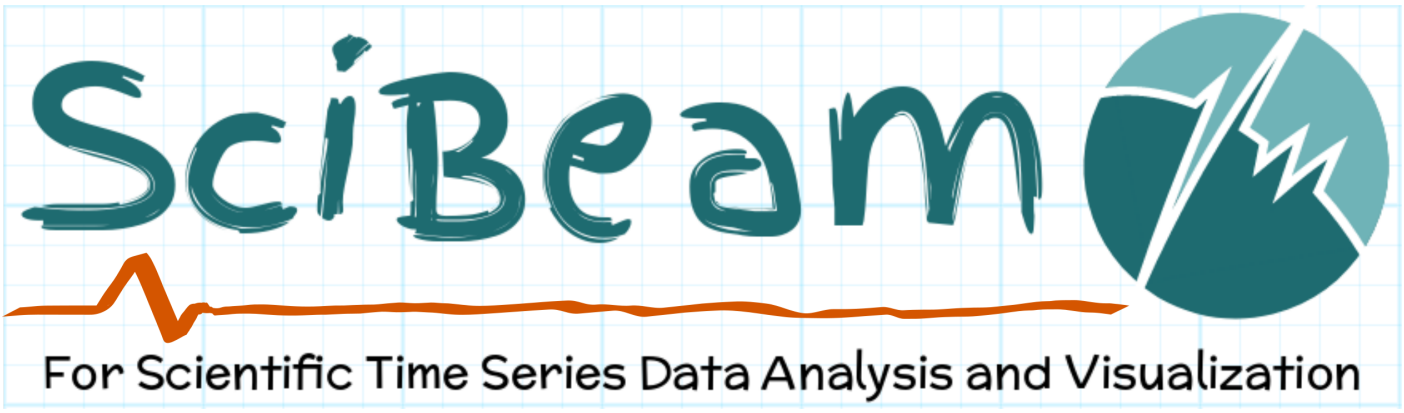
scibeam Documentation

Release 0.1.1

Yu Lu

Aug 27, 2018

GENERAL



SciBeam is a python package build on top of pandas, numpy, sicpy and matplotlib. It is aimed for quick and easy scientific time-series data analysis and visualization in physics, optics, mechanics, and many other STEM subjects.

In the context of scientific data analysis, there are a lot of situations that people have to deal with time-series data, such as time dependent experiment(e.g. temperature measurement), dynamic processes(e.g. beam propagation, chemical reaction), system long/short term behavior(e.g. noise), etc. Quite often is that data taking and result analysis is gaped by some time and effort, which could result in complains or regrets during the data analysis, like "I wish I took another measurement of … so than I could explain why …". As such, the general guidline of scibeam is to bridge the gap between measurement and data analysis, so that time-series related experiment can be done in a more guided way.

The basic features of scibeam include but not limited to: beam propagation, single or multi-dimentional time depedent measurement, data file auto indexing, noise reduction, peak analysis, numerical fittings, etc.

Note: scibeam doesn't support python 2.7, make sure you have the right python version (>=3.4).

CHAPTER

ONE

ABOUT

1.1 This document

This document is created uisng Sphinx and autodoc. The general rule of the static html is configured in reStructured-Text and api document is generated by Sphinx-apidoc, configure in conf.py.

The general html structure looks like below:

```
html/
├── about.html
├── conf.html
├── genindex.html
├── index.html
├── install.html
├── _modules
│   ├── index.html
│   └── scibeam
│       ├── core
│       │   ├── base.html
│       │   ├── common.html
│       │   ├── descriptor.html
│       │   ├── formatter.html
│       │   ├── gaussian.html
│       │   ├── numerical.html
│       │   ├── peak.html
│       │   ├── plot.html
│       │   ├── regexp.html
│       │   ├── tofframe.html
│       │   └── tofseries.html
│       └── tests
│           ├── test_base.html
│           ├── test_common.html
│           ├── test_formatter.html
│           ├── test_imports.html
│           ├── test_regexp.html
│           └── test_tofseries.html
├── modules.html
├── objects.inv
├── py-modindex.html
├── scibeam.core.html
├── scibeam.html
├── scibeam.tests.html
├── search.html
├── searchindex.js
```

(continues on next page)

