# Vobi One v1.02 user's manual

get support at int-support-vobione[at]univ-amu.fr

December 2013

# Chapter 1

# Setting up *Vobi One*

## 1.1 Installing and getting familiar with BrainVISA

BrainVISA can be downloaded from the dedicated website at `http://www.brainvisa.info`. Installation instructions come with the downloaded file, and are also available in the BrainVISA handbook at `http://brainvisa.info/doc/axon/bv_man/en/html/`. We urge users to read this handbook to get familiar with the BrainVISA environment before going further with *Vobi One*.

Note that the tests and validation of *Vobi One* were performed under a 64 bit linux operating system using BrainVISA V4.3.0. Therefore, using the same version of BrainVISA is mandatory. We also recommend a 64 bit linux operating system to use *Vobi One*, even if there is a high chance that everything should work fine with others.

To install BrainVISA, simply download the archive into the directory of your choice and unpack it.

```
> cd /mydirectory
> tar xvfj brainvisa-Mandriva-2008.0-x86_64-4.3.0-2012_09_03.tar.bz2
```

We recommend that you choose `/mydirectory` to have the full permissions to read and write what you need (for instance, it can be your home directory `/home/login`). In particular, we recommend not to install BrainVISA system-wide as root, which will ease some further steps. Once you have executed this `tar` command, the file `brainvisa-Mandriva-2008.0-x86_64-4.3.0-2012_09_03.tar.bz2` is no longer needed; you can remove it, unless you want to reinstall BrainVISA.

A directory called `brainvisa-4.3.0` will be created. We will refer to `/mydirectory/brainvisa-4.3.0` as the BrainVISA installation directory. You will find installation instructions and other useful information in a `README` file in that directory. To launch BrainVISA, simply run the corresponding executable by typing its full path. Add the & sign at the end to keep an active shell prompt (optional).

```
> /mydirectory/brainvisa-4.3.0/bin/brainvisa &
```

## 1.2 Installing *Vobi One*

*Vobi One* can be downloaded from the dedicated website at `https://trac.int.univ-amu.fr/vobi_one`. Once you have downloaded the toolbox file into `/mydirectory`, the first thing to do is to unpack the downloaded archive:

```
> cd /mydirectory
> tar xvfz vobi_one_vX.XX_cYYY.tgz
```

This will create a directory called `vobi_one_src`. To install *Vobi One*, you simply need to go to this directory, and then copy some content inside the BrainVISA installation directory.

```
> cd /mydirectory/vobi_one_src
> cp -rf lib/ brainvisa/ /mydirectory/brainvisa-4.3.0
```

To check that the installation was successfull, you should restart BrainVISA. You will be asked to update a database, and you should proceed. Then make sure that the *Vobi One* icon now appears in the toolbox panel on the left side of the main BrainVISA interface, as illustrated on Fig. 1.1.

Finally, you need to update the documentation of BrainVISA to include the *Vobi One* documentation. This process, which will take several minutes, can be launched by typing the following command:

```
> /mydirectory/brainvisa-4.3.0/bin/brainvisa --noMainWindow --updateDocumentation
```

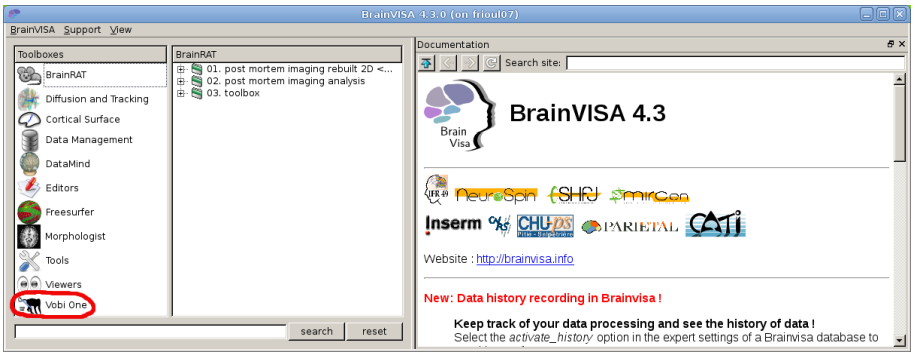


Figure 1.1: If the installation of *Vobi One* is successfull, you should see the icon circled in red.

