

SIGMA USER MANUAL

Takfarinas MEDANI & Aurélien BAELD

Version 1.0

February 2018



SIGMA
The open box for
SIGnal processing and MACHine learning

Sommaire

Aucune entrée de table des matières n'a été trouvée.

1.	Outline of the Toolbox	3
1.1.	Introduction	3
1.2.	What is SIGMAbox	3
1.3.	Why SIGMAbox	4
1.4.	General description	4
1.4.1.	Diagram of the main feature	5
1.4.2.	Parameter and hyper parameter	6
2.	Download and installation of the SIGMAbox	6
3.	The Graphical User Interface of the SIGMAbox	7
3.1.	Data Loading	9
3.1.1.	Data Format	9
3.2.3.	Frequency Bands selection	14
3.2.4.	Edit Frequency Bands	14
3.3.	Feature Extraction	15
3.3.1.	Available Methods for Feature extraction	16
3.4.	Feature Selection	21
3.5.	Data Classification	28
3.6.	Tools	33
3.7.	Apply Model	36

In which journal should we publish this & the associated paper :

<https://www.software.ac.uk/which-journals-should-i-publish-my-software>

1. Outline of the Toolbox

1.1.Introduction

Significant progress was made in hardware and software for recording and analyzing bio-physiological signals (EEG, MEG, EMG, ...). This manual describes our new open-source Matlab-based toolbox, designed to help with EEG data processing. SIGMAbox (SIGNAL processing and MACHine learning toolbox) gathers several pre-configured methods and algorithms for signal processing, statistics, classification and data visualization.

1.2.What is SIGMAbox

SIGMAbox is a MATLAB Toolbox allowing user to perform EEG data (and other type of data) classification using machine learning algorithms. SIGMAbox encapsulates a collection of new and existing Matlab functions.

The parameters of the implemented methods are initialized in a specific file. Their values are validated on our databases. However, some of them could be chosen by the user if necessary. Advanced users can change the parameters and the hyperparameters and add their own contributions.

The main features of SIGMA box are :

- Loading of EEG signals and other data provided they comply with the required format
- Processing of EEG signals :
 - Filtering on the different EEG bands
- Feature extraction algorithms
- Feature ranking algorithms including the probe variable method and Orthogonal Forward Regression (OFR)
- Data classification with four different algorithms :
 - Linear Discriminant Analysis (LDA)
 - Quadratic Discriminant Analysis (QDA)
 - Support Vector Machine (SVM)
- Numerous visualizations methods (plots, charts, tables) for data exploration and results presentation

1.3. Why SIGMAbox

SIGMAbox can be used with by no expert in programming, signal processing and Machine learning, but a minimal background is necessary.. The idea is to ‘Get all in one’ with an easy way to use. The idea of SIGMAbox is designed to deal with most of the following issues of to the most available tools, which are :

- Designed for experts with advanced programming skills [2,5]
- Some of these tools are too specific for some applications and methods [3,4]
- In many cases, the user must be a tinkerer, mixing between different tools/codes to get the desired results of one experiment
- Medical doctors and psychologists are not using these tools
- None or few students are using these software during their studies and projects
- Without GUI or the GUI is not intuitive difficult to use [2,3,4,5]

1.4. General description

SIGMAbox is an open source Matlab code developed by the Brain Computer Interface Team, Brain Plasticity Laboratory, UMR CNRS 8249, ESPCI Paris, PSL Research University in order to help people dealing with EEG and electrophysiological signals.

1.4.1. Diagram of the main feature

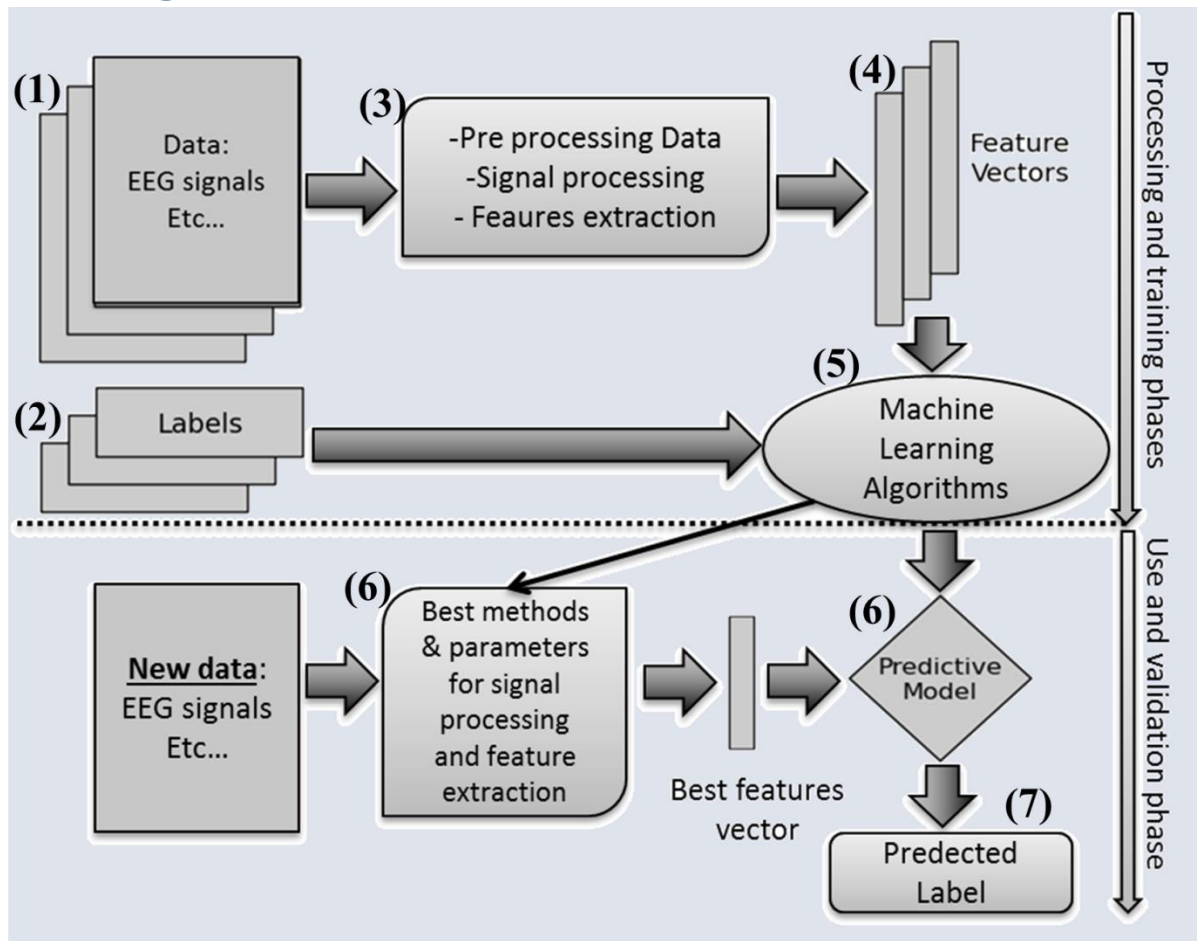


Figure 1 : General organisation of SIGMA

Data analysis in SIGMABOX is divided into three phases : a training phase, a validation phase, and a test phase. The user selects the database, and chooses the suitable method(s) for preprocessing, feature extraction and selection, and classification.

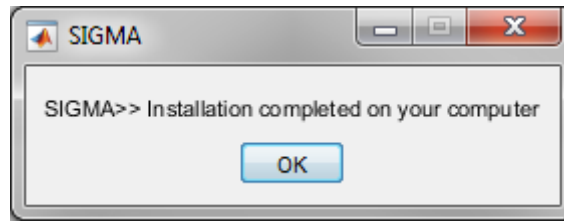
(1) Data : The input data has a specified format; for ease of use of SIGMAbox, functions for data conversion are included.

(2) Labels : Contain the “output” or the label defining the two classes

(3) Data processing and feature extraction: The core of the SIGMAbox : a set of methods for data processing and feature extraction and selection. The implemented feature extraction methods include time, spectral and statistical analyses, complexity and synchrony measures.

(4) Feature (Matrix) : Contains the numerical value of the extracted features from the data. Methods for feature selection and identification methods are included in the toolbox.

(5) Machine Learning : The present version of the toolbox allows the design of two-class classifiers for EEG data [1]. SIGMAbox uses the built-in Matlab toolboxes for data classification including linear discriminant analysis (LDA), quadratic discriminant analysis (QDA) and support vector machines (SVM).



Now, you can use the SIGMAbox on your computer.

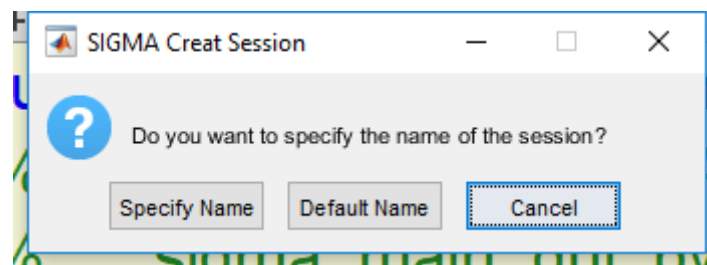
3. The Graphical User Interface of the SIGMAbox

When the installation of SIGMAbox is completed, user can either use this tool from the command window or the graphical user interface (GUI) . in this section we explain how to use SIGMAbox from its GUI

To start the user interface of SIGMAbox, just type on the command window :

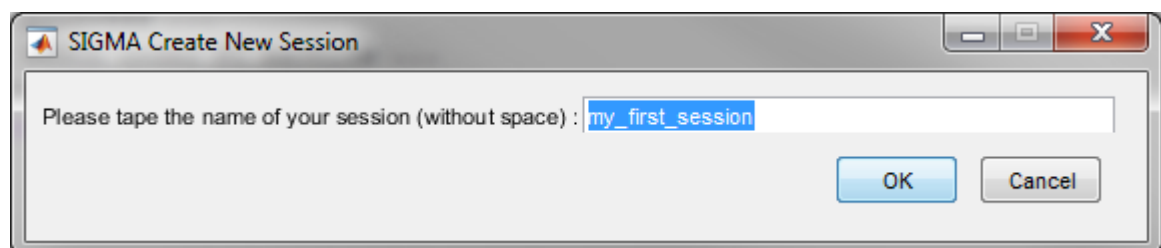
```
>> SIGMAbox
```

The following dialogue box appears :



This box asks the user to give a name for the session. The user can specify a name or choose the default name.

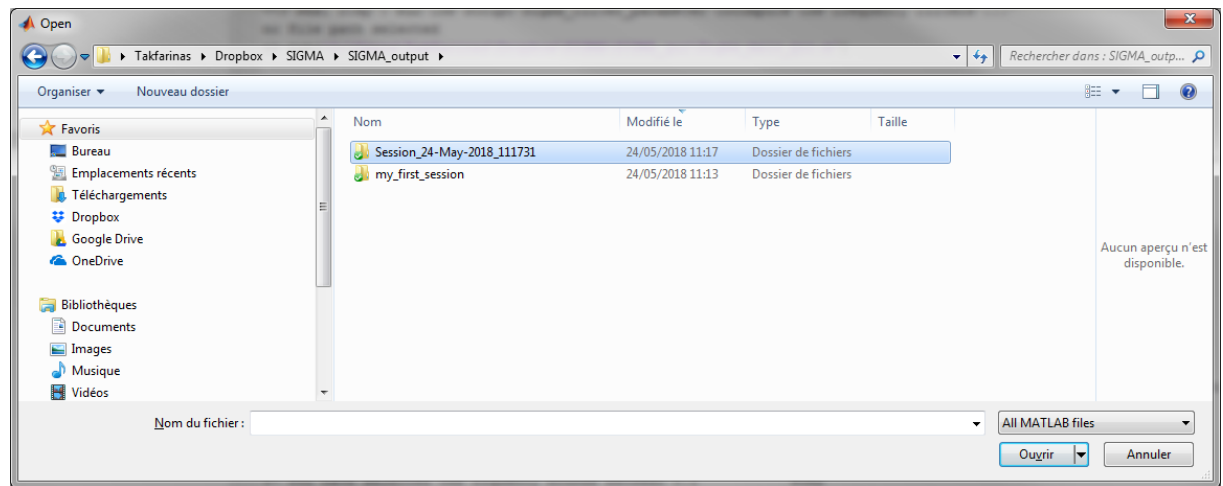
If the user specify the name, the following box will appear and the user can put the desired name.



In this case, the user choose 'my_first_session' as the name of his session.

If the default name is chosen, the session will have a name containing the date and the time of the creation. if we create a session on the May, 24, 2018, at 11 : 17 AM, the generated session name is '*Session_24-May-2018_111731*'.

In the both cases, SIGMAbox will create a subfolder with the session name in the SIGMA_output folder.



All the data and results related for a session will be saved on session's folder during all the session.

If the session is correctly created, the main graphical user interface (GUI) will appear on the screen. The main user interface of the SIGMAbox is presented in the following figure.

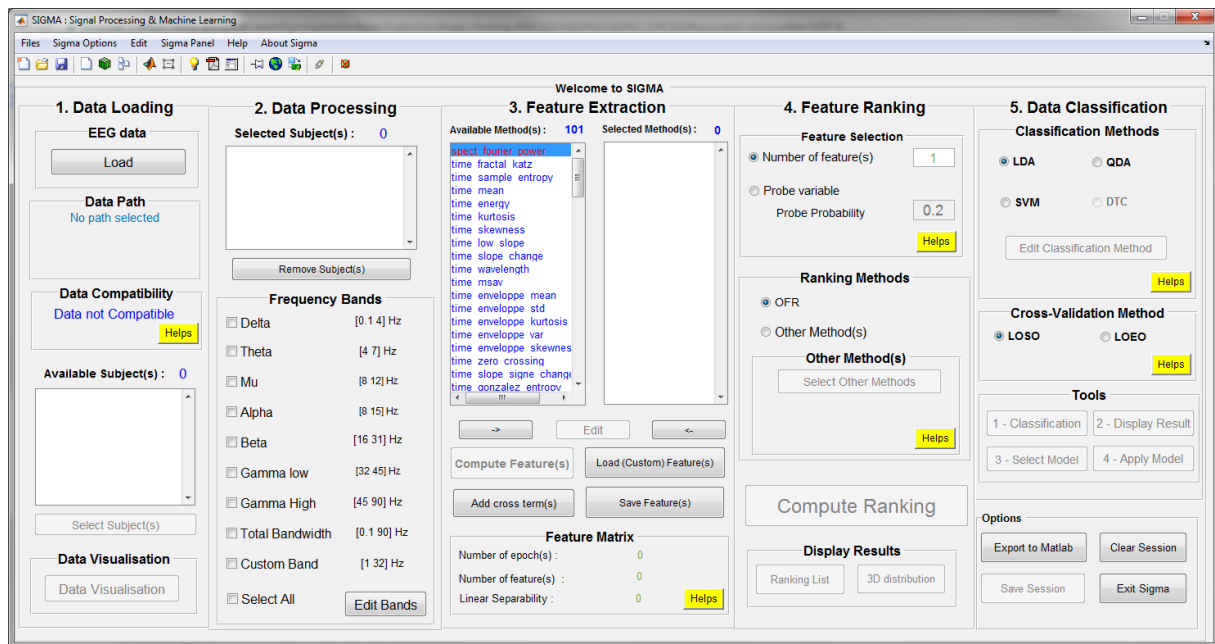


Figure 2 : The Main Graphical User Interface Of SIGMA

The GUI interface is designed to represent all the process explained in the figure 1. The left part of the GUI is dedicated to signal and data processing, the middle part to feature extraction and selection and the right part to data classification. The user can naturally navigate in the GUI from left to right. If conditions are not met or if parameters are missing, the GUI prevents the user from proceeding further.