(continued from previous page)

```
├── setup.html
├── _sources
│   ├── about.rst.txt
│   ├── conf.rst.txt
│   ├── index.rst.txt
│   ├── install.rst.txt
│   ├── modules.rst.txt
│   ├── scibeam.core.rst.txt
│   ├── scibeam.rst.txt
│   ├── scibeam.tests.rst.txt
│   ├── setup.rst.txt
│   ├── structure.rst.txt
│   └── tutorial.rst.txt
├── _static
│   ├── ajax-loader.gif
│   ├── alabaster.css
│   ├── basic.css
│   ├── comment-bright.png
│   ├── comment-close.png
│   ├── comment.png
│   ├── css
│   │   ├── badge_only.css
│   │   └── theme.css
│   ├── custom.css
│   ├── doctools.js
│   ├── documentation_options.js
│   ├── down.png
│   ├── down-pressed.png
│   ├── file.png
│   ├── fonts
│   │   ├── fontawesome-webfont.eot
│   │   ├── fontawesome-webfont.svg
│   │   ├── fontawesome-webfont.ttf
│   │   ├── fontawesome-webfont.woff
│   │   ├── fontawesome-webfont.woff2
│   │   ├── Lato
│   │   │   ├── lato-bold.eot
│   │   │   ├── lato-bolditalic.eot
│   │   │   ├── lato-bolditalic.ttf
│   │   │   ├── lato-bolditalic.woff
│   │   │   ├── lato-bolditalic.woff2
│   │   │   ├── lato-bold.ttf
│   │   │   ├── lato-bold.woff
│   │   │   ├── lato-bold.woff2
│   │   │   ├── lato-italic.eot
│   │   │   ├── lato-italic.ttf
│   │   │   ├── lato-italic.woff
│   │   │   ├── lato-italic.woff2
│   │   │   ├── lato-regular.eot
│   │   │   ├── lato-regular.ttf
│   │   │   ├── lato-regular.woff
│   │   │   └── lato-regular.woff2
│   │   └── RobotoSlab
│   │       ├── roboto-slab-v7-bold.eot
│   │       ├── roboto-slab-v7-bold.ttf
│   │       ├── roboto-slab-v7-bold.woff
│   │       ├── roboto-slab-v7-bold.woff2
```

(continues on next page)

(continued from previous page)

```
│   │   │   ├── roboto-slab-v7-regular.eot
│   │   │   ├── roboto-slab-v7-regular.ttf
│   │   │   ├── roboto-slab-v7-regular.woff
│   │   │   └── roboto-slab-v7-regular.woff2
│   │   ├── jquery-3.2.1.js
│   │   ├── jquery.js
│   │   ├── js
│   │   │   ├── modernizr.min.js
│   │   │   └── theme.js
│   │   ├── minus.png
│   │   ├── plus.png
│   │   ├── pygments.css
│   │   ├── searchtools.js
│   │   ├── underscore-1.3.1.js
│   │   ├── underscore.js
│   │   ├── up.png
│   │   ├── up-pressed.png
│   │   └── websupport.js
│   ├── structure.html
│   └── tutorial.html

11 directories, 101 files
```

1.2 reStructuredText Sturcture

The reStructuredText files are the source that these htmls are build on top of. Most of the text related .rst fils are wrote in the corresponding mark up formart, other module related .rst are build using autodoc, which automatically looks in to the doc strings in python source files.

In this project, the document style in the pyhon source files are following numpy style, which is rendered by Spnhinx externsion napoleon.

The structure of .rst folder structure:

```
├── about.rst
├── conf.py
├── index.rst
├── install.rst
├── modules.rst
├── scibeam.core.rst
├── scibeam.rst
├── scibeam.tests.rst
├── _static
├── structure.rst
└── _templates

2 directories, 11 files
```

1.3 Package structure

The package structure of scibeam is

```
scibeam
├── core
│   ├── base.py
│   ├── common.py
│   ├── descriptor.py
│   ├── dictfunc.py
│   ├── formatter.py
│   ├── gaussian.py
│   ├── __init__.py
│   ├── numerical.py
│   ├── peak.py
│   ├── plot.py
│   ├── regexp.py
│   ├── tofframe.py
│   └── tofseries.py
├── data
│   ├── examples
│   └── test
├── __init__.py
├── tests
│   ├── __init__.py
│   ├── __pycache__
│   ├── test_base.py
│   ├── test_common.py
│   ├── test_formatter.py
│   ├── test_imports.py
│   ├── test_regexp.py
│   └── test_tofseries.py
└── util
    ├── folderstruct.py
    ├── __init__.py
    ├── io.py
    ├── multiframe.py
    └── pipeline.py
```

10 directories, 34 files

Where:

- core: main part of the pacaage
- tests: unittests
- util: extral add ons for the package
- data: test data and example data files

CHAPTER

TWO

INSTALL

Install scibeam is easy, one can choose either install using pypi or from source code using python setuptools.

