CCA-fMRI Toolbox - SPM 2

User's Manual

Version 1.01

Linköping University

Content

Trademarks

Matlab® is a registered trademark of The MathWorks, Inc. (http://www.mathworks.com).

Linux® is the registered trademark of Linus Torvalds in the U.S. and other countries.

Windows® is registered trademarks of Microsoft Corporation in the U.S.A and/or other countries.

Credits

Author

The toolbox and its documentation were developed by Ph.D. Nils Paulsson, Center for Medical Image Science and Visualization (CMIV), Linköping University, with much appreciated input from Professor Magnus Borga and Ph.Lic. Joakim Rydell, Department of Biomedical Engineering, Linköping University, Sweden.

Scientific work

This toolbox is based on the work in, among others, the following references:

1. Borga M., Learning Multidimensional Signal Processing., Ph.D. thesis, Linköping University, SE-581 83 Linköping, Sweden, 1998. Dissertation No 531, ISBN 91-7219-202-X, http://www.imt.liu.se/mi/Publications/Papers/M_Borga_thesis.pdf

2. Buxton R., Wong E. and Frank L., Dynamics of Blood Flow and Oxygenation Changes During Brain Activation: the Balloon Model.,Magnetic Resonance in Medicine, 39(6):855-864, 1998.

3. Das S. and Sen P., Asymptotic distribution of restricted canonical correlations and relevant resampling methods., Journal of Multivariate Analysis, 56(1):1-19, 1996.

4. Friman O., Borga M., Lundberg P. and Knutsson H., Adaptive Analysis of fMRI Data, NeuroImage 19(3):837-845, July 2003.

5. Friman O., Borga M., Lundberg P. and Knutsson H, Detecting Neural Activity in fMRI Using Maximum Correlation Modeling, NeuroImage, 15(2):386-395, February 2002.

6. Friston K. J., Mechelli A., Turner R. and Price C. J., Nonlinear responses in fMRI: the balloon model, Volterra kernels,and other hemodynamics. NeuroImage, 12:466-477, 2000.

7. Hotelling H., Relations between two sets of variates., Biometrika, 28:321-377, 1936.

8. Rydell J., Adaptive Spatial Filtering of fMRI Data, Linköping Studies in Science and Technology, Thesis No. 1200, Linköping University, SE-581 83 Linköping, Sweden, LiU-TEK-LIC-2005:55, ISBN 91-85457-43-4.

9. Zheng Y., Martindale J., Johnston D., Jones M., Berwick J. and Mayhew J., A model of the hemodynamic response and oxygen delivery to brain., Neuroimage 16: 617-37, 2002.

1. Background

The *CCA-fMRI Toolbox* implements the use of canonical correlation analysis (CCA) for detecting brain activity patterns recorded by functional magnetic resonance imaging (fMRI). CCA was developed by Hotelling [7] and is a method for finding the maximum correlation between linear combinations of two sets of variables. In the *CCA-fMRI Toolbox* CCA is used in a two step process. In the first step, CCA is used to construct a low pass filter that adapts to the environment of each voxel in the brain volume analyzed. In the second step, CCA compare the temporal intensity change of the filtered voxels with an expected activation pattern to determine the level of correlation and, hence, the level of activation. The method is thoroughly described in reference [4].

SPM is a well known and free (GNU General Public License) software package for analysis of brain imaging data sequences. It has a long history and has continuously been released in new versions since at least 1994.

2. Requirements

Hardware requirements

- Any hardware platform that is supported by both Matlab® version 7 (7.1 - 7.3) and SPM 2.
- Minimum 1 GB RAM. For 3-dimensional analysis a minimum of 1.5 GB RAM is recommended.
- CPU running at 2 GHz or above.
- Minimum 1.5 GB available temporary disk storage.

Software requirements

- Matlab® 7, i.e. versions 7.1 – 7.3.
- SPM software package version 2 (http://www.fil.ion.ucl.ac.uk/spm/).
- Linux®, Windows® XP or any other operating system that is supported by both SPM 2 and Matlab® 7.

Restrictions

- This specific version (the 1.x branch) of the *CCA-fMRI Toolbox* was written for SPM 2. For SPM 5 please use version 2.x
- The *CCA-fMRI Toolbox* was developed and tested using 32-bit Matlab®, versions 7.1, 7.2 and 7.3. Correct operation under 64-bit Matlab® has neither been tested nor verified.
- SPM 2 is officially only supported on Matlab® versions 6.0 - 6.5. However, besides having successfully developed the *CCA-fMRI Toolbox* using Matlab® 7.1-7.3 and SPM 2, there are several external sources reporting that this is indeed a working combination.
- The *CCA-fMRI Toolbox* will **not** work on Matlab® versions prior to 7.1.

3. Installation & Upgrade

1. If not already done, install Matlab® and SPM 2 according to their respective installation instructions.

2. Unzip the *CCA-fMRI Toolbox* zip-file in a temporary directory.

3. Move the extracted folder, named `CCA_fMRI`, to the toolbox folder in the installed SPM 2 directory tree, e.g.

   `<MatlabPath>\toolbox\spm2\toolbox`.

   Be sure to have write access to the toolbox folder. If prompted to overwrite or replace an existing `CCA_fMRI` directory and its files acknowledge doing so.

4. Start Matlab® and add the path of the *CCA-fMRI Toolbox*-folder, e.g. `<MatlabPath>\toolbox\spm2\toolbox\CCA_fMRI`, to the Matlab® path by selecting `Set Path...` from the `File`-menu in Matlab's® console window.

5. Install the *CCA-fMRI Toolbox* by entering:

   ```
   >> CCA_fMRI('install');
   ```

   at the Matlab® prompt. When the installation is done the following text will be displayed:

   ```
   CCA-fMRI for SPM has now been installed and the
   toolbox can be reached from within SPM.
   ```

The *CCA-fMRI Toolbox* is now installed and ready to be used. In case SPM 2 is used on a computer where each user has a profile of their own, step 5 above has to be repeated for all users that want to use the *CCA-fMRI Toolbox*. In other words, repeat the following steps for each user:

1. Log on the user
2. Start Matlab®
3. Run `CCA_fMRI('install');`

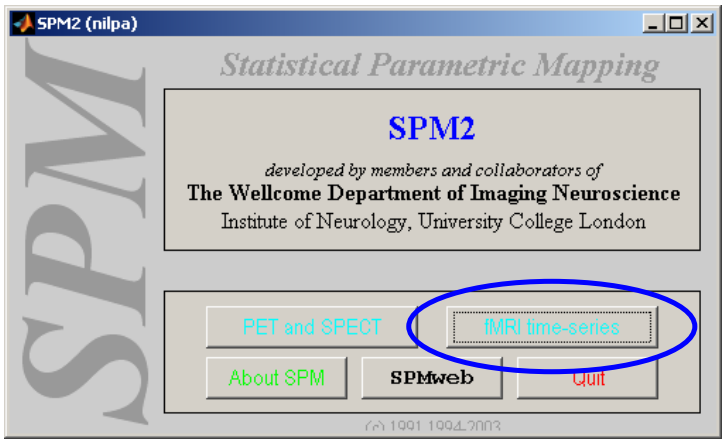
For more information regarding usage of the *CCA-fMRI Toolbox* please see the following chapters in this User's Manual.

4. Start up

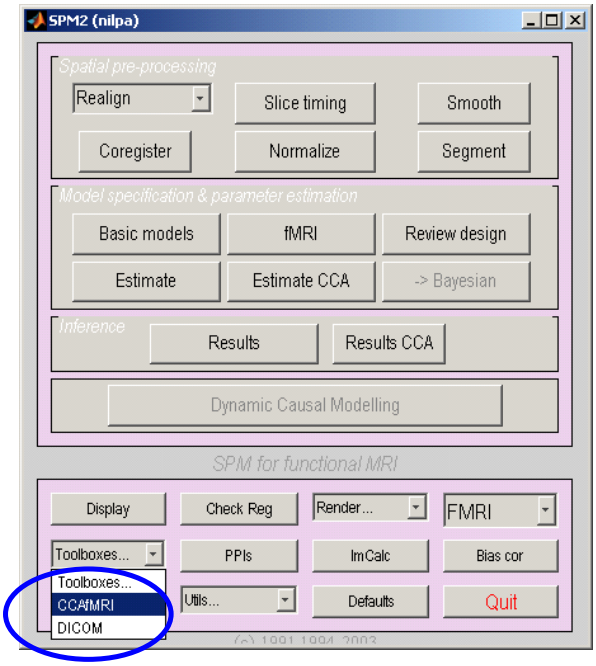
The *CCA-fMRI Toolbox* can be reached from the SPM GUI in the same way as any other SPM toolbox. First, start SPM from the Matlab® prompt:

```
>> spm
```

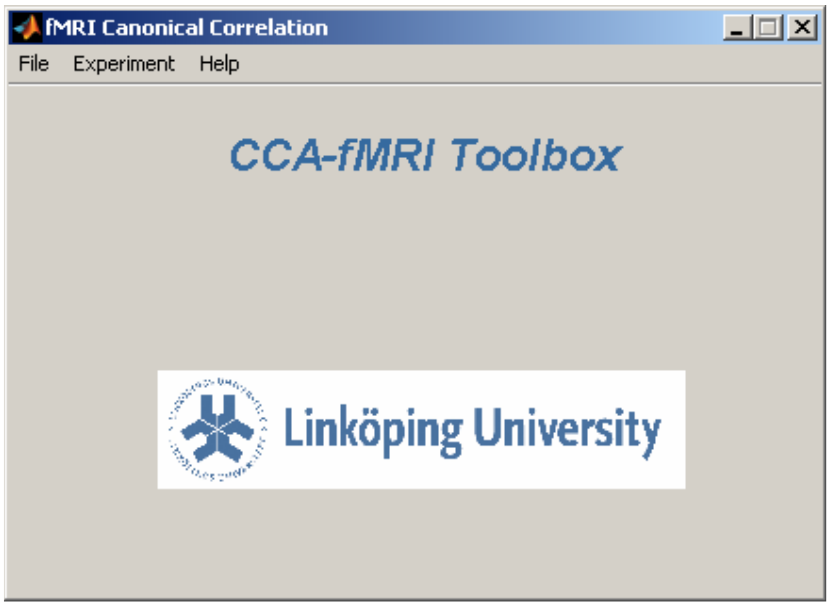
Click the `fMRI time-series` button in the main window:



Three new windows are opened. Open the drop down list named `Toolboxes...` in the lower left corner of the fMRI main window and select `fMRICCA`.

