R-software multiMS-toolbox

(User Guide)

Pavel Cejnar[1], Štěpánka Kučková[2]

[1]Department of Computing and Control Engineering, Institute of Chemical Technology, Technická 3, 166 28 Prague 6, Czech Republic, E-mail: cejnarp@vscht.cz

[2]Department of Biochemistry and Microbiology, Institute of Chemical Technology, Technická 3, 166 28 Prague 6, Czech Republic, E-mail: kuckovas@vscht.cz

Introduction

multiMS-toolbox is a software toolbox to efficiently search for differences in mass-spectrometry samples from long-term experiments. It is supposed to have several runs for each sample. Then the software allows you to:

- simplify matching of appropriate peaks in the spectra of different runs and different samples,
- do a principal component analysis (PCA [Pearson 1901]),
- export and analyze each PCA component and draw graphs of the most important PCA factors and the most important peaks in the spectra,
- draw other statistic graphs, like the most important changes of absolute and relative intensities and area factors of the peaks and peak clusters.

Mass spectrometry data are preprocessed first – the intensities in each spectrum can be normalized against some template spectrum. Then the corresponding peaks in each spectrum are identified. Then, based on the configuration parameter, the peaks in clusters can be replaced by one group peak. Then the PCA analysis is run and the principal components are exported (including their factors and „influence“). Several graphs showing main factors, their influence, relative and absolute changes are shown.

Installation on MS Windows

The software is distributed in ZIP archive containing the „multiMS-toolbox“ application folder. The toolbox main file is “multiMS-toolbox.R”. To use it you must first install R-software system (http://www.r-project.org/) on your computer.

When you run the R-software, then install the toolbox by running a command

> source("multiMS-toolbox.R")

Then you can run the demo by a command

> demo()

or

> demoNormalized()

Or use any function implemented (see the help for each function in the source file multiMS-toolbox.R).

Current version of *multiMS-toolbox* was successfully tested on Windows 7 64-bit with R 2.15.1 (both 32-bit and 64-bit environment).

Implemented functions

init(name,normalizedTemplateSpectrum="",intensityBased=0, deisotoping=1)

Fulfills the initial param structure.

Parameters:

name – character string to print in graphs and to use for file names

normalizedTemplateSpectrum – filename of the spectrum, which will be use as template. If none, then spectra are not normalized. The normalization coefficient will be computed as median of ratios of spectrum intensities in several selected m/z points.

intensityBased – 1 means intensities of peaks will be used, 0 means areas of the peaks will be used. Assuming Gaussian distribution of peak intensities, areas are computed as (full width at half maximum) * (peak intensity).

deisotoping – 0 means all peaks are used, 1 means clusters are replaced by only one peak having the m/z value as first peak in the cluster. Peak intensity is sum of the processed intensities of all the peaks within the cluster. Peak fwhm parameter is computed to keep the sum of all the peaks in the cluster.

Returns the initial parameters structure.

readFiles(params, csvfile="filesAll.csv", sn_cut=1.0)

Reads the spectrum peaks from all the files listed in the csvfile, higher than given signal to noise parameter.

Parameters:

params – initial parameters structure

csvfile – Excel’s csv file with the semicolon (;) separator. The file should have column headers on the first line. The file should have at least one column – ***filesName*** containing the names of files to process.

sn – signal to noise ratio threshold

Data files, whose names are listed in the csvfile, should have the structure:

First line should contain the names of the data columns, at least mz, ai, int, sn, fwhm, deisotoping_grp, then each line contains data for one peak.

Data file columns:

mz – m/z of the peak

ai – absolute intensity of the peak

int – peak intensity after preprocessing (from the level sn=0.0)

sn – signal to noise ratio of the peak

fwhm – full width at half maximum of the peak

deisotoping_group – either “None”, if not a part of any peak cluster, or the number of the peak cluster.

Returns the structure with read peaks ($data), filenames ($files) and zero level values for each file ($matchInit).

Remarks:

When deisotoping, peaks are read and the peak cluster is constructed from subsequent peaks having the same deisotoping_group number. If the peak with the deisotoping_group=None is read, then it is assumed that this peak doesn’t belong to the constructed peak cluster and is read as the separate (simple) cluster. If the peak with the different deisotoping_group number is read, then the previous constructed cluster is finished and the construction of the new peak cluster is started.

readNormalizedFiles (params, csvfile="filesAll.csv", sn_cut=1.0, normalizeLowMz=900.0, normalizeHighMz=2000.0)

Reads the spectrum peaks from all the files of given pattern, higher than given signal to noise parameter, and normalizes them to the given template spectrum. For the basic parameters see the help for the *readFiles* function.