Once you have done all this successfully, you can clean up some files and directory that are not needed anymore (unless you want to reinstall *Vobi One*):

```
> rm -rf /mydirectory/vobi_one_src
> rm /mydirectory/vobi_one_vX.XX_cYYY.tgz
```

## 1.3 Setting up BrainVISA to use Vobi One

Before being able to use *Vobi One*, you need to define a BrainVISA database. We recommend reading the documentation of the *Data Management* toolbox available in the BrainVISA interface to learn more about BrainVISA databases.

In order to create a database, the first time you launch BrainVISA, go to the following menu item: BrainVISA - Preferences - Databases - Add..., and select (or create) the root directory for your database (let's call it `/brainvisadatabase`). Make sure this directory is located on a ressource where enough space is available to store all the raw data, the intermediary files and the results of your future analyses.

You are now ready to go!

## 1.4 Preparing the demo dataset

A demonstration dataset is available at `https://trac.int.univ-amu.fr/vobi_one`. Once you have downloaded it, unpack the archive in the directory of your choice. We strongly recommend that this should not be in the root directory you defined for your BrainVISA database.

```
> cd /rawdatadirectory
> tar xvfj vobi_one_demo_data.tar.bz2
```

# Chapter 2

# Tutorial: running the linear model with the GUI

This is the easiest tutorial which only uses the Graphical User Interface (GUI) of *Vobi One*, as integrated in the main BrainVISA interface. Each of the following steps correspond to launching one of the processes available in *Vobi One*.

## 2.1 Import / Import BLK File

This process imports the data of a trial stored in BLK format into the BrainVISA database, and converts it to a Nifti file. Open the process and follow these instructions:

- Click on the file browser to select the *input* file. Choose the BLK file which is in the `/rawdatadirectory/vobi_one_demo_data/single_blk_file` directory.
- Click on the red icon to select the *subject*, and then:
  - in the top-left panel, use the filter on attributes to choose the *database* that you previously defined by selecting it in the available drop-down list
  - you do not need to specify anything for *Data type* and *File format* entries
  - define the *protocol* name by directly typing in the textfield (for instance: `protocol_tutorial01`)
  - define the *subject* name by directly typing in the textfield (for instance: `subject_tutorial01`)
  - a single entry should appear in the top-right panel; to choose it, select it (single click) and press OK (bottom-right); or you can also double click on it;
- Choose the file *format* (compressed or uncompressed Nifti); use uncompress for now;
- For now, you can leave the default values for sampling *period*, *temporal binning* and *spatial binning*.

You will see that the *ouput* file name is automatically defined from the choice you have made. Now you can simply click *Run* to launch the process. The file will be imported in the BrainVISA database and stored at a precise location in the directory architecture.

## 2.2 Session Pre-analysis / Construct model

This process constructs all the regressors that will be used in the multiple regression, i.e it builds the linear model that will be used to fit the data. Here, we will use a set of parameters that are given in a file that comes with the demo data. Here is a short description of the parameters

The parameters that define the models are :

- the *sampling frequency*, in Hertz, after importing (do not forget to take into account the temporal binning used when importing the data);
- the *trial duration*, in seconds;
- the value of *tau* : dye-bleaching time constant, in seconds;
- the *frequencies*, in Hertz, of the oscillatory noise components you want to model;

- the *Fourier orders*, i.e the number of harmonics of the Fourier serier used to model each oscillatory noise components; the length of this vector should be equal to the number of frequencies listed above;
- $L$ is the number of regressors that will be used to model the shape of the response and its potential variations, which are defined by the alpha parameters below
- alpha-min[sec] : minimum values for the alpha parameters (see Chapter 4 for their definition)
- alpha-max[sec] : maximum values for the alpha parameters The sum of the first six values should be lower than the trial duration. The seventh and eighth values are expressed relative amplitude with respect to 1.