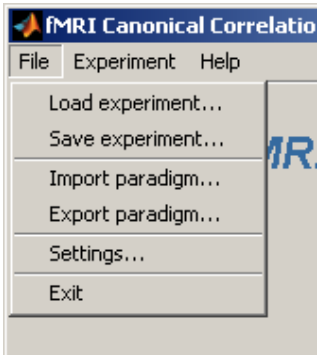


The *CCA-fMRI Toolbox* starts up and opens the main window. The toolbox is now ready to be used.



5. The File Menu

The `File` menu's main purpose is to provide the abilities to save and load experiment settings (see *Setting up an experiment*), import/export paradigms and optimize memory usage.



The `Load Experiment...` loads a previously saved experiment and makes it the currently defined experiment.

The `Save Experiment...` menu stores the current experiment setup to an external Matlab® file (MAT-format) using a user provided file name. The file holds a Matlab® structure named `Experiment` having the following fields:

```
Experiment

    ParadigmDesign: [1x1 struct]
    BalloonLimits: [1x1 struct]
    GammaDiffLimits: [1x1 struct]
    FilterSettings: [1x1 struct]
    ImageFiles: [1x1 struct]
```

`ParadigmDesign` - holds the parameters defining the paradigm of the experiment. For more informaton see `ValidateDesign.m` and the paradigm design dialog box.

`BalloonLimits` - specifies the base and threshold values of the hemodynamic response model used in the experiment for cases when being based on the balloon model. For more information see `ValidateBalloonLimits.m` and the balloon settings dialog box.

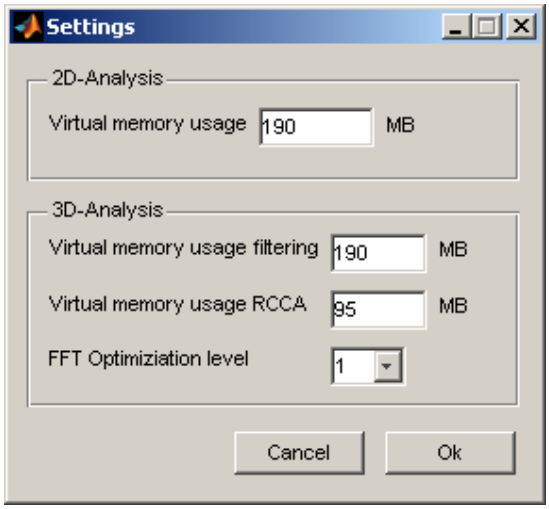
`GammDiffLimmits` - specifies the base and threshold values of the hemodynamic response model used in the experiment for cases when being based on the differential gamma function. For more information see `ValidateGammaDiffLimits.m` and the GammaDiff settings dialog box.

`FilterSettings` - specifies the filter settings of the experiment. For more information see `SteerableBasisFilter3D.m` and the steerable filter settings dialog box.

`ImageFiles` - specifies the set of image files that are analyzed in the experiment. For more information see `frm_ImageFiles.m`.

At times the same paradigm is used in several experiments. To facilitate the experiment setup the two menu items `Import paradigm...` and `Export paradigm...` allows the user to exclusively save and load the paradigm part of an experiment.

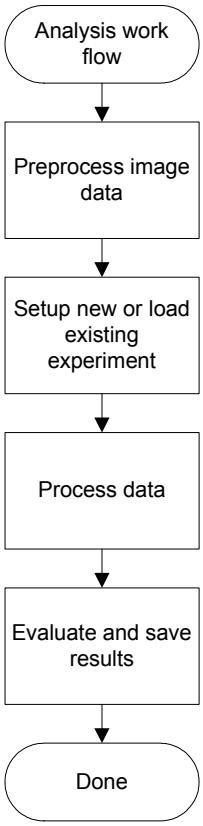
The `Settings` menu opens up the dialog box for managing memory usage and FFT optimization levels. There are different settings depending on if 2-dimensional or 3-dimensional analysis is performed (see *Setting up an experiment*). For 2-dimensional analysis `Virtual memory usage` has to be set to be large enough to hold the entire image set of an experiment.



For 3-dimensional analysis the `Virtual memory usage filtering` parameter has to be set to be large enough to hold the entire image set of an experiment. The `Virtual memory usage RCCA` setting has to be large enough to hold at least one image file. Finally, the `FFT Optimization level` sets the level of optimization performed during filtering with steerable filters. The lowest level is 1 and the highest level is 4. The higher levels may produce lower initial performance than the lower levels but over time Matlab®'s FFT module will learn how to perform an optimal Fourier transform. To determine the size of an image in bytes use the following formula *ImageSizeX* * *ImageSizeY* * *ImageSizeZ* * 4. The image size is expressed as number of pixels. To get the size of an image set just multiply the image size with the number of images.

6. Analysis workflow

The workflow of the CCA-fMRI analysis is straight forward and consists of preprocessing, experiment setup, data processing and finally evaluation of the result.



In the preprocessing step the raw image data is realigned, normalized, etc. to facilitate the subsequent data processing. However, **lowpass filtering must not be performed**. That would significantly degrade the quality of the results. The *CCA-fMRI Toolbox* performs its own lowpass filtering using adaptive filters. In the second step, experiment setup, parameters such as paradigm, BOLD model settings, filter properties and image data set are specified. Experiments can also be stored/loaded from secondary storage (hard drive, CD-ROM, etc). The step of processing data consists of several phases but the process is entirely automatic. Once started there is no need for user interaction until the processing has finished. This is usually the most time consuming step. A 2D-analysis usually takes less than 10 minutes and a 3D-analysis less than an hour, depending on size of data set, amount of available RAM, etc. In the last step the result can be visualized as well as exported to secondary storage. With the exception of preprocessing, all steps are performed within the *CCA-fMRI Toolbox* user interface.

7. Preprocessing

The *CCA-fMRI Toolbox* will usually produce better results when analyzing preprocessed data. Commonly applied preprocessing procedures in SPM are:

- Setting origin
- Realignment
- Reorientation

The only prerequisite of using preprocessed images in the toolbox is that the data is available as standard SPM2 image files, i.e. ANALYZE files. This is seldom a problem since SPM usually applies preprocessing to the image data files directly or by creating new preprocessed versions of existing image files.

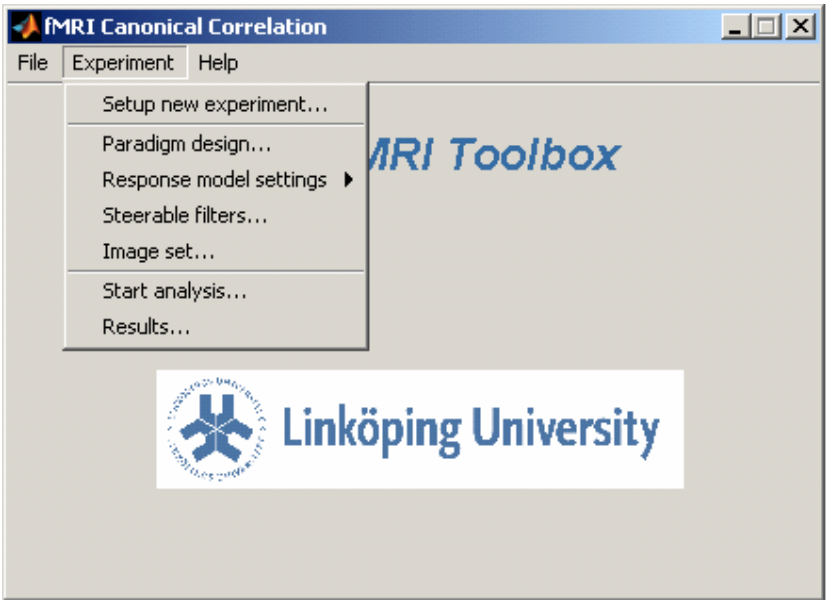
NOTE
Smoothing, or low pass filtering, must not be applied to the data set during preprocessing. Doing so would interfere with the adaptive filtering applied later on by the *CCA-fMRI Toolbox* and significantly degrade the quality of the final analysis results.