2.1 Requirements

The scibeam package requires:

- Python(>= 3.4)
- Numpy
- Scipy
- Pandas
- matplotlib

Note: scibeam doesn't support python 2.7, make sure you have the right python version (>=3.4).

2.2 Using PyPI

Scibeam is avaliable on PyPI, one can install under python3 environment using:

```
pip install scibeam
```

Scibeam can then be imported as:

```
import scibeam
```

2.3 Using Setuptools

To install using python setuptools, simply clone the source code:

```
git clone git@github.com:SuperYuLu/SciBeam.git
```

Then change into the SciBeam folder:

```
cd SciBeam
```

Under SciBeam folder, install by typing:

```
python setup.py install
```

scibeam package name should be then available in the python environment, to import:

```
import scibeam
```

or:

```
from scibeam import *
```

CHAPTER

THREE

HOW TO USE

How to use

CHAPTER

FOUR

SCIBEAM.CORE PACKAGE

4.1 Submodules

4.2 scibeam.core.base module

Base functions for mixin classes and module width constants

scibeam.core.base.**_mixin_class**
list(str) – Specify allowed mixin class for method chain. The two basic data structures are TOFSeries and TOFFrame, current.

Note: TODO: Move Defaults to a seperate config.py file for easy configuration

class scibeam.core.base.**Defaults**
Bases: object

Module level default values

Settings for global default values

Note: TODO: realize these using a seperate config.py file

```
data_file_extenstion = '.lvm'
data_file_num_column = 2
file_regex = '.*_(\\d+\\.?\\d+).*.lvm$'
subfolder_regex = '.*(\\d+\\.?\\d+).*'
```

4.3 scibeam.core.common module

Common functions used across classes and modules

scibeam.core.common.**winPathHandler**(*args*)
A windows path string handler

Convert windows path string variables to python/linux compatible Path

Parameters **args** (*string*) – A single or list of strings of path

Returns Reformated string of list of strings

Return type string

scibeam.core.common.**loadFile**(*filename*, *cols=2*, *usecols=None*, *skiprows=0*, *kind='txt'*, *sep='\t'*)
File loader

Loading txt / lvm data files

Parameters

- **filename** (*string*) – Filename string (including the full path to the file)
- **cols** (*int*) – Total number of columns to be loaded, default 2
- **usecols** (*int*) – Column to be used, if None then load all. Default None
- **skiprows** (*int*) – Number of rows to skip when loading data, this is specifically designed for the case that there is header in the file
- **kind** (*string*) – File format, default 'txt'. Currently only works for txt-like files
- **sep** (*string*) – Seperator of the data column, default ' '

Returns data loaded as numpy ndarray, default 2D array

Return type numpy.ndarray

Raises

- FileNotFoundError – File not found with given filename string
- ValueError – Data loading didn't finish

4.4 scibeam.core.descriptor module

class scibeam.core.descriptor.**DescriptorMixin**(*descriptor_cls*)
Bases: object

Meta class for method chain mixin

This is a meta class to realize method chain in other classes Read-only descriptor for class cross reference

4.5 scibeam.core.formatter module

scibeam.core.formatter.**format_dict**(*rawdict*, *alphabetical=True*, *digits=2*)
dictionary to string format

Format dictionarys to strings as a list of key value pairs in each row , meant for printing, annotation on plot, etc.

Parameters

- **rawdict** (*dictionary*) – raw input dictionary
- **alphabetialy** (*bool*) – if true (default) arrange dict key alphabetical
- **digits** (*int*) – number of digits to keep if the value the key is numerical

Returns Formated string in seperate rows

Return type string

4.6 scibeam.core.gaussian module

class scibeam.core.gaussian.**Gaussian**
Bases: object

Class for numerical gaussian funciton application

A collections of methods for gaussian analysis on the data, such as single gaussian function, single gaussian 1d fitting, double gaussian, double gaussian fitting, etc.

static doubleGaus(*x*, *a1*, *x1*, *sigma1*, *a2*, *x2*, *sigma2*, *y0=0*)
Gaussian function of two independent variables

Double gaussian function with offset :: y = a1 * exp((x - x1)^2 / (2 * sigma1^2) + a2 * exp((x - x2)^2 / (2 * sigma2^2))

Parameters

- **x** (*float*) – Input variable for the double gaussian function
- **a1** (*float*) – Amplitude of the first gaussian variable peak
- **x1** (*float*) – Peak center for the first variable gaussian peak
- **sigma1** (*float*) – Sigma vlaues for the two gaussian peaks
- **a2** (*float*) – Amplitude of the second gaussian variable peak
- **a2** – Amplitude of the first gaussian variable peak
- **x2** (*float*) – Peak center for the first variable gaussian peak
- **sigma2** (*float*) – Sigma vlaues for the two gaussian peaks
- **y0** (*float*) – Y offset, optional, default y0 = 0

Returns

Return type Numerical value of the double gaussian function

static doubleGausFit(*x*, *y*, *guessPara*, *offset=False*)
Two independent variable gaussian fitting

Fit the data with a double gaussian function base on given x, y data and initial guess parameters.

