pyccutof Documentation

Release 2018

Adam Birdsall

Nov 28, 2018

Contents

1 Installation 3

2 Example usage 5

3 Main module 7

4 Indices and tables 15

Python Module Index 17

pyccutof is a Python package for working with JEOL AccuTOF mass spectrometer data files.

The project repository is hosted on Github.

CHAPTER 1

Installation

```
pip install pyccutof
```

CHAPTER 2

Example usage

Given the FFC and FFT files, along with a previously determined calibration curve, create a pandas DataFrame containing all mass spectra in the FFT file, with m/z value as the index and mass spectrum timestamp as column name:

```
import numpy as np
import pyccutof as pt

FFC_FN = "MsData.FFC"
FFT_FN = "MsData.FFT"
cal_curve = np.poly1d([
                8.97324469e-19,
                -5.01087595e-14,
                3.47362513e-07,
                4.90251082e-03,
                1.73793870e+01
                ])

numpts, index_records = pt.readffc(FFC_FN)
fft = pt.read_fft_lazy(FFT_FN, index_records)
mz = pt.apply_mz_cal(fft, cal_curve)
df_specs = pt.create_df_specs(fft, mz)
```

The DataFrame `df_specs` can then be further analyzed or saved to disk (e.g., `df_specs.to_csv()`, `df_specs.to_pickle()`,...).

CHAPTER 3

Main module

Import pyccutof using `import pyccutof`. This imports the entirety of `pyccutof.main`, which contains the following (descriptions auto-generated from docstrings):

exception pyccutof.main.**ParserError**
Raise exception when parser didn't work correctly.

pyccutof.main.**apply_mz_cal**(*fft*, *cal_poly*)
Given fft array and calibration polynomial, create array of m/z values.

This means every bin of counts in the fft is given a corresponding m/z.

pyccutof.main.**calc_baseline_const_before**(*eic*, *peak_time*, *start_earlier=15*, *window=10*)
Create constant baseline function from mean value in region before peak.

This is the baseline-finding algorithm used by eic_areas_from_raw(), because it does not rely on the (often inaccurate) peak edges detected by the scipy peak_prominences algorithm.

Parameters

- **eic** (*pd.Series*) – Extracted ion chromatogram, intensities indexed by timestamp.
- **peak_time** (*float*) – Time, in units of int_region index, where peak is located.
- **start_earlier** (*float*) – Time before peak_time, in units of eic index, at which baseline region begins.
- **window** (*float*) – Length of time, in units of eic index, over which baseline region extends. Must be smaller than start_earlier so that baseline region doesn't include peak, other ValueError is raised.

Returns **baseline** – Constant-output function giving baseline intensity as a function of time.

Return type np.poly1d

See also:

eic_areas_from_raw(), *calc_baseline_fit_surround()*, *calc_baseline_two_points()*

pyccutof.main.**calc_baseline_fit_surround**(*eic*, *leftend*, *rightstart*, *leftwindow=60*, *rightwindow=60*, *fitorder=2*)

Calculate baseline via linear fit to regions surrounding peak.

This is one option for going between detect_peak_heights() and integrate_area(), but note the algorithm is sensitive to the accuracy of the two base values. The base values detected by the scipy peak_prominences algorithm (used in detect_peak_heights()) is often inaccurate and will lead to a faulty baseline. For this reason, calc_baseline_const_before() is preferred.

Parameters

- **eic** (*pd.Series*) – Extracted ion chromatogram, intensities indexed by timestamp.
- **rightstart** (*leftend,*) – Timestamps defining the end of the baseline region preceding the peak and the start of the baseline region following the peak, in the units of the eic index.
- **rightwindow** (*leftwindow,*) – Width of the baseline windows to fit to, before and after the peak, in units of the eic index.
- **fitorder** (*int*) – Order of polynomial fit to use. Depending on size of window and stability of baseline, order 1 or 2 is likely most appropriate to avoid overfitting.

Returns **baseline** – Polynomial function giving baseline intensity as a function of time.

Return type np.poly1d

See also:

calc_baseline_const_before(), *calc_baseline_two_points()*

pyccutof.main.**calc_baseline_two_points**(*eic*, *leftbase*, *rightbase*)

Define a linear baseline from two timestamps of a series of intensities.

This is one option for going between detect_peak_heights() and integrate_area(), but note the left and right base detected by the scipy peak_prominences algorithm (used in detect_peak_heights()) is often inaccurate and will lead to a faulty baseline. calc_baseline_const_before() is preferred.

Parameters

- **eic** (*pd.Series*) – Extracted ion chromatogram, intensities indexed by timestamp.
- **rightbase** (*leftbase,*) – Timestamps of the start and end of the linear baseline.