8. Setting up an experiment

An analysis setup is defined in terms of an experiment, which defines the parameters necessary for the analysis of a certain fMRI data set. An experiment is defined by the four main property groups:

- Paradigm design
- Response model (BOLD) settings
- Settings of the steerable filters
- Name and location of the image data set to be analyzed

All properties are accessible from the `Experiment` menu in the *CCA-fMRI Toolbox* main window.

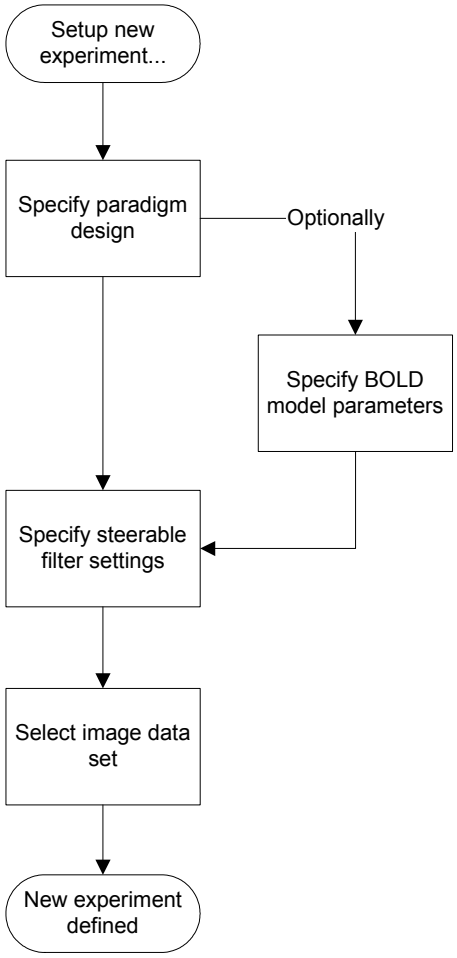


The paradigm design specifies at what time points and for how long the subject is exposed to stimuli. The response model describes how the expected BOLD response, resulting from the stimuli, should look like. The steerable filter settings give the shape and size of the adaptive low pass filters used during the data analysis and the data set points out which image files to analyze.

The Experiment menu

Setup new experiment…

This menu item takes you through all the steps necessary to setup a new experiment, i.e. the paradigm design, the response model (BOLD) settings, the filter settings and the image data file selection. The setup workflow is as follows:

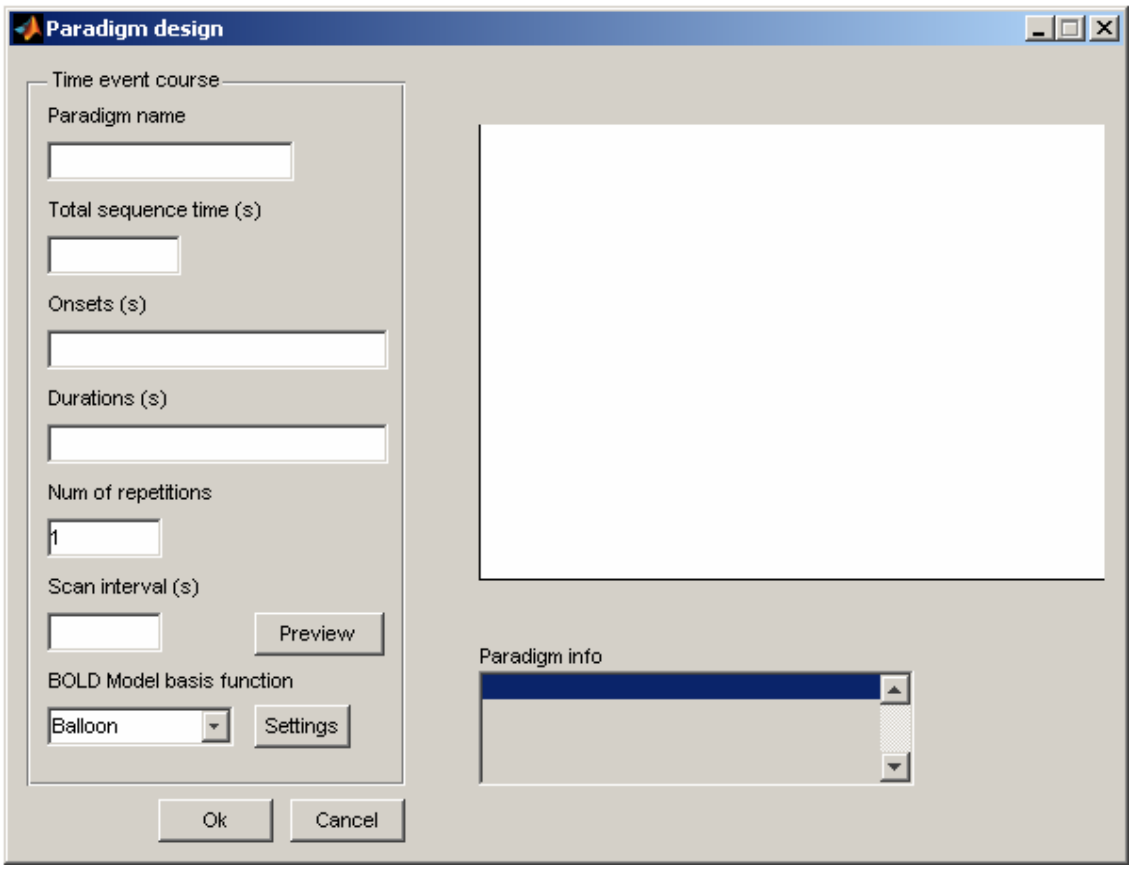


Setting up a new experiment this way is entirely transactional, meaning that the user has to respond `Ok` to all the dialog boxes that appear during the setup procedure. If the user selects `Cancel`, i.e. click the `Cancel`-button, in any of the steps above, none of the settings are saved and any previously specified experiment and results are still intact.

The separate settings are also available under their corresponding menu headline in the `Experiment` menu. Consequently, an experiment can also be setup in a more manual fashion by accessing the menu items `Paradigm design`…, `Response model settings`, `Steerable filters` and `Image set`…. Please see below for more information about how to use the separate settings dialog boxes.

Paradigm design…

The paradigm design outlines at what times and for how long the CCA-fMRI data analysis should look for activity, i.e. BOLD responses, in the fMRI data.



On the left side of the dialog box the paradigm properties are entered and the right side shows how the paradigm looks as well as some basic information about it. The paradigm parameters are:

`Paradigm name` – Any name you want to give the paradigm.

`Total sequence time` – The total time of the paradigm in integer seconds.

`Onsets` – The time points at which stimuli are presented to the subject investigated. `Onsets` are specified in integer seconds and the different time points are separated by space when entered in the field.

`Durations` – The length of the stimulation period beginning at the corresponding `Onset` time points. `Durations` are specified in integer seconds and the different time periods are separated by space when entered in the field.

`Num of repetitions` – Specifies the number of times the `Onset/Duration` settings should be repeated **within** the total sequence time. As such, the `Onset/Duration` settings may not be repeated beyond the total sequence time.

`Scan interval` – The sampling interval at which fMRI image volumes are recorded by the MR scanner. `Scan interval` is given in decimal seconds.

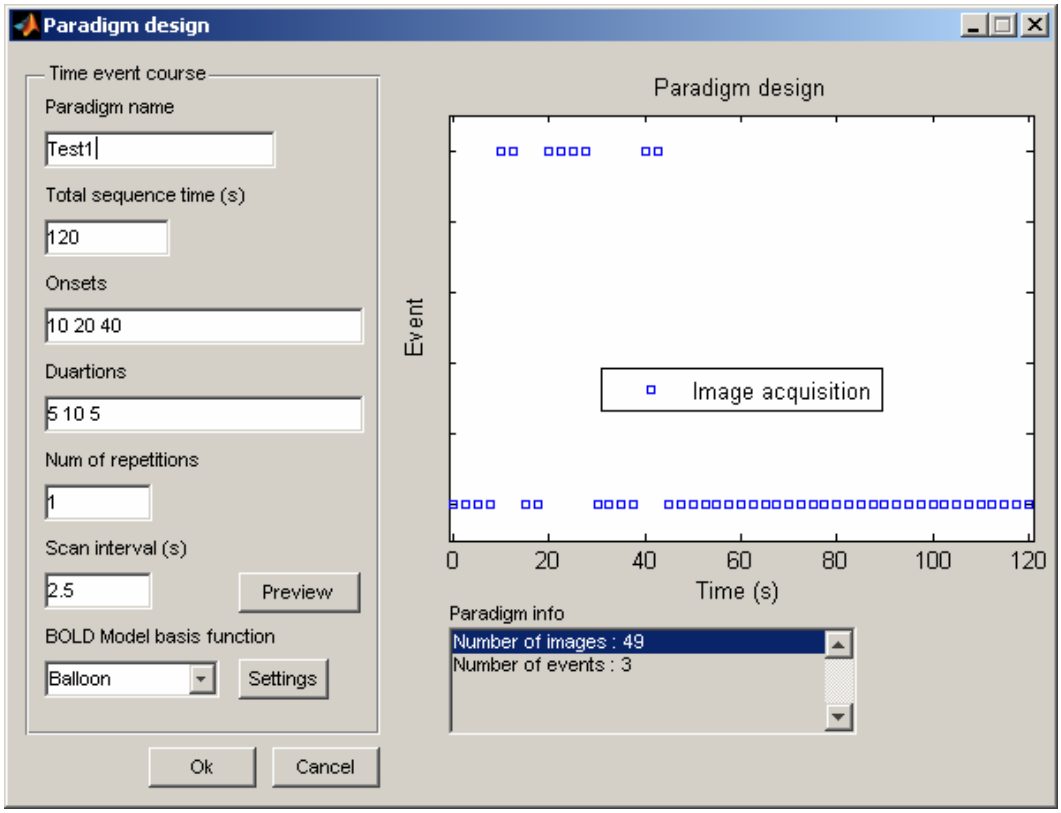
`BOLD Model basis function` – Specifies what mathematical model should be used to approximate the BOLD response. The two options are the *Balloon model* and the *Differential gamma model*. Default model selection is the *Balloon model*.

