fredpy Documentation

Release 2.0.1

Brian C. Jenkins

Aug 25, 2016

Contents:

FREDPY.SERIES CLASS

class `fredpy.`**`series`**(*series_id=None*)

Creates an instance of a *`fredpy.series`* object that stores information about the specified data series from FRED with the unique series ID code given by `series_id`.

Parametersseries_id (*`string`*) – unique FRED series ID. If `series_id` equals None, an empy *`fredpy.series`* object is created.

Attributes:

data(numpy ndarray) – data values.

daterange(string) – specifies the dates of the first and last observations.

dates(list) – list of date strings in YYYY-MM-DD format.

datetimes(numpy ndarray) – array containing observation dates formatted as datetime objects.

freq(string) – data frequency. 'Daily', 'Weekly', 'Monthly', 'Quarterly', or 'Annual'.

idCode(string) – unique FRED series ID code.

season(string) – specifies whether the data has been seasonally adjusted.

source(string) – original source of the data.

t(integer) – number corresponding to frequency: 365 for daily, 52 for weekly, 12 for monthly, 4 for quarterly, and 1 for annual.

title(string) – title of the data series.

units(string) – units of the data series.

updated(string) – date series was last updated.

Methods:

apc (*log=True*, *method='backward'*)

Computes the percentage change in the data over one year.

Parameters

•**log** (*`bool`*) – If True, computes the percentage change as $100 \cdot \log(x_t/x_{t-1})$. If False, compute the percentage change as $100 \cdot (x_t/x_{t-1} - 1)$.

•**method** (*`string`*) – If 'backward', compute percentage change from the previous period. If 'forward', compute percentage change from current to subsequent period.

Returns`fredpy.series`

bpfilter (*low=6*, *high=32*, *K=12*)

Computes the bandpass (Baxter-King) filter of the data.

Parameters

>   • **low** (*integer*) – Minimum period for oscillations. Select 24 for monthly data,
>   6 for quarterly data (default), and 3 for annual data.
>   • **high** (*integer*) – Maximum period for oscillations. Select 84 for monthly data,
>   32 for quarterly data (default), and 8 for annual data.
>   • **K** (*integer*) – Lead-lag length of the filter. Select, 84 for monthly data, 12 for
>   for quarterly data (default), and 1.5 for annual data.
>
>   **Returns** *fredpy.series*

Note: In the returned series, the following attributes are different from the input series:
> **dates**(list) – Removes K values from each end of the original series.
> **datetimes**(numpy ndarray) – Removes K observations from each end of the original
>   series.
> **daterange**(string) – Corrects for the shorter date range.
> **data**(pandas.core.frame.DataFrame) – Changes the data attribute to a pandas
>   DataFrame with the following columns:
>   **actual**(numpy ndarray) – unfiltered series with K observations removed from
>   each end.
>   **cycle**(numpy ndarray) – cyclical component of series.
>   **trend**(numpy ndarray) – trend component of series.

---

cffilter (*low=6*, *high=32*)
> Computes the Hodrick-Prescott filter of the data.
>   **Parameters**
>   • **low** (*integer*) – Minimum period for oscillations. Select 6 for quarterly data
>   (default) and 1.5 for annual data.
>   • **high** (*integer*) – Maximum period for oscillations. Select 32 for quarterly data
>   (default) and 8 for annual data.
>   **Returns** *fredpy.series*

Note: In the returned series, the data attribute is a Pandas DataFrame with the following
columns:
>   **actual**(numpy ndarray) – unfiltered data series.
>   **cycle**(numpy ndarray) – cyclical component of series.
>   **trend**(numpy ndarray) – trend component of series.

---

copy ()
> Returns a copy of the *fredpy.series* object.
>   **Parameters**
>   **Returns** *fredpy.series*

divide (*series2*)
> Divides the data from the current fredpy series by the data from `series2`.
>   **Parameters** **series2** (*fredpy.series*) – A `fredpy.series` object.
>   **Returns** *fredpy.series*

firstdiff ()
> Computes the first difference filter of original series.
>   **Parameters**
>   **Returns** *fredpy.series*

Note: In the returned series, the following attributes are different from the input series:
> **dates**(list) – Removes the first value from the original series.
> **datetimes**(numpy ndarray) – Removes the first value from the original series.
> **daterange**(string) – Corrects for the shorter date range.

