

Spectra analysis system - documentation

Brian Burford, Ilia Nouretdinov, Dmitry Devetyarov,
Zhiyuan Luo, Alexey Chervonenkis, Volodya Vovk, Alex Gammerman

Contents

1	Introduction	1
1.1	Preprocessing overview and motivations	2
2	Input file formats	3
3	Batch Preprocessing	4
4	Major functions	4
4.1	PROC_SPEC	4
4.2	SMOOTH	5
4.3	BASELINE_ESTIMATION / BASELINE_CORRECTION / SPLINEFUN	5
4.4	NORMALISE	7
4.5	FILE_FINDER	7
4.6	PEAK_FINDER	8
4.7	SET_UP_PEAKS	8
4.8	PEAKALIGN2	9
4.9	MAKE_PEAK_INT_LIST / GEN_PEAK_LIST	9
4.10	GOSASGO	10
4.11	CAL_SPEC	11
4.12	GOCALGO	11
4.13	PLOT_PEAK / PLOT_ALL	11
	4.13.1 plot_peak	12
	4.13.2 plot_all	12
5	Triplet / matched controls specific functions	12
5.1	GET_TRIPS	12
5.2	NEW_PVALUES	13
5.3	GEN_TRIP_LISTS_NOVAL	13

1 Introduction

This document instructs the user in the use of the matlab system for preprocessing and analysis of mass spectra. This system is set up to be compatible

with the MRC pilot study data but will also work with any other combatable file formats as outlined below.

1.1 Preprocessing overview and motivations

We begin with resampling (binning) the data - this has the advantage of removing some noise and reducing the number of points that represent the spectra. The reduced number of points result in an increase in speed for later processing. The amount of resampling (i.e. the percentage reduction in the number of points) that is done must be carefully monitored: a small percentage may not provide a significant advantage; whereas a large percentage would run the risk of losing signal. We lose signal because we have basically assumed that there is more noise than there actually is and we 'over smooth'.

This is followed by a smoothing procedure that attempts to remove noise that results from random gaussian fluctuations in the signal. As with restamping there are parameters to consider, in this case we use a moving window average to smooth the spectra, this requires careful selection of the window size parameter. If we select a window size that is small then there would be very little change between the original noisy spectra and the new, apparently smoothed spectra. Similarly if a window size is selected that is too large we run the risk of smoothing over useful signal in the data. Empirical investigation has indicated that large window sizes can remove small signals that may be useful later on; it was found that smooth spectra that still contain small signals can be obtained by using a small window size and repeating the smoothing process a number of times.

As a result of materials used in the analysis of samples, and some other factors, the spectra's 'baseline' does not sit on the horizontal axis. We therefore need to subtract this positive baseline from the spectra in order to extract the true signal. However this requires the difficult task of estimation of the baseline. Our method is based on performing rough peak identification and using the minimum points between these peaks as a basis for the baseline. We begin by connecting these minima using a sophisticated interpolation procedure. We then make adjustments to this baseline based on some criteria, which include the restriction that the baseline should not be above the spectra for a large range. Once these adjustments are completed the interpolation procedure is performed once more to produce the final estimation for the base line. Subtraction of this from the smoothed spectra is then trivial.

Several factors including small fluctuations in the amount of sample analysed can result in two spectra from the same set of samples having different total signal. This means if we sum up the height of all the points in the two spectra the total would not necessarily be the same, or indeed very close. This can of course lead to false results where discrimination may be due to

the amount of ions in a sample rather than class discrimination. We combat this through normalisation: we first calculate the sum of all points, we then divide each point in the spectra by this sum, this makes the total ion count 1 - we can of course multiply this up to make the numbers easier to deal with, the result will be that each spectra has an identical total signal.

Each spectra has now been through the filtration steps that are required to extract the true signal from a noisy spectra with a varying baseline. The next step is the identification of useful signal in each spectra. We are interested in the maxima of each spectra that have a high signal and signal-to-noise ratio. We call these points peaks, each peak represents a mass where there was a high abundance of material in that sample at that mass. Briefly peaks are identified by first locating all local maxima and filtering out those with low signal and low signal-to-noise ratio. Our final list of peaks will be those local maxima that are superior to other local maxima with respect to these constraints.

We now have a list of peaks for each spectra in the sample set. Our next step is to evaluate this list and create a general list of peaks. We can create such a list by clustering peaks that are close to one another. We of course do not allow two peaks that exist in the sample spectra to be present in the same group of peaks, we therefore favour the peak with the greatest height. Clearly we may not find every peak in each spectra, only the most common peaks will be found in all spectra, this is however not an issue, we do however ignore peaks that are below a certain threshold of commonality, usually defined as a proportion of the total number of spectra. This list of common peaks tells us which peaks are present in most or all spectra. Due to signal and signal-to-noise ratio requirements for local maxima in the peak identification stage the case may exist such that a peak with a signal or SNR only shingly below the threshold is ignored. However if this is a common peak then we need not be concerned as other spectra that contain this peak will compensate.