The dialog box also has four buttons. Besides the standard `Ok/Cancel`-buttons, there is also a `Preview`-button for plotting the resulting paradigm design and a `Setting`-button that allows direct access to the dialog boxes for adjusting the parameters of the selected BOLD model basis function. Please see menu item `Response model settings` for more information.

Examples

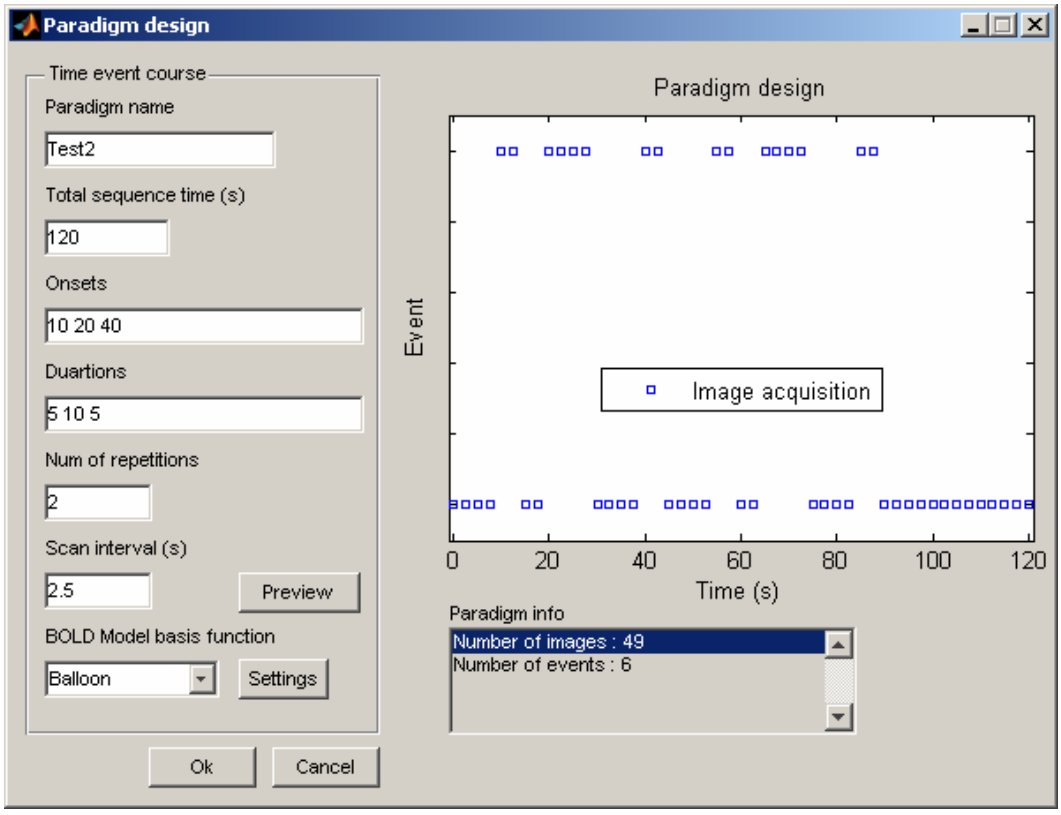
Example 1.

The total sequence time is 120 seconds. Three events occur at 10 s, 20 s and 40 seconds respectively (onsets). The duration of the first event is 5 seconds, the duration of the second event is 10 seconds and the duration of the final event is 5 seconds. The MR-scanner samples a new image volume every 2.5th second. The paradigm is entered in the dialog box the following way:



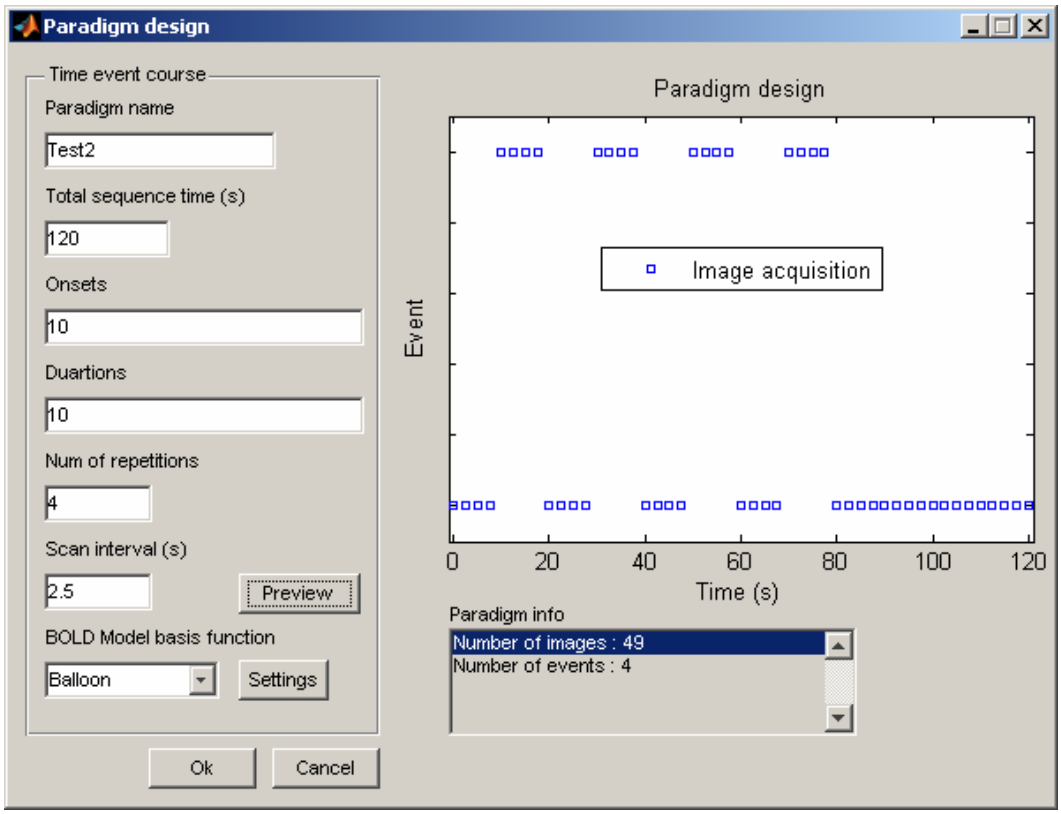
Onsets are entered after each other separated by a space character, as are the corresponding durations. Click the `Preview`-button to plot the sampling points.

Example 2.
In this example almost all parameters are the same as in Example 1 with the notable exception of the `Num of repetitions`. This time we would like to repeat the `Onset/Duration` sequence owing to repetitive patterns of stimuli presentations. By setting `Num of repetitions` to 2 the Onset/Duration sequence will be repeated one time directly after the last data point of the last stimuli event, ending at 45 seconds (Onset 40 s + Duration 5 s = 45 seconds).



Again, click the `Preview`-button to plot the new paradigm. The limitation of the `Num of repetitions`-parameter is that the `Onset/Duration` sequence can't be repeated beyond the `Total sequence time`-setting.

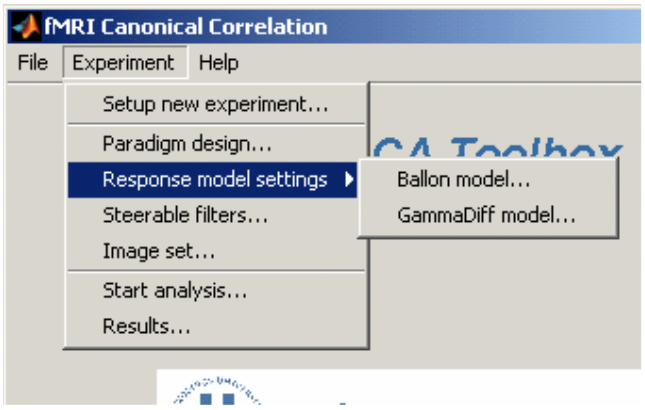
Example 3
A consequence of having the possibility to repeat the sequence of events, the setting `Onset` = [10 30 50 70] and `Duration` = [10 10 10 10] can also be defined as `Onset`=10 s, `Duration`=10 s and `Num of repetitions`=4 :



Response model settings

The response model describes the expected BOLD response for a given paradigm. In other words, the model is considered to be the approximate true activation pattern for any subject exposed to the stimuli paradigm used in the experiment. The level of activation in a brain voxel is, simplified, determined by comparing the temporal intensity change of the voxel with the intensity change of the response model. High correlation means high level of activation and vice versa. This comparison is performed for each voxel in the sample data.

