrsig**(*y1*, *y2*, *nsim=1000*, *method='isospectral'*, *alpha=0.5*)

Estimates the significance of correlations between non IID time series by 3 independent methods:

1. 'ttest': T-test where d.o.f are corrected for the effect of serial correlation
2. 'isopersistent': AR(1) modeling of x and y.

3) 'isospectral': phase randomization of original inputs. (default) The T-test is parametric test, hence cheap but usually wrong except in idyllic circumstances. The others are non-parametric, but their computational requirements scales with nsim.

Parameters

- **y2** (*y1,*) –
- **nsim** (*int*) –
- **method** (*str*) –
- **alpha** (*float*) –

Returns

correlation between x and y

signif (int): true if significant; false otherwise

p (real): Fraction of time series with higher correlation coefficents than observed (approximates the p-value).

Note that signif = True if and only if p <= alpha.

Return type r (real)

CHAPTER

SIX

TIMESERIES FUNCTIONS

This module allows the manipulation of timeseries.

pyleoclim.Timeseries.**bin**(*x*, *y*, *bin_size=''*, *start=''*, *end=''*)

Bin the values

Parameters

- **x** (*array*) – the x-axis series.
- **y** (*array*) – the y-axis series.
- **bin_size** (*float*) – The size of the bins. Default is the average resolution
- **start** (*float*) – Where/when to start binning. Default is the minimum
- **end** (*float*) – When/where to stop binning. Defulat is the maximum

Returns

binned_values - the binned output

bins - the bins (centered on the median, i.e., the 100-200 bin is 150)

n - number of data points in each bin

error - the standard error on the mean in each bin

pyleoclim.Timeseries.**interp**(*x*, *y*, *interp_step=''*, *start=''*, *end=''*)

Linear interpolation onto a new x-axis

Parameters

- **x** (*array*) – the x-axis
- **y** (*array*) – the y-axis
- **interp_step** (*float*) – the interpolation step. Default is mean resolution.
- **start** (*float*) – where/when to start the interpolation. Default is min..
- **end** (*float*) – where/when to stop the interpolation. Defaul is max.

Returns

xi - the interpolated x-axis

interp_values - the interpolated values

pyleoclim.Timeseries.**onCommonAxis**(*x1*, *y1*, *x2*, *y2*, *interp_step=''*, *start=''*, *end=''*)

Places two timeseries on a common axis

Parameters

- **x1** (*array*) – x-axis values of the first timeseries
- **y1** (*array*) – y-axis values of the first timeseries
- **x2** (*array*) – x-axis values of the second timeseries
- **y2** (*array*) – y-axis values of the second timeseries
- **interp_step** (*float*) – The interpolation step. Default is mean resolution of lowest resolution series
- **start** (*float*) – where/when to start. Default is the maximum of the minima of the two timeseries
- **end** (*float*) – Where/when to end. Default is the minimum of the maxima of the two timeseries

Returns

xi - the interpolated x-axis

interp_values1 - the interpolated y-values for the first timeseries interp_values2 - the intespolated y-values for the second timeseries

pyleoclim.Timeseries.**standardize**(*x*, *scale=1*, *axis=0*, *ddof=0*, *eps=0.001*)

Centers and normalizes a given time series. Constant or nearly constant time series not rescaled.

Parameters

- **x** (*array*) – vector of (real) numbers as a time series, NaNs allowed
- **scale** (*real*) – a scale factor used to scale a record to a match a given variance
- **axis** (*int or None*) – axis along which to operate, if None, compute over the whole array
- **ddof** (*int*) – degress of freedom correction in the calculation of the standard deviation
- **eps** (*real*) – a threshold to determine if the standard deviation is too close to zero

Returns the standardized time series (z-score), Z = (X - E[X])/std(X)*scale, NaNs allowed mu (real): the mean of the original time series, E[X] sig (real): the standard deviation of the original time series, std[X]

Return type z (array)

References

1.Tapio Schneider's MATLAB code: http://www.clidyn.ethz.ch/imputation/standardize.m

2.The zscore function in SciPy: https://github.com/scipy/scipy/blob/master/scipy/stats/stats.py

@author: fzhu

CHAPTER

SEVEN

LIPD UTILITIES

This modules allow basic manipulation of LiPD files

7.1 Creating Directories and saving

pyleoclim.LipdUtils.**createDir**(*path*, *foldername*)
Create a new folder in a working directory

Create a new folder in a working directory to save outputs from Pyleoclim.