The final stage in preprocessing involves using this list of peaks to extract the intensity of the signal for each common peak from each spectra. This is performed by, for each peak in the peak list, extracting the height for each repetition of that peak in each spectra. Hence any peaks that we missed in the peak alignment stage are still located here.

2 Input file formats

Each file must be a table with 2 columns, the first column is the m/z value or clock tick and the second column must be the corresponding intensity for each m/z value or clock tick. For the pilot study filenames must start with the string '00Z' followed by a 2 character code uniquely identifying the sample. As the pilot study used higher and lower mass ranges these are

indicated by '*a_1.ascii' and '*a_2.ascii' respectively. Also standard spectra used for calibration must contain the string '*-Cal_1.ascii' for the lower mass range and '*-Cal_2.ascii' for the higher mass range.

3 Batch Preprocessing

Details of all the functions are outlined below, if the user wishes to run a batch preprocessing script then see function GOSASGO, in section 4.10.

4 Major functions

This section lists the major preprocessing function in the system. Some functions are grouped together for convenience.

4.1 PROC_SPEC

The system is mainly intended for the possessing of many mass spectra and the identification of common peaks. However it is also possible to evaluate a single spectra. Perhaps the user would find this beneficial when tuning parameters or just to get a feel for the data and the system. In order to process a single spectra X, use the function:

[X P P] = PROC_SPEC(X,SAMPLE_INDEX, MR)

Output: Xp returns the processed spectra, p returns information on the identified peaks. Input arguments: X is the input spectra; sample_index is a optional field used to identify spectra that should be considered as on e.g. when different mass ranges are used. For single spectra this can be set to 0. MR indicates which mass range to use. This is a specific feature for the pilot study data and and other such data with multiple mass ranges - by default this will select the first can (lower mass range in this instance)—advanced users can add and remove cases as required for their specific data sets.

Most of the parameters used in this system are set in the proc_spec function. Explanations of each and reasonable default values are given here. Note that default value were selected for the pilot study data set and may be unsuitable for new data and will need to be changed:

- resample_percentage (default = 50, range: (0, 100] real value)—set the percentage reduction of the number of data points through binning.
- smooth_reps (default = 3, range:[1,∞) integer value)
- smooth_window_size (default = 5, range:[1,number of points in the spectra] integer value)
- baseline_param (default = 0.01, range: (0, ∞) real value)

- `num_baseline_corrections` (default = 1, range: $[1, \infty)$ integer value)
- `normalisation_constant` (default = 10^7 , range: $(0, \infty)$ real value)
- `snr_window_size` (default = 450, range: $[1, \text{number of points in the spectra}]$ integer value)
- `minimum_intensity_threshold` (default = 250, range: $[0, \text{highest point in the spectra}]$ real value)

4.2 SMOOTH

This function smooths the spectra by using a moving window of pre determined size. If this function is being used directly it is advised that it is called several times—a single pass does not usually produce a very smooth looking spectra especially for highly noisy data.

Input:

- `X`: spectra to be smoothed.
- `windowSize`: width of the moving window.

Output:

- `X_new`: The smoothed spectra.

4.3 BASELINE_ESTIMATION / BASELINE_CORRECTION / SPLINEFUN

The baseline estimation calculates and subtracts a baseline from the spectra. The baseline is estimated by creating a ‘rough’ spectra and then running corrections improve the estimation. The function `BASELINE_ESTIMATION` call 2 other functions, the details of which are given below.

BASELINE_ESTIMATION

The main function called in order to perform baseline estimation and subtraction on a spectra.

Input:

- `X`: Input spectra for baseline removal
- `percent_diff`: Parameter of the algorithm, which describes the maximum mass separation between dominant local maxima when fitting the baseline.
- `iters`: Number of times the correction procedure is performed.

Output:

- B: The spectra with the baseline removed.
- Base: The estimated baseline.

BASELINE_CORRECTION

This procedure is used to correct and parts of the baseline that do not conform to some requirements. In this implementation the requirement is that the baseline must not be greater than the spectra. This functions has been isolated to allow the user to define their own requirements if they prefer.

Input:

- base: The current baseline
- point_present: this flags the points in the spectra that make up the baseline.
- X: The original spectra.
- pd: The baseline separation parameter as above.