Additional parameters:

normalizeLowMz – the m/z start value of the spectrum normalization interval

normalizeHighMz – the m/z end value of the spectrum normalization interval

Spectrum file columns:

The spectrum is read from first two data columns (assuming no header line). In the first column are m/z values, in the second column are spectrum values (absolute intensities). The spectrum data files needn't to be sampled in exactly the same m/z points.

Remarks:

For the normalization spectrum values are interpolated in approximately (*normalizeHighMz* – *normalizeLowMz*) points and then the median of the ratios of this values divided by the template spectrum interpolated values is selected as the normalization coefficient.

matchPeaks(data, files, matchInit, maxDistance1=0.3, maxDistance2=0.4)

Matches the peaks in the peaks read from different files. If no peak is found for given m/z value in selected file, then the appropriate (zero level) value from the matchInit array is used.

Parameters:

data – the data vector with three columns:

1. m/z value
2. peak intensity or approximation of the peak area (=fwhm*intensity), based on the intensityBased and deisotoping parameter from the initial parameters structure when read
3. filename, from which was the peak read

files – the names of all the files read, from which are the peaks matched. The parameter is required for correct indexing into the matchInit array and for correct dimension of the constructed matched data vector.

matchInit – the array of same size as the files parameter, containing the values to use when no appropriate peak from given file is found

maxDistance1 – The maximal m/z distance to suppose that the peaks are of the same m/z value

maxDistance2 –The maximal m/z distance to suppose that the *matched* peaks (matched among several files) are of the same m/z value. The value is used only if it is higher than the maxDistance1 value.

Returns the m/z values ($mzVector) of all matched peaks and their absolute intensities/approximated areas ($matchedVectors) in every sample file. The m/z value is the m/z value of the peak with highest intensity/approximated area found in the sample files.

Remarks:

The peak matching algorithm first sorts all the peaks read from the files according their intensities/approximated areas and then tries to match the peaks (the highest peaks first) to peaks read from other files. It stores the m/z value of the selected peak and goes through all the peaks in the neighborhood of this peak searching for peaks that were not matched to any peak up to now (!) and that are at most in the maxDistance1. If any peak is found, it is marked as matched to current peak (processed) and its value is written to the column belonging to file from which it was read. If there is more than one peak from one file in given m/z distance, then the highest value is stored, however both peaks are marked as matched (processed). If no peak from a file is found in the given m/z distance, then the appropriate value from matchInit is used.

plotPCAGraphs(params, files, pr , filesColourProperty, filesCharProperty)

When the PCA is made, having the matched peak intensities/approximated areas from files as data points, the function plotPCAGraphs draws XY graph of data points in PCA1 component x PCA2 component coordinates and also stores this graph to a pdf file. Then the graph of PCA component variances is drawn and stored to a pdf file.

Parameters:

params – initial parameters structure

files – the names of all the files read, assuming one data point from one file

pr – the PCA structure containing the $x array with the PCA coordinates of each data point

filesColourProperty – vector of string representations of graph colours for given files, the same string for two different files means, that its data points will be drawn with the same colour

filesCharProperty – vector of string representations of graph point shapes for given files, the same string for two different files means, that its data points will be drawn with the same shape

Doesn't return any value.

Remarks:

To draw the smart legend, it is assumed that the file names are of the pattern Colour-AgeRun.data, where the Colour is a text string specifying the agent in the sample and the Age is the number specifying the age of the sample. The Run is any string value that is omitted. However, all the data points of the same colour are drawn with the same shape in the graph and all the data point of the same age are drawn with the same colour.

computeHighestDifference(files, numDim, pr)

For each data point (file) computes its highest difference from any other data point (averaged difference in PCA component1 and PCA component2 coordinates) is found and then it is divided by the number of dimensions of each data point (i.e. the number of matched peaks) and displayed. This could lead to an assumption of the size of unimportant disturbations (should be smaller in the order of magnitude) in the input data.

Parameters:

files – the names of all the files read, assuming one data point from one file

numDim – the number of dimensions for each datapoint

pr – the PCA structure containing the $x array with the PCA coordinates of each data point

Doesn’t return any value.

plotAbsoluteValueAndRelativeValueChangeGraph(params, dataMatched)

Plots the graph showing the peaks where the intensity/approximated areas changed a lot in the absolute value. Then plots the graph showing the relative value change (in percent) for that peaks. Both graphs are also stored to a pdf file.