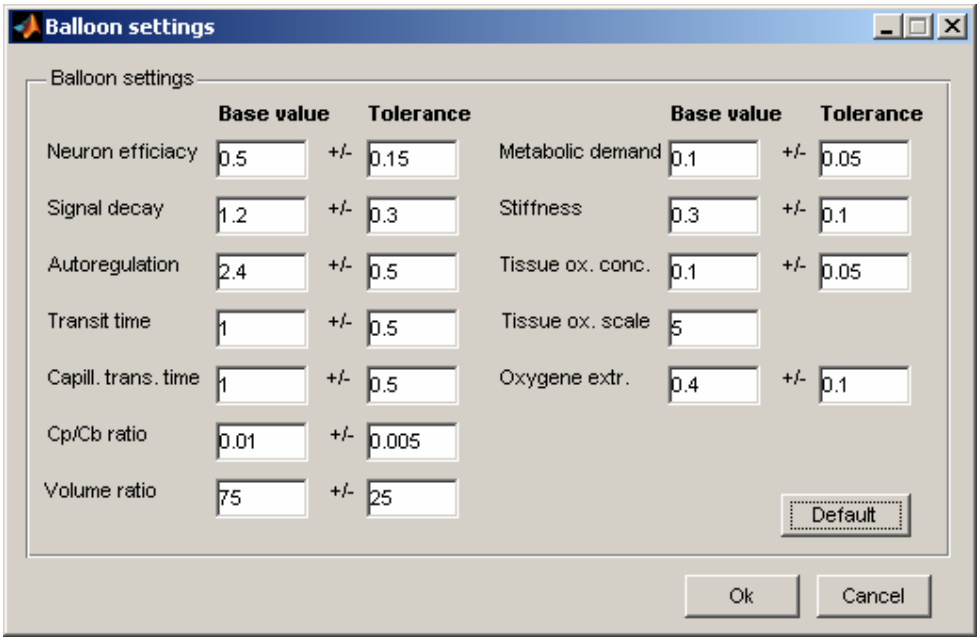
There are two different BOLD basis functions available in the toolbox; the *Balloon* model and the Differential Gamma model, also called *GammaDiff* in the toolbox. Of the two, the *Balloon* model is considered more accurate but also somewhat more demanding for the computer to generate. For a fairly up to date computer there is little incentive not to use the *Balloon* model.



The two models have their own sets of adjustable parameters, accessible from the `Response model settings` menu. Disregarding what basis function is used, the BOLD model is generated in the same way. First, 500 plausible response curves are generated by randomizing the values of corresponding parameters. To assure plausible curves the parameters are only allowed random variations within a specific tolerance interval. The 500 plausible responses are then reduced, by principal component analysis, to a compressed format, which is also the expected BOLD response used in the subsequent data analysis.

Balloon model settings

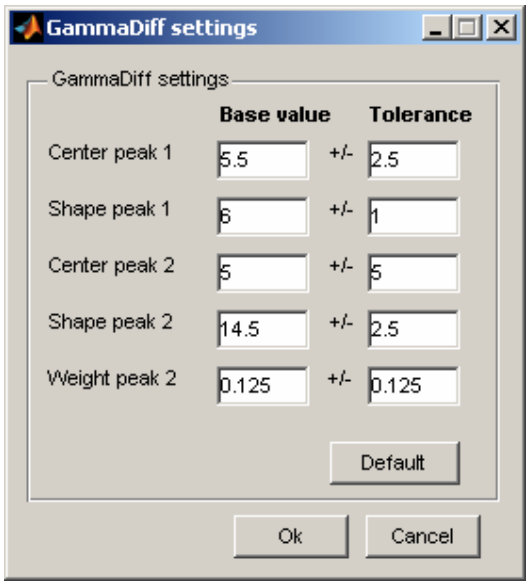
The *Balloon* model is a fairly complicated model having a number of adjustable parameters corresponding to, among other things, properties of an expanding blood vessel forming a local balloon of oxygenated blood. The model, and its parameters, is explained in references [2] and [6].



The default parameter settings are adequate in most cases and have been compiled from data in references [2], [6] and [9].

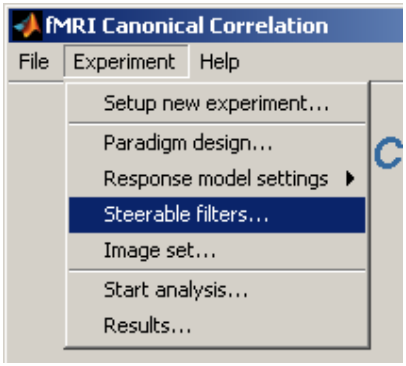
GammaDiff model settings

The *GammaDiff* model is a simpler model and uses the difference of two Gamma functions to model the BOLD response. The default settings are adequate for most cases and have been compiled from data in reference [5].

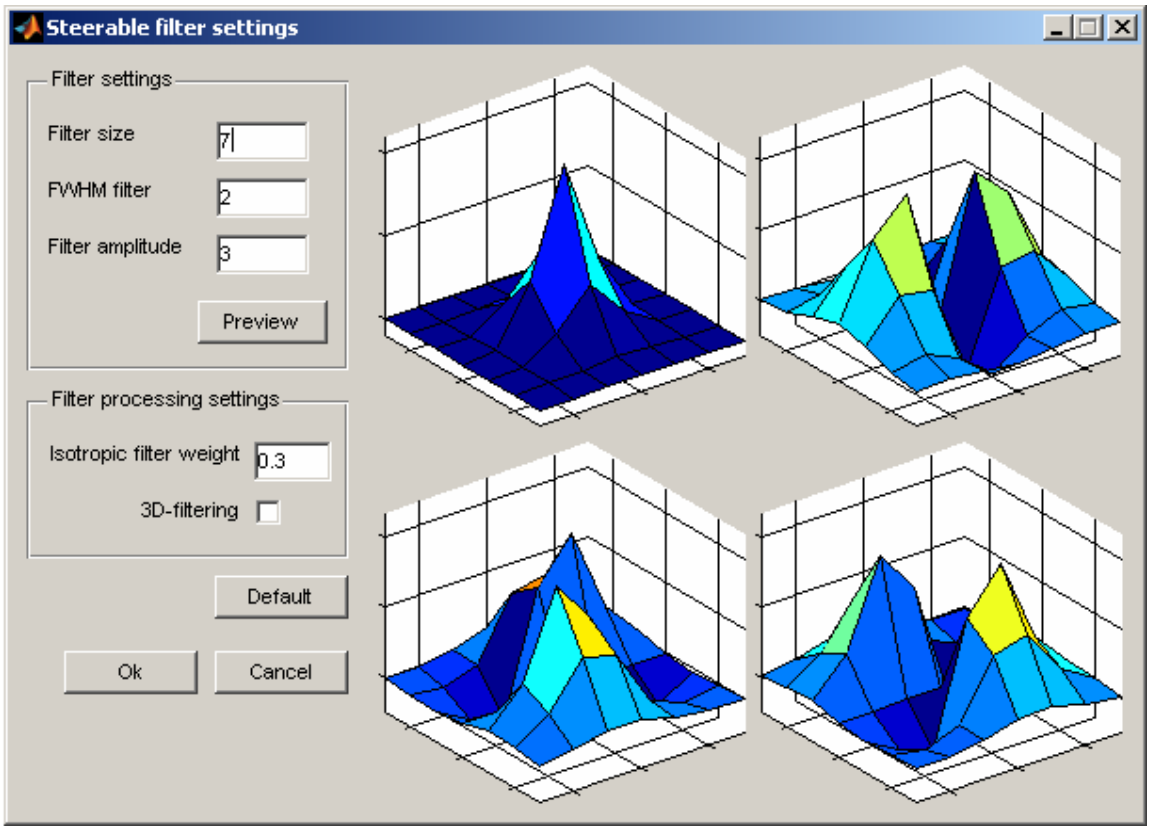


Steerable filters…

A steerable filter, or adaptive filter, is a special set of spatial low pass filter kernels that can be combined and optimized in accordance with the nature of the data on which they are applied. The *CCA-fMRI Toolbox* utilizes these properties by only applying filter configurations that are favorable to the CCA-fMRI analysis.



From the perspective of the *CCA-fMRI Toolbox* user they have similar set of parameters as ordinary low pass filters, e.g. filter matrix size and FWHM (Full Width Half Maximum).



In the left side of the dialog box the filter and processing parameters are set and the right side shows previews of the filter kernel shapes. The filter kernel views are updated by clicking on the `Preview`-button using the currently entered values.

NOTE
Even when selecting 3-dimensional analysis the filter kernels will still be drawn as 2-dimensional kernels in the previews.

Filter settings

The filter parameters are:

`Filter size` – The size of the adaptive filter given as the number of pixels (or voxels) of one side if the filter kernel. The filter size is symmetric in all directions forming a squared matrix or cube depending on whether 2-dimensional or 3-dimensional analysis should be performed. Only odd sizes are allowed.

`FWHM filter` – The full width half maximum of the filter kernels expressed as number of pixels. This is the same as the effective width of the filter.