It may not be clear what the difference between base, point_present and X may be, so to clarify: X is the original spectra; point_present indicate which points in X are sitting on the base line; base is the baseline—some points are in contact with the original spectra others were found through interpolation of these points in the original spectra.

Output:

- point_present: Updated set of points through which the baseline much pass.

splinefun

The function calls the matlab function PCHIP used for cubic spline interpolation of the points on the baseline.

Input:

- B: Set of points which make up the spectra - usually the original spectra but will later be changed to define the baseline only.
- point_present: Indicates which set of points will be on the baseline.

Output:

- base: Cubic hermit spline interpolation of the points defined in the spectra.

- `point_present`: Updated set of points on the baseline—in this version this vector will not change in this function; therefore this output is not a necessity.

4.4 NORMALISE

This function simply normalises the spectra by making the total ion count `C`.

Input:

- `X`: Spectra to be normalised.
- `C`: Constant for scaling spectra.

Output:

- `X`: The normalised spectra.

4.5 FILE_FINDER

This function searches the specified directory for files containing a specified string. This is useful for finding all sample spectra or standard sample spectra.

Input:

- `path`: The location of the file to be searched.
- `bit`: (A string) the function will look for files that contain this string. Note that the matching is exact, i therefore if you wish to search for all files with the string `S` you must use `*`'s to denote wild cards: `*S*`; for example `*.txt` to find all of the `.txt` files.

Note: If `bit` is left blank the default is `*`, meaning all files and folders will be found. In this case (that of `bit = *`) the first 2 directories: `./.` and `../.` will be ignored. Also note that this function is recursive, all subdirectories will also be searched.

Output:

- `f_list`: Struct containing information on the files found, the struct contains the following elements:
 - `fpath`: The full path name of the file.
 - `fdirpath`: The location of the directory that contains the file.
 - `fname`: The file name.

4.6 PEAK_FINDER

This function is responsible for finding peaks in a spectrum. Given a spectrum and some set of parameters it will return the set of peaks.

Input:

- X: Preprocessed spectra.
- ws: Window size for calculating the signal-to-noise ratio.
- thold: The minimum intensity threshold, all peaks below this threshold will be excluded.
- sample_index: it may be the case that one blood sample results in more than one spectra, for example having a lower mass range and a higher mass range, this parameter provides the facility for several spectra to be related to each other through a common index.

Output:

- p: A matrix containing information on the peak from the spectra. The matrix has 5 columns detailed as follows:
 1. sample index.
 2. index for the peak in this spectra—this information is not very important but is kept from an older version of the system in case of backwards compatibility requirements.
 3. m/z-value.
 4. signal-to-noise ratio (intensity of the peak divided by the average intensity in the window defined by ws).
 5. intensity.

4.7 SET_UP_PEAKE

This function collects saved peak lists into one table, no extra calculations are performed.

Input:

- path: Location of the file containing peak information for each spectra.
- start: This counts the spectra providing a unique id for each. By default this is set to 0.

Output:

- peaks: This matrix is the concatenation of all peaks from all spectra.

4.8 PEAKALIGN2

Given a set of peaks this function will find peak groups based on a set of input parameters. The format of the input list of peaks is the same as the output from SET_UP_PEAKEs.

Input:

- `pks`: The matrix of peaks generated by `set_up_peaks`
- `first_SNR`: The signal-to-noise ratio.
- `second_SNR`: This can be used to set the second level signal-to-noise however this should be left blank in this version as this variable is not important with respect to the set of peaks generated.
- `massSep`: The mass separation parameter for clustering peaks into groups.

Output:

- `peak_groups`: Matrix containing information on the peak groups found, the matrix can have the following columns:
 1. `PeakID`: A unique id for the peak.
 2. `Highest peak mass location`: The m/z value of the tallest peak.
 3. `Mean Mass location`: The mean of all m/z values of all the peaks in the group.
 4. `Min mass`: The peak in the group that is furthest left in the spectra (minimum m/z value).
 5. `Max mass`: The peak in the group that is furthest right in the spectra (maximum m/z value).
 6. `Number of pks`: Number of peaks above the signal-to-noise threshold in the group.
 7. `Max Intensity`: The maximum intensity in the group.

4.9 MAKE_PEAK_INT_LIST / GEN_PEAK_LIST

After desired preprocessing is complete a matrix containing samples on rows and peaks on columns can be created using the `MAKE_PEAK_INT_LIST` function.

MAKE_PEAK_INT_LIST

Input:

- pg: The peak group matrix generated by calling peakalign2.
- massSep: The mass separation parameter.
- proc_path: Location of the processed spectra.