Now, let's open the process window itself.

- Click on the file browser (and not the green icon) to select the *parameters*. Choose the `param.npz` file which is in the `/rawdatadirectory/vobi_one_demo_data/lm_parameters` directory. This will fill in all the parameters that define the model.
- Click on the red icon to select the *output*, and then:
  - select the *database*, *protocol*, *subject* and *session_date* from the corresponding drop-down lists (with the same values you defined when importing the data);
  - define the *secondlevel_analysis* name by directly typing in the textfield (for instance: `_model1`)
  - a new entry should appear in the top right panel; select it;
- Click *Run* to launch the process.

Once the process is finished, you can visualize a summary of the model by clicking on the eye icon on the *output* line. In particular, the bottom figure displays some representative example shapes that can be taken by the neural response using this model. Click once again on the eye icon to make the figure disappear.

## 2.3 Trial Analysis / LM Based Denoising / Apply Linear Model

This process fits the linear model to a given trial. It estimates the model parameters for the timeseries at each pixel of the image, and produces a graph to display the result of this estimation.

- Click on the green icon to select the *glm*. Either you can select it from the list of available *OI GLM Design Matrix* entries in the top right panel (for now, it is the easiest way since only one should be there; just double click on it!); or, if a lot of them are availeable, you can use the top left panel to filter what appears in the top right panel, by choosing the *database*, *protocol*, *subject*, *session_date* and finally *secondlevel_analysis*;
- Click on the green icon to select your input *data* file, and use the same filtering strategy before only one *OI Raw Imported Data* file appears in the top right panel;
- Choose the *format* file of your output data (NIFTI-1 image by default, but it can be compressed; it is recommended to keep the same one all along an analysis)
- Choose the region of interest (*ROI*) for which a figure will be generated, presenting the components of the estimated model averaged on all pixels of this region. You can specify your *ROI* in two ways:
  - as a *Rectangular ROI (from coordinates)*; in that case you have to specify the (x,y) coordinates of the bottom-left corner (*corner0*) and the top-right corner (*corner1*) of the rectangle; try it with $corner0 = (125, 250)$ and $corner1 = (200, 300)$;
  - as a *Binary mask (from image)*; in that case you need to specify a Nifti image which is a binary mask of your region of interest (1 within the ROI, 0 outside); this mask need to have the same size as your data; try it by choosing, with the file browser (and not the green icon) the mask `/rawdatadirectory/vobi_one_demo_data/roi_masks/region1.nii`.

Once you have run the process, you can visualize this *data_graph* (use the eye icon) to check whether the model you have defined fits the data correctly in this region of interest.

# Chapter 3

# Tutorial: looping over all trials of a session

The *iterate* capability of BrainVISA makes it possible to repeat the same operation on numerous files. This tutorial aims at demonstrating how to use it by i) importing all the files of a given session, and ii) running the same linear model on these trials. In all *processes*, the *iterate* button is present next to the *run* button. In order to use it, the principle is to click on the *iterate* button at the right time, i.e only once everything that's common to all the files that have to be processed has been set-up.

Here is an example, step by step, to import several files and run the same linear model on all the imported trials.

## 3.1 Import / Import BLK File

Here, we will run the same *process* as in Chapter 2, but using the *iterate* function to set-up a loop. First, open the *Import BLK File* window, and then:

- DO NOT click on the file browser to select the *input* file. This is what will be iterated.
- Click on the red icon to select the *subject*, and do it as described in Chapter 2, except change something so that the data ends up being imported in a different place (for instance, use *protocol* = `protocol_tutorial02` and/or *subject* = `subject_tutorial02`);
- Choose the file *format* (compressed or uncompressed Nifti), sampling *period*, *temporal binning* and *spatial binning* as previously.