`Filter amplitude` – The amplitude of the final filter.

Large settings for `FilterSize` and `FWHM filter` tend to favor large regions of activation and suppress small. Small settings allow small regions of activation to remain but also increase the spatial noise level.

Filter processing settings

The filter processing parameters defines how the adaptive filter is applied during the data analysis phase. The parameters are:

`Isotropic filter weight` – Defines the importance of the center voxel during filtering. Weight 0.0 means that the center voxel is given no special importance, i.e. only anisotropic filtering, and weight 1.0 gives the center voxel the same importance as the anisotropic filter.

`3D-filtering` – Enables 3-dimensional filtering. When unchecked, 2-dimensional filtering is used.

Using 3-dimensional filtering is the preferred setting since that would take into account the entire 3-dimensional neighborhood. 2-dimensional only takes into account the neighborhood in the X/Y-plane, which typically is the horizontal plane through the brain.

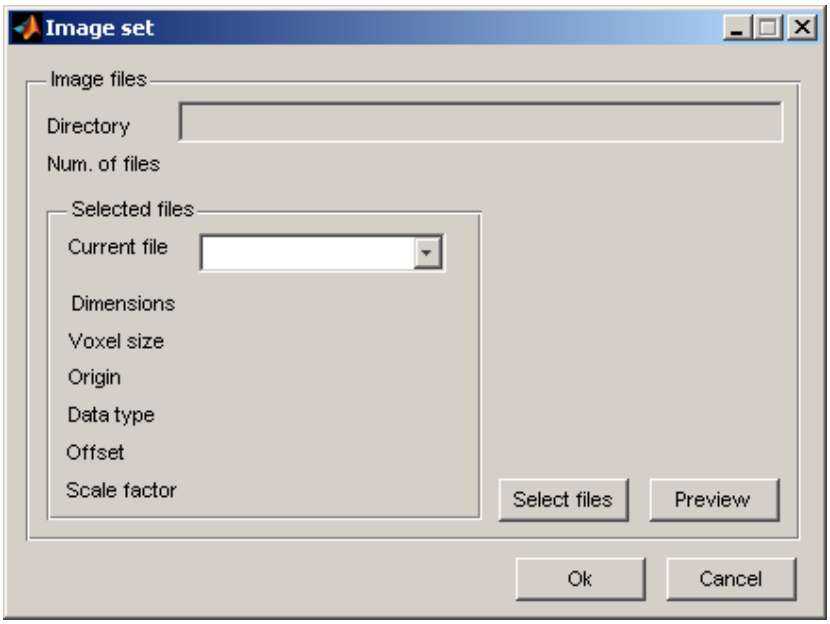
The drawback of using 3-dimensional filtering is that the complexity of the fMRI analysis grows exponentially meaning significantly longer processing time, compared to 2-D filtering. On a fairly modern computer a 2-dimensional analysis would take 5-

10 minutes. The same analysis performed in 3 dimensions would take from 40 minutes up to an hour.

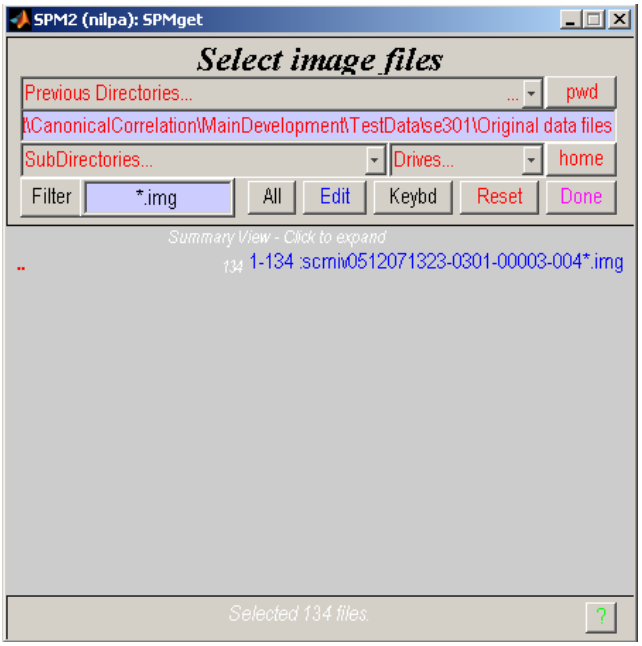
NOTE
The most important factor for maximizing the 3-D performance is to have a lot of random access memory (RAM). The analysis generates between 1 and 2 GB of temporary data. Large amount of RAM prevents data from being written to disk during the analysis, hence increasing performance. See *Requirements* for more information regarding the recommended hardware platform.

Image set …

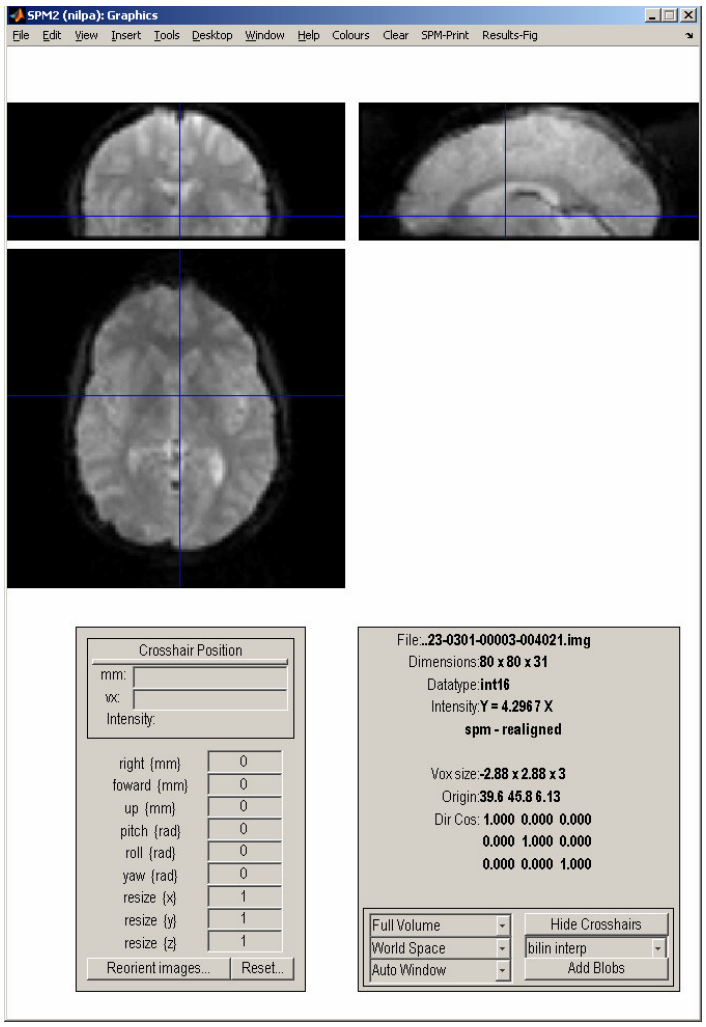
The `Image set` window is used for selecting the image set to be analyzed by the *CCA-fMRI Toolbox*. The data set is defined by clicking on the `Select files` button.



The CCA-fMRI Toolbox uses the standard SPM file dialog box to select the image set. In case the images have been preprocessed be sure to select the image files having the correct name prefix.

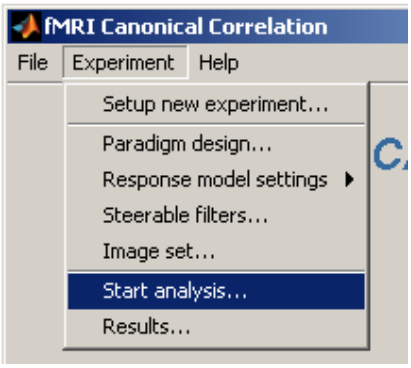


Information about separate image files in the set, selectable from the drop down list `Current file`, is displayed in the panel. The currently selected image can also be viewed using the `Preview`-button.

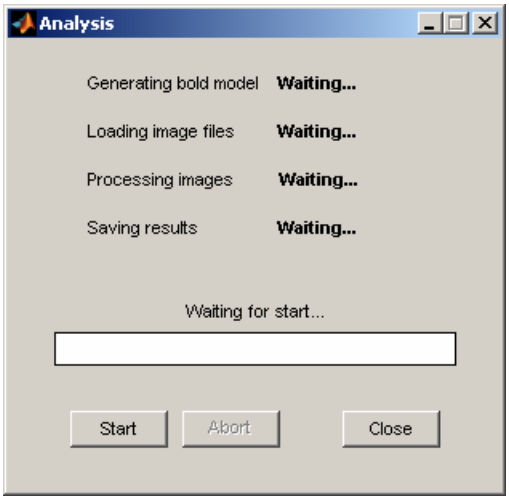


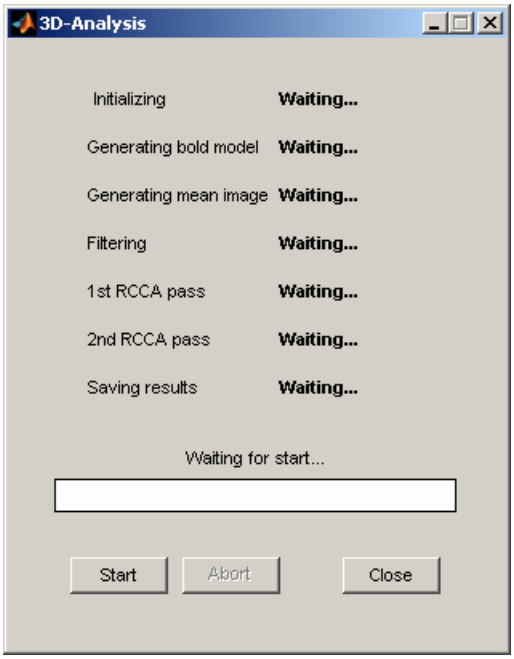
9. Data analysis

The data analysis is an automated process with little need for manual intervention. Depending on whether 2-dimensional or 3-dimensional analysis has been chosen one of the dialog boxes below will be opened.