Output:

- peak_list: A matrix containing rows of samples. Each column in the matrix is a peak in pg.

GEN_PEAK_LIST

Input:

- proc_spec_LMR: Spectra for the lower mass range of the sample
- proc_spec_HMR: Spectra for the higher mass range of the sample.
- peak_set: Used to define a subset of peaks to use (Not used in this form in the pilot study data).
- lab: The class label.
- peakGroups: Matrix of peak groups.
- pd: The mass separation parameter.

Output:

- peak_list: list containing the intensity of each peak.

4.10 GOSASGO

This function runs the preprocessing functions above and saves the data in a specified location. If the user wishes to do mass automated preprocessing of many spectra then this is the function to use.

Input:

- MR: A flag for targeting a specific mass range.
- raw_path: Location of the spectra to process.
- proc_path: Location to save the processed spectra.

Output:

- There are no specific output arguments however the function does save information such as: peak groups and peak list.

4.11 CAL_SPEC

The function calibrates a raw sample relative to a standard sample through the use of calibrant markers. This function will calibrate on spectra at a time.

Input:

- raw: the sample spectra to be calibrated.
- cal: the standard sample spectra.
- cal_peaks: the calibrant markers
- massSep: mass separation parameter for the standard sample spectra
- ws: window size for calculating signal to noise ratio.

The level of signal-to-noise can also be changed in the CAL_SPEC.M file, it has been given a default of 3 and does not require in depth tuning—remember we are only running a quick quality check as this stage.

Output:

- Xc: The calibrated raw sample spectra.

4.12 GOCALGO

The preprocessing above can also be applied to standard calibrant samples and therefore the user may wish to with until all other preprocessing is complete before doing calibration. This function allow for automated calibration of all spectra.

Input:

- raw_path: Location of the spectra to be calibrated
- cal_path: Location of the standard spectra used for calibration.
- cal_peaks: Set of calibrant markers used to calibrate the spectra.

Output:

- There are no specific output arguments however the function does save information on calibrated spectra.

4.13 PLOT_PEAK / PLOT_ALL

These functions are for visualisation of the peaks. The user can either plot one peak at a time using PLOT_PEAK or many peaks using PLOT_ALL

4.13.1 plot_peak

Given the m/z value of a peak this function will overlay all spectra that have that peak. The spectra will also be cut-off at +/- some predefined value (d, below).

Input:

- mz: m/z location of the peak to plot.
- d: Width of the window to plot.
- proc_path: Location of the spectra from which to plot the peak.
- col: Any matlab symbol for colour or style of the line to plot.

Output:

- The functions output is a figure containing the plot.

4.13.2 plot_all

For mass plotting of many peaks, this functions should be provided with a list of peaks, and the location of mass spectra in which the peaks will be looked for. Note; if a spectra exists at that location that does not have the peak then it is simply ignored, not problems will occur, allowing for more that one mass range to be stored in one file.

Input:

- pg: Peak groups matrix of all peaks to plot.
- plot_path: Location to save the figures.
- proc_path: Location of the spectra to plot peaks from.

Output:

- The function saves all figures, 1 per peak group.

5 Triplet / matched controls specific functions

5.1 GET_TRIPS

Input:

- T: Time in months before the last moment to consider.

Output:

- trips: A struct containing the following fields:
 - trips.case: A struct with the following fields:
 - * trips.case.peaks: List of peak intensities for the case sample.
 - * trips.case.CA: CA125 value for the sample
 - * trips.case.id: Sample ID for the case sample—used to uniquely identify the triplet.
 - trips.cont1: A struct with the following fields:
 - * trips.cont1.peaks: List of peak intensities for the first matched control sample.
 - * trips.cont1.CA: CA125 value for the sample
 - trips.cont2: A struct with the following fields:
 - * trips.cont2.peaks: List of peak intensities for the second matched control sample.
 - * trips.cont2.CA: CA125 value for the sample

5.2 NEW_PVALUES

Input:

- T: Time in months to the time of diagnosis.
- Four parameters can be set in the function:
 1. set_W1: Set of weights for W1.
 2. set_W2: Set of weights for W2.
 3. num_peaks: Number of peak to use.
 4. N: number of repetitions for the Monte-Carlo method.

Output:

- res: struct containing information on the triplet analysis such as error and p-values.

5.3 GEN_TRIP_LISTS_NOVAL

Input:

- trips: The set of triplets.
- set_W1: Set of weights for W1, passed from new_pvalues.
- set_W2: Set of weights for W2, passed from new_pvalues.

- num_of_peaks: Number of peak to use, passed from new_pvalues.

Output:

- max_mat: Information used for the Monte-Carlo method