At this point, everything that is common to all the files has been set-up, it is therefore time to click on the *iterate* button. Here you find yourself in front of a new window with the same entries as before. Do not touch anything that you have just selected (even if not visible, it has already been set and will be used for all files). Your goal is to iterate (i.e loop) over data files, so open the file browser for *input*; go to `/rawdatadirectory/vobi_one_demo_data/raw_blk_data`, select all the files present in this directory and press Open. Validate the different windows by clicking OK twice (without changing anything) until you can Run the iteration process. This will import all files of this session. You will be able to follow the execution of all the processes, and, by clicking on one of them, you will be able to access its detailed inputs and ouputs, and eventually use the different viewers from there.

## 3.2 Session Pre-analysis / Create Conditions File

This process will create the Conditions File, a text file that summarizes the information about all trials that have been imported in the database for a given session. There is just one entry to fill up: click on the red icon and select the *database*, *protocol*, *subject* and *session_date*; select the file in the right panel and run the process. Once the file has been created, you can visualize it by clicking on the eye icon. It contains the file name of all the imported files for this session, the experience number, the trial number, the experimental condition. The last column, which is by default filled with ones, allows to discard a trial if needed (for example because it is corrupted by large artefacts that prevent using it in further interpretation of the dataset); for now, in order to do so, you need to manually edit the `conditions.txt` file and replace this one by a zero. In this tutorial, you can leave this column filled with ones.

## 3.3 Session Pre-analysis / Construct model

We will here use the same model parameters as in Chapter 2. Note two differences:

- now that a previous model has already been run, you can use the green icon to select it from the database (or use the file browser to retrieve it as in Chapter 2);
- when defining the *output*, make sure you use the parameters (*database*, *protocol*, *subject* and *session_date*) for the current study and not the one from Chapter 2.

The model is now defined at the session level, and will therefore be available for all trials of this session.

## 3.4 Trial Analysis / LM Based Denoising / Apply Linear Model

We will iterate this process to fit the model for all trials that have just been imported.

- Click on the green icon to select the *glm* that you just created with the Construct model process;
- Define your region of interest.

At this point, everything that is common to all the files has been set-up, it is therefore time to click on the *iterate* button. Use the green icon to select the *data* files from the database. Use the top left-panel to filter the results displayed in the top-right panel (select the (*database*, *protocol*, *subject* and *session_date*), until only the raw data you have just imported can be selected. Proceed with your selection in the top-right panel, and click on Ok. Note that before running the iteration process, you can examine the individual processes by click on them; this is a good way to check that everything is set up correctly before launching a large amount of computation. Once you've checked that everything is OK, launch the iteration by clicking on the Run button.

## 3.5 Session Post-Analysis / Visualization of Trials variability

This process will create a figure displaying the set of all estimated single-trial denoised responses for the session we are currently working on, or for a subset of trials corresponding to a given list of specified experimental conditions.

- Click on the green icon and select the Conditions File created previously, which corresponds to the desired session, by using the left panel as filters;
- Choose the *model* (the default is Linear Model, which is what we want here);
- Select the *analysis_name* from the drop-down list (it should be the *secondlevel_analysis* you defined above);
- Define the *conditions_list*, i.e the list of experimental conditions for which you want to see the results displayed; try it with [6] to only display the trials from condition 6, or with [1,2,3,4,5,6] to display the trials for these six conditions;
- Define your region of interest as you now know how to do, and Run the process.

Once finished, click on the eye of the *data_graph*. You will see the averaged denoised response for each trial that belongs to the list of experimental conditions you chose, and their mean in red.

## 3.6 Session Post-Analysis / Comparison of ROIs

This process will create a figure displaying the average denoised responses in different regions of interest, for a given analysis and a chosen list of experimental conditions.

- As in the previous process, select the Conditions File with the green icon, choose the *model* type (Linear Model), the *analysis_name* and the *conditions_list* (try it with [5,6] for instance);
- Select the type for *ROI_1* (mandatory, because you need at least on region!): choose Binary mask, and select `v1center.nii` as previously;
- If you now select the type for *ROI_2*, it will add another region to the analysis: choose Binary mask, and select `v1periphery.nii`;
- Now select the type of *ROI_3* as Rectangular ROI and set $corner0 = (125, 250)$ and $corner1 = (200, 300)$.