> **data**(pandas.core.frame.DataFrame) – Changes the data attribute to a pandas
> DataFrame with the following columns:
>> **actual**(numpy ndarray) – unfiltered series with the first observation removed from the series.
>> **cycle**(numpy ndarray) – cyclical component of data values.
>> **trend**(numpy ndarray) – trend component of data values.

hpfilter(*lamb=1600*)

Computes the Hodrick-Prescott filter of the data.

> **Parameters****lamb** (*integer*) – The Hodrick-Prescott smoothing parameter. Select 129600 for monthly data, 1600 for quarterly data (default), and 6.25 for annual data.
> **Returns***fredpy.series*

> **Note:** In the returned series, the data attribute is a Pandas DataFrame with the following columns:
>> **actual**(numpy ndarray) – unfiltered data series.
>> **cycle**(numpy ndarray) – cyclical component of series.
>> **trend**(numpy ndarray) – trend component of series.

lintrend()

Computes a simple linear filter of the data using OLS.

> **Parameters**
> **Returns***fredpy.series*

> **Note:** In the returned series, the data attribute is a Pandas DataFrame with the following columns:
>> **actual**(numpy ndarray) – unfiltered data series.
>> **cycle**(numpy ndarray) – cyclical component of series.
>> **trend**(numpy ndarray) – trend component of series.

log()

Computes the natural log of the data.

> **Parameters**
> **Returns***fredpy.series*

ma1side(*length*)

Computes a one-sided moving average with window equal to `length`.

> **Parameters****length** (*integer*) – `length` of the one-sided moving average.
> **Returns***fredpy.series*

ma2side(*length*)

> Computes a two-sided moving average with window equal to 2 times `length`.
>> **param integer length**half of `length` of the two-sided moving average. For example, if `length = 12`, then the moving average will contain 24 the 12 periods before and the 12 periods after each observation.
>> **return***fredpy.series*

minus(*series2*)

Subtracts the data from `series2` from the data from the current fredpy series.

> **Parameters****series2** (*fredpy.series*) – A `fredpy.series` object.
> **Returns***fredpy.series*

monthtoannual(*method='average'*)

Converts monthly data to annual data.

> **Parameters****method** (*string*) – If 'average', use the average values over each twelve

month interval (default), if 'sum,' use the sum of the values over each twelve month interval, and if 'end' use the values at the end of each twelve month interval.

> **Returns** *fredpy.series*

monthtoquarter (*method='average'*)

Converts monthly data to quarterly data.

> **Parameters method** (*string*) – If 'average', use the average values over each three month interval (default), if 'sum,' use the sum of the values over each three month interval, and if 'end' use the values at the end of each three month interval.
>
> **Returns** *fredpy.series*

pc (*log=True*, *method='backward'*, *annualized=False*)

Computes the percentage change in the data from the preceding period.

> **Parameters**
>
> • **log** (*bool*) – If True, computes the percentage change as $100 \cdot \log(x_t/x_{t-1})$. If False, compute the percentage change as $100 \cdot (x_t/x_{t-1} - 1)$.
>
> • **method** (*string*) – If 'backward', compute percentage change from the previous period. If 'forward', compute percentage change from current to subsequent period.
>
> • **annualized** (*bool*) – If True, percentage change is annualized by multiplying the simple percentage change by the number of data observations per year. E.g., if the data are monthly, then the annualized percentage change is $4 \cdot 100 \cdot \log(x_t/x_{t-1})$.
>
> **Returns** *fredpy.series*

percapita (*total_pop=True*)

Transforms the data into per capita terms (US) by dividing by one of two measures of the total population.

> **Parameters total_pop** (*string*) – If `total_pop == True`, then use the toal population (Default). Else, use Civilian noninstitutional population defined as persons 16 years of age and older.
>
> **Returns** *fredpy.series*

plus (*series2*)

Adds the data from the current fredpy series to the data from `series2`.

> **Parameters series2** (*fredpy.series*) – A `fredpy.series` object.
>
> **Returns** *fredpy.series*

quartertoannual (*method='average'*)

Converts quarterly data to annual data.

