Cluster4Metanalysis: User notes - 03.Jan.2012 – Version 1.0

Cluster4Metanalysis is a MATLAB package. It is a collection of Software to perform a meta-analysis, "clustering-based", of neuroimaging data. It is constituted of 4 modules, to each of which a separate folder is assigned:

A. Talairach2MNI: converts the peak coordinates from the Talairach atlas to the MNI atlas.
B. AnatomicalSegregation: allows seaprating the peaks in different groups according to the anatomical areas to which they belong to; the next clustering step is then performed disjointed on the different grops (*NOTICE: the integration of this module with the rest of this software package has still to be completed).*
C. Clustering: it groups the peaks of activity in data sets (clusters) that are spatially close on to the other. This is the main module.
D. Label&Visualize: it assigns to each obtained cluster, an anatomical label and it creates an image of ellipsoidal blobs that represent the clusters.

The data input and output files are contained in the folder MyData. These files can be used as example data to test the software.

A. Talairach2MNI

The coordinates of the activation peaks, collected in the neuroimaging studies and published in the literature, are generally reported in one of the two stereotactic reference atlases: Talairach or MNI. To analyze collectively the data expressed in these two atlas, a normalization operation is required. This module allows converting the peaks referred to the Talairach atlas into coordinates expressed in the MNI atlas.
To launch such conversion, the user has to go to the folder: Talairach2MNI and launch the script:

launchTal2MNIConv

The input and output files for this software module can be defined editing the corresponding lines inside the launchTal2MNIConv.m :

```
% Input and output files are in this folder
IOfolder = '../myData';
% Input File
infile = 'mixedCoord.txt';
% Output Files
outfile = 'MNICoord.txt';
% The input file must contain the original coordinates in subsequent rows;
% the first column of each row is to be set to 1 if the corresponding
% coordinate is in Talairach space (and therefore needs to be converted), 0
% if it is in MNI space (the coordinate will be left untouched). The output
% file will contain the converted (and untouched) coordinates - note that
% the column with the 0/1 code is not included in the output.
```

B. Anatomical Segregation

Once the final data set has been obtained (that is once the conversion from Talairach atlas to MNI atlas has been performed, if required), the peaks inside the data set are fed to clustering. In some cases, an additional step of "anatomical segregation" could be required. That is, in some cases, it is better to distribute the peaks to macro-areas before launching the clustering, to avoid that peaks, belonging to anatomical areas not compatible among them on the basis of the user knowledge, are grouped inside the same cluster. This step is performed by this module. We explicitly remark that in the present software realease, 1.0, this software has not been integrated yet with the other software modules.
This module is constituted of 3 main functions: a function to edit the groups of peaks, a function to group the peaks, and a function to cluster the groups of peaks. The clustering function used is that provided by Matlab, before it was re-written by Isabella Cattinelli. This is one of the reasons for which this module is separated by the other 3 modules.

1) Group editor

This function allows creating distinct functional groups, on the basis of the anatomical subdivisions operated by the AAL template. The function is launched by the following command:
groups_editor4
The function does not require any parameter. The different identified regions are saved in an xml file, specified by the interface. An example of such file is *groupsExample.xml* contained inside the directory *test*. This is its basic structure:

```
1 <? xml version ="1.0" encoding =" utf -8"? >
2 <grouping >
3 <comment > Space for a comment </ comment >
4 <groups >
5 <group name =" group 1 name ">
6 <region >AAL label of the region 1</ region >
7 <region >AAL label of the region 2</ region >
8 [...]
9 <region >AAL label of the region n </ region >
10 </group >
11 <group name =" group 2 name ">
12 <region >AAL label of the region 1</ region >
13 <region >AAL label of the region 2</ region >
14 <region >AAL label of the region 3</ region >
15 [...]
16 <region >AAL label of the region n </ region >
17 </group >
18 [...]
19 </ groups >
20 </ grouping >
```

In the case in which not all the anatomical regions have been selected as belonging to one group, these are assigned in two different ways:
"**All the remaining regions in another group**": a single group is created, called autoRemainingRegions, that contains all the regions that have not been selected.
"**Keep the two hemispheres separated**": 3 groups are created: autoRemainingRegionsLeft, autoRemainingRegionsRight, for the regions of the two hemispheres, not assigned to any group; and a third group that contains the regions not assigned to the cerebellum and to the cerebellar vermis.