Now, launch the process, and visualize the resulting figure with the eye icon: you will see a comparison of the mean denoised responses for these three regions.

# Chapter 4

# Tutorial: setting up and running the linear model with scripts

In this tutorial, we will use the provided example scripts to run a similar analysis as the one that was performed in Chapter 3. The demo scripts are available in `/rawdatadirectory/vobi_one_demo_data/example_scripts`. To run a script, just follow these instructions (which are available in the documentation of the script, at the beginning of the file):

- copy the script in the directory of your choice and go to this directory
- change the parameters in the "PARAMETERS TO BE DEFINED BY THE USER" section of the script
- launch BrainVISA in shell mode from the unix prompt by typing `/mydirectory/brainvisa-4.3.0/bin/brainvisa --noMainWindow --shell`
- run this script in the BrainVISA python shell by typing `%run scriptname.py`

## 4.1 Running `script0_import_and_preanalysis.py`

This script imports some raw data in .blk format, and estimates noise noise parameters from blank trials.

It successively calls four Vobi One processes:

- Import / Import BLK File
- Session Pre-analysis / Create Conditions File
- Session Pre-analysis / Spectral analysis
- Session Pre-analysis / Tau and heartbeat frequency estimation

Finally, it displays the graphs created by the two last steps

At this point, you already know what the first two processes do. The *Spectral analysis* process helps you identify peaks in the the Fourier spectrum of the timeseries that are induced by noise-related oscillatory components present in the raw signal. The *Tau and heartbeat frequency estimation* allows estimating the rate of the exponential decay in the timeseries (dye bleaching), as well as the strongest sinusodal component present in the signal (often caused by the heart beat, and not visible on the power spectrum because at a very low frequency). These two processes produces graphs from which you can extract noise parameters that will be use in the construction of the linear model in the next step.

You need to edit the section of the script called "PARAMETERS TO BE DEFINED BY THE USER", and in particular the locations of the raw .blk files and the BrainVISA database root directory. All other parameters are customizable. In particular, it is important to give meaningful names to the *protocol* and *subject*, and adequately use the binning parameters to optimize disk usage. Once this is done, just type `%run script0_import_and_preanalysis.py` in the BrainVISA python shell to launch the execution of the script.

## 4.2 Running `script1_estimate_linearmodel.py`

This script starts by constructing the design matrix of the model itself and then it fits the model on all imported trials.

It successively calls two Vobi One processes:

- Session Pre-analysis / Construct model
- Trial Analysis / LM Based Denoising / Apply Linear Model

Again, you need to edit the section of the script called "PARAMETERS TO BE DEFINED BY THE USER", filling in the same parameters as with the previous script. Several other parameters need to be chosen, like the *analysis_name* (for which it is, once again, important to give a meaningful name). After that, the *param* structure defines all the regressors that will be included in the design matrix of the linear model, as described in Chapter 2. In particular:

- the figure produced by the process *Tau and heartbeat frequency estimation* suggests a value for $Tau$, as well as the heartbeat frequency $F_h$ that you can enter in the list of frequencies; here we chose $tau = 0.7$ and $F_h = 2.0Hz$;
- the figure produced by the process *Spectral analysis* displays a list of frequencies for which there is a peak in the spectrum; you can include the largest of them; here, we chose to include $10Hz$, $28Hz$, $41Hz$ and $50Hz$.

The list of orders (here all at 1) is the number of harmonics you include in the Fourier series that models the periodic component for each frequency. You can increase it if you think the contribution of this frequency is not purely sinusoidal.