Parameters:

params – initial parameters structure

dataMatched – the data points with the matched peaks and their intensities/approximated areas and a vector of m/z values of the matched peaks

Doesn’t return any value.

showComponents(params, files, pr, mzVector, matchedVectors, n=4)

Parameters:

params – initial parameters structure

files – the names of all the files read, assuming one data point from one file

pr – the PCA structure with data points transformed to new coordinates

mzVector – the vector of m/z values of the matched peaks.

matchedVectors – the data points with the intensities/approximated areas of the matched peaks

n – the number of PCA components to show

Doesn’t return any value.

Remarks:

For each component this function:

- displays the index of data point having the smallest and the larges computed coordinate in given PCA component\
- exports the appropriate data for each component to a csv file
- draws graphs showing most important (with highest influence) peaks and their values of PCA factor, PCA influence and PCA value (peak intensity of approximated peak area) change
- saves the plotted graphs to a pdf file

To the csv file, for each PCA component there are exported the data for each matched peak:

Data columns of each PCA component csv file:

mz – the m/z value of the matched peak

center of PCA origin in old coords

minPCAComponentX projected sample – the intensities/approximated areas of matched peaks in a pure projected data point having the same PCA component coordinate as the datapoint with the smallest coordinate

maxPCAComponentX projected sample – the intensities/approximated areas of matched peaks in a pure projected data point having the same PCA component coordinate as the datapoint with the largest coordinate

PCAvectorX (not normalized) – PCA factor for the given peak (not normalized to 0..1 interval)

PCAvectorX influence (i.e. (max-min sample)*vector) – the influence of the matched peak (i.e. the PCA factor multiplied by the change in the intensity/approximated area between the minPCAComponentX and the maxPCAComponentX)

Existing minPCAcomponentX – the intensities/approximated areas of matched peaks in the data point having the smallest PCA component coordinate

Existing maxPCAcomponentX – the intensities/approximated areas of matched peaks in the data point having the largest PCA component coordinate

Functions for faster data processing

extractFilesPropertiesToCSVfile(pattern=".*\\.data", outputCSVfile="filesAll.csv")

Lists all the files of given pattern and if the files have the pattern

[colour:chars][concentration:numbers]-[time:numbers][run:chars].extension

then it creates the semicolon separated csvfile with the columns filesName, filesColour, filesAge, filesRun, filesConcentrations. You can rename or copy the data from any listed column to the columns filesColourProperty, filesCharProperty and to use them for selecting the colour and shape of the data points in the PCA graph.

Parameters:

pattern – files to proccess

outputCSVfile – Excel's csv file with the semicolon (;) separator.

Doesn't return any value.

prepairNewDataReadFromAllDataRead (csvfile="files.csv", dataReadAll)

If you read all the data files by the readFiles or readNormalizedFiles, you can use this function to read only a subset of given files from dataReadAll structure (returned from readFiles or readNormalizedFiles). This makes the processing faster than reading all the files from disk by readFiles or readNormalizedFiles. The data have the same parameters and s/n threshold as read by the readFiles or readNormalizedFiles function.

Parameters:

csvfile – Excel’s csv file with the semicolon (;) separator. The file should have column headers on the first line. The file should have at least one column – *filesName* containing the names of files to process.

dataReadAll – data read by the readFiles or readNormalizedFiles

Returns the structure with read peaks ($data), filenames ($files) and zero level values for each file ($matchInit).

demoNewDataReadFromAllDataRead (csvfile="files.csv", label="", dataReadAll,n=1)

If you read all the data files by the readFiles or readNormalizedFiles, you can use this function to read only a subset of given files from dataReadAll structure (returned from readFiles or readNormalizedFiles), plot their PCA graph and show *n* PCA components. This makes the processing faster than reading all the files from disk by readFiles or readNormalizedFiles again. The data have the same parameters and s/n threshold as read by the readFiles or readNormalizedFiles function. This function calls the prepairNewDataReadFromAllDataRead.

Parameters:

csvfile – Excel’s csv file with the semicolon (;) separator. The file should have column headers on the first line. The file should have at least one column – *filesName* containing the names of files to process.

label – new character string to print in graphs and to use for file names, for reinintializing the param structure

dataReadAll – data read by the readFiles or readNormalizedFiles

n – the number of PCA components to show

Doesn’t return any value.

License