2) Grouping

The activation peaks (the data) are subdivided among the different regions identified by the “Group editor”. This function is launched with the command: steppedhc_grouping(file_in1, file_in2, file_out). For instance: steppedhc_grouping(‘motIm.txt’,‘groupsExample.xml’,‘groupingxml.xml’), where:

file_in1, is the input file that contains the coordinates of the peaks inside an ASCII file: each row contains the coordinates of one peak.

file_in2, is the XML input file that contains the groups defined through the “group editor”.

file_out, is the XML output file that contains the result: a group number is associated to each peak, by analyzing the anatomical area of the AAL atlas in which the peak falls. This is an example of output file:

<? xml version ="1.0" encoding =" utf -8"? >
2 <grouping >
3 <names >
4 <name groupid =" -1" > none </ name >
5 <name groupid ="1" > group 1 name </ name >
6 <name groupid ="2" > group 2 name </ name >
7 [...]
8 <name groupid ="n"> group n name </ name >
9 </names >
10 <peaks >
11 <peak >
12 <coords > datapoint 1 coordinates </ coords >
13 <groupid >id corresponding to the group
14 this datapoint has been put into </ groupid >
15 </peak >
16 [...]
17 <peak >
18 <coords > datapoint n coordinates </ coords >
19 <groupid >id corresponding to the group
20 this datapoint has been put into </ groupid >
21 </peak >
22 </peaks >
23 </ grouping >

In case the peak does not fall inside any region, the user is prompted to decide what to do:

1 The datapoint [x y z] doesn ’t belong to any group .
2
3 Groups available :
4 1: group 1 name
5 2: group 2 name
6 ...
7 n: group n name
8 Specify the code of the group to which you want to add
9 that datapoint (ENTER to skip this , # to skip all):

At the end of this procedure, all the peaks have been assigned to one region, eventually also to the region “autoRemainingRegion”.

3) Clustering.

One the region to which the peak belongs to has been determined, clustering can be launched. This can be launched through the function: steppedhc_clustering(file_in, file_out1, file_out2). For instance: steppedhc_clustering(groupingxml, clusteringDataOut, clusteringIdsOut).

file_in is the XML file produced by the grouping procedure and it contains the region to which each peak belongs to.
For the parameters, please look at the C module: “Clustering” described hereafter.
file_out1 is a text file that contains the coordinates of the center of each cluster, the standard deviation of the cluster on the 3 axes and the cluster cardinality. It contains as many lines as the number of clusters.
file_out2 is a text file that contains the number of the cluster associated to each peak and it contains as many lines as the number of peaks.

Notice: in a future version we will insert into a separate function, the code segment that extracts from the file file_in the peaks belonging to the same region, and send them to clustering.

C. Clustering

This is the main module. It clusters the activation peaks that receives as input.
To launch clustering, the user has to go to the folder Clustering, where he can launch the script launchClustering

The input and output file for this module, as well as the parameters related to the clustering procedure that the user has chosen, must be defined editing the corresponding rows inside the file launchClustering.m :

The following rows regard the definition of the **I/O:**
% Input and output files are in this folder
IOfolder = '../myData';
Input File
datafile = 'exampleCoord.txt';
% Output Files
filename1 = 'clusteringData.txt';
filename2 = 'clusteringIDs.txt';

The **input** file is an ASCII file that contains the coordinates, X Y Z, of one peak in each row.
The output file **filename1** is a text file that contains the coordinates of the center of each cluster, the standard deviation of the cluster on the 3 axes and the cluster cardinality. It contains as many lines as the number of clusters.
The output file **filename2** is a text file that contains the number of the cluster (cluster ID) associated to each peak and it contains as many lines as the number of peaks.
Therefore, the first number in the file is the cluster number associated to the first peak, the second number is the cluster number associated to the second peak and so forth.
Note: *If the data for the metanalysis are, as usual, contained inside an Excel spreadsheet, the data in filename2 can be copied in a new spreadsheet column so that each activation peak can be made in correspondence with the clustered. It is then sufficient to filter the data according the value contained in this column (clusterID) to obtain the list of all the data, along with their information, associated to the selected cluster.*

The following rows allow setting the clustering **options**. More precisely, they allow choosing the clustering algorithm that has to be performed and to set its parameters.