Parameters

- **path** (*str*) – the path to the new folder.
- **foldername** (*str*) – the name of the folder to be created

Returns newdir - the full path to the new directory

pyleoclim.LipdUtils.**saveFigure**(*name*, *format='eps'*, *dir=''*)
Save a figure

Save the figure in the directory. If not given, creates a folder in the current working directory.

Parameters

- **name** (*str*) – name of the file
- **format** (*str*) – One of the file extensions supported by the active backend. Default is "eps". Most backend support png, pdf, ps, eps, and svg.
- **dir** (*str*) – the name of the folder in the LiPD working directory. If not provided, creates a default folder called 'figures'.

7.2 LiPD files

pyleoclim.LipdUtils.**enumerateLipds**()
Enumerate the LiPD files loaded in the workspace

pyleoclim.LipdUtils.**promptForLipd**()
Prompt for a LiPD file

Ask the user to select a LiPD file from a list Use this function in conjunction with enumerateLipds()

Returns The index of the LiPD file

7.3 Handling Variables

pyleoclim.LipdUtils.**promptForVariable**()
Prompt for a specific variable

Ask the user to select the variable they are interested in. Use this function in conjunction with readHeaders() or getTSO()

Returns The index of the variable

pyleoclim.LipdUtils.**xAxisTs**(*timeseries*)
Prompt the user to choose a x-axis representation for the timeseries.

Parameters **timeseries** – a timeseries object

Returns

x_axis - the values for the x-axis representation,

label - returns either "age", "year", or "depth"

pyleoclim.LipdUtils.**checkXaxis**(*timeseries*, *x_axis=''*)
Check that a x-axis is present for the timeseries

Parameters

- **timeseries** – a timeseries
- **x_axis** (*str*) – the x-axis representation, either depth, age or year

Returns

x - the values for the x-axis representation,

label - returns either "age", "year", or "depth"

7.4 Handling timeseries objects

pyleoclim.LipdUtils.**enumerateTs**(*timeseries_list*)
Enumerate the available time series objects

Parameters **timeseries_list** – a list of available timeseries objects. To use the timeseries loaded upon initiation of the pyleoclim package, use pyleo.time_series.

pyleoclim.LipdUtils.**getTs**(*timeseries_list*)
Get a specific timeseries object from a dictionary of timeseries

Parameters **timeseries_list** – a list of available timeseries objects. To use the timeseries loaded upon initiation of the pyleoclim package, use pyleo.time_series.

Returns A single timeseries object

7.5 Linking LiPDs to the LinkedEarth Ontology

pyleoclim.LipdUtils.**LipdToOntology**(*archiveType*)
standardize archiveType

Transform the archiveType from their LiPD name to their ontology counterpart

Parameters **archiveType** (*STR*) – name of the archiveType from the LiPD file

Returns archiveType according to the ontology

CHAPTER

EIGHT

SUMMARY PLOTS

This module handles some basic return for the summary plots.

Requires a LiPD file

pyleoclim.SummaryPlots.**getMetadata**(*timeseries*)
Get the necessary metadata to be printed out automatically

Parameters **timeseries** – a specific timeseries object.

Returns

archiveType

Authors (if more than 2, replace by et al.

PublicationYear

Publication DOI

Variable Name

Units

Climate Interpretation

Calibration Equation

Calibration References

Calibration Notes

Return type A dictionary containing the following metadata

pyleoclim.SummaryPlots.**TsData**(*timeseries*, *x_axis=''*)
Get the PaleoData with age/depth information

Get the necessary information for the TS plots/necessary to allow for axes specification

Parameters

- **timeseries** – a single timeseries object. By default, will prompt the user
- **x-axis** (*str*) – The representation against which to plot the paleo-data. Options are "age", "year", and "depth". Default is to let the system choose if only one available or prompt the user.