Besides the `Start`-button, which commences the analysis process, there is also an `Abort`-button for interrupting an analysis in progress and a progress bar showing the progress of each separate phase.





2-Dimensional analysis ***3-Dimensional analysis***

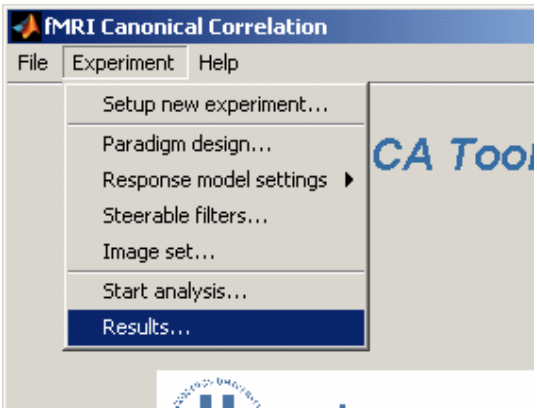
The analysis is divided into separate phases, shown in the dialog boxes above. The phases are listed in the order of execution and each phase has a status displayed to the right. The valid statuses are `Waiting...`, `Working...`, `Done` and `Aborted`. The toolbox keeps track of these statuses meaning that when a previously aborted analysis

is restarted only the phases not yet finished, i.e. not having the `Done`-status, will be reprocessed. Some parts of the analysis process may be reprocessed in cases when experiment settings influencing the phase have been changed. Which phases to reprocess is entirely handled by the *CCA-fMRI Toolbox*. Once the analysis has finished the result is reachable from the `Results…` dialog box. The results are also stored in two standard SPM image files in the same directory as the original image file sets analyzed. The two images are named `CorrelationMap_#TIMESTAMP#` and `MeanVolume_#TIMESTAMP#`, where `#TIMESTAMP#` is on the format MMM-DD-HH-MM-SS, e.g. `CorrelationMap_Jan-15_14-27-23.img` (or `.hdr`).

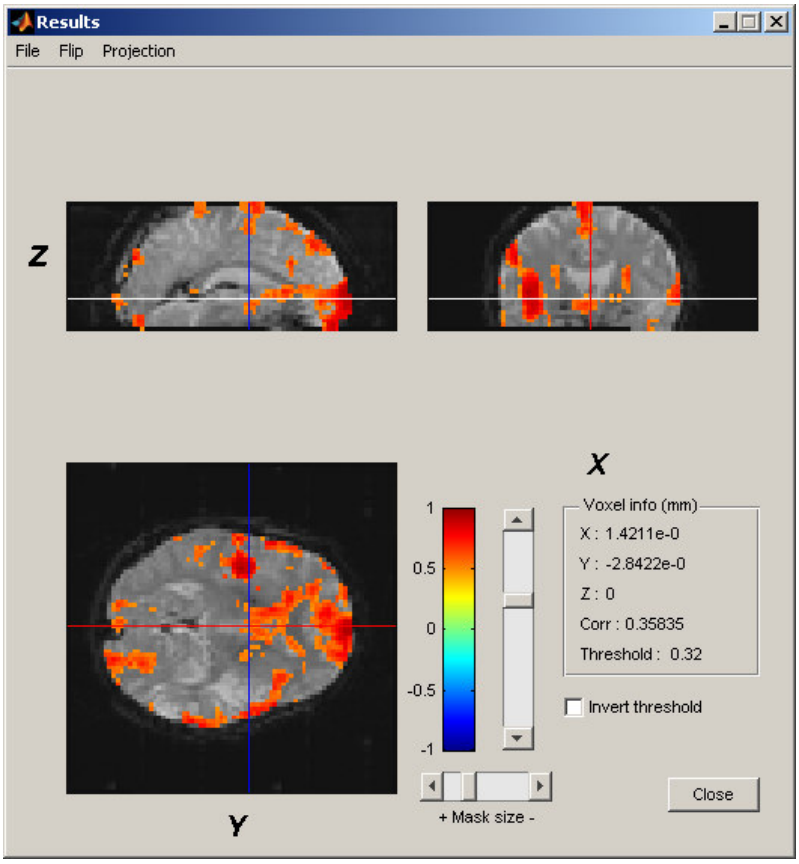
NOTE
When aborting an analysis the toolbox usually takes a while before actually aborting. The reason is that owing to limitations in Matlab® the abort of the analysis has to be synchronized with an update of the horizontal progress bar which only happens a limited number of times during each phase. The toolbox will, however, in time abort the analysis process.

10. Results

The `Results...` menu provide basic functionality for displaying, printing and saving the analysis results from the current experiment.

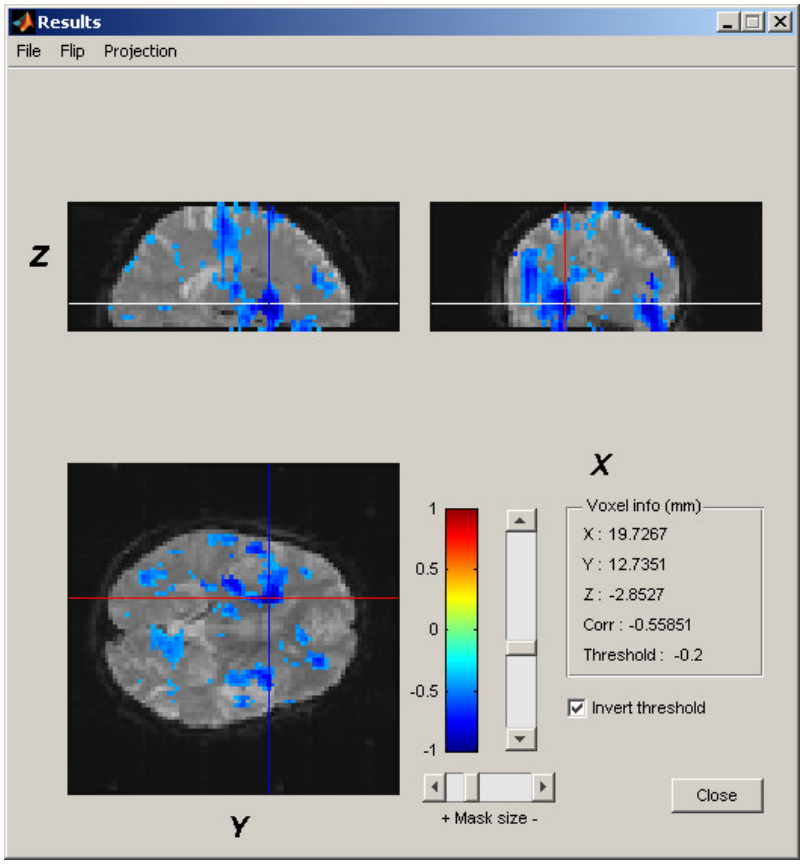


The `Results` dialog box displays the correlation as color coded levels superimposed on a background image of the brain analyzed. The brain is segmented along the X-, Y- and Z- dimensions forming 3 image segments. The lower left image is the Y/X-segment, the upper left is the Y/Z-segment and the right is the X/Z-segment.



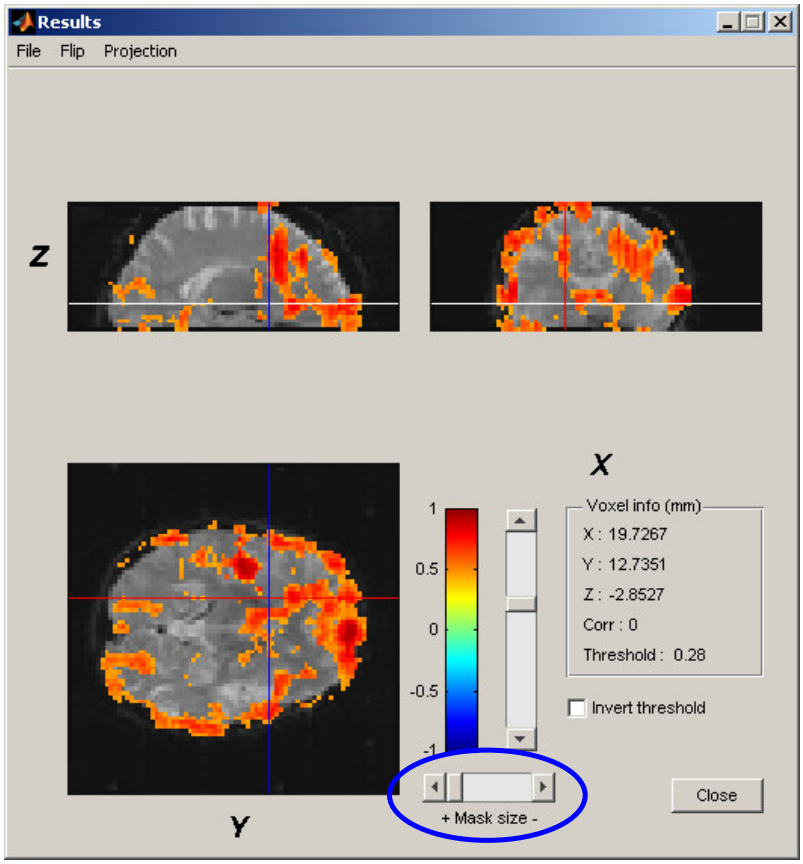
The background image is the mean image of the entire image set. The projection honors reorientation and other preprocessing procedures applied in advance of the canonical correlation analysis.