Returns **baseline** – Linear function giving baseline intensity as a function of time.

Return type function

See also:

calc_baseline_const_before(), *calc_baseline_fit_surround()*

pyccutof.main.**create_cal_poly**(*cal_times*, *cal_mz*, *deg=4*)

Create a calibration polynomial from fitting times to known m/z.

pyccutof.main.**create_df_specs**(*fft*, *mz*)

Create DataFrame of all spectra in fft, with shared m/z index.

Assumes fft array is in the format passed from read_fft_lazy(), in which each row contains the timestamp in the first entry, the total ion count in the last two entries, and the mass spectrum in between.

Array of m/z values can be constructed using apply_mz_cal().

Parameters

- **fft** (*2D numpy array*) – Array of FastFlight data, as output from read_fft_lazy.

- **mz** (*1D numpy array*) – Array of m/z values corresponding to intensities in each fft spectrum.

Returns **df** – DataFrame in which each row is a single m/z (m/z value is index), and each column is a different chromatogram time (time is column name).

Return type pd.DataFrame

pyccutof.main.**detect_peak_heights**(*eic*, *num_peaks=1*, *use_flat_baseline=True*, *bl_kwargs={}*, *make_plot=True*, *ax=None*)

Find a given number of largest peaks in a chromatogram.

Parameters

- **eic** (*pd.Series*) – Chromatogram of intensities indexed by timestamps, as output by extract_eic().
- **num_peaks** (*float*) – Number of peaks to extract, starting with largest. Returned peaks are kept in chromatogram order (i.e., largest peak not necessarily returned first). If None, all peaks are returned.
- **use_flat_baseline** (*Boolean*) – Whether to calculate height from a flat baseline extracted from before the peak, using calc_baseline_const_before. Otherwise, use peak prominences from scipy.signal.peak_prominences(). Default True.
- **bl_kwargs** (*dict*) – Dict of arguments to pass to calc_baseline_const_before. Allowed keys are 'start_earlier' and 'window'. Default empty. Only used if use_flat_baseline is True.
- **make_plot** (*Boolean*) – Whether to make a diagnostic plot showing the peaks, heights, and bases.
- **ax** (*matplotlib.Axis*) – Axis to plot on. If None, make new axis.

Returns **df_heights** – DataFrame with each entry containing the peak height, along with its timestamp and the left and right bases to which the prominences are measured.

Return type pd.DataFrame

pyccutof.main.**eic_areas_from_raw**(*data_fldr*, *unit_mzs*, *calpoly*, *chrom_center=None*, *chrom_window=1*, *make_plot=False*, *ffc_fn='MsData.FFC'*, *fft_fn='MsData.FFT'*, *bl_kwargs={}*)

Calculate extracted ion chromatogram peak areas from raw data.

Convenience function. Assumes working with unit m/z resolution. Uses calc_baseline_const_before to define baseline and integrate_area to integrate.

Parameters

- **data_fldr** (*str*) – Path to folder containing FFC and FFT data.
- **unit_mzs** (*list of floats*) – List of m/z from which to extract unit EICs (i.e., from -0.5 to +0.5 m/z of provided values).
- **calpoly** (*np.poly1d*) – Calibration curve to convert from TOF bin to m/z.
- **chrom_center** (*float or None, optional*) – Time, in minutes, within chromatogram for which peak window is defined. If None, entire chromatogram is searched. Default None. Note if not None, only a portion of the FFT file is ever read, which can significantly reduce memory usage for a large FFT file.
- **chrom_window** (*float, optional*) – Time, in minutes, for width of peak window within chromatogram. Only used if chrom_center is not None.
- **make_plot** (*bool, optional*) – Whether to make a plot showing each EIC and the detected peak. Default False.

- **ffc_fn** (*str, optional*) – Name of FFC file in *data_fldr*. Default 'MsData.FFC', which is the default output file name from the mass spec software.
- **fft_fn** (*str, optional*) – Name of FFT file in *data_fldr*. Default 'MsData.FFT', which is the default output file name from the mass spec software.
- **bl_kwargs** (*dict, optional*) – Dict of keyword arguments that are passed to the baseline function, calc_baseline_const_before(). If empty dict, will use default values.

Returns **eic_areas** – Dict of peak heights and optional figure for each EIC. Each peak height entry has format {'p###': area}, where *###* is m/z value (not fixed number of digits) and area is peak area (float). If *make_plot* is True, the figure is the value with key "fig".