Returns

x - the x-valus

y - the y-values

archiveType - the archiveType (for plot settings)

x_label - the label for the x-axis

y_label - the label for the y-axis

label - the results of the x-axis query. Either depth, year, or age

pyleoclim.SummaryPlots.**agemodelData**(*timeseries*)

Get the necessary information for the agemodel plot

Parameters **timeseries** – a single timeseries object. By default, will prompt the user

Returns

depth - the depth values

age - the age values

x_label - the label for the x-axis

y_label - the label for the y-axis

archiveType - the archiveType (for default plot settings)

CHAPTER

NINE

SPECTRAL FUNCTIONS

This module allows analysis in the frequency domain

pyleoclim.Spectral.**ar1_fit**(*ys*, *ts=None*, *detrend='no'*)

Returns the lag-1 autocorrelation from ar1 fit OR persistence from tauest.

Parameters

- **ys** (*array*) – the time series
- **ts** (*array*) – the time axis of that series
- **detrend** (*str*) – 'no' - the original time series is assumed to have no trend; 'linear' - a linear least-squares fit to *ys* is subtracted; 'constant' - the mean of *ys* is subtracted

Returns lag-1 autocorrelation coefficient (for evenly-spaced time series) OR estimated persistence (for unevenly-spaced time series)

Return type g (float)

pyleoclim.Spectral.**ar1_sim**(*ys*, *n*, *p*, *ts=None*, *detrend='no'*)

Produce p realizations of an AR1 process of length n with lag-1 autocorrelation g calculated from *ys* and *ts*

Parameters

- **ys** (*array*) – a time series
- **p** (*n,*) – dimensions as n rows by p columns
- **ts** (*array*) – the time axis of that series
- **detrend** (*str*) – 'no' - the original time series is assumed to have no trend; 'linear' - a linear least-squares fit to *ys* is subtracted; 'constant' - the mean of *ys* is subtracted

Returns n rows by p columns matrix of an AR1 process

Return type red (matrix)

pyleoclim.Spectral.**wwz**(*ys*, *ts*, *tau=None*, *freqs=None*, *c=0.012665147955292222*, *Neff=3*, *nMC=200*, *nproc=8*, *detrend='no'*, *method='Kirchner_f2py'*)

Return the weighted wavelet amplitude (WWA) with phase, AR1_q, and cone of influence, as well as WT coeeficients

Parameters

- **ys** (*array*) – a time series, NaNs will be deleted automatically
- **ts** (*array*) – the time points, if *ys* contains any NaNs, some of the time points will be deleted accordingly
- **tau** (*array*) – the evenly-spaced time points
- **freqs** (*array*) – vector of frequency

- **c** (*float*) – the decay constant, the default value 1/(8*np.pi**2) is good for most of the cases
- **Neff** (*int*) – effective number of points
- **nMC** (*int*) – the number of Monte-Carlo simulations
- **nproc** (*int*) – the number of processes for multiprocessing
- **detrend** (*str*) – 'no' - the original time series is assumed to have no trend; 'linear' - a linear least-squares fit to *ys* is subtracted; 'constant' - the mean of *ys* is subtracted
- **method** (*str*) – 'Foster' - the original WWZ method; 'Kirchner' - the method Kirchner adapted from Foster; 'Kirchner_f2py' - the method Kirchner adapted from Foster with f2py

Returns the weighted wavelet amplitude. AR1_q (array): AR1 simulations coi (array): cone of influence freqs (array): vector of frequency tau (array): the evenly-spaced time points, namely the time shift for wavelet analysis Neffs (array): the matrix of effective number of points in the time-scale coordinates coeff (array): the wavelet transform coefficents

Return type wwa (array)

pyleoclim.Spectral.**wwz_psd**(*ys*, *ts*, *freqs=None*, *tau=None*, *c=0.001*, *nproc=8*, *nMC=200*, *detrend='no'*, *Neff=3*, *anti_alias=False*, *avgs=2*, *method='Kirchner_f2py'*)

Return the psd of a timeseries directly using wwz method.

