GenePattern

PeakMatch Documentation

Module name:	PeakMatch
Description:	LC-MS peak clustering using Gaussian mixture model
Author:	Jake Jaffe, D. R. Mani and Vincent Fusaro (Broad Institute) manidr@broad.mit.edu
Reference:	Jaffe et al., *Mol. Cell. Proteomics.,* 5:1927 (2006)

Summary: Peak matching attempts to group similar features (or peaks) across multiple LC-MS sample runs by incorporating m/z and retention time (RT) variation. Although peak matching can be performed on virtually any type of LC-MS data, it is typically performed after running LandmarkMatching (another GenePattern module).

The peak matching process starts with the union of all charge-identified peaks from all the LC-MS runs under consideration. Each peak is defined by its m/z, RT, and charge (z). Each peak also has an observed intensity from its run and might have a corresponding sequence identity (or landmark).

1. The first step is to use the sequence identified landmark peptides to determine the m/z and RT tolerances.
2. The peaks are then sorted by m/z and partitioned into m/z strips such that each strip is less than the m/z tolerance. Each strip spans the entire RT range and may represent multiple peptides.
3. Peak matching is performed using model-based clustering using the Expectation Maximization algorithm for Gaussian mixture modeling.
4. The final result of peak matching and alignment is an intensity table whose rows represent features that are matched peaks and whose columns represent samples (or runs).

Peak matching is part of the Platform for Experimental Proteomic Pattern Recognition (PEPPeR) pipeline.

The table below summarizes the different options available and which parameters are required.

Parameter	Input
peakList.filename	Required
sampleInfo.filename	Required
LandmarkMatchOutput.filename	Optional
MZtolerance	Optional
RTtolerance	Optional
outputName	Required (default = PeakMatchOutput
numberProcesses	Optional (default = 1)

GenePattern

Parameters

Name	Description
peakList.filename	A zip archive containing a set of peak list files, each of which is generated according to the specified format below. Each file name stub must be specified in the sampleInfo.filename. See below for important naming convention information regarding the zip archive structure.
sampleInfo.filename	A comma delimited file with a header that specifies the following: experiment ID, sample, class. The peak list and retention time files must be named according to the experiment ID. See the file format below.
LandmarkMatchOutput.filename	The zip file output from the LandmarkMatch module. This is a zipped directory of all the experiments and their corresponding processed files.
MZtolerance	Optional: Specify the m/z tolerance in ppm. This is only used if a LandmarkMatch file is not specified. If a LandmarkMatch file is specified the MZtolerance is calculated based on the data.
RTtolerance	Optional: Specify the RT tolerance in minutes. This is only used if a LandmarkMatch file is not specified. If a LandmarkMatch file is specified the RTtolerance is calculated based on the data.
outputName	File prefix for the output files
numberProcesses	Specify the number of processors to use. Designed for LSF clusters only.

File naming convention for files in zip archives:

Each file in the zip archives specified above (as supplied for peakList.filename) must use the following naming convention:

<raw_data_stub>.pgr (for peak list files)

Where the <raw_data_stub> portions of the files are the first column entries in the sample information file (see below). There must be one file for each experiment listed in the sample information file in each archive.

Example: given raw data VARMIX_A_01.raw, you would have:

VARMIX_A_01.pgr (peaklist)

as members of the zip archive submitted at run time.
See the associated example data files for further treatment on this topic. The names of zip archives themselves do not follow any convention other than ending in '.zip'

File Formats:

Peak list: The peak list file is a tab delimited generic peak list format. The number of columns must remain the same (you can leave the columns blank if you don't have all the data to fill in the columns). The number of rows is unlimited. The header must be included. If the **mz_err**, **rt_err**, and **carbons** data are unknown, these columns can be zero-filled.

feature	m/z	rt	z	abund	mz_err	rt_err	carbons
1	373.1936	40.82	2	70.8841	0.00501	0.1157	34
2	373.7314	23.61	2	269.1172	0.00443	0.0668	33
4	374.9436	19.98	4	126.9451	0.00673	0.0361	59
5	375.4211	21.21	4	156.4705	0.00733	0.0368	55

Sample information file: this is a comma delimited file that specifies the details of the experiment. The header must be included. The experiment names must match the file names for peak list. For example, there should be a peak list file called "VARMIX_A_01.pgr."

experiment	sample	class
VARMIX_A_01	A_01	Mix1
VARMIX_A_02	A_02	Mix2

References:

- Jaffe J.D., Mani D.R., et al. (2006) "PEPPeR, a Platform for Experimental Proteomic Pattern Recognition," Mol Cell Proteomics, 5(10) 1927-1941.

Return Value:
There are many outputs from this module. Due to the GenePattern format they are all displayed. Only a few files are actually important.

1. *.gct – A gct file that can be used in other GenePattern modules for further analysis.
2. *.cls – class file that can be used in conjunction with the gct file for other modules within GenePattern
3. stdout or lsf_log.txt – should be inspected for any obvious errors during processing.

Platform dependencies:

Task type: Proteomics
CPU type: any
OS: Windows, Linux
Java JVM level: 1.4
Language: Perl, R