Unlike the 1D gaussian fitting function, one hase to provide initial guess parameters to make sure optimal parameters could be found.

The fitting method is based on least square method, fitted parameters and their covariance matrix is returned.

Parameters

- **x** (*1D array*) – Input data x value
- **y** (*1D array*) – Input data y value
- **guessPara** (*array-like*) – Initial guess parameter list[a1, x1, sigma1, a2, x2, sigma2, y0]

Returns

- *array1* – Fitted parameter array [a1, x1, simga1, a2, x2, simga1]
- *array2* – Cnveriance matrix of fitted parameters

static gaus(*x*, *A*, *x0*, *sigma*, *offset=0*)
gaussian function with or without offset

General form of a 1D gaussian function, with variable as first parameter and other associate parameters followed. Can be used for fitting or line plotting after fitting is done.

The function generally follow the form :: y = A * exp(-(x - x0)^2 / (2 * sigma^2)) + offset (optional)

Handles the case with and without offset seperatelly, since for fitting without offset at all one has to force the function to be of not offset.

Parameters

- **x** (*float*) – variable x in gaussian function
- **A** (*float*) – Peak value
- **x0** (*float*) – Center coordinates
- **sigma** (*float*) – Standard deviation
- **offset** (*float*) – overall offset, default 0

static gausFit(*x*, *y*, *offset=False*, *plot=False*)
Perform gaussian fit on given data

Fit data with 1D gausian function :: y = a * exp((x - x0)^2 / (2 * sigma)) + y0(optional)

The function generates initial guesses automatically based on given data, the algorithm is based on scipy curve_fit function

Parameters

- **x** (*array-like*) – X values of the input data
- **y** (*array-like*) – Y values of the input data
- **offset** (*bool*) – Wether fit gaussian with offset or not Default False
- **plot** (*bool*) – Wether plot the fitting result or not Default False

Returns

- *array1* – Array of optmized best fit data [a, x0, sigma, y0]
- *array2* – A 4 x 4 covariant matrix of the corresponding optmized data

Raises RuntimeError – When optimized parameters not found within max depth of iteration

4.7 scibeam.core.numerical module

scibeam.core.numerical.**bandPassFilter**(*data*, *tStep=None*, *lowFreq=0*, *highFreq=10000.0*)
band pass filter based on fourier transform

Filter the noise in time series data with given frequency range.

The data has to be in numpy array. If only 1D array is provided, one also needs to provide time step size. If 2D array is provided, the 0th column will be used to calculate time step size, while the 1st column will be treated as the signal value.

Parameters

- **data** (*numpy array*) – The input time series data. 1d array is treated as the signal value, which requires input parameter tStep to be not None.

- **tStep** (*float*) – Time step size in seconds of the time series data. If None (default), 0st columns in data will be treated as time and time step size will be extracted from there
- **lowFreq** (*float*) – Lower bound of the bandpass filter, default 0 Hz
- **highFreq** (*float*) – Upper bound of the bandpass filter, default 1e4 Hz

---

Note: The data has to be uniformly sampled, e.g. same time gap between each data point, all parameters here are supposed to be SI unit.

---

scibeam.core.numerical.**integrate**(*x=None*, *y=None*, *kind='numerical'*, *func=None*, *args=()*)
numerical / functional integration

Perform integration on either numerical data or on a function.

The numerical intergration is based on given parameter x and y, based on numpy function trap; while the functional integration is based on given function and numpy function quad.

Parameters

- **x** (*1D array*) – THe x axis values for numerical data, default None
- **y** (*1D array*) – The y axis values for numerical data, default None
- **kind** (*string*) – Specify the integration method, options are: 'numerical', 'function' default 'numerical'
- **func** (*function*) – The function to be integrated, default None
- **args** – arguments for function quad

4.8 scibeam.core.peak module

class scibeam.core.peak.**SeriesPeak**(**args*, ***kwargs*)
Bases: pandas.core.series.Series

Peak analysis on 1D labeled / unlabeled data

Build on top of pandas.Series, this adds more methods on peak analysis for pandas series data. The any class instance of SeriesPeak can still access all pandas sereis methods.