Parameters

- **ys** (*array*) – a time series, NaNs will be deleted automatically
- **ts** (*array*) – the time points, if *ys* contains any NaNs, some of the time points will be deleted accordingly
- **freqs** (*array*) – vector of frequency
- **tau** (*array*) – the evenly-spaced time points, namely the time shift for wavelet analysis
- **c** (*float*) – the decay constant, the default value 1e-3 is good for most of the cases
- **nproc** (*int*) – the number of processes for multiprocessing
- **nMC** (*int*) – the number of Monte-Carlo simulations
- **detrend** (*str*) – 'no' - the original time series is assumed to have no trend; 'linear' - a linear least-squares fit to *ys* is subtracted; 'constant' - the mean of *ys* is subtracted
- **method** (*str*) – 'Foster' - the original WWZ method; 'Kirchner' - the method Kirchner adapted from Foster; 'Kirchner_f2py' - the method Kirchner adapted from Foster with f2py

Returns power spectral density freqs (array): vector of frequency psd_ar1_q95 (array): the 95% quantile of the psds of AR1 processes

Return type psd (array)

pyleoclim.Spectral.**plot_wwa**(*wwa, freqs, tau, Neff=3, AR1_q=None, coi=None, levels=None, tick_range=None, yticks=None, ylim=None, xticks=None, xlabels=None, figsize=[20, 8], clr_map='OrRd', cbar_drawedges=False, cone_alpha=0.5, plot_signif=False, signif_style='contour', plot_cone=False, ax=None*)

Plot the wavelet amplitude

Parameters

- **wwa** (*array*) – the weighted wavelet amplitude.

- **freqs** (*array*) – vector of frequency
- **tau** (*array*) – the evenly-spaced time points, namely the time shift for wavelet analysis
- **Neff** (*int*) – effective number of points
- **AR1_q** (*array*) – AR1 simulations
- **coi** (*array*) – cone of influence
- **levels** (*array*) – levels of values to plot
- **tick_range** (*array*) – levels of ticks to show on the colorbar
- **yticks** (*list*) – ticks on y-axis
- **ylim** (*list*) – limitations for y-axis
- **xticks** (*list*) – ticks on x-axis
- **figsize** (*list*) – the size for the figure
- **clr_map** (*str*) – the name of the colormap
- **cbar_drawedges** (*bool*) – whether to draw edges on the colorbar or not
- **cone_alpha** (*float*) – the alpha value for the area covered by cone of influence
- **plot_signif** (*bool*) – plot 95% significant area or not
- **signif_style** (*str*) – plot 95% significant area with *contour* or *shade*
- **plot_cone** (*bool*) – plot cone of influence or not

Returns the 2-D plot of wavelet analysis

Return type fig (figure)

pyleoclim.Spectral.**plot_wwadist**(*wwa*)

Plot the distribution of wwa with the 95% quantile line.

Parameters **wwa** (*array*) – the weighted wavelet amplitude.

Returns the 2-D plot of wavelet analysis

Return type fig (figure)

pyleoclim.Spectral.**plot_psd**(*psd, freqs, lmstyle=None, linewidth=None, xticks=None, xlim=None, ylim=None, figsize=[20, 8], label='PSD', plot_ar1=True, psd_ar1_q95=None, psd_ar1_color='#d9544d', ax=None*)

Plot the wavelet amplitude

Parameters

- **psd** (*array*) – power spectral density
- **freqs** (*array*) – vector of frequency
- **xticks** (*list*) – ticks on x-axis
- **xlim** (*list*) – limitations for x-axis
- **figsize** (*list*) – the size for the figure

Returns the 2-D plot of wavelet analysis

Return type fig (figure)

CHAPTER

TEN

INDICES AND TABLES

- genindex
- modindex
- search

INDEX