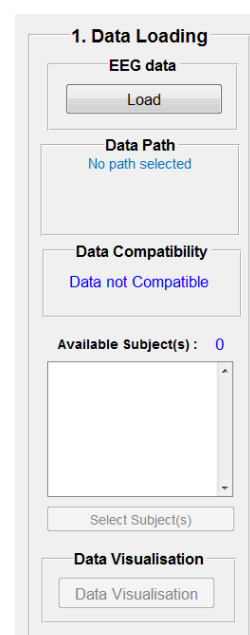
Unlike to the other available tools [4][5], advanced skills in programming are not needed to use SIGMAbox. All required hyperparameters can be reached from the main GUI. The toolbox offers options that users can select and run to get the desired results, together with helpful graphical displays.

3.1.Data Loading

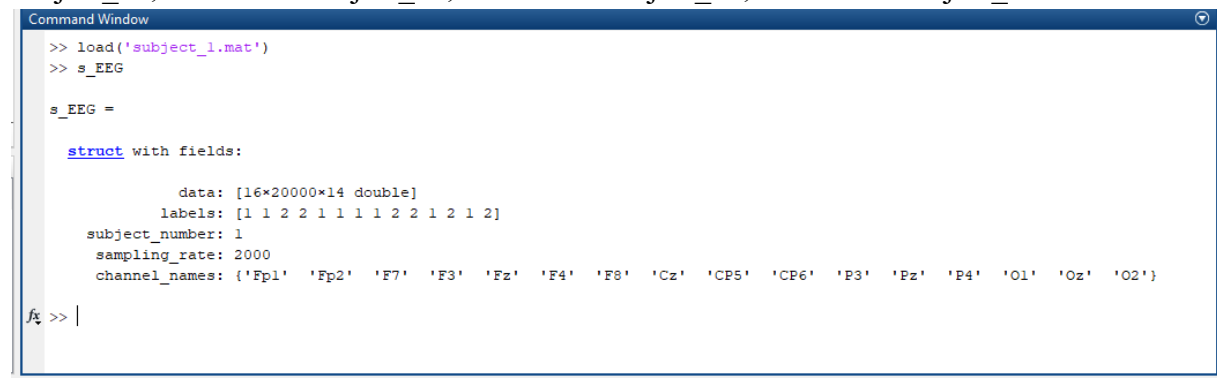
To use correctly the GUI, a sequence of steps should be performed, as shown; the first step is data loading. Clicking on ‘Load’ allows user to select the directory containing his/her data. A browser window will open and then user can choose the data path. In the download file, user can find the directory named ‘SIGMA_data’ with samples to use this manual.

3.1.1. Data Format

The SIGMAbox uses a specific data format. The data file is a Matlab *mat* format presented as follow. The name of the data file should be ‘subject_xx.mat’ where xx should be the number of the



subject, this number should be an integer with at least two digit for example subject_01, subject_02, subject_07, subject_18 ...



```

Command Window
>> load('subject_1.mat')
>> s_EEG

s_EEG =

    struct with fields:
        data: [16x20000x14 double]
        labels: [1 1 2 2 1 1 1 1 2 2 1 2 1 2]
        subject_number: 1
        sampling_rate: 2000
        channel_names: {'Fp1' 'Fp2' 'F7' 'F3' 'Fz' 'F4' 'F8' 'Cz' 'CP5' 'CP6' 'P3' 'Pz' 'P4' 'O1' 'Oz' 'O2'}
fx >> |

```

This file must contain a structure named s_EEG, which should contain at least these fields :

<i>data</i>	:	<i>[n×m×k]</i>	<i>double</i>
<i>labels : [k×1]</i>			
<i>subject_number</i>	:		<i>xx</i>
<i>sampling_rate</i>	:	<i>sampling rate value</i>	<i>(Hz)</i>
<i>channel_names</i>	:	<i>{1×n}</i>	<i>cell</i>

where :

data : is the three dimensional matrix, with size *n×m×k*. The size of the first dimension is equal to the number of EEG channels, denoted by n, that the size of the second dimension is equal to the number of samples of each epoch, denoted by m, and that the size of the third dimension is the number of epochs, denoted by k. The stored values are the amplitude of the signal at each time sample.

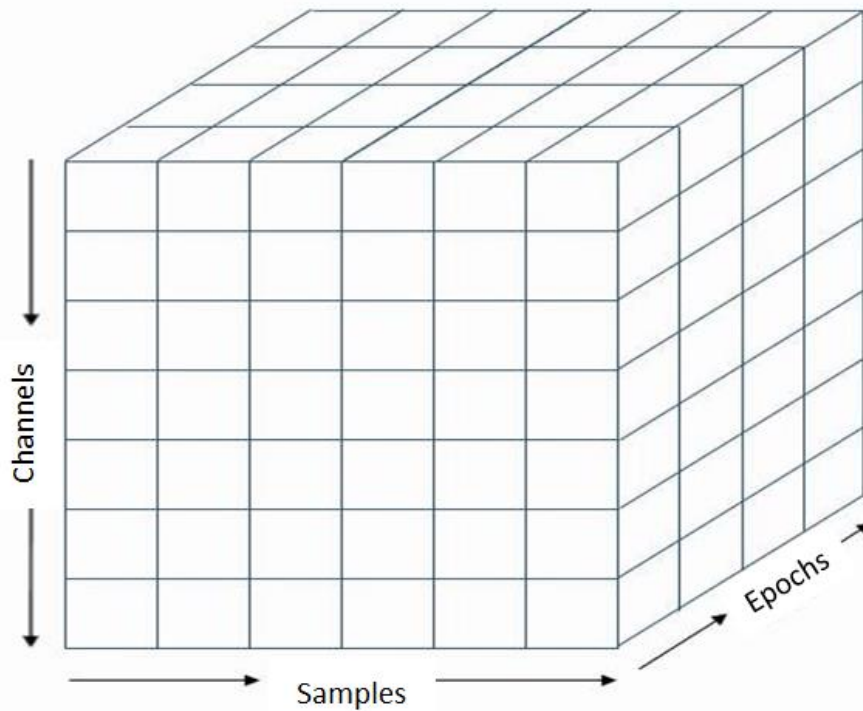


Figure 4 : The shape of the data used by SIGMA

labels : a vector containing the 'Label' or the expected 'Output' of the epoch. If the labels are different from typical value, SIGMABox will change them to the typical labels ('1' and '-1'). The length of the labels vector should be the same as the length of the examples (epochs) in the cube (k).

subject_number : the index of the subject and should be the same in the name of the Matlab file, if the Matlab file is *subject_xx*, subject number should be *xx*

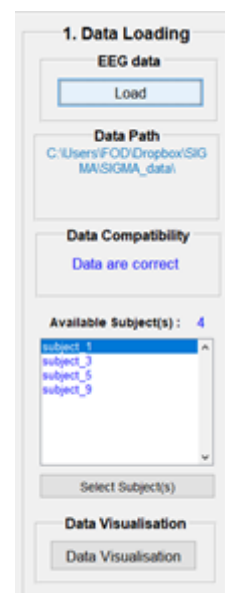
sampling_rate : integer, as the name indicates, it's the sampling rate of the signal in Hz.

channel_names : cell containing the name of the channels, the size should be the same as the number of the channels (n).

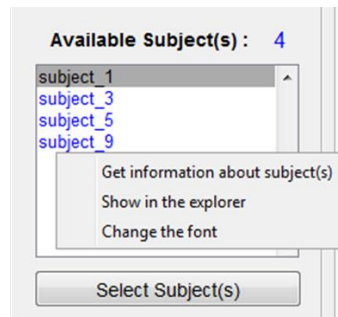
3.1.2. Data Compatibility

When a data file is provided, the toolbox checks its compatibility with the internal data format. If the data format is incorrect, the file is rejected and an error message is displayed. If the format of the data is respected, the field 'Data Compatibility' will display a message 'Data are correct' and the path of the data is shown in the field 'Data Path'. The folder can be accessed by right-clicking on the name of the path and clicking on 'Open the folder'. The number of the subject will appear and the list of available subjects is displayed at the bottom of the panel.

3.1.3. Available Subject



The toolbox displays the subjects that can be used for current session. Each subject file can comprise several epochs and classes, that can be viewed by right clicking on the subject name and select among the context menu as shown in the following figure :



- Get information about subject(s): opens a window and displays all the information about the loaded subject, a screen shot of the window is shown in this figure.

Loaded subject on SIGMA

Infos about the loaded subject(s)

	Subject number	nb_channel	nb_sample	sampling_rate (Hz)	time_duration (s)	nb_epoch	class_one	class_two	amount_class_one	amount_class_two	rate_class_one	rate_class_two
subject1	1	16	20000	2000	10	14	1	2	8	6	57.1429	42.8571
subject2	3	16	20000	2000	10	9	1	2	5	4	55.5556	44.4444
subject3	5	16	20000	2000	10	21	1	2	11	10	52.3810	47.6190
subject4	9	16	20000	2000	10	17	1	2	12	5	70.5882	29.4118

- Show in the explorer : will open the folder containing the subjects' files.
- Change the font (optional) : change the text font.

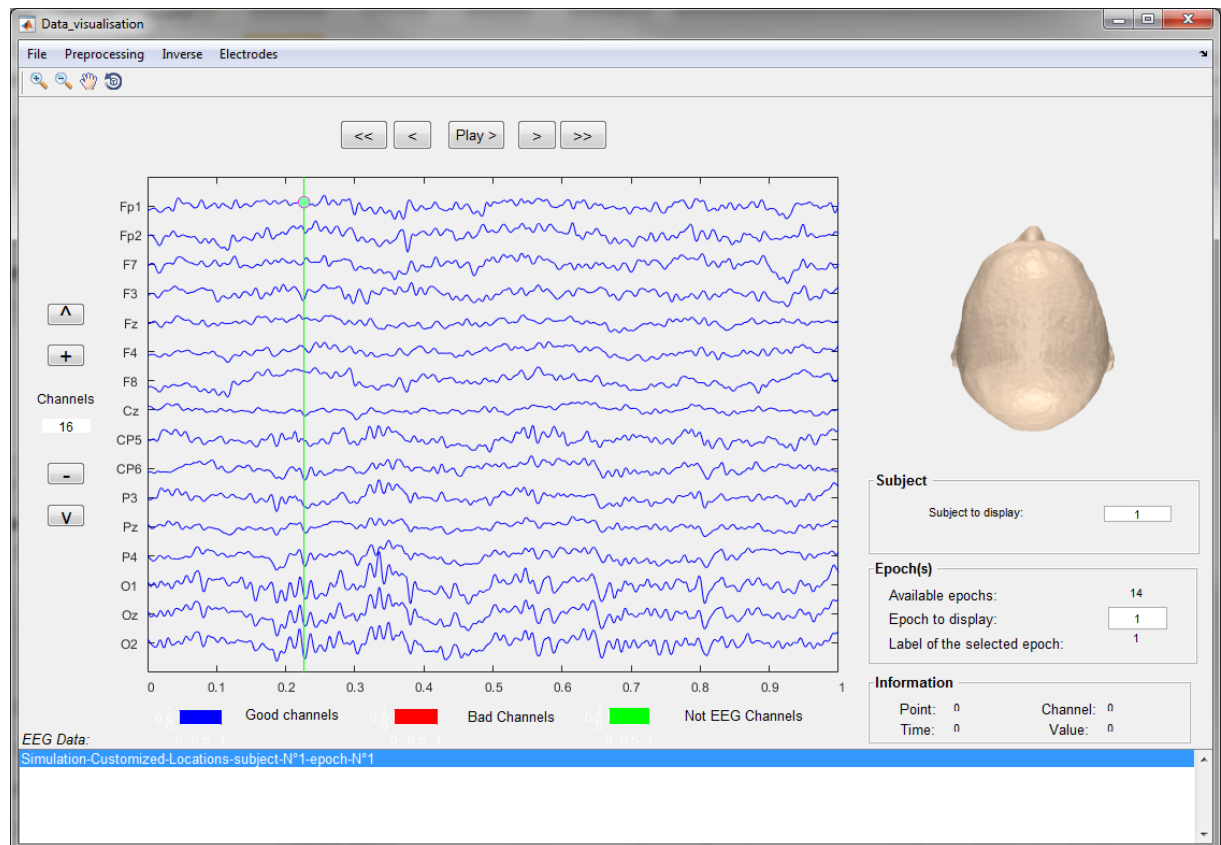
3.1.4. Data Visualisation

Before using subjects as inputs for the computation of machine learning algorithms, signals of each subjects can be visualized using the “Data Visualisation” button. This opens another GUI that allows the user to view the subject signals.

Once the subjects are correctly loaded, user can use the push button ‘Data Visualisation’ to view his data, the signals and some other features. The following explains the different functions that these tools can use.

3.1.5. Link to the Manual of the data Visualisation

When you click on the « Visualize data » on the Sigma GUI, if your data are under the good format (the one the Sigma toolbox is using), it will open a GUI as this one below :



For more details about this panel please refers to this report [link](#).

3.2.Data Processing

This section is dedicated to the processing of selected data before the feature selection and the training of the machine learning algorithms.

To select subject from the previous panel to the current panel, user has just to double click on the subject on the available subject list.

3.2.1. Selected Subject

The selected subjects list correspond to the subjects that will be used to compute the machine learning algorithms. To select a subject, the user can double click on the subject name in the “available subject” list or using the right arrow behind this list. On the contrary, if the user wants to remove a subject from the selected subject list, he can double click on the subject name on the “selected subject” list or click on the left arrow behind the “selected subject” list.

2. Data Processing

Selected Subject(s) : 0

Remove Subject(s)

Frequency Bands

<input type="checkbox"/> Delta	[0.1 4] Hz
<input type="checkbox"/> Theta	[4 7] Hz
<input type="checkbox"/> Mu	[8 12] Hz
<input type="checkbox"/> Alpha	[8 15] Hz
<input type="checkbox"/> Beta	[16 31] Hz
<input type="checkbox"/> Gamma low	[32 45] Hz
<input type="checkbox"/> Gamma High	[45 90] Hz
<input type="checkbox"/> Total Bandwidth	[0.1 90] Hz
<input type="checkbox"/> Custom Band	[1 32] Hz
<input type="checkbox"/> All	

3.2.2. Frequency Bands

Once the subjects are selected, the user must select the frequency band of the signal that will be used to compute the machine learning algorithm. A frequency band is selected through filtering of the raw signals. The user can select several frequency bands corresponding to broadly accepted definitions in the EEG community. Not selecting any frequency band will block the progression of the user in the GUI, as at least one frequency band must be selected to train a machine learning model.

In this section, '2 – Data Processing', user can select the desired subject to study by double click on the 'Available subject' list to add them on the 'Selected subject' list. Also He can display information about the selected subject by right clicks on the listbox > Get information about selected subject.

3.2.3. Frequency Bands selection

In this section user can select the frequency bands in which he wants to extract the feature/descriptor. The core algorithm uses the selected band to apply filter (pass band filter) already computed on the main code. The frequency bands are set by default to the common used values in the literature [reference, <https://en.wikipedia.org/wiki/Electroencephalography>].