Finally, a set of eight $\alpha$ parameters define the shape of the neural response $r(t)$ as given by:

$$r(t) = \begin{cases} 0 & \text{if } t \leq \alpha_1, \\ \frac{\alpha_7}{2}(\cos(\pi\frac{t-\alpha_1}{\alpha_2}) - 1) & \text{if } \alpha_1 < t \leq \alpha_1 + \alpha_2, \\ \frac{1-\alpha_7}{2} - \frac{1+\alpha_7}{2}\cos(\pi\frac{t-\alpha_1-\alpha_2}{\alpha_3}) & \text{if } \alpha_1 + \alpha_2 < t \leq \alpha_1 + \alpha_2 + \alpha_3, \\ 1 & \text{if } \alpha_1 + \alpha_2 + \alpha_3 < t \leq \alpha_1 + \alpha_2 + \alpha_3 + \alpha_4, \\ \frac{1-\alpha_8}{2} + \frac{1+\alpha_8}{2}\cos(\pi\frac{t-\alpha_1-\alpha_2-\alpha_3-\alpha_4}{\alpha_5}) & \text{if } \alpha_1 + \alpha_2 + \alpha_3 + \alpha_4 < t \leq \alpha_1 + \alpha_2 + \alpha_3 + \alpha_4 + \alpha_5, \\ -\frac{\alpha_8}{2}(1 + \cos(\pi\frac{t-\alpha_1-\alpha_2-\alpha_3-\alpha_4-\alpha_5}{\alpha_6})) & \text{if } \alpha_1 + \alpha_2 + \alpha_3 + \alpha_4 + \alpha_5 < t \leq \alpha_1 + \alpha_2 + \alpha_3 + \alpha_4 + \alpha_5 + \alpha_6, \\ 0 & \text{if } t > \alpha_1 + \alpha_2 + \alpha_3 + \alpha_4 + \alpha_5 + \alpha_6 \end{cases}$$

The first six $\alpha$s are timing parameters (or lags), and the last two are relative amplitude values. The shape model is illustrated on Fig. 4.1.

You have to specify the minimum and maximum values for each $\alpha$ to define the range of shapes that can be taken by the neural response.

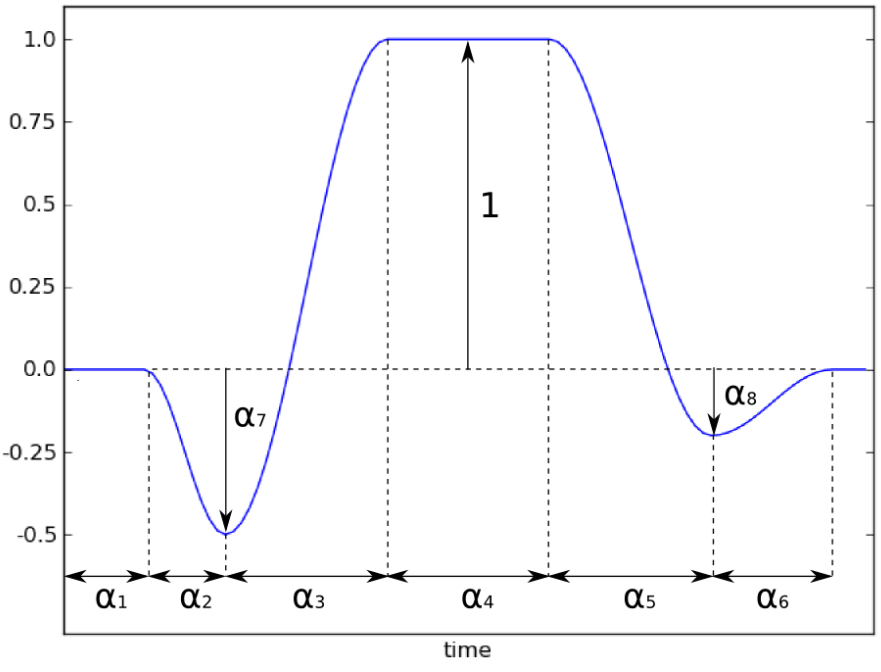


Figure 4.1: The model of the neural response shape. By varying the parameters, one can obtain a wide variety of shapes.