By default, the indexes is treated as the time axis, while the values of series is the data value.

Additionally, SeriesPeak is also designed as a mixin class which can be used as a method chain in other pandas dataframe / sereis based data formats.

self
pandas series – pandas series data

__init__(**args*, ***kwargs*)
assign value to initialize SeriesPeak

The initlization of this class can be done exactly as one initlize pandas series, for more information please pandas series documentation.

area(*gauss_fit=False*, *offset=False*)

autocrop(*n_sigmas=4*, *offset=False*)

fwhm(*gauss_fit=False*, *offset=False*)
Full-Width-Half-Maximium

Find the Full-Width-Half-Maximium (FWHM) of the peak, from gaussian fitting or direction calculation.

Parameters

- **gauss_fit** (*bool*) – If true, fwhm is get from gaussian fitting If false (default), fwhm is from direction calculation
- **offset** (*bool*) – If True, the gaussian fitting will consider also fit the data offset. If False (default), the fitting procedure will assume that the data has 0 offset.

Returns peak full-width-half-maximium value

Return type float

gausFit(*plot=False*, *offset=False*)
Fit series with gausssian function

This gasussian fit function assumes the time or x-axis values is given by series index, while the measurement data or y-axis values is given by the values of sereis.

Optionally the one can choose to plot the fitted gaussian curve together with the raw data to visuallize the fitting property.

Parameters

- **plot** (*bool*) – If True a plot will be generated with raw data and fitted gaussian curve. Others no plot will be generated. Default False.
- **offset** (*bool*) – If True, the gaussian fitting will consider also fit the data offset. If False (default), the fitting procedure will assume that the data has 0 offset.

Returns

- **popt** (*1D array*) – optimized parameters of gaussian fitting. [A, x0, sigma, y0(optional)] Where A: peak height of gaussian function x0: peak center x coordinates sigma: standard deviation y0: offset. Only exist if parameter offset is set to be 'True'
- **pcov** (*2D array*) – Covariance matrix of fitted parameters corresponding to popt

height(*gauss_fit=False*, *offset=False*)
calculate peak height

Calculated the peak height, either by gaussian fitting (if gauss_fit true), or simply return the maximium as the peak height (default)

Parameters

- **gauss_fit** (*bool*) – If true, the peak height is get by performing a gaussian fit If false, simply the maximium value in the given data
- **offset** (*bool*) – If True, the gaussian fitting will consider also fit the data offset. If False (default), the fitting procedure will assume that the data has 0 offset.

Returns Peak height

Return type float

idx(*gauss_fit=False*, *offset=False*)
find x-axis value corresponding to peak

This funciton is to locate the corresponding x corrdinate or 'index' of the peak. Depend on the value of parameter 'gauss_fit', the x coordinate of peak can either come from the max value or gaussian fitting.

The index of series is treated as the x coordinate of the data.

Parameters

- **gauss_fit** (*bool*) – If true, the x coordinate corresponding to peak is get by performing gaussian fitting on the data, as in member method gausFit. If false, the maxmium value of data will be treated as the 'peak', and its corresponding x coordinate will be returend.
- **offset** (*bool*) – If True, the gaussian fitting will consider also fit the data offset. If False (default), the fitting procedure will assume that the data has 0 offset.

Returns The x coordinate that corresponding to the peak

Return type float

nidx(*gauss_fit=False*, *offset=False*)
number index of the peak

Similar to member method idx, this one returns the number index rather than the real index, which means self.index[nidx] = idx

region(*n_sigmas=4*, *plot=False*, *offset=False*)
Auto find the peak region

Locate the region where there exists a peak and return the lower and upper bound index of the region.

sigma(*n_sigmas=1*, *gauss_fit=False*, *offset=False*)
Find peak width

Find the peak width, with specified mutiples of standard deviation. The width can be obtained by literally calculate the full-width-half-max or by gaussian fitting, depend on the parameter value 'gauss_fit' to be true of false.

Parameters

- **n_sigmas** (*integer*) – Multiplies of standard deviations of the peak width is wanted
- **gauss_fit** (*bool*) – If true, the peak width is obtained from gaussian fitting If False (default), the peak width is calculated from literally full-width-half-max.
- **offset** (*bool*) – If True, the gaussian fitting will consider also fit the data offset. If False (default), the fitting procedure will assume that the data has 0 offset.