3.2.4. Edit Frequency Bands

The user can define its own frequency band or changes the existing frequency band by clicking on the "Edit" button at the bottom of the "Frequency band" panel. This action opens another GUI dedicated to the modification of the frequency bands.

The image shows a window titled "Frequency_band_edit_gui" with a table of frequency bands. Each band has input fields for "Minimal Frequency" and "Maximum Frequency". At the bottom are "OK" and "Cancel" buttons.

	Minimal Frequency	Maximum Frequency
Delta	0.1	4
Theta	4	7
Mu	8	12
Alpha	8	15
Beta	16	31
Gamma	32	45
Gamma High	45	90
Total Bandwidth	0.1	90
Custom Bandwidth	1	32

This GUI allows the user to select minimum and maximum values of each bands. A custom band can be designed to fit specific purposes of advanced users. Clicking on “OK “ saves the new frequency values, clicking on “Cancel” reset the original values to frequency bands boundaries.

3.3.Feature Extraction

When performing analysis of complex data one of the major problems stems from the number of variables involved. Feature extraction is useful when data sizes’ are large and a reduced feature representation is required. Feature extraction is related to dimensionality reduction. This process extract the interesting parts of the data as a compact feature vector/matrix.

This process starts from an initial set of measured data and constructs derived values (feature) facilitating the learning from the original data, and in some cases leading to better human interpretations.

Using the frequency band extracted from raw signals, the ‘Feature Extraction’ process in SIGMA contains the methods that are used to extract features from selected subject signals within the selected frequency bands. The aim of feature extraction is to extract relevant features (like mean, kurtosis,

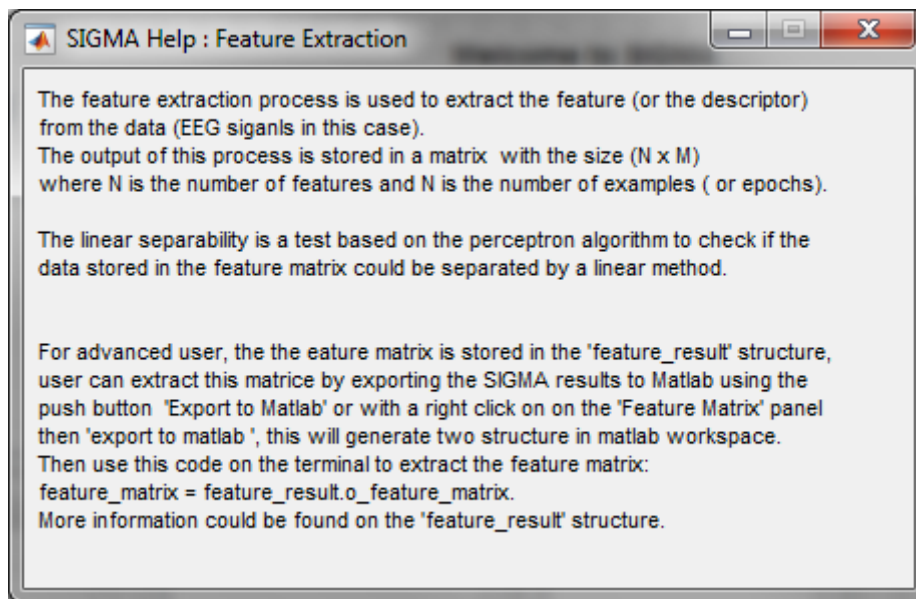
The image shows a window titled "3. Feature Extraction Methods". It has two lists: "Available Method(s): 101" and "Selected Method(s): 0". The available methods list includes: spect, fourier, power, time, fractal, katz, time, sample, entropy, time, mean, time, energy, time, kurtosis, time, skewness, time, low, slope, time, slope, change, time, wavelength, time, msav, time, envelope, mean, time, envelope, std, time, envelope, kurtosis, time, envelope, var, time, envelope, skewness, time, zero, crossing, time, slope, signe, change, time, oonzalez, entropy. Below the lists are buttons: ">", "Edit", "<", "Compute Feature(s)", "Load (Custom) Feature(s)", "Add cross term(s)", and "Save Feature(s)". At the bottom is a "Feature Matrix" section with fields: "Number of epoch(s) : 0", "Number of feature(s) : 0", and "Linear Separability : 0". There is a "Helps" button next to the last field.

enveloppe etc) that will be used as inputs to the machine learning algorithm.

It is difficult to know in advance which methods will be relevant. The next step (Feature Ranking) provide informations on the relevance of methods toward the dataset. Another way to select feature extraction methods is to rely on the existing literature of you research topic.

The output of this section is a list of features, one feature corresponding to a frequency band and a specific method.

The 'Helps' button allows to display this panel



3.3.1. Available Methods for Feature extraction

Available methods are the list of the methods available and not selected by the user. A method can be selected by right clicking on its name or by click on it and then clicking on the right arrow on the bottom of the “Available method” list.

The methods are listed and sorted by ‘family’. The method could be recognized by the font color or by the prefix as follow :

time_xxx : for method applied on the time domaine (blue color)

spect_xxx : fro the spectral methods (red color)

wt_xxx : for the wavelet methods (green color)

synchro_xxx : for the synchronisation methods (brown color)

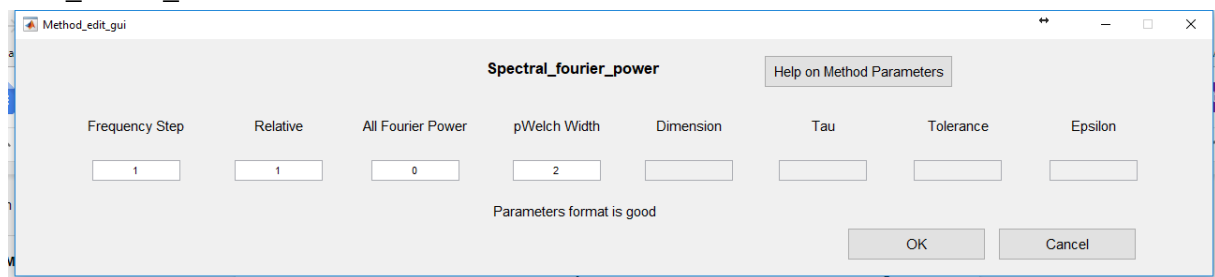
User can add his own method just by following the steps explained on the annex (template matlab to create new code for feature extraction).

3.3.2. Selected Methods

Selected methods are those selected by the user that will be used in the feature extraction algorithms. To remove a method from the selected method list, the user can double click on a method name or select it and click on the left arrow at the bottom of the list. Many machine learning practitioners believe that properly optimized feature extraction is the key to effective model construction.[2]

3.3.3. Edit

This opens a new panel allowing the user to edit method's parameters. The parameters differ from one method to another. Some of the method could be edited by user by a simple click on the EDIT button, the editable method are the : spect_fourier_power and the time_fractal_katz.



where :

Frequency Step : step of the power integration for the frequency Used in the Fourier power computation (refers to pwelch Matlab function)

pWelch Width : window width for the pwelch function

Dimension : embedded dimension used by the Sample entropy function

Tau : used by the Sample entropy function

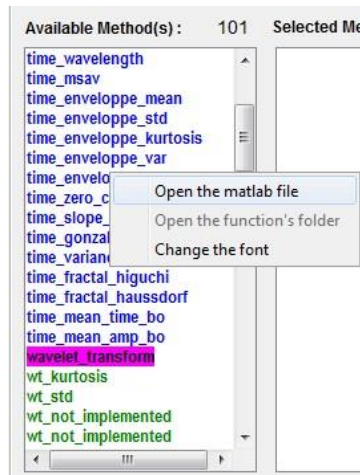
Tolerance : used by the Sample entropy function

Epsilon : used by the low slope method

Relative : Compute or not the relative Fourier power per band

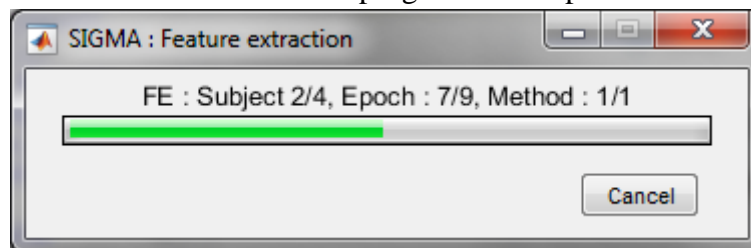
All frequency Power : Compte all the power on the all signal

For advanced users, additional parameters could be reached from the matlab function Sigma_method_initialisation.m, this file could be opened by a right click on the context menu as shown in this picture :



3.3.4. Compute Feature

By clicking on this button, the user launch the computation process of the features selected; this process can be long due to the computational cost of some methods. So, the user shall not expect an immediate response from the GUI as long as computation is not over. A waitbar is displayed to indicate to the user the progress of computation.



3.3.5. Load Feature

Some people would like to use custom data in the toolbox, that do not come from an EEG setup. Some example are temperature or psychological questionnaires. To add these data to the SIGMAbox, users can use the “Load Feature” button provided they use a compatible data format.

3.3.6. Custom feature data format

In order to use this additional function on SIGMA, user should pre-process the data and save it in the specific format as explained below.

The data should be saved in matlab data file (*.mat) and this mat file should contains at least the following fields.

Assume that the matlab file with the name ‘custom_feature.mat’

```
>> load('custom_feature.mat')
```

```
Command Window
>> data = load('custom_feature.mat')

data =

    struct with fields:

        feature_value: [32x23 double]
        feature_name: {32x1 cell}
        label: [-1 -1 1 1 -1 -1 -1 -1 1 1 -1 1 -1 1 1 -1 1 -1 -1 1 -1 -1 1]
```

feature_name, feature_value, label should be in the file, as in this example

feature_value : is a matrix with a size n*m. with n is number of feature and m the number of examples (epochs).

feature_name : cell of string, containing the name of feature. The size of this matrix should be the same as the number of feature. (n in this case)

label : vector of the outputs, containing the labels mainly 1 and -1, the size is the same as the number of the epochs.

3.3.7. How to use SIGMA with custom feature

User can use SIGMA only from this step. Custom feature could be used to compute new model without any previous step or signal from loaded subjects.

As Custom feature could be added to a previous computed feature from subject, in this case, the size of the added feature (custom) must fit to the current computed feature from the selected subject. (User should ensure that the labels are the same in the case of adding feature)

3.3.7.1. Custom feature from included on the subject

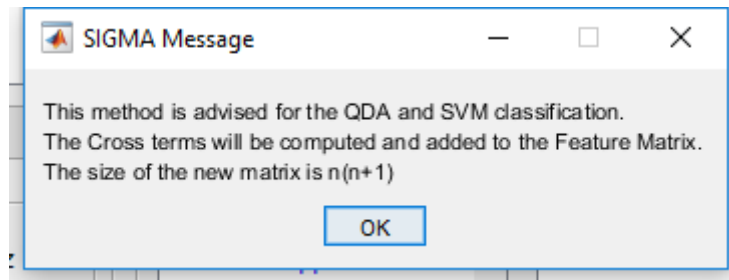
Custom feature could be included on subject. in this case the subject matlab file should contain additional fields and the user should check the option in order that SIGMA could include the custom feature.

% Add checkbox as an option in which user can add the custom feature included on the subject (check box disabled if no custom feature) ⇒ add method 99

3.3.8. Add Cross term

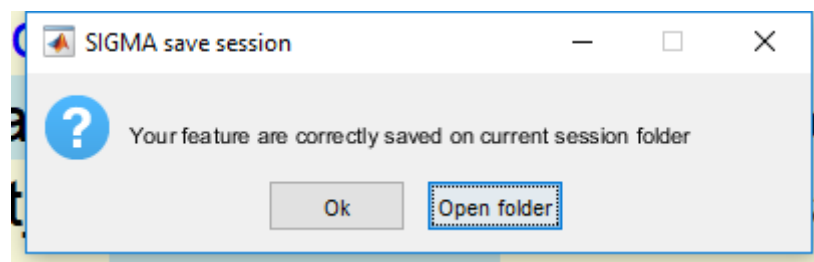
In order to improve the efficiency of the Feature Ranking process and the classification, the user can request multiplications of feature to be added to the standard single feature data structure. These secondary features are known as “cross terms” or ‘interaction terms’ and can prove very useful in uncovering non-linear relationships between features and output. For example, if the user selects F1, F2 and F3, three features from the feature extraction panel, the standard procedure will compute F1, F2 and F3 from the data. If it also requests the cross terms, the output of the feature extraction will be F1, F2 and F3, and also F1*F2, F1*F3 and F2*F3.

Adding the cross term could be more interesting and could bring more informations then the simple terms. If user choose this option the number of feature will be increased from n to $n(n+1)$, a message box will appears with information about this option.



3.3.9. Save Feature

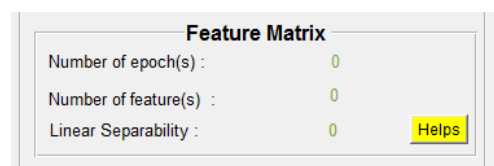
As the computation of feature can be time consuming, the user can save the computed feature before using the rest of the process on Sigma, and can load these feature next time. The 'Save Feature(s)' push button allow the user to save the feature in a matlab file which is saved on the session folder. A message box is displayed on the screen and user can open the folder session.



Feature can be loaded by loading the resulting file from the session using the 'Load (custom) Feature(s)' button.

3.3.10. Feature Matrix

This panel displays general informations about the output of the feature extraction process :



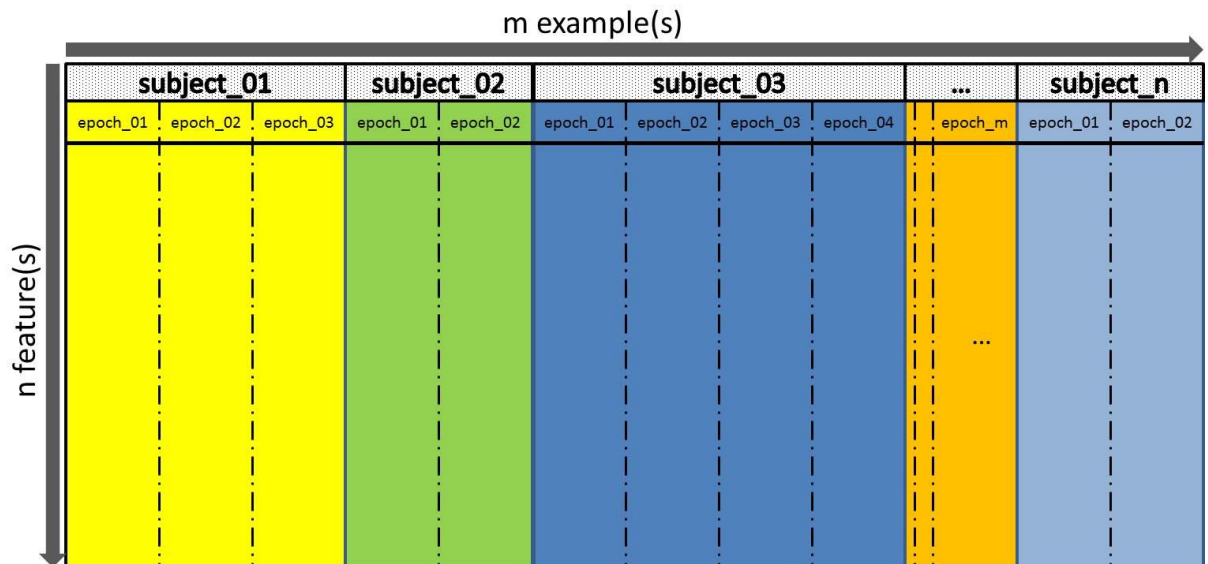
- Number of epoch(s) : the total number of epoch on which feature have been computed. This number is the same as the number of labels on the all selected subject's data.
- Number of feature(s) : Corresponds to the number of the computed (or loaded) feature. In the case of the computed feature, this number depend on the used method, number of the frequency band and the number of the channel.
 - For the simple methods (single channel methods), the number of feature N is :

$$N = \text{number of method(s)} * \text{number of channel} * \text{number of frequency band}$$
 - For the double channel (synchrony methods), the total number N is :

$$N = \text{number of method} * \text{number of channel} * (\text{number of channel} - 1) / 2 * \text{number of the selected frequency band}.$$

On some other methods, the number of feature could be different.