# Chapter 5

# Tutorial: running a standard blank subtraction analysis with scripts

In this tutorial, we will use the provided example script to run a standard analysis, the so-called *blank subtraction*. The script is also available in `/rawdatadirectory/vobi_one_demo_data/example_scripts`.

## 5.1 Running `script2_blank_subtraction.py`

This script performs all the steps needed to run a standard analysis, assuming the data has already been imported into the BrainVISA database and the Conditions File has already been created.

The different steps consist in calling the following Vobi One processes:

- Session Post-analysis / Average Trials, to average all the blank trials
- Trial Analysis / Blank Based Denoising / Frame0 Division, to perform this on the average of blank trials
- Trial Analysis / Blank Based Denoising / Frame0 Division, performed on each trial
- Trial Analysis / Blank Based Denoising / Blank Subtraction, performed on each trial
- Trial Analysis / Blank Based Denoising / Linear Detrend, performed on each trial

You need to edit the section of the script called "PARAMETERS TO BE DEFINED BY THE USER", and change the values of the same parameters as in the previous tutorials (the BrainVISA database root directory, the *protocol* and *subject*). Moreover, you need to define the time windows used to define the Frames0 (i.e frames before stimulus onset that will be used to define the baseline) and the Frames1 (i.e frames at the end of the trial when the signal is back to baseline level); these will be used for the Frame0 division and for the linear detrending operations. Once this is done, just type `%run script2_blank_subtraction.py` in the BrainVISA python shell to launch the execution of the script.

# Chapter 6

# Tutorial: comparing results from the linear model and the blank subtraction with the GUI

You have now fully processed the same dataset with two different methods, the linear model and the blank subtraction framework. We will here use a process that allows comparing the results obtained with these two methods.

## 6.1 Session Post-Analysis / Comparison of analyses

This process will create a figure displaying the average denoised responses in a single region of interest, for different analyses, or different options of the same type of analyses, or a different set of experimental conditions, or even data acquired in different sessions or on different days. In this tutorial, we will use the same set of experimental conditions with the two analysis methods that were ran in the previous tutorials.

- First start by defining the common ROI: for instance choose Binary mask for the type of ROI, and select the `v1center.nii` file;
- Select the first type of analysis (*model_1*) by clicking *Change to add a new analysis*; select *Linear model (GLM)* from the drop-down list
- Select the *conditions_file* with the green icon
- Select the *analysis_name* that you chose when running the linear model, and the *conditions_list* (try it with [5,6] for instance);
- Now repeat the same thing for the second type of analysis (*model_2*): click on *Change to add a new analysis*: select *Blank subtraction + Detreding (BkSD)* from the drop-down list
- Select same *conditions_file* as above
- Select the *analysis_name* that you chose when running the blank subtraction, and the same *conditions_list* as above.

Now, launch the process, and visualize the resulting figure with the eye icon: you will see a comparison of the mean denoised responses for given condition list between the two types of analyses. Note that you could have added more types of analyses.

# Chapter 7

# Practical advice

In this section, we give practical recommandations on how to best use *Vobi One* and additional information that could be useful. These recommandations come from the way we have been using it locally.

## 7.1 Naming conventions of .blk and .rsd files

To import files in .blk or .rsd format into *Vobi One*, their names has to respect the following conventions:

- a .blk file needs to be called like `tcXXDDMMYY??????_eEEbTTT.blk`
- a .rsd file needs to be called like `stmXX-MMDD-TTT(?).rsd`

where:

- XX is a code number for the experimental condition (type of stimulus for this trial);
- the date is given with YY (year), MM (month), DD (day); note that the year is not present in the name of .rsd file, and will be asked in the import process;
- EE is the experience number conducted at that date (not present in .rsd files; will be set to 01);
- TTT is the trial number (which can also be encoded on four digits, TTTT, thus supporting at most 9999 trials per session date);
- ? can be any character (they are unused).

If your raw files do not follow this convention, the easiest solution is to rename them before trying to import them. In all cases, the imported Nifti file name will be `sYYMMDD_eEE_tTTTT_cXX.nii`.