Returns The peak width in terms of multiples of standard deviatioins

Return type float

class scibeam.core.peak.**FramePeak**(**args*, ***kwargs*)
Bases: pandas.core.frame.DataFrame

Peak analysis on 1D labeled / unlabeled data

area(***kwargs*)

fwhm(***kwargs*)

height(***kwargs*)

idx(***kwargs*)

nidx(***kwargs*)

region(***kwargs*)

sigma(***kwargs*)

4.9 scibeam.core.plot module

class scibeam.core.plot.**PlotTOFFrame**(*dataframe*, *lowerBound=None*, *upperBound=None*, *index_label=None*, *column_label=None*)

Bases: object

Plot dataframe with time as index and another numerical variable as column labels

contour(*n_contours=5*, *n_sigma=2*, *xlabel='time'*, *ylabel='value'*, *title='contour plot'*, *label=None*, *ax=None*, *image=False*, ***kwargs*)

contour plots for 2D self.data

contourf(*n_contours=5*, *n_sigma=2*, *xlabel='time'*, *ylabel='value'*, *title='contour plot'*, *label=None*, *ax=None*, ***kwargs*)

contourf plots for 2D self.data

data

image(*sideplots=True*, *contour=False*, ***kwargs*)

image plot of tof data measured multiplot positions

class scibeam.core.plot.**PlotTOFSeries**(*dataseries*, *lowerBound=None*, *upperBound=None*, *index_label=None*, *column_name=None*)

Bases: object

Plot dataframe with time as index and another numerical variable as column labels

data

plot(*ax=None*, *gauss_fit=True*, *gauss_fit_offset=0*, *print_fit_params=True*, *title=None*, *xlabel=None*, *ylabel=None*, *label=None*, *params_digits=3*, ***kwargs*)

4.10 scibeam.core.regexp module

class scibeam.core.regexp.**RegMatch**(*regStr*)

Bases: object

match(*strings*, *group=1*, *asNumber=True*)

Match a single or list of regularizations to a single or list of strings Return as a dictionary

matchFolder(*folder_path*, *asNumber=True*, *group=1*)

Match files in the folder content with self.regex if two regex are in the self.regex, then the match is done in a recursive way, that first regex get matched, and the 2nd regex is applied to the match result from the first one.

regex

static single_regex_match(*regStr*, *strings*, *group=1*, *asNumber=False*)

Match python regex pattern in a given string or list of strings Based on python re package and uses group to locate the value

returns pairs of (value, string) matched pairs

4.11 scibeam.core.tofframe module

class scibeam.core.tofframe.**TOFFrame**(**args*, ***kwargs*)

Bases: pandas.core.frame.DataFrame

Time-Of-Flight (TOF) DataFrame

Subclassing pandas.DataFrame with extral methods / properties for time-series analysis

Parameters

- **data** (*numpy ndarray (structured or homogeneous), dict, or DataFrame*) – Dict can contain Series, arrays, constants, or list-like objectsSingle time-of-flight data analysis Value of measurement, e.g. voltage, current, arbiturary unit signel, shape(len(labels), len(times))
- **index** (*numpy ndarray, iterables*) – Time axis for time-of-flight
- **columns** (*str, int, or float*) – label of different tof measurement, e.g. pressure, temperature, etc

static find_time_idx(*time*, **args*)
Generator of time index for a given time value args: can be 1,2,3, or [1,2] or [1,2,3]

classmethod from_file(*filePath*, *lowerBound=None*, *upperBound=None*, *removeOffset=True*, *offset_margin_how='outer'*, *offset_margin_size=20*, *skiprows=0*, *sep='\t'*)
Generate TOFFrame object from a single given file

classmethod from_matchResult(*path*, *matchDict*, *lowerBound=None*, *upperBound=None*, *removeOffset=True*, *offset_margin_how='outer'*, *offset_margin_size=20*, *skiprows=0*, *sep='\t'*)
Creat TOFFrame from a RegMatch resutl dictionary

classmethod from_path(*path*, *regStr*, *lowerBound=None*, *upperBound=None*, *removeOffset=True*, *offset_margin_how='outer'*, *offset_margin_size=20*, *skiprows=0*, *sep='\t'*)
Buid TOFFrome instance from given file folder Current only works for ' ' seperated txt and lvm file

inch_to_mm(***kwargs*)

microsec_to_sec(***kwargs*)

mm_to_inch(***kwargs*)

peak
alias of *scibeam.core.peak.FramePeak*

plot2d
alias of *scibeam.core.plot.PlotTOFFrame*

reduce(***kwargs*)

static remove_data_offset(*data*, *lowerBoundIdx=None*, *upperBoundIdx=None*, *how='outer'*, *margin_size=10*)
remove offset in 1D array data

sec_to_microsec(***kwargs*)

selectTimeRange(***kwargs*)

selectTimeSlice(***kwargs*)

sum(***kwargs*)

scibeam.core.tofframe.**read_folder**(*path*, *regStr*, *lowerBound=None*, *upperBound=None*, *removeOffset=True*, *offset_margin_how='outer'*, *offset_margin_size=20*, *skiprows=0*, *sep='\t'*)
Create TOFFrame class instance by reading in group of files in a folder matched by regex