- Linear separability : this indicate if the data set (loaded or computed features) can be classified and separated by a linear classifier. The process is based on the linear perceptron [\[ref\]](#)



3.4.Feature Selection

Analysis with a large number of variables generally requires a large amount of memory and computation power; also it may cause a classification algorithm to overfit. Large amount of data is also suspected to be redundant (e.g. the same measurement in both feet and meters), then it can be transformed into a reduced set of features (also named a feature vector (on example) or feature matrix (multiple examples)). Feature selection techniques should be distinguished from feature extraction. Feature extraction creates new features from functions of the original data/features, whereas feature selection returns a subset of the features.

Once user has extracted feature matrices, it is necessary to look for the best features in the set, usually termed as “supervised feature selection”. The key idea is to rank the variables according to their correlation with the expected output.

In SIGMA, user can choose between selecting the Nth best features (“Number of feature” in Feature Selection”) or the subset of features that performs better than a random variable

4. Feature Ranking

Feature Ranking Methods

☒ Number of feature(s)

☐ Probe variable

Probe Probability

[Helps](#)

Ranking Methods

☒ OFR

☐ Other Method(s)

Other Method(s)

[Select Other Methods](#)

[Compute Ranking](#)

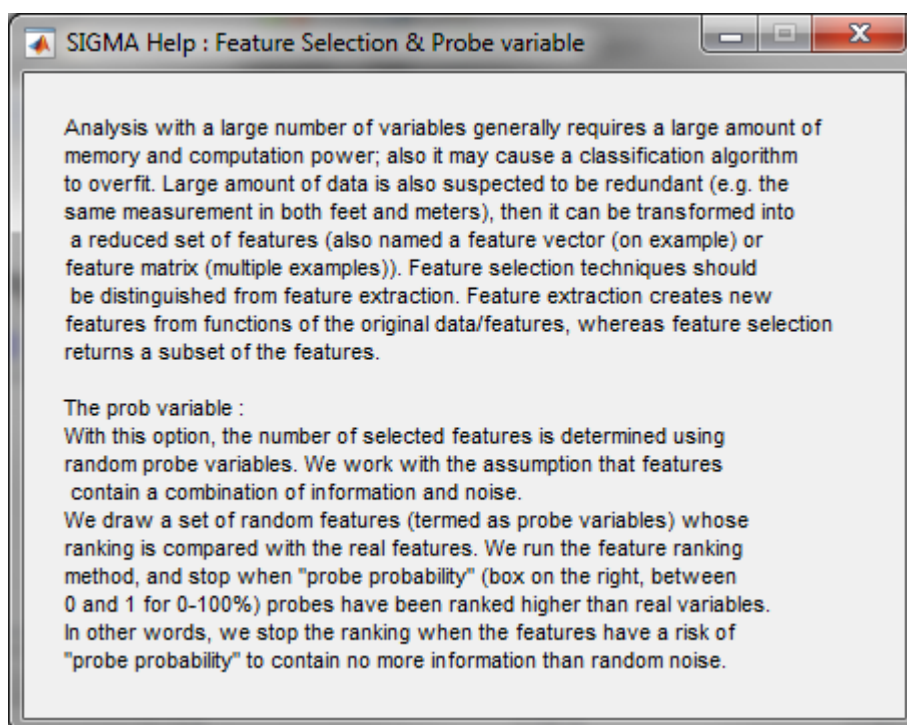
Display Results

[Ranking List](#) [3D distribution](#)

included in the whole matrix feature set, with a given probability ("Probe variable") that the user can set.

The prob variable : With this option, the number of selected features is determined using random probe variables. We work with the assumption that features contain a combination of information and noise. We draw a set of random features (termed as probe variables) whose ranking is compared with the real features. We run the feature ranking method, and stop when "probe probability" (box on the right, between 0 and 1 for 0-100%) probes have been ranked higher than real variables. In other words, we stop the ranking when the features have a risk of "probe probability" to contain no more information than random noise.

If the user clicks on the 'Helps' push button, this will display a panel with the help about the feature selection and the probe variable.



The selected features are expected to contain the relevant information from the input data, so that the desired task can be performed by using this reduced representation instead of the complete initial data.

3.4.1. Ranking Method

Ranking methods are the methods used to compute the "proximity" between features and the output. This notion of "best" is defined through the ranking. This is done from a

variety of methodologies and algorithms which are provided but we strongly recommend to stick to the Orthogonal Forward Regression (OFR) [\[ref article de gérard\]](#) in a first approach.

Orthogonal Forward Regression (OFR)

The Feature are selected according to the highest projection on the output vector. The OFR performs the following steps:

- Compute the angle $\varphi(F_i, O)$ between each candidate feature F_i and the quantity to be modeled O and select the candidate feature F_j that has the smallest angle with O , i.e. the candidate feature that is most correlated to O :

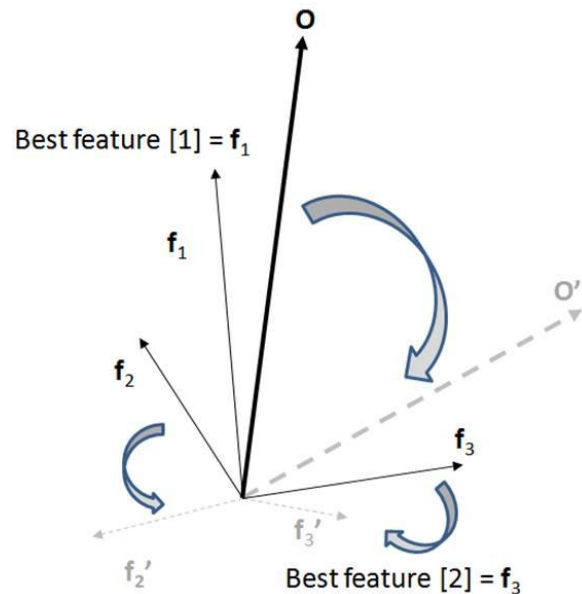
$$F_j = \operatorname{argmax}(\cos^2 \varphi(F_i, O))$$

)

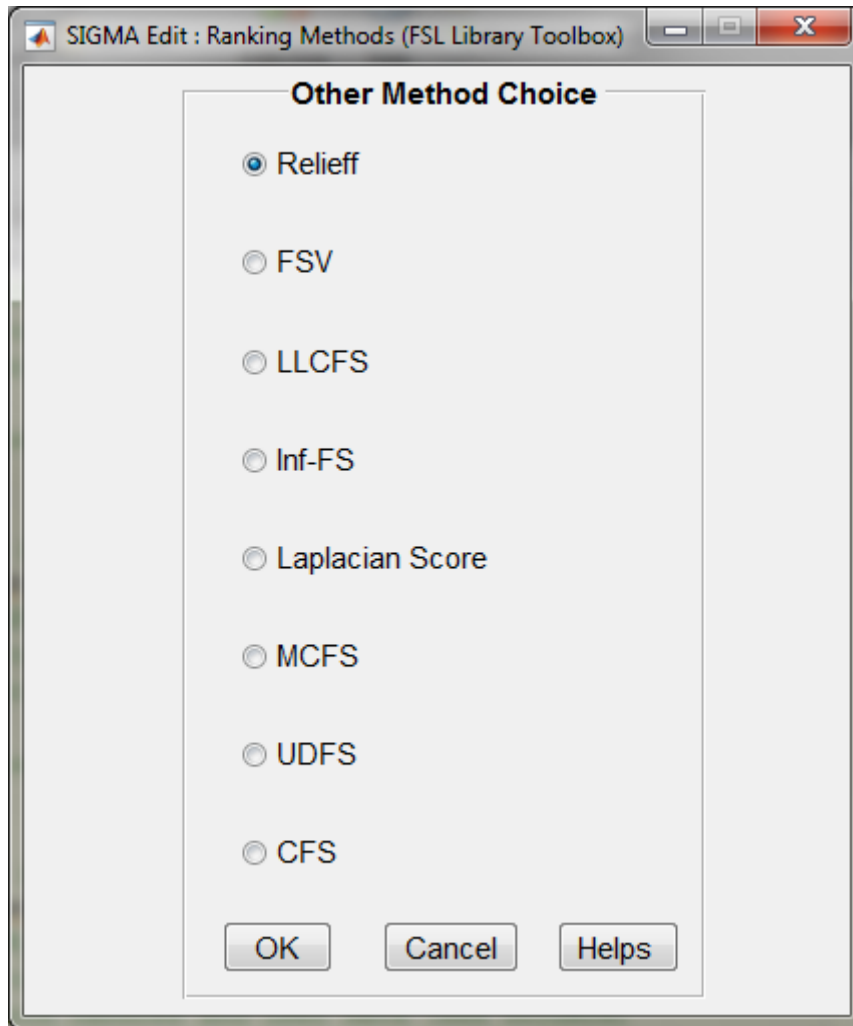
- Projects O and all the remaining candidate features onto the null space of the selected feature;

The above two steps can be iterated in subspaces of decreasing dimensions until all candidate features are ranked.

As you can see on the illustration, this ranking procedure removes the inter-correlation of the feature vectors, therefore removing useless redundancies, and providing a more reliable estimate of the best possible combination of features for a classification. The feature f_3 is selected as second-best, because f_2 is redundant with f_1 .



Advanced users can explore other possibilities provided by the “Other methods” panel.



For more details about these methods, please refer to this paper or the FSL library [\[Feature Selection Library \(MATLAB Toolbox\)\]](#).

ref : G Roffo, 'Feature selection library (MATLAB toolbox)', arXiv preprint [arXiv:1607.01327](https://arxiv.org/abs/1607.01327), 2016 - arxiv.org

The help button will open the pdf file, which is the published paper about the FSL library toolbox.

3.4.2. Compute Ranking

This button allows the user to launch the feature ranking algorithm according to the methods selected on the listbox 'Selected method', either the OFR or one of the 'Other Method(s)' with the associated methods.

3.4.3. Display Result

This section allows the user to visualize the ranking results in multiple ways. The ranking list or the 3D distribution of the three best features. These two possibilities are exposed in the following sections.

3.4.4. Ranking List

The ranking list displays the ranked features by order best projection to the output vector. The 'cos(theta)' being the measure of $\cos\angle(F_i, 0)$, is displayed for each features. The channel and frequency band are also displayed in order to identify the origin of the feature. In the case of the synchrony feature, the couple of channels are displayed.

SIGMA : Feature Ranking

Feature(s) Ranking & Identification

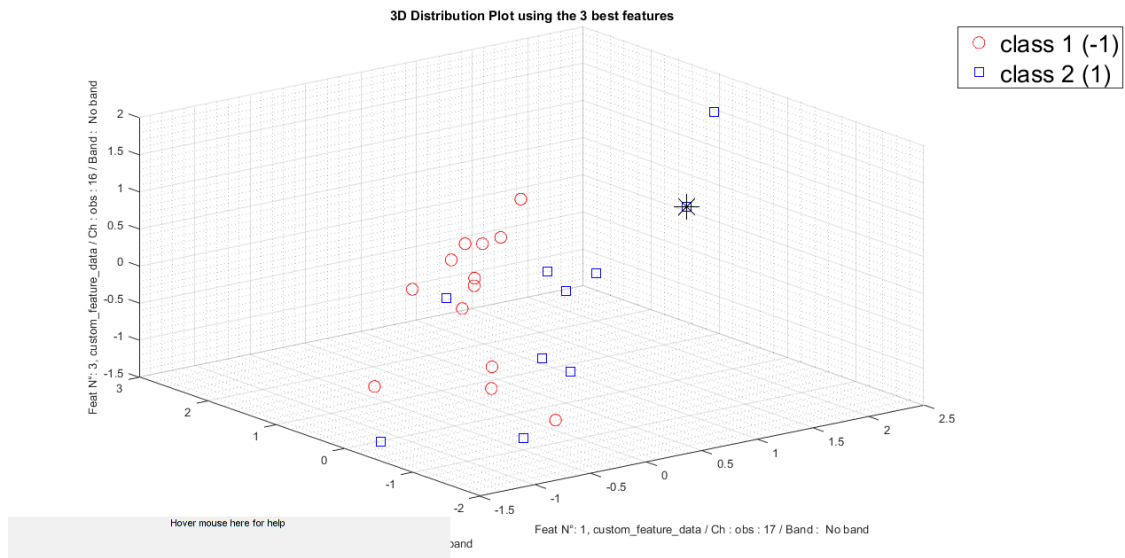
	Method (Number)	Channel(s)/Observation(s)	Frequency	Score (cos(theta))
1	time_mean (method N°: 4)	Channel(s) : 4 ; (F3)	Delta : [0.1 4] Hz (band N°:1)	0.0402
2	time_mean (method N°: 4)	Channel(s) : 1 ; (Fp1)	Delta : [0.1 4] Hz (band N°:1)	-0.0213
3	time_mean (method N°: 4)	Channel(s) : 3 ; (F7)	Delta : [0.1 4] Hz (band N°:1)	0.0290
4	time_mean (method N°: 4)	Channel(s) : 14 ; (O1)	Delta : [0.1 4] Hz (band N°:1)	0.0544
5	time_mean (method N°: 4)	Channel(s) : 10 ; (CP6)	Delta : [0.1 4] Hz (band N°:1)	-0.0078
6	time_mean (method N°: 4)	Channel(s) : 15 ; (Oz)	Delta : [0.1 4] Hz (band N°:1)	-0.0183
7	time_mean (method N°: 4)	Channel(s) : 11 ; (P3)	Delta : [0.1 4] Hz (band N°:1)	-0.0278
8	time_mean (method N°: 4)	Channel(s) : 9 ; (CP5)	Delta : [0.1 4] Hz (band N°:1)	-0.0196
9	time_mean (method N°: 4)	Channel(s) : 5 ; (Fz)	Delta : [0.1 4] Hz (band N°:1)	-0.0026
10	time_mean (method N°: 4)	Channel(s) : 6 ; (F4)	Delta : [0.1 4] Hz (band N°:1)	0.0108
11	time_mean (method N°: 4)	Channel(s) : 13 ; (P4)	Delta : [0.1 4] Hz (band N°:1)	-0.0021
12	time_mean (method N°: 4)	Channel(s) : 8 ; (Cz)	Delta : [0.1 4] Hz (band N°:1)	-0.0197
13	time_mean (method N°: 4)	Channel(s) : 12 ; (Pz)	Delta : [0.1 4] Hz (band N°:1)	-0.0194
14	time_mean (method N°: 4)	Channel(s) : 2 ; (Fp2)	Delta : [0.1 4] Hz (band N°:1)	0.0151
15	time_mean (method N°: 4)	Channel(s) : 16 ; (O2)	Delta : [0.1 4] Hz (band N°:1)	0.0050
16	time_mean (method N°: 4)	Channel(s) : 7 ; (F8)	Delta : [0.1 4] Hz (band N°:1)	-0.0263

Display Cross term ranking 3D Feature Distribution Load Feature Ranking Results

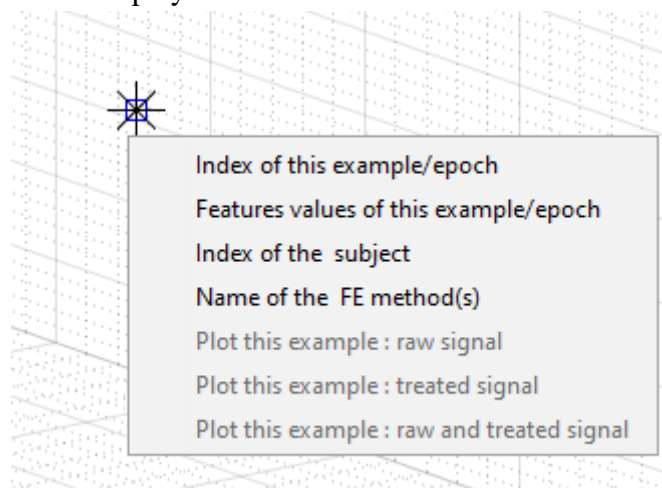
From this panel, user can also display the 3D distribution of the feature. If any previous feature are saved in a session file, user can load and display these feature from this panel using the button 'Load Feature Ranking Results'.