In the present software version, the user can choose among these three clustering algorithms:

1. Hierarchical clustering with Ward dissimilarity measure (and distances measured as squared Euclidean distances). Check of unicity of the solution can be performed by selecting a flag. The default option is to check for unicity.
2. Clussical hierarchical clustering.
3. K-means.

Each algorithm requires to set its characteristic parameters.

For algorithm **1**, the only parameter to be set is the cut threshold of the dendrograms. To set this threshold, you have to modify the variable *cutThreshold*:

cutThreshold = 7.5;

For algorithm **2,** the following raws have to e commented:

algorithm = @findOptimizedSolution;
cutThreshold = 7.5;
varargin = {data, cutThreshold};

and de-comment the raws:

% algorithm = @classicalLinkage;
% method = 'ward';
% distance = 'euclidean';
% cutThreshold = 7.5;
% varargin = {data, method, distance, cutThreshold};

The parameters *method*, *distance*, and *cutThreshold* can be mofidied acting directly to the related raws.

Lastly, for algorithm **3**, the raws hereabove (related to the algorithm findOptimizedSolution and classicalLinkage) have to be commented and de-commented those that invoke the K-means function.

% algorithm = @kmeans;
% K = 30;
% varargin = {data, K};

D. Label & Visualize

This module assigns an anatomical label to each cluster returned at the previous step according to the position of its centroid. The anatomical label is extracted from the AAL or the Broadman atlas made available in the software MRICro. Moreover, an image is created that contains ellipsoidal blobs, each representing one of the obtained clusters. The position of the blob will be associated to the cluster position and its axes dimension will be associated to the cluster standard deviation.

To use this module, you have to go to the folder Label&Visualize and launch the script:
launchMapping

The input and output files of this script should be defined editing the adequate rows of the file launchMapping.m :

```
IOfolder = '../myData';
datafile = 'clusters.txt';
% NOTE: input data format must be:
% MeanX MeanY MeanZ StDevX StDevY StDevZ Card
% (no headers, just plain INTEGER values)
% Output Files
labelfile = 'labels.txt';
imagename = 'blobs';    % do not provide a file extension here!
```

The input file is a text file containing one row for each cluster. The row contains the mean position, the standard deviation on the three axes and the cardinality of the cluster. The output file, *labelfile*, will contain, for each cluster, its anatomical label. Two image files are generated. They are named: *imagename*, respectively with extension ".img" e ".hdr". The last one is the header file. Each blob, represented in the image, is centered in its average position and has semi-axes equal to the standard deviation of the cluster on each axis. The gray level is proportional to the cardinality of the cluster.
To visualize the image of the blobs overlayed to a structural image of the brain, you may use the MRICro software. To do so, you may open the template (File → Open Template) ch2 (o ch2bet for a 3D rendering) and select from the menu Overlay → Load Functional Overlay, with the image of the blobs just created. Modifying from the meny Overlay, the Color Scheme item, you may also visualize the blobs with different chromatic scales.

In the same folder a second script is present:
launchMappingCat

This second script should be used after the user has been carried out her own analysis of the clusters and she has thus established to which functional category (e.g. Word-related, Pseudoword-related) assign each cluster. This module allows creating a new imagine of the blob, where all the clusters belonging to the same functional category are represented with the same gray level.

The input and output files of this script should be defined editing the adequate rows of the file launchMappingCat.m :

```
IOfolder = '../myData';
 datafile = 'catClust.txt';
% NOTE: input data format must be:
```

% CategoryID CategoryLabel MeanX MeanY MeanZ StdX StdY StdZ
% (no headers, just plain INTEGER values; CategoryID must be a progressive
% ID from 1 to MAX)

% Output File
imagename = 'catBlobs'; % do not provide a file extension here!

The input file is a text file that reports for each cluster, besides its position and its standard deviation, also a categorical label (e.g. word, pseudo, …), preceded by an ID associated uniquely to a category (for instance, 1 = word, 2 = pseudo, 3 = …). The image output file will report the colored blobs such that the clusters belonging to different functional categories can be distinguished.