Parameters

- **path** (*str*) – folder path, linux style or windows style as "raw string", e.g. r"C:UserDocumentFolderName"
- **lowerBound** (*int or float*) – time axis lower boundrary limit for data
- **upperBound** (*int or float*) – time axis upper boundrary limit for data
- **removeOffset** (*bool*) – if True (default) remove data offset (set floor to 0 in no-signal region)
- **offset_margin_how** (*{"outer", "outer left", "out right", "inner", "inner left", "inner right"}, default "outer"*) – Specify the way to handle offset margin, offset floor value is calculated by averaging the value in a given range relative to data lower and upper boundrary, with avaliable options:
  - "outer" (default): from both left and right side out of the [lowerBound, upperBound] region
  - "outer left": like "outer" but from only left side
  - "outer right": like "outer" but from only right side
  - "inner": from both left and right side inside of the [lowerBound, upperBound] region
  - "inner left": like "inner" but from only left side
  - "inner right": like "inner" but from only left side
- **offset_margin_size** (*int*) – Number of values to use for averaging when calculating offset
- **skiprows** (*int*) – number of rows to skip when read in data
- **sep** (*str, defult " "*) – seperator for columns in the data file
- **Returns** –
- **--------** –
- **of class TOFFrame** (*Instance*) –

scibeam.core.tofframe.**read_regexp_match**(*path*, *matchDict*, *lowerBound=None*, *upperBound=None*, *removeOffset=True*, *offset_margin_how='outer'*, *offset_margin_size=20*, *skiprows=0*, *sep='\t'*)

Create instance of TOFFrame from regular expression match result dictionary using scibeam class RegMatch

Parameters

- **path** (*str*) – path of the targeted data folder
- **matchDict** (*dictionary*) – result dictionary form scibeam.regexp.RegMatch, or user specified dictionary with key as measurement label, value as file name string
- **lowerBound** (*int or float*) – time axis lower boundrary limit for data
- **upperBound** (*int or float*) – time axis upper boundrary limit for data
- **removeOffset** (*bool*) – if True (default) remove data offset (set floor to 0 in no-signal region)
- **offset_margin_how** (*{"outer", "outer left", "out right", "inner", "inner left", "inner right"}, default "outer"*) – Specify the way to handle offset margin, offset floor value is calculated by averaging the value in a given range relative to data lower and upper boundrary, with avaliable options:

- ”outer” (default): from both left and right side out of the [lowerBound, upperBound] region
- ”outer left”: like “outer” but from only left side
- ”outer right”: like “outer” but from only right side
- ”inner”: from both left and right side inside of the [lowerBound, upperBound] region
- ”inner left”: like “inner” but from only left side
- ”inner right”: like “inner” but from only left side

- **offset_margin_size** (*int*) – Number of values to use for averaging when calculating offset
- **skiprows** (*int*) – number of rows to skip when read in data
- **sep** (*str, defult " "*) – seperator for columns in the data file

Returns

Return type Instance of TOFFrame

4.12 scibeam.core.tofseries module

class scibeam.core.tofseries.**TOFSeries**(**args*, ***kwargs*)

Bases: pandas.core.series.Series

static find_time_idx(*time*, **args*)

classmethod from_file(*file_path*, *lowerBound=None*, *upperBound=None*, *removeOffset=True*, *cols=2*, *usecols=None*, *offset_margin_how='outer'*, *offset_margin_size=20*, *skiprows=0*, *sep='\t'*)

Buid TOF instance from given file Current only works for ‘ ‘ seperated txt and lvm file

gausCenter(*offset=False*)

gaus fit center

gausFit(*offset=False*)