> **Parameters method** (*string*) – If 'average', use the average values over each four quarter interval (default), if 'sum,' use the sum of the values over each four quarter interval, and if 'end' use the values at the end of each four quarter interval.
>
> **Returns** *fredpy.series*

recent (*N*)

Restrict the data to the most recent N observations.

> **Parameters N** (*integer*) – Number of periods to include in the data window.
>
> **Returns** *fredpy.series*

recessions (*color='0.5'*, *alpha = 0.5*)

Creates recession bars for plots. Should be used after a plot has been made but before either (1) a new plot is created or (2) a show command is issued.

> **Parameters**
>
> • **color** (*string*) – Color of the bars. Default: '0.5'.
>
> • **alpha** (*float*) – Transparency of the recession bars. Must be between 0 and 1. Default: 0.5.

Returns

times(*series2*)

Multiplies the data from the current fredpy series with the data from `series2`.

Parameters series2 (`fredpy.series`) – A `fredpy.series` object.

Returns *fredpy.series*

window(*win*)

Restricts the data to the most recent N observations.

Parameters win (*list*) – is an ordered pair: `win = [win_min,win_max]` where `win_min` is the date of the minimum date desired and `win_max` is the date of the maximum date. Date strings must be entered in either 'yyyy-mm-dd' or 'mm-dd-yyyy' format.

Returns *fredpy.series*

ADDITIONAL `FREDPY` FUNCTIONS

fredpy.**date_times**(*date_strings*)

> Converts a list of date strings in yyyy-mm-dd format to datetime.

>> **Parameters** **date_strings** (*list*) – a list of date strings formated as: 'yyyy-mm-dd'.

>> **Returns** numpy ndarray

fredpy.**divide**(*series1*, *series2*)

> Divides the data from `series1` by the data from `series2`.

>> **Parameters**

>>> • **series1** (*fredpy.series*) – A `fredpy.series` object.

>>> • **series2** (*fredpy.series*) – A `fredpy.series` object.

>> **Returns** *fredpy.series*

fredpy.**plus**(*series1*, *series2*)

> Adds the data from `series1` to the data from `series2`.

>> **Parameters**

>>> • **series1** (*fredpy.series*) – A `fredpy.series` object.

>>> • **series2** (*fredpy.series*) – A `fredpy.series` object.

>> **Returns** *fredpy.series*

fredpy.**quickplot**(*fred_series*, *year_mult=10*, *show=True*, *recess=False*, *save=False*, *filename='file'*, *linewidth=2*, *alpha = 0.75*)

> Create a plot of a FRED data series

>> **Parameters**

>>> • **fred_series** (*fredpy.series*) – A `fredpy.series` object.

>>> • **year_mult** (*integer*) – Interval between year ticks on the x-axis. Default: 10.

>>> • **show** (*bool*) – Show the plot? Default: True.

>>> • **recess** (*bool*) – Show recession bars in plot? Default: False.

>>> • **save** (*bool*) – Save the image to file? Default: False.

>>> • **filename** (*string*) – Name of file to which image is saved *without an extension*. Default: `'file'`.

>>> • **linewidth** (*float*) – Width of plotted line. Default: 2.

>>> • **alpha** (*float*) – Transparency of the recession bars. Must be between 0 and 1. Default: 0.7.

Returns

`fredpy.`**`minus`**(*series1*, *series2*)

Subtracts the data from `series2` from the data from `series1`.

Parameters

- **`series1`** (`fredpy.series`) – A `fredpy.series` object.

- **`series2`** (`fredpy.series`) – A `fredpy.series` object.

Returns *fredpy.series*

`fredpy.`**`times`**(*series1*, *series2*)

Multiplies the data from `series1` with the data from `series2`.

Parameters

- **`series1`** (`fredpy.series`) – A `fredpy.series` object.

- **`series2`** (`fredpy.series`) – A `fredpy.series` object.

Returns *fredpy.series*

`fredpy.`**`window_equalize`**(*series_list*)

Adjusts the date windows for a collection of fredpy.series objects to the smallest common window.

Parameters **`series_list`** (*list*) – A list of *fredpy.series* objects

Returns

INDICES AND TABLES

- genindex
- modindex
- search

F