Return type dict

See also:

calc_baseline_const_before(), *integrate_area()*, *eic_peaks_from_raw()*

pyccutof.main.**eic_peaks_from_raw**(*data_fldr*, *unit_mzs*, *calpoly*, *chrom_center=None*, *chrom_window=1*, *make_plot=False*, *ffc_fn='MsData.FFC'*, *fft_fn='MsData.FFT'*, *bl_kwargs={}*)

Calculate extracted ion chromatogram peak heights from raw data.

Convenience function. Assumes working with unit m/z resolution.

Parameters

- **data_fldr** (*str*) – Path to folder containing FFC and FFT data.
- **unit_mzs** (*list of floats*) – List of m/z from which to extract unit EICs (i.e., from -0.5 to +0.5 m/z of provided values).
- **calpoly** (*np.poly1d*) – Calibration curve to convert from TOF bin to m/z.
- **chrom_center** (*float or None, optional*) – Time, in minutes, within chromatogram for which peak window is defined. If None, entire chromatogram is searched. Default None. Note if not None, only a portion of the FFT file is ever read, which can significantly reduce memory usage for a large FFT file.
- **chrom_window** (*float, optional*) – Time, in minutes, for width of peak window within chromatogram. Only used if chrom_center is not None.
- **make_plot** (*bool, optional*) – Whether to make a plot showing each EIC and the detected peak. Default False.
- **ffc_fn** (*str, optional*) – Name of FFC file in *data_fldr*. Default 'MsData.FFC', which is the default output file name from the mass spec software.
- **fft_fn** (*str, optional*) – Name of FFT file in *data_fldr*. Default 'MsData.FFT', which is the default output file name from the mass spec software.
- **bl_kwargs** (*dict, optional*) – Dict of keyword arguments that are passed to the baseline function, calc_baseline_const_before(). If empty dict, will use default values.

Returns **eic_peaks** – Dict of peak heights and optional figure for each EIC. Each peak height entry has format {'p###': height}, where *###* is m/z value (not fixed number of digits) and height is peak height (float). If *make_plot* is True, the figure is the value with key "fig".

Return type dict

pyccutof.main.**extract_counts_from_spectrum**(*word_list*, *clean_after_end=True*)

Use boolean mask to extract counts from spectrum.

Return array includes first bin (timestamp) and last two bins (total ion count) of spectrum, even though they don't correspond to "real" ion counts. If the word list extends past a frame flagged as the last frame in a spectrum (as seen for first entry in a sample FFC), strip off all further words if *clean_after_end*.

Requires brittle assumptions about structure of frames in spectrum:

```
ffffff # start of frame
ffxxxx # frame header: length + position in spectrum
xxxxxx # spectrum protocol + timestamp (first frame only)
ff8000 # extended address tag
xxxxxx # extended address value
ff0000 # data tag offset of 0 from extended address
xxxxxx # first data value
...... # continuation of contiguous data (no ffxxxx words skipping bins)
xxxxxx # last data value
xxxxxx # total ion sum, low word (last frame only)
xxxxxx # total ion sum, high word (last frame only)
```

NB: The "total ion sum" will not equal the sum of the raw data values in the spectrum. According to the FastFlight manual (Sect. 3.15 "Data Compression"), the total ion count is calculated using the net peak area above background, where peaks and background are determined by the digital signal processor, regardless of whether data compression is on or off.

pyccutof.main.**extract_data_record**(*word*)
bits 0-14 are value and bit 15 is sign

pyccutof.main.**extract_eic**(*spec_df*, *mz_min*, *mz_max*)
Calculate extracted ion chromatogram over given m/z range.

Parameters

- **spec_df** (*pd.DataFrame*) – DataFrame of all spectra, as output from create_df_specs().
- **mz_max** (*mz_min,*) – Values of m/z between which the EIC is limited to.

Returns **eic** – Extracted ion chromatogram, intensities indexed by chromatogram timestamp.

Return type pd.Series

pyccutof.main.**extract_tic**(*fft*)
Create dataframe of total ion chromatogram from fft.

pyccutof.main.**filter_index_recs_time**(*index_recs*, *time_center*, *time_window*)
Filter FFC index records to those in a particular time window (in min).

Parameters

- **index_recs** (*numpy.recarray*) – recarray of index records located in FFT file, in format produced by *readffc()*.
- **time_center** (*float*) – Center of time region, in minutes, to restrict *index_recs* values to.
- **time_window** (*float*) – Width of time region, in minutes, to restrict *index_recs* values to.

Returns **index_recs_filtered** – Slice of input *index_recs*, limited to those entries for which the spectrum time is within the defined window.