1D gauss fit

gausStd(*offset=False*)

gaus fit std

peak

alias of *scibeam.core.peak.SeriesPeak*

plot1d

alias of *scibeam.core.plot.PlotTOFSeries*

static remove_data_offset(*data*, *lowerBoundIdx=None*, *upperBoundIdx=None*, *how='outer'*, *margin_size=10*)

remove offset in 1D array data

sec_to_microsec(*offset_sec=0*, *inplace=False*)

convert seconds in index to microseconds

selectTimeRange(***kwargs*)

selectTimeSlice(***kwargs*)

scibeam.core.tofseries.**read_file**(*file_path*, *lowerBound=None*, *upperBound=None*, *removeOffset=True*, *cols=2*, *usecols=None*, *offset_margin_how='outer'*, *offset_margin_size=20*, *skiprows=0*, *sep='\t'*)

Read from sngle file and create an instance of TOFSeries

Parameters

- **file_path** (*str*) – path to file
- **lowerBound** (*int or float*) – time axis lower boundrary limit for data
- **upperBound** (*int or float*) – time axis upper boundrary limit for data
- **removeOffset** (*bool*) – if True (default) remove data offset (set floor to 0 in no-signal region)
- **cols** (*int*) – Total number columns in the data file
- **usecols** (*int*) – The index of column that will be used out of total number of columns cols
- **offset_margin_how** (*{"outer", "outer left", "out right", "inner", "inner left", "inner right"}, default "outer"*) – Specify the way to handle offset margin, offset floor value is calculated by averaging the value in a given range relative to data lower and upper boundrary, with avaliable options:
  - "outer" (default): from both left and right side out of the [lowerBound, upperBound] region
  - "outer left": like "outer" but from only left side
  - "outer right": like "outer" but from only right side
  - "inner": from both left and right side inside of the [lowerBound, upperBound] region
  - "inner left": like "inner" but from only left side
  - "inner right": like "inner" but from only left side
- **offset_margin_size** (*int*) – Number of values to use for averaging when calculating offset
- **skiprows** (*int*) – number of rows to skip when read in data
- **sep** (*str, defult " "*) – seperator for columns in the data file
- **Returns** –
- **--------** –
- **of class TOFSeries** (*Instance*) –

4.13 Module contents

CHAPTER

FIVE

CONTRIBUTE

As a open source project, scibeam is under active development towards version 1.0, thus we need contributors from the conmunity.

5.1 Steps

- Read the documents
- Join the slack channel(https://scibeam.slack.com)
- Report issure / bug on Github
- Look for open issues
- Create new pull request

5.2 Help needed

- Write unittest for better coverage
- Finish document “how to use” part
- Add slack channel badge to Readme
- Add more file read in format support
- Add plotly extension for better visualization
- Many more

CHAPTER

SIX

SCIBEAM.TESTS PACKAGE

6.1 Submodules

6.2 scibeam.tests.test_base module

class scibeam.tests.test_base.**TestFunctions**(*methodName='runTest'*)
Bases: unittest.case.TestCase

setUp()
Hook method for setting up the test fixture before exercising it.

test_is_mixin()

6.3 scibeam.tests.test_common module

class scibeam.tests.test_common.**TestFunctions**(*methodName='runTest'*)
Bases: unittest.case.TestCase

Test core.common.py

test_loadFile()

test_winPathHandler()

6.4 scibeam.tests.test_formatter module

class scibeam.tests.test_formatter.**TestFunctions**(*methodName='runTest'*)
Bases: unittest.case.TestCase

testDict = {'a': 1, 'b': 2, 'c': 3.1415926, 'd': 4, 'e': 5}

test_formart_dict()

6.5 scibeam.tests.test_imports module

class scibeam.tests.test_imports.**TestImports**(*methodName='runTest'*)
Bases: unittest.case.TestCase

test_import_Gaussian()

test_import_PlotTOFSeries()

test_import_RegMatch()

test_import_TOFFrame()

test_import_TOFSeries()

test_import_read_file()

test_import_read_folder()

6.6 scibeam.tests.test_regexp module

class scibeam.tests.test_regexp.**TestRegmatch**(*methodName='runTest'*)
Bases: unittest.case.TestCase

setUp()
Hook method for setting up the test fixture before exercising it.

test_match()

test_matchFolder()

test_single_regex_match()

6.7 scibeam.tests.test_tofseries module

class scibeam.tests.test_tofseries.**TestFunctions**(*methodName='runTest'*)
Bases: unittest.case.TestCase

test_read_defaults()

test_read_with_bounds()

test_read_without_offset()

class scibeam.tests.test_tofseries.**TestTOFSeries**(*methodName='runTest'*)
Bases: unittest.case.TestCase

setUp()
Hook method for setting up the test fixture before exercising it.

test_find_time_idx()

test_init()

test_remove_data_offset()

test_selectTimeSlice()

6.8 Module contents

CHAPTER

SEVEN

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

S