In the lower right area information about the current voxel is displayed, i.e. voxel coordinate in millimeters as well as its corresponding correlation coefficient. The current voxel is selected using the mouse to select and clicking in one of the three projections. The voxel information and hair cross are updated accordingly. The vertical slider is used to manually determine the smallest correlation threshold value for which correlation should be superimposed on the background image. The threshold level can also be inverted by checking the `Invert threshold` checkbox. This can be useful in cases where e.g. anticorrelation is of interest.



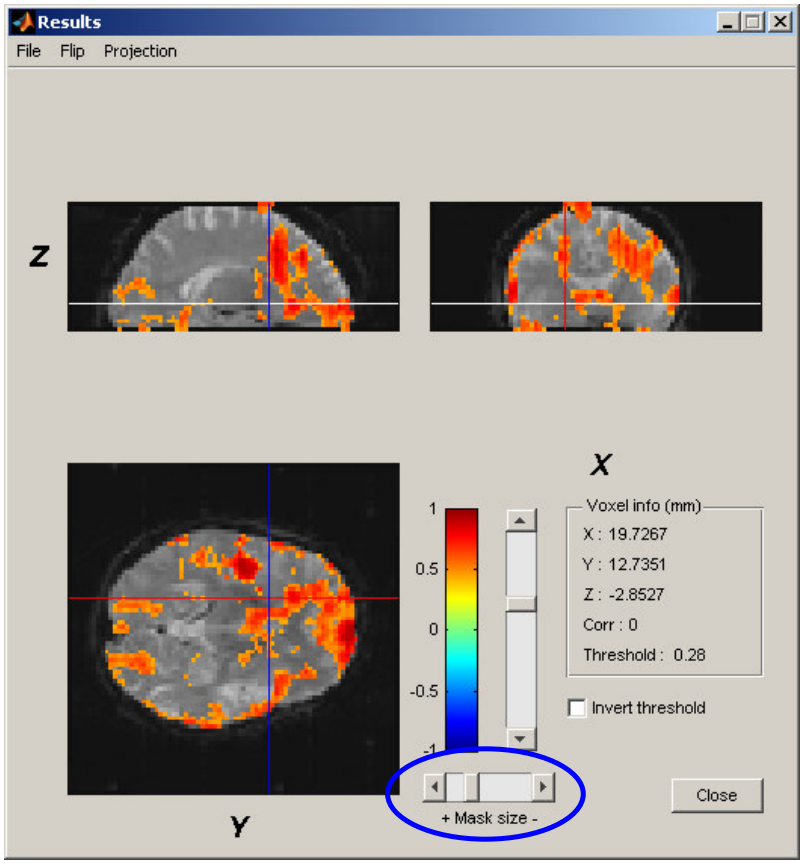
Inverted threshold

Before the CCA-calculation the image volume is segmented with respect to the brain. To reduce the possibility of accidentally excluding brain voxels the segmentation algorithm usually extracts a volume slightly larger than the brain itself. As a result, the analysis also includes voxels slightly outside the brain. Using the horizontal `Mask size` slider the segmentation mask can be reduced in size allowing the correlation map to better correlate with the brain tissue.



Original brain segmentation mask

The original brain segmentation mask is used when the `Mask size`-slider is positioned to the far left. Moving the slider to the right reduces the mask size stepwise.

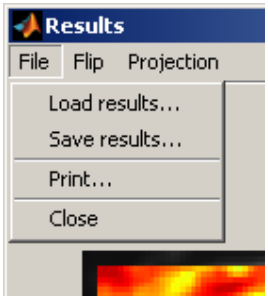


Reduced brain segmentation mask

The `Mask size`-slider only affects the displayed results and not the segmentation mask stored in the result's data file.

The File Menu

In the `File` menu current results can be saved to disk and already saved results can be loaded and displayed.



Load results…

Loads and displays the results from a Matlab® data file previously saved using `Save results...` Results from a current experiment setup is **not** erased by this. The next time the `Results` window is opened existing results from an ongoing experiment is re-displayed.

Save results…

Saves the currently displayed results in a Matlab® data file. The format of the data saved is a Matlab® structure (`Results`) having the following structure and fields:

```
Results
        SPM: [1x1 struct]
        xSPM: [1x1 struct]
        CorrelationMapFile: [1x1 struct]
        CorrelationMap: [COLxROWxDEPTH single]
        MeanImageFile: [1x1 struct]
        MeanImageVolume: [COLxROWxDEPTH single]
```

`Results.SPM` is the `SPM` structure associated with the correlation map. Please see the SPM documentation for more information.

`Results.xSPM` is the `xSPM` structure associated with the correlation map. Please see the SPM documentation for more information.

`Results.CorrelationMapFile` is the `SPM` file header structure pointing to the file where the correlation map is stored externally. The file is a standard SPM image file.

`Results.CorrelationMap` is the resulting 3-dimensional correlation map calculated during the analysis step organized as a 3-dimensional Matlab® matrix.

`Results.MeanImageFile` is the SPM file header structure pointing to the file where the mean image of all the image volumes in the set is stored externally. The file is a standard SPM image file.

`Results.MeanImage` is the resulting 3-dimensional mean image calculated during the analysis step organized as a 3-dimensional Matlab® matrix.

Print…

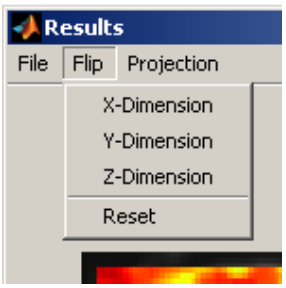
Prints a copy of the currently displayed results.

Close

Closes the results dialog box and returns to the main window.

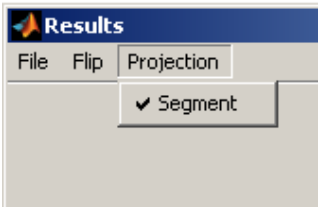
The Flip Menu

The `Flip` menu provides the ability to flip the dimensions of the displayed projections 180 degrees. The `Reset` menu item resets the dimensions to default orientation.



The Projection Menu

The `Projection` menu allows to change between various ways of displaying the results. Currently only segmented projection is supported.



11. Scripting

Great efforts have been made to keep the graphical user interface separated from the actual image processing modules. As a consequence it is straight forward to use the *CCA-fMRI Toolbox*, and its modules, in Matlab® scripts to simplify repetitive analyses. An advantage of running the toolbox in script mode is that repetitive tasks for large analysis series can be automated. Script mode also reduce the memory footprint significantly thereby allowing larger datasets to be analyzed than is the case with the graphical user interface.

Below are two examples of how to use the toolbox in scripts. The first example is a 2-dimensional analysis and the second example is a 3-dimensional analysis. The two sample scripts are located in the `SampleScripts` folder located within the *CCA-fMRI Toolbox* directory. The wrapper functions are located in the main toolbox directory and have names prefixed by `scr`.

NOTE
Make sure that the path to the sample directory has been added to the Matlab® path before trying to use the samples.

Script for 2-dimensional analysis

The script `Sample2DAnalysis.m` uses analysis wrappers functions, also provided in the toolbox, for the actual processing. Their usage and calling parameters and are described in *Appendix A – Scripting wrapper functions*. The wrapper functions used for 2-dimensional analysis are:

1. `scrCreateHemydynModel()`
2. `scrRCCA()`

```
function [CorrelationMap MeanImageVolume] = Sample2DAnalysis()


   % GET IMAGE VOLUME FILES TO PROCESS
   disp('Getting image files...');

   % Query the user for the files in the data set. The is not the usual way
   % of specifying what files to analyze in a script...
   warning off;
   ImageFiles = spm_get([],'.img','Select image files',pwd);
   spm_Headers = spm_vol(ImageFiles);
   warning on;


   % CREATE HEMODYNAMIC RESPONSE MODEL
   disp('Creating hemodyn model...');

   % Setup parameters for balloon model
   BalloonLimits.NeuronEffBase = 0.5;
   BalloonLimits.NeuronEffMinMax = 0.15;
   BalloonLimits.SigDecayBase = 1.2;
   BalloonLimits.SigDecayMinMax = 0.3;
   BalloonLimits.AutoRegBase = 2.4;
   BalloonLimits.AutoRegMinMax = 0.5;
   BalloonLimits.TransTimeBase = 1;
   BalloonLimits.TransTimeMinMax = 0.5;