3.4.5. 3D Distribution of the three best feature

The 3D features distribution plot on a 3D plan the distribution of the feature according to the three bests ranked features computed from the ranking. Each point on the plot represent one epoch (example). User can click on each point and show more informations about the selected example.



A context menu is possible to use in this figure. Right click on the point and the the following menu will be displayed.



The allowed options on the context menu are :

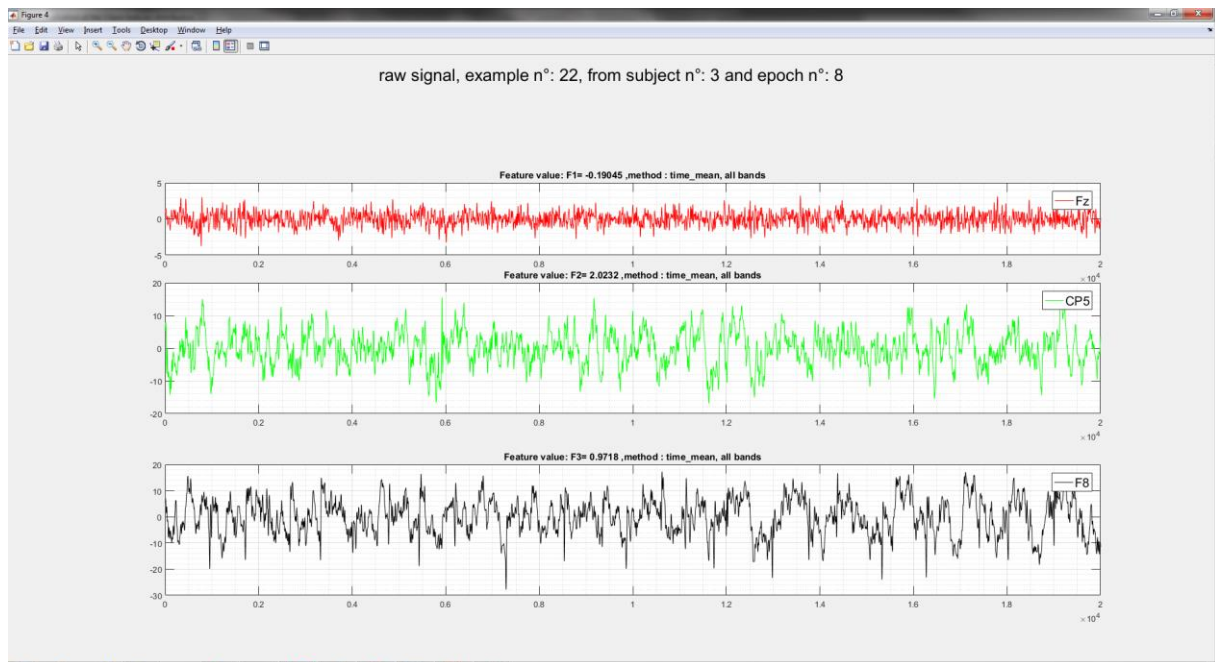
Index of this example/epoch : Displays on the figure the index of the example

Feature value of this example/epoch : Displays the value of the current feature

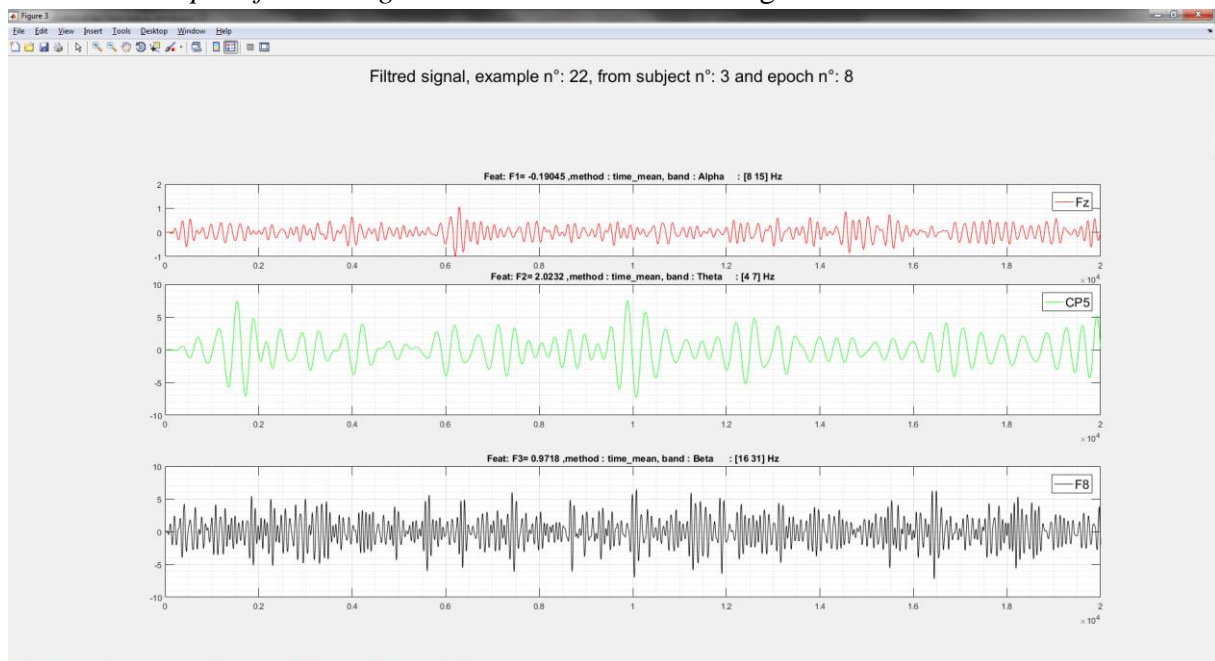
Index of the subject : Displays the index of the subject

Name of the FE method : Display the name of the used method to extract the feature

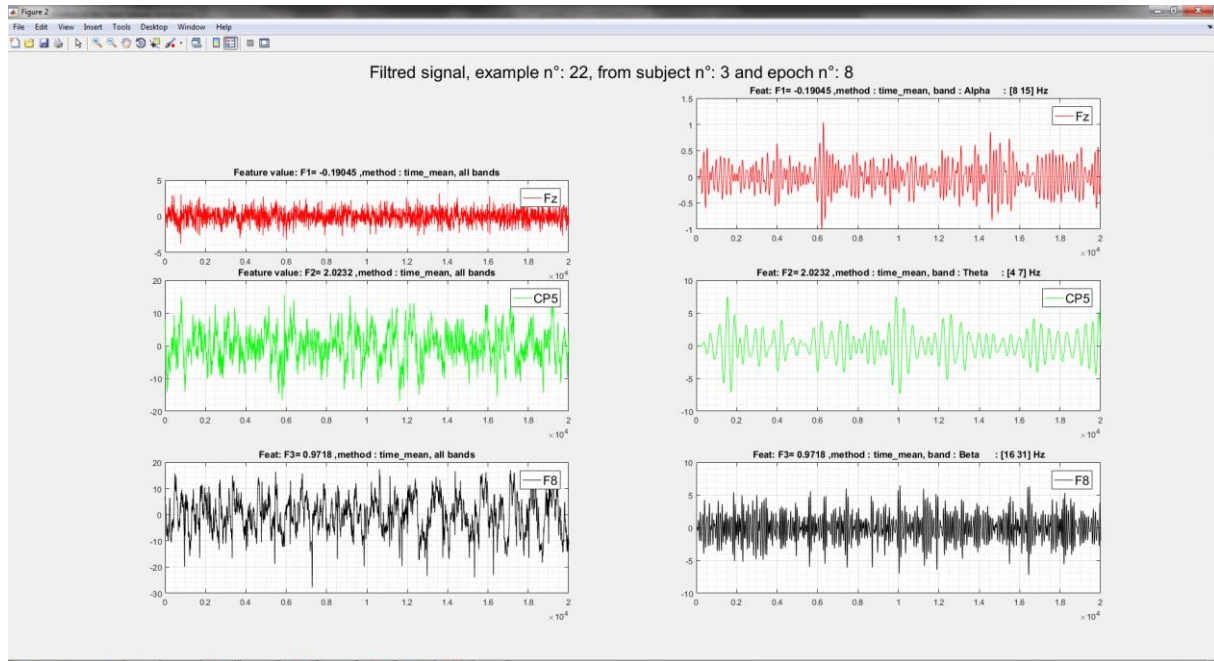
Plot this example : raw signal : Plot the EEG raw signal



Plot this example : filtered signal : Plot the filtered EEG signal



Plot this example : raw and filtered data : Plot both signal EEG signal (raw and filtered)



Note : If the feature is coming from custom feature, there is no signal to plot. Also, if the features are coming from a synchrony measure, the signal plot is not implemented in this version.

3.5.Data Classification

Once the best features are extracted, user can classify the feature sets and estimates the performance of the built classifier model.

This section is dedicated to these purposes. User can select a classification methods and a cross validation method from this panel.

5. Data Classification

Classification Methods

☒ LDA
 ☐ QDA

☐ SVM
 ☐ DTC

Edit Classification Method

Cross-Validation Method

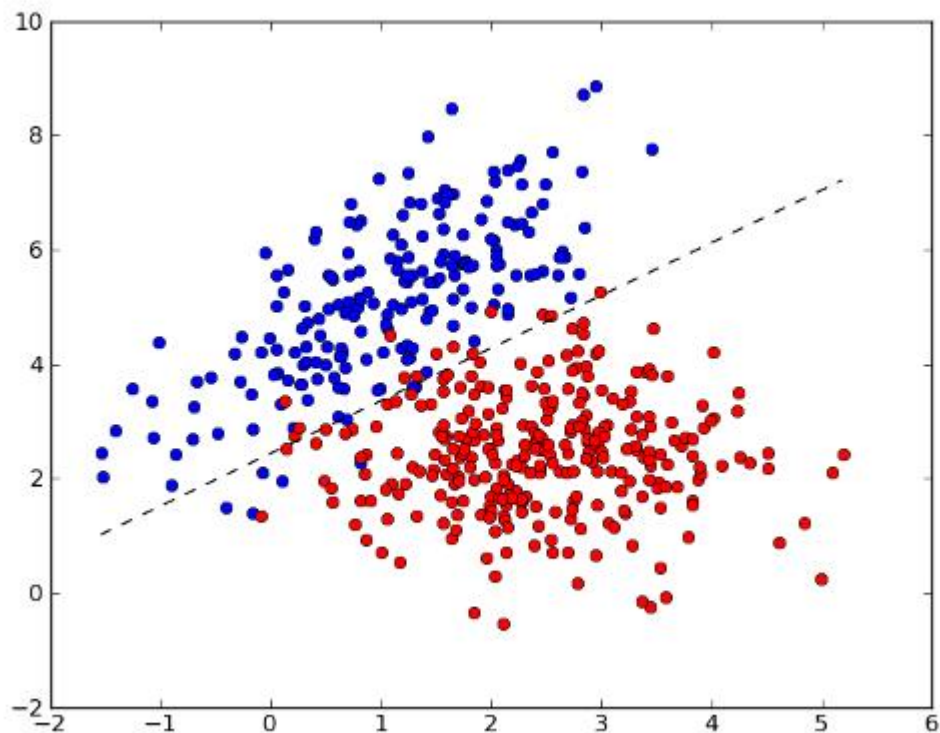
☒ LOJO
 ☐ LOEO

?

?

3.5.1. Classification Method

The classification method is the mathematical algorithm that is used to classify the data. It is the “function” that is used to draw a boundaries between differents classes of examples (epochs). In the following figure, this function compute the equation of the dashed line separating the two set of data.



Three methods are implemented in this version SIGMAbox, these methods are :

- Linear discriminant analysis (LDA) is a generalization of Fisher's linear discriminant [reference], a method used in statistics, pattern recognition and machine learning to find a linear combination of features that characterizes or separates two or more classes of objects or events. LDA attempt to express one dependent variable as a linear combination of other features or measurements, LDA analysis has a categorical dependent variable (i.e. the class label). Discriminant analysis is used when groups are known a priori (supervised learning).
Matlab has an implementation of the LDA algorithm: the 'fitcdiscr' function. LDA in SIGMA box is based on this function.
- Quadratique Discriminant analysis (QDA) is quadratic classifier used in machine learning and statistical classification to separate measurements of two or more classes of objects or events by a quadric surface. It is a more general version of the linear classifier. Quadratic discriminant analysis (QDA) is closely related to linear discriminant analysis (LDA). QDA creates a longer measurement vector from the old one by adding all pairwise products of individual measurements (cross terms ref). Finding a quadratic classifier for the original measurements would then become the same as finding a linear classifier based on the expanded measurement vector. As for the LDA, QDA in SIGMA box is based on built in Matlab function.
- Support Vector Machines (SVM), also known as support vector networks, is also a supervised learning models with robuste learning algorithms that analyze data used for

classification. An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall.

In addition to performing linear classification, SVMs can efficiently perform a non-linear classification using different kernel, implicitly mapping their inputs into high-dimensional feature spaces. ([Vapnik-Chervonenkis dimension](#)).^[2]

3.5.2. Unsupervised/supervised learning :

When data are not labeled, supervised learning is not possible, and an unsupervised learning approach is required, which attempts to find natural clustering of the data to groups, and then map new data to these formed groups. The supervised learning, is based on set of training examples, each example is marked as belonging to one or the other of two categories.

3.5.3. Edit Classification Method

For this version of SIGMA, only the the Support Vector Machine (SVM) method can be edited by user. In fact, this method involve numerous parameters that can be adjusted by the user. These parameters are implemented on the SVM algorithm: the ‘fitsvm’ function. A separate panel is accessible through “Edit Classification Method”.

Kernel Parameters	Low	Step	High
Gaussian kernel std range	0.5	0.5	4
Polynomial kernel degree range	2	1	5
Soft margin Hyper-Parameter range	0.1	0.1	5

The editable parameters are :

- Type of kernel : User can select one or more kernel function for the SVM computation. The possible kernel are :

- 'gaussian' or 'rbf' : Gaussian or Radial Basis Function (RBF) kernel,
- 'linear' : Linear kernel,
- 'polynomial' : Polynomial kernel
- Kernel parameters : Gaussian kernel standard deviation range
 - Polynomial kernel degree range
 - Soft margin Hyper Parameter range
- KKT : Karush-Kuhn-Tucker complementarity conditions violation tolerance
 - KKT Tolerance
 - KKT Violation Level
- Optimization
 - Number of optimizations of hyper parameters
 - Numbers of optimizations for the final model

Explanation of the influence of these parameters could be reached from the help push button or found within this link <https://fr.mathworks.com/help/stats/fitcsvm.html>

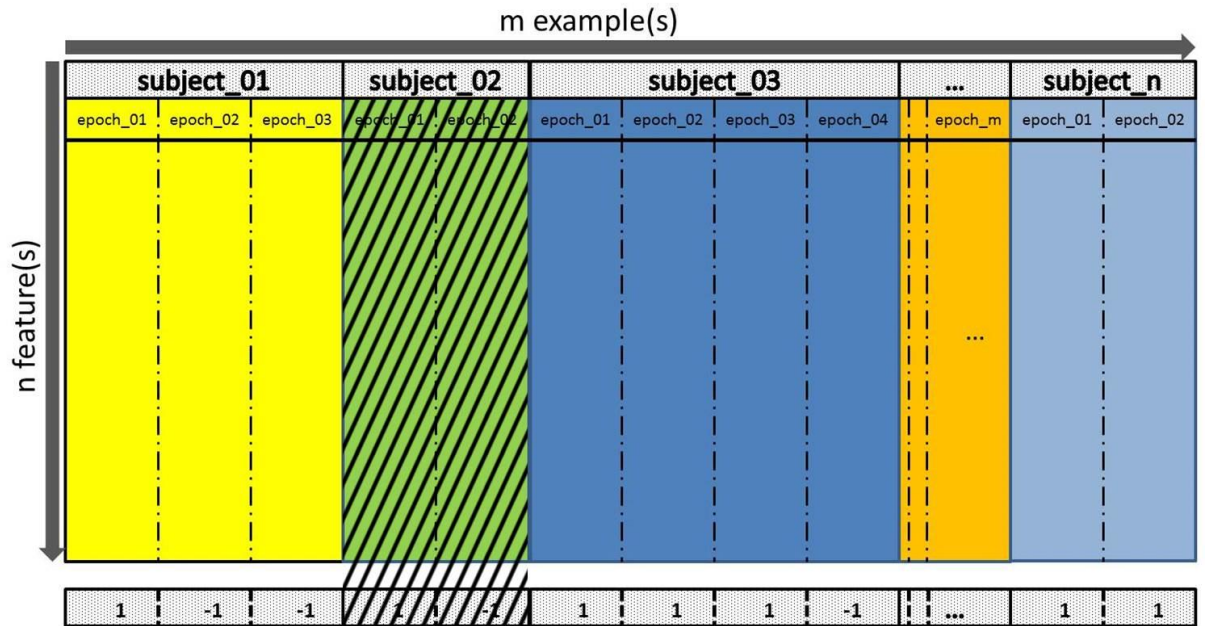
3.5.4. Cross-Validation Method

Cross-validation is a technique that is used for the assessment of how the results of statistical analysis generalize to an independent data set. The holdout method is the simplest kind of cross validation. A round of cross-validation comprises the partitioning of data into complementary subsets, called the training set and the testing set. The test data is removed before training begins, then when training is done, the data that was removed can be used to test the performance of the learned/trained model. This model is asked to predict the output values for the data in the testing set (it has never seen these output values before). A complete and efficient model is based on a multiple rounds of cross-validation using many different partitions and then an average of the results are taken. Cross-validation is a powerful technique in the estimation of model performance technique.

In this version of SIGMA, the 'Leave-One-Out' cross validation is implemented. This is based on two approaches.

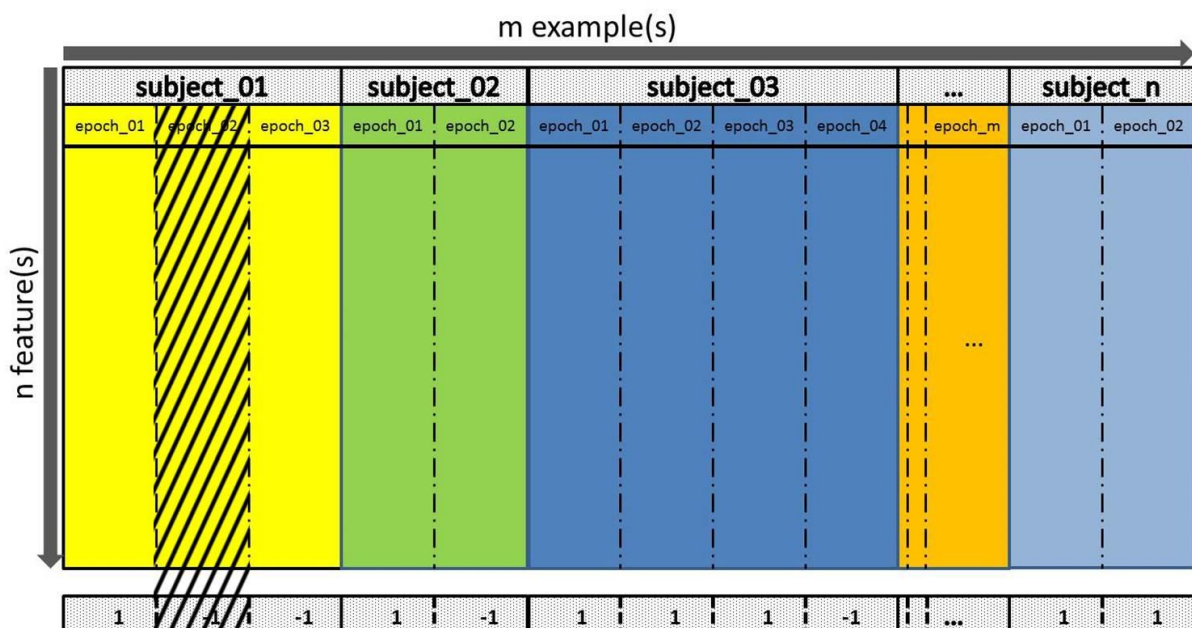
- LOSO : Leave One Subject Out
 - This method train a model using all the subject except one and try to predict the outputs of the leaved out subject.

A subject is supposed to have more than one example, else the LOSO will be the same as the LOEO.



The process of the LOEO is repeated **n** times until the subject will be all processed.

- LOEO : Leave One Epoch/Example Out
 - Leave only one example out and train with all the rest of the example. When the training is over, the prediction is computed on the one leaved example.

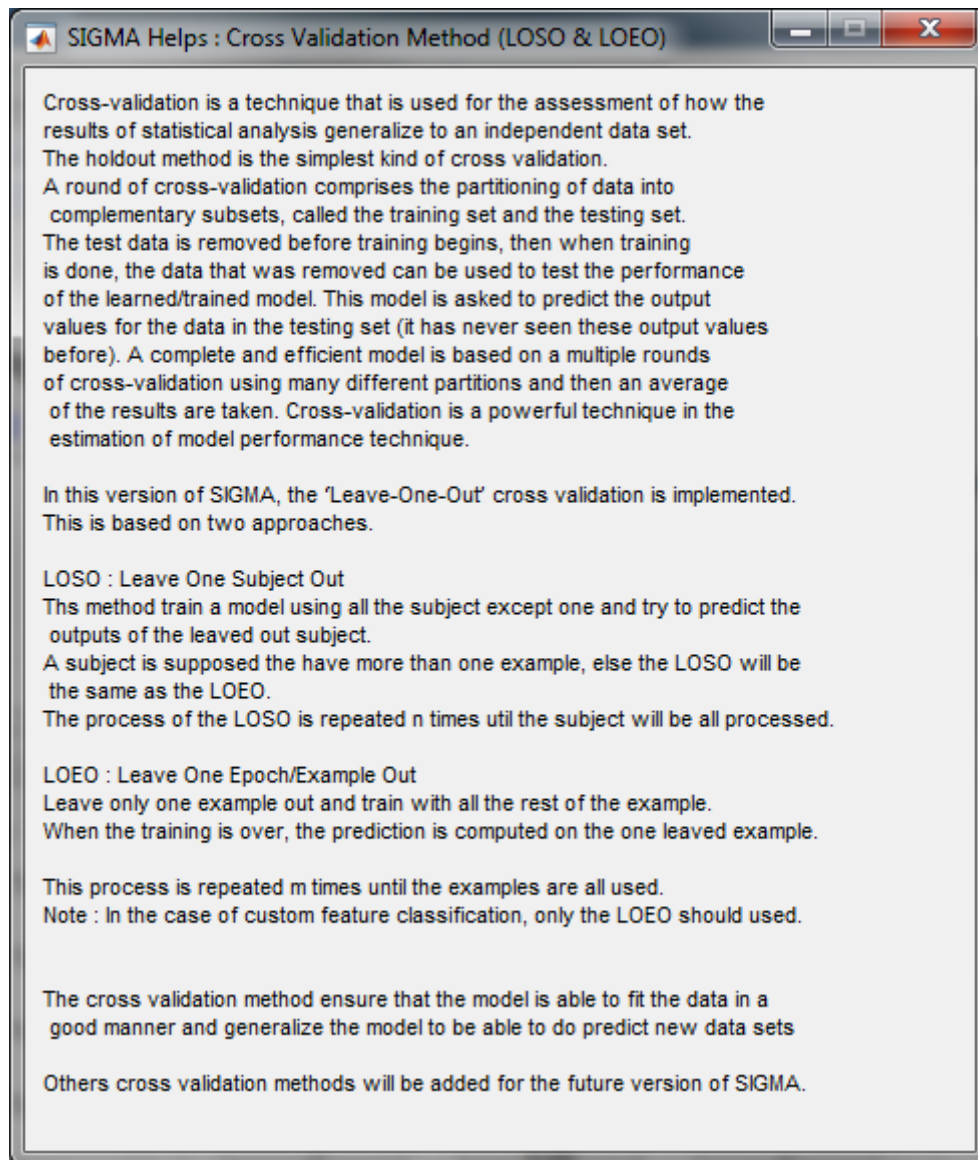


This process is repeated **m** times until the examples are all used.

Note : In the case of custom feature classification, only the LOEO should be used.

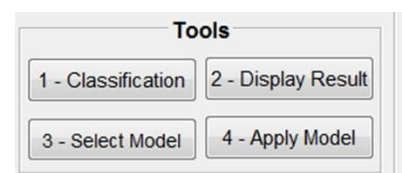
The cross validation method ensure that the model is able to fit the data in a good manner and generalize the model to be able to do predict new data sets. Others cross validation methods will be added for the future version of SIGMA.

When the user click on the 'Helps' button, this panel displays on the screen



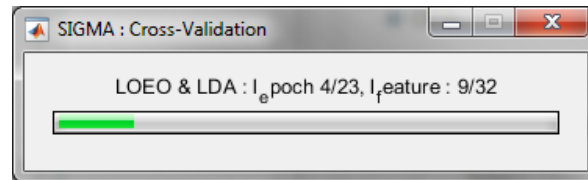
3.6.Tools

This Panel of is part of the “Data Classification”, in the following section, the functions of each pushbutton is explained.



3.6.1. Classification

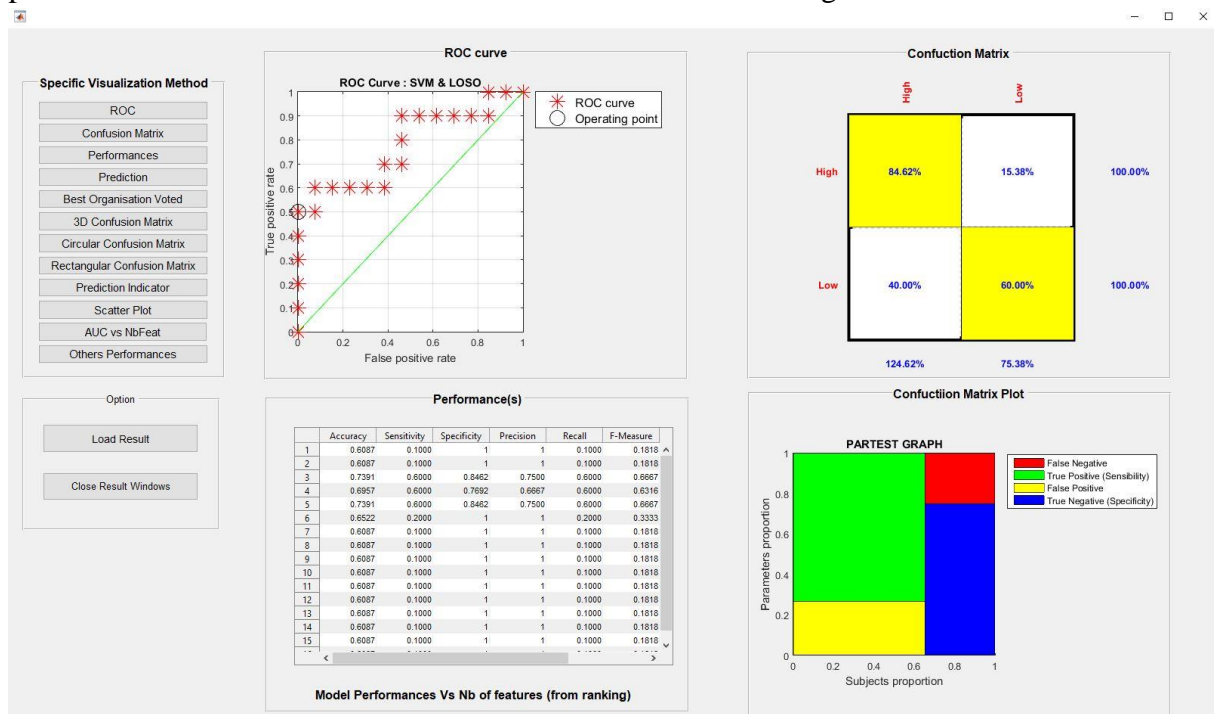
This button allows the user to compute the classification using the selected feature matrix, classification method and the cross-validation method. This computation can take a lot of time according to the different parameter cited above. A waitbar will appear and information related to the process is displaying during all the process.



The classification is computed using a loop over all the selected feature. For each loop prediction on the datasets is done, at the end of the cross validation , all the outputs are predicted. From these score and results are computed to evaluate the performance of the classifiers models.

3.6.2. Display Result

Once the classification has been computed, SIGMA computes a set of measures. These measures are computed using the expected outputs (labels) and the predicted outputs. The results can be visualized by clicking on the “Display Result” button. This opens another panel that allows the user to visualize results as shown in this figure:



This panel provides an estimate of how the classifier should perform when user apply it on new data. The list of methods on left side of the panel allow the user to select additional visualizations. The main panel displays these four criteria.

- **ROC : Receiver Operating Characteristic**, is a graphical plot that illustrates the ability of a binary classifier system as its discrimination threshold is varied. The ROC curve is created by plotting the true positive rate (TPR) against the false positive rate (FPR) at various threshold settings. The true-positive rate is also known as sensitivity, recall or probability of detection [1] in machine learning. The false-positive rate is also known as the fall-out or probability of false alarm[1] and can be calculated as $(1 - \text{specificity})$.

- **Confusion Matrix** : a confusion matrix, also known as an error matrix,[4] is a specific table layout that allows visualization of the performance of an algorithm, typically a supervised learning one (in unsupervised learning it is usually called a matching matrix). Each row of the matrix represents the instances in a predicted class while each column represents the instances in an actual class (or vice versa).[2] The name stems from the fact that it makes it easy to see if the system is confusing two classes (i.e. commonly mislabelling one as another).
- **Performance** : displaying table with various characteristics measure versus the number of feature used by the model
 - Specificity :
 - Precision :
 - Recall
 - F-Measure
 - G-Mean
 - AUC

Patient: positive for disease

Healthy: negative for disease

True positive (TP) = the number of cases correctly identified as patient

False positive (FP) = the number of cases incorrectly identified as patient

True negative (TN) = the number of cases correctly identified as healthy

False negative (FN) = the number of cases incorrectly identified as healthy

Accuracy: The accuracy of a test is its ability to differentiate the patient and healthy cases correctly. To estimate the accuracy of a test, we should calculate the proportion of true positive and true negative in all evaluated cases. Mathematically, this can be stated as:

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN}$$

Sensitivity: The sensitivity of a test is its ability to determine the patient cases correctly. To estimate it, we should calculate the proportion of true positive in patient cases. Mathematically, this can be stated as:

$$\text{Sensitivity} = \frac{TP}{TP+FN}$$

Specificity: The specificity of a test is its ability to determine the healthy cases correctly. To estimate it, we should calculate the proportion of true negative in healthy cases. Mathematically, this can be stated as:

$$\text{Specificity} = \frac{TN}{TN+FP}$$

		True condition			
Total population		Condition positive	Condition negative	Prevalence = $\frac{\sum \text{Condition positive}}{\sum \text{Total population}}$	Accuracy (ACC) = $\frac{\sum \text{True positive} + \sum \text{True negative}}{\sum \text{Total population}}$
Predicted condition	Predicted condition positive	True positive , Power	False positive , Type I error	Positive predictive value (PPV), Precision = $\frac{\sum \text{True positive}}{\sum \text{Predicted condition positive}}$	False discovery rate (FDR) = $\frac{\sum \text{False positive}}{\sum \text{Predicted condition positive}}$
	Predicted condition negative	False negative , Type II error	True negative	False omission rate (FOR) = $\frac{\sum \text{False negative}}{\sum \text{Predicted condition negative}}$	Negative predictive value (NPV) = $\frac{\sum \text{True negative}}{\sum \text{Predicted condition negative}}$
		True positive rate (TPR), Recall, Sensitivity, probability of detection = $\frac{\sum \text{True positive}}{\sum \text{Condition positive}}$	False positive rate (FPR), Fall-out, probability of false alarm = $\frac{\sum \text{False positive}}{\sum \text{Condition negative}}$	Positive likelihood ratio (LR+) = $\frac{\text{TPR}}{\text{FPR}}$	Diagnostic odds ratio (DOR) = $\frac{\text{LR+}}{\text{LR-}}$ $F_1 \text{ score} = \frac{2}{\frac{1}{\text{Recall}} + \frac{1}{\text{Precision}}}$
		False negative rate (FNR), Miss rate = $\frac{\sum \text{False negative}}{\sum \text{Condition positive}}$	True negative rate (TNR), Specificity (SPC) = $\frac{\sum \text{True negative}}{\sum \text{Condition negative}}$	Negative likelihood ratio (LR-) = $\frac{\text{FNR}}{\text{TNR}}$	

- Sensibility
- Accuracy
- True Positive
- True Negative
- False Positive
- False Negative
- Confusion Matrix Plot : Plot representation of the confusion matrix.

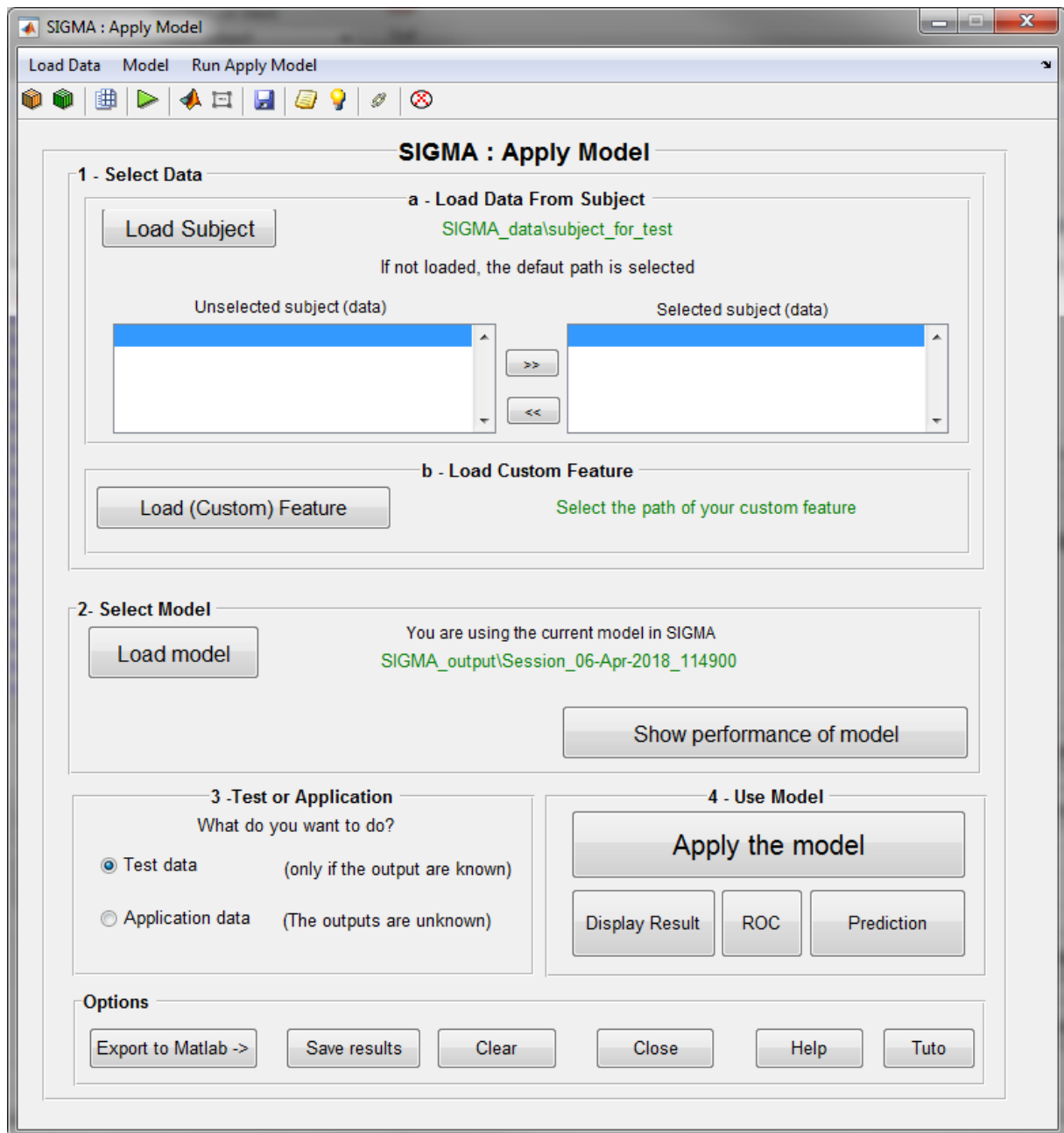
Note : The evaluation may depend heavily on which data used in the training set and which end up in the test set, and thus the evaluation may be significantly different depending on how the division is made.

3.6.3. Select Model

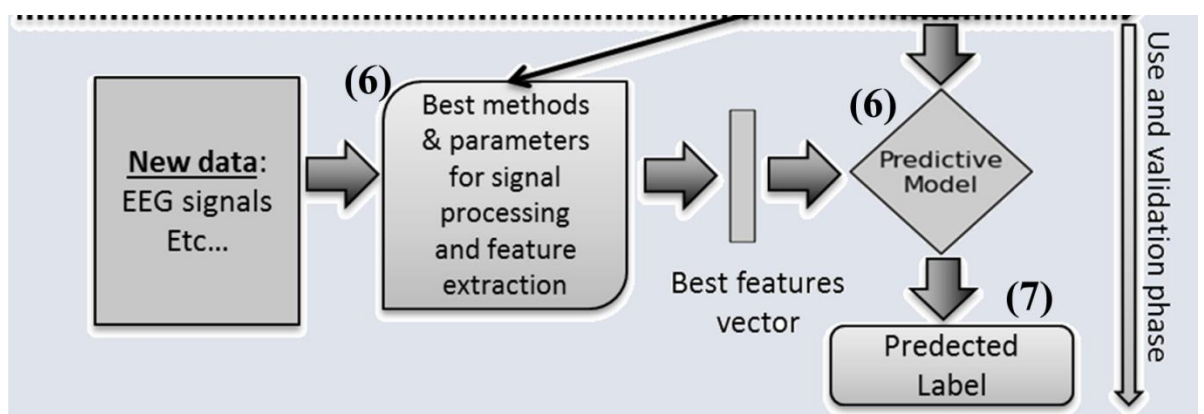
From all models computed during the classification step, based on the number of feature taken in account, the best model can be selected by clicking on the “Selected Model” button. The selection of the model is based according to the best AUC. Then the Apply model button is enabled.

3.7. Apply Model

When user select his model according to the best AUC, the Apply Model push button is activated. New panel could be displayed from this button allowing the user to use the selected model on new data, apply the selected model and visualize results.



The process implemented in this section follow this scheme :



(6) Predictive model : The trained model obtained from the classification method and the chosen method for the feature extraction. User can compare the performance of the different classifiers, and display various parameters (sensitivity, specificity, ROC curves...)

(8) The prediction/classification : Results (predicted labels) of the classification of new data using the predictive model obtained from the training/validation phase.

3.7.1. Select data

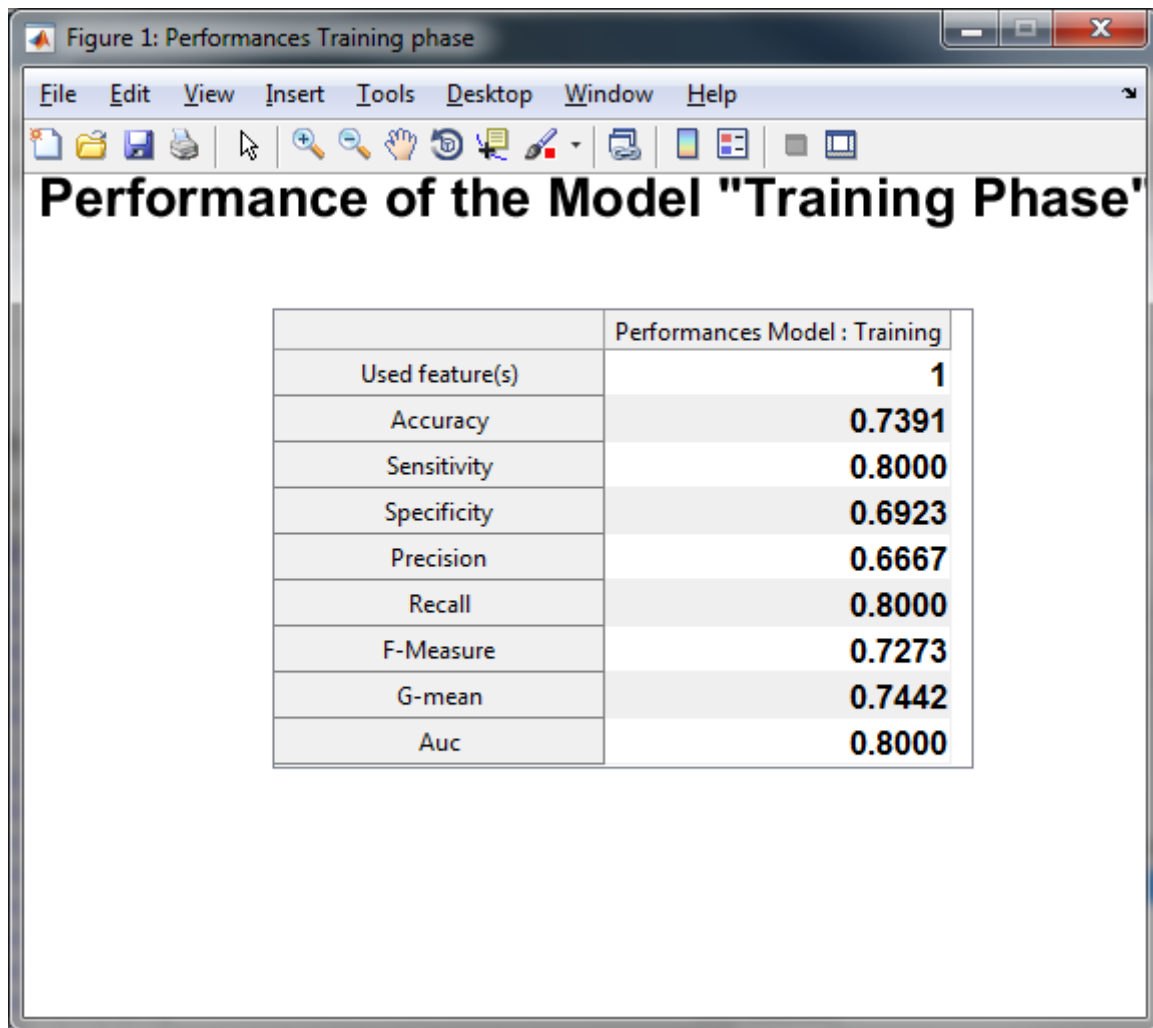
To apply the model, user can directly use the default parameters, in which the default path of new subject is pointed to the directory : ‘Sigma_data\subject_for test’ in which the user can put the subject to test or apply his model. If the subject are not located in this path, user can choose his own path using the ‘Load Subject’ push button, then the full list of available subject will be displayed on the listbox. From this list, user can select the desired subject(s) by double clicking on subject names or using the right and left arrows push button.

If user is dealing with custom feature, the user can load the feature to classify with the ‘Load (Custom) Feature’.

3.7.2. Select Model

If a best model has been chosen in the principal GUI, it is automatically loaded in the Apply Model GUI. Otherwise, the user can load a session containing the desired model he wants to apply.

The “Show performance of model” displays general informations on the model performance.



3.7.3. Test or Application

User can either use the model for test data or application data. In the case of the test data the output (the labels) are known and then the prediction are done, in this case performance on the test data are computed. If user select application data, tu output are unknown and in this case prediction are computed on these data.

Test data : as data already have labels, the Toolbox will compute the test error rate to assess the performance of the model on new data. This correspond to the third step, after Training and Validation.

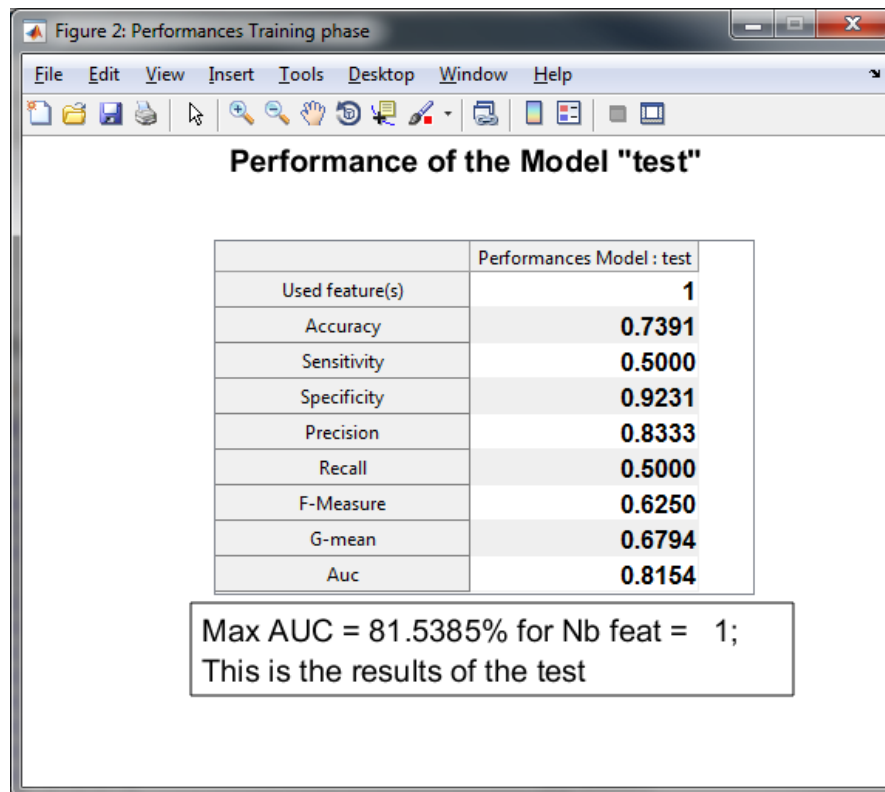
Application data : the Toolbox will apply the model on new data to compute the predicted labels. This correspond to the true application of the model on new data.

The "Apply the model" button launch the computation on the model on new data.

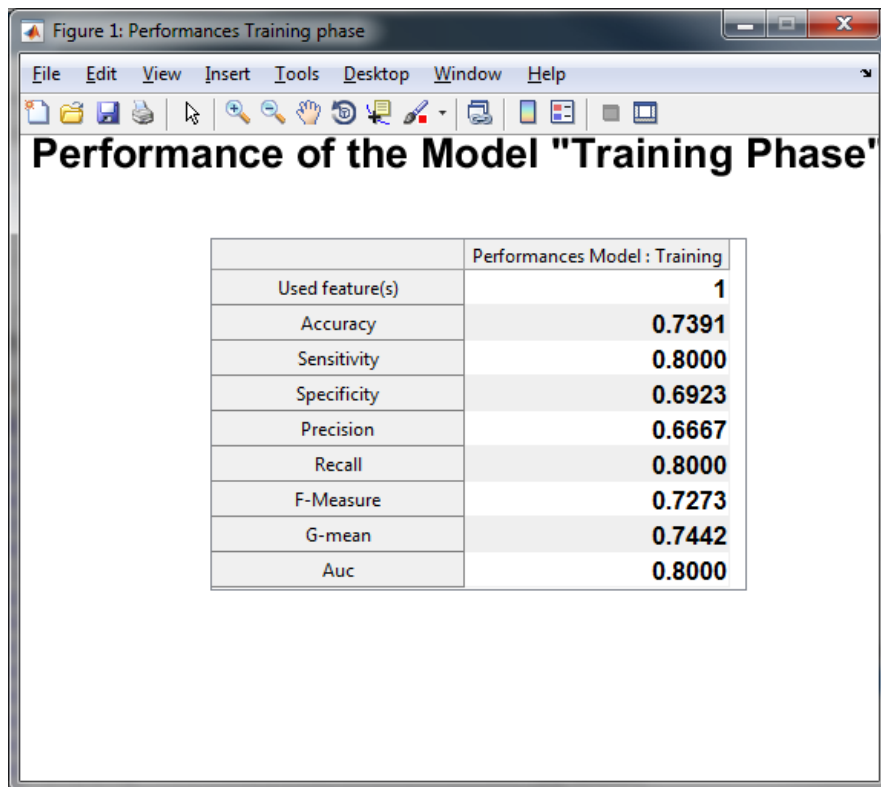
3.7.4. Use the Model

In this panel, user can launch the computation of the apply model on the selected data. From this user can also displays the different results once the computation is done.

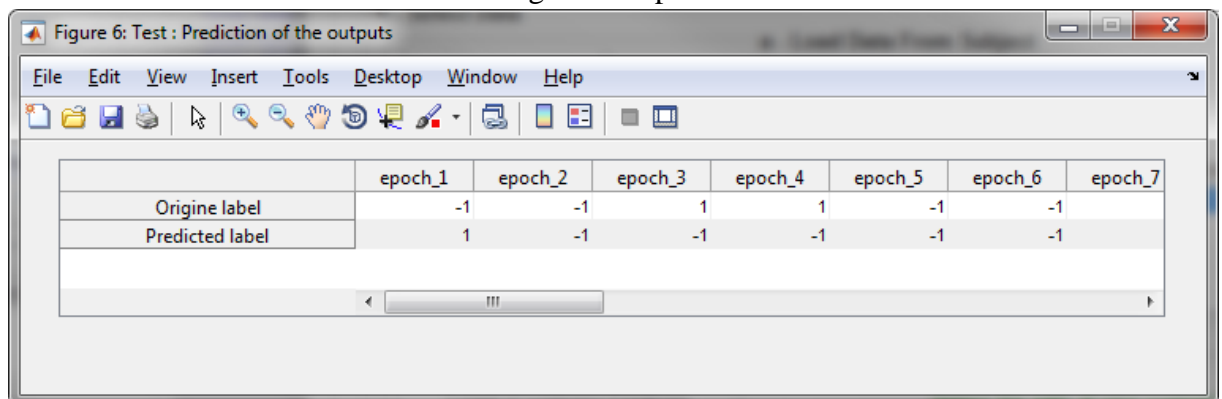
- Display result : This display a table with the performance of the classifier,
 - if user selects 'test data', in this case performance are computed on these data



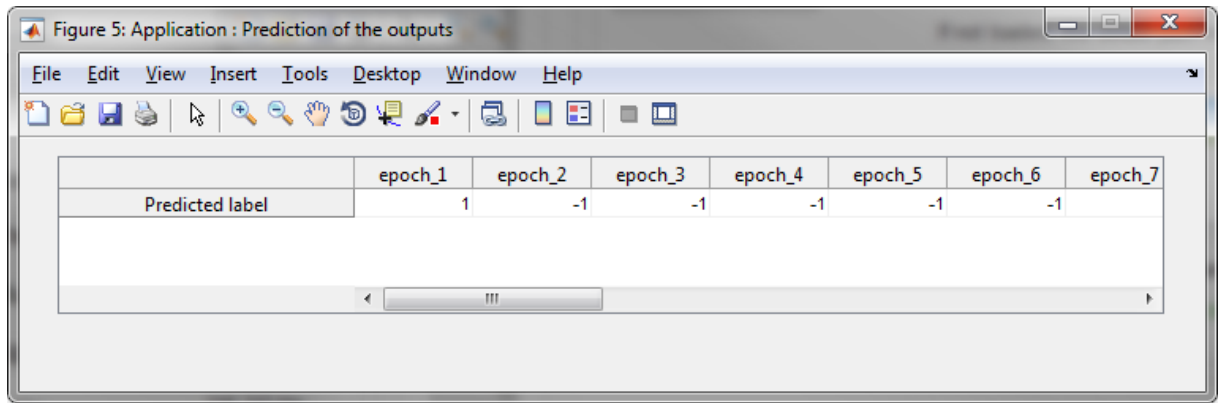
- if user selects application data', in this case the training performance are shown on the figure



- ROC : This display a roc curve resulting from the application of the data. As in previous, if user select test data, the roc curve is the results of the test data otherwise the curve displays the results of the training.
- Prediction : This display the class prediction, either for the test or the application data.
 - For test data a table of original and predicted labels is shown

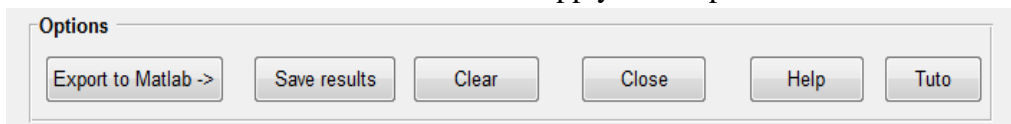


- for application data, only a table of the prediction is displayed.



3.7.5. Option of the Apply Model Panel

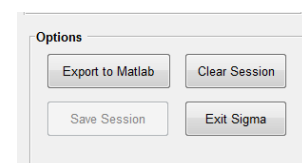
This panel contains some function related to the Apply model panel.



- Export to matlab : this export to matlab the structure of data used by the apply model panel
- Save results : save the differents structure of SIGMA in the session directory
- Clear : clear all the session loaded on the panel
- Close : close the Apply model Panel
- Help ; display a quick help how to use the panel
- Tuto : open a pdf file containing the tutoriel about the Apply model panel

3.7.6. Option of Main Sigma Panel

- **Export to Matlab** : At any moment during the use of the toolbox, the user can export the data and model structure to the MATLAB workspace for close investigation.
- **Clear Session** : The “Clear session” button will delete all settings and data loaded in the current session.
- **Save Session** : At any moment during the use of SIGMAbox, the user can save its current session by clicking on the “Save session” button. A session can be loaded via the Menu/Load session.
- **Exit Sigma** : By clicking on this button, the use choose to exit SIGMAbox and all setting not saved through the “Save session” button.



Menu bar

Files

This menu allows to create load or save sessions.

Settings

This menu allows to close all waitbar, install the toolbox files or exit the toolbox.

Other windows

This menu allows to open all visualization panels available in the Toolbox :

- Data Visualization
- Feature Ranking Results
- Display Results
- Apply Model

Edit

This menu allows to open all options panels available in the Toolbox :

- Frequency Bands edit
- Feature Extraction methods
- Feature Selection methods
- Classification methods

Help

This menu allows to access to user manual and tutorials

About Sigma

1. Toolbar

New session

Allows to create a new session, erasing the current not saved data and parameters

Load Session

Allows to load an existing session.

Save Session

Allows to save the current session

Clear Session

Clear the current session, but do not create new one.

Load EEG data

Load Custom Feature

Export to Matlab

Export all data structure to MATLAB for easy inspection by the user

Print Screen

How to Use Sigma

Manual of Sigma

Tutorial of Sigma

About Sigma

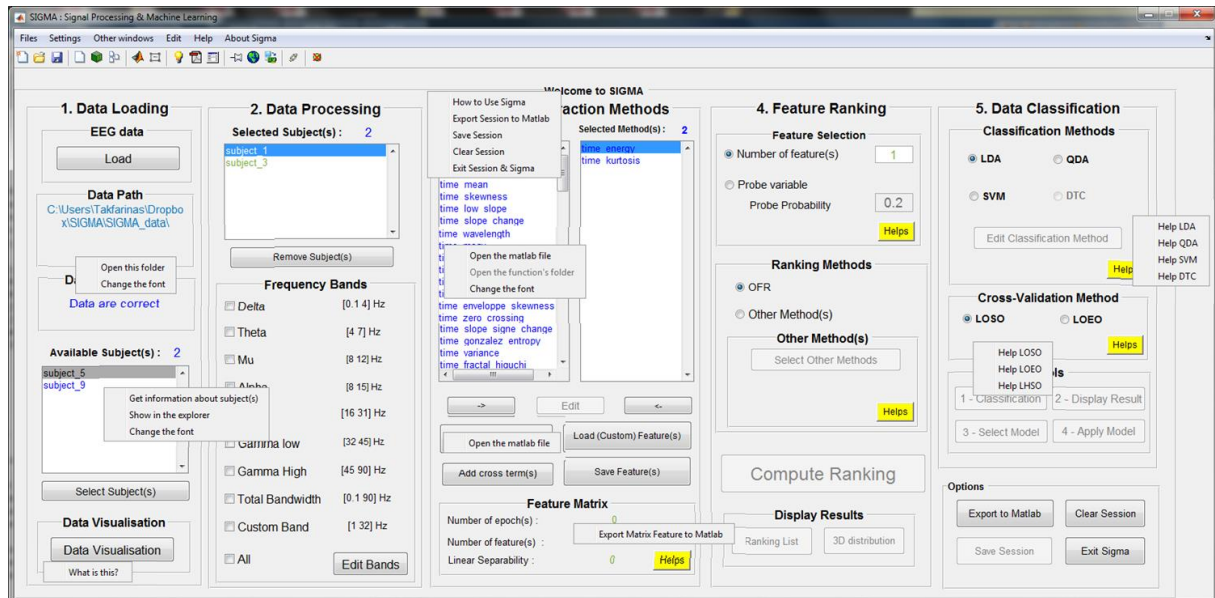
Visit SIGMA Web Page

Contact the developers team

Close all wait bar

Exit Sigma

2.



3. Definitions

GUI : Graphical User Interface

Data format

K-fold cross validation is one way to improve over the holdout method. The data set is divided into k subsets, and the holdout method is repeated k times. Each time, one of the k subsets is used as the test set and the other k-1 subsets are put together to form a training set. Then the average error across all k trials is computed. The advantage of this method is that it matters less how the data gets divided. Every data point gets to be in a test set exactly once, and gets to be in a training set k-1 times. The variance of the resulting estimate is reduced as k is increased. The disadvantage of this method is that the training algorithm has to be rerun from scratch k times, which means it takes k times as much computation to make an evaluation. A variant of this method is to randomly divide the data into a test and training set k different times. The advantage of doing this is that you can independently choose how large each test set is and how many trials you average over.

Quick tutorial in order to use the Apply model GUI

To use correctly this panel (apply model), you are supposed to already computed the feature, the CV and selected the best model from the main GUI of SIGMA (the button Select best

model), the selected model is chosen according to the best AUC. Then you click on the Apply Model Button.

This figure contain many functionalities,

1-Select your model: You can load the model that you want to apply, but by default, and to continue the previous process, the selected model is loaded in the panel. You can display the performance of you model by clicking on the push button (Show Performance of the model) , this figure shows the results of the training phase.

You can load also your model, but it should respect the SIGMA format, see the structure of selected_model variables, by exporting the results to matlab.

2-Select your data : Here you can select the data on which you want to apply the model, by default the selected data are located on the SIGMA_data\subject_to_test, you are supposed to put your test data in this path, otherwise you can load your own path of data.

3-Use the model : Here you are going to choose your simulation,

-: you can apply the model on data for validation (in this case the data should have an output label) and performance of validation will be computed on the new data.

-: you can apply the model for for new and unknown data, (no label), in this case you are going to predict the label of the data.

Use model panel : This is the core of the computation, the selected model is applied on the selected data, the feature extraction and classification is computed. The results of the application can be displayed using the pushbutton on he panel

Subsequently, in order to select the optimal number of features from their ranked list, the random probe method is applied. A set of “probes”, i.e. realizations of random variables, are computed and appended to the feature set. A risk level P is defined, which corresponds to the risk that a feature might be kept although, given the available data, it might be less relevant than the probe.

reference : http://scikit-learn.org/stable/modules/cross_validation.html#leave-one-out-loo

The parameters for the implemented methods are initialized according to the best ones found in the literature and validated on our databases. However, some of the hyper parameters can be chosen by the user if necessary. Advanced users can change parameters and add their own contributions.