Return type numpy.recarray

See also:

readffc()

Notes

timemultiplier for the FFC time values is assume to be 2.5 ms, which is the case when the digital signal processor is not installed.

pyccutof.main.**ibits**(*i*, *pos*, *l*)
Extract l bits of i, starting from pos.

Convenience function, like fortran *ibits*.

pyccutof.main.**import_fft**(*fn*, *index_recs*)
Read FFT file and yield spectra of raw words.

Requires filename and list of offsets (from FFC file).

Assume the spectrum lasts for the length of the corresponding 'reserved' value in the FFC. Sample data shows that the 'reserved' value is usually okay, except it's too big for the first entry. Don't clean up issues like this here; instead, use *extract_counts_from_spectrum()*.

pyccutof.main.**int_edges_from_baseline**(*int_region*, *bl_function*, *peak_time*)
Define edges of integration region from crossings with baseline.

For left edge, use last crossing before the peak. For right edge, since decay is slower, wait for two consecutive values below the baseline. If that is never observed in int_region, use the first value before the baseline. As a final fallback, use the end of int_region.

Parameters

- **int_region** (*pd.Series*) – Series of intensities, with index values of chromatogram time.
- **bl_function** (*function*) – Function that returns baseline intensity as a function of chromatogram time.
- **peak_time** (*float*) – Time, in units of int_region index, where peak is located.

Returns **(left_edge, right_edge)** – Chromatogram times, in units of int_region index, defining left and right edge of peak integration region.

Return type tuple of floats

See also:

integrate_area()

pyccutof.main.**integrate_area**(*int_region*, *bl_function=None*, *left_edge=None*, *right_edge=None*, *make_plot=True*, *ax=None*)
Integrate Series values, with optional baseline and/or index cutoff.

Parameters

- **int_region** (*pd.Series*) – Series of intensities, with index values of chromatogram time.
- **bl_function** (*function or None*) – Function that returns baseline intensity as a function of chromatogram time. If provided, baseline is subtracted before integration is performed.
- **right_edge** (*left_edge,*) – Chromatogram times over which to integrate, if provided, defined as (left_edge, right_edge]. If None, integrate over entire int_region.
- **make_plot** (*Boolean*) – Make plot for visual check. Only helpful if baseline and/or edges defined.
- **ax** (*matplotlib.Axis or None*) – Axis to plot on. If None, create new axis.

Returns **integral** – Integral of region.

Return type float

See also:

`int_edges_from_raw()`

Notes

Integral is calculated using Scipy implementation of composite trapezoidal rule.

`pyccutof.main.`**`read_jeoldx`**(*fn*)
Read jeol-dx *.jmc* file as pandas DataFrame.

`pyccutof.main.`**`readffc`**(*fn*)
Read an FFC file from the filename.

The FFC record starts with a 4-byte header, and then continues with index records providing information about each spectrum in the corresponding FFT file.

Parameters **`fn`** (*`str`*) – Filename of FFC file to read.

Returns

- **numpts** (*int*) – Header of FFC, "NumberOfPoints", converted to int.
- **index_records** (*numpy.recarray*) – recarray of spectra in corresponding FFT file, with field names 'protocol', 'time', 'reserved', 'counts', 'offset', and 'timemultiplier'.

`pyccutof.main.`**`readmassrange`**(*acqfn*)
Fetch StartMass and EndMass from acquisition data database file.

CHAPTER 4

Indices and tables

- genindex
- modindex
- search

Python Module Index

p

pyccutof.main, 7

Index

A

apply_mz_cal() (in module pyccutof.main), 7

C

calc_baseline_const_before() (in module pyccutof.main), 7
calc_baseline_fit_surround() (in module pyccutof.main), 7
calc_baseline_two_points() (in module pyccutof.main), 8
create_cal_poly() (in module pyccutof.main), 8
create_df_specs() (in module pyccutof.main), 8

D

detect_peak_heights() (in module pyccutof.main), 9

E

eic_areas_from_raw() (in module pyccutof.main), 9
eic_peaks_from_raw() (in module pyccutof.main), 10
extract_counts_from_spectrum() (in module pyccutof.main), 10
extract_data_record() (in module pyccutof.main), 11
extract_eic() (in module pyccutof.main), 11
extract_tic() (in module pyccutof.main), 11

F

filter_index_recs_time() (in module pyccutof.main), 11

I

ibits() (in module pyccutof.main), 12
import_fft() (in module pyccutof.main), 12
int_edges_from_baseline() (in module pyccutof.main), 12
integrate_area() (in module pyccutof.main), 12

P

ParserError, 7
pyccutof.main (module), 7

R

read_jeoldx() (in module pyccutof.main), 13
readffc() (in module pyccutof.main), 13
readmassrange() (in module pyccutof.main), 13