## 7.2 Orders of regressors for a linear model

When running a multiple regression, several regressors are included in a design matrix $X$, which is stored in the session and trial directories in a file called `glm.txt`. The matrix has $N$ lines, where $N$ is the number of regressors, and $T$ columns, where $T$ is the length of the imported timeseries (i.e after a potential temporal binning). Here is the order of the regressors:

- the first regressor has a constant one value (it accounts for the mean of the timeseries);
- after that are the oscillatory noise components modelled as Fourier series (each consecutive pair of line contains a sinus and cosinus components, at a given fondamental frequency, and for a given harmonics order; those are specified when constructing the model);
- the next one (if present) is the decaying exponential with time constant $Tau$ (it $Tau$ was set to zero when constructing the model, this regressor does not exist);
- finally, the last $L$ regressors are the ones that model the neural response.

When the linear model has been estimated, you will find a file called `sYYMMDD_eEE_tTTTT_cXX_betas.nii` in the trial directory. This file contains the $N$ spatial maps of $\beta$ weights that are the results of the model fit at each pixel. The order of these maps follows the same order as the regressors in the `glm.txt` file.

## 7.3 Using compressed or uncompressed Nifti files

The choice here is between:

- faster computation but higher disk usage with uncompressed files, and
- slower computation and lower disk usage with compressed files.

Here, our advice is to use uncompressed Nifti files when setting up an analysis and testing it on a very limited number of files. Because you are using a small number of files, the extra disk space taken will not be too large. And at this point, you probably are at a stage where you work in an interactive way, waiting for the result to be computed, so you want to optimize the computational time and thus avoid the extra compression/uncompression time.

Once all your parameter choices have been finalized, you can launch your imports and analyses on a very large number of files using compressed files. The computation will probably last for several hours, and the extra time used to compress/uncompress the files will not be very important. However, seeing the large number of files you probably want to process, it is important to minimize the disk usage, hence the use of compressed Nifti files.

## 7.4 Using scripts vs. using the GUIs

Although the iterate capability of BrainVISA allows setting up loops over several files through the user interface, it can become difficult to use when working with a very large number of files (several hundreds or more). It then becomes almost necessary to use scripts for the operations that need to be repeated over each trial. These operations notably include the importing steps and the trial analysis processes (linear model or standard blank subtraction).

For all other operations (and notably all the post-processing visualization operation), it is possible to run them both through the GUI or by writing scripts (although it can sometimes be more convenient to use the GUI).

You can therefore settle in a working mode where you switch back and forth between using the GUI or scripts. In that case, every time you have run one or several scripts and you want to go to the GUI for the next opration, it is necessary to update the database in the GUI before running anything else. For this, select the *Data Managmenet* toolbox in the left panel and run the *Update Databases* process.

## 7.5 Importing other data formats

If you have data that is not in formats currently supported by *Vobi One*, you will need to develop a new importing routine for this format. We here list the name of the files and corresponding functions that will need to be added or modified, in a hierarchical manner

- `brainvisa/toolboxes/vobi_one/processes/Import/import_xxx_file.py`; this is the process of the BrainVISA toolbox itself, which defines the user interface for this process; most of the code for a new importer should be copy-pasted from existing ones;
- `lib/python2.5/site-packages/oidata/oitrial_processes.py`, function `import_external_data_process`; this function is the generic function called by all Importing processes; this is where the file name convention for each format is defined
- `lib/python2.5/site-packages/oidata/oitrial.py`, function `load_external_file`; this function is called by the previous one; it basically is a switch on the data format, and it instanciates an object of the corresponding class and read the data through the corresponding method;
- `lib/python2.5/site-packages/oidata/xxx_file.py`; this file defines a class for the corresponding data format; this class should implement a few methods (see for instance rsd_file.py for an example), among those one of them reads the header information to read the metadata, one of the reads the actual data, and another shapes the data into the correct configuration to be stored into a Nifti object.

We are of course available to help you with implementing any new Importer (int-support-vobione[at]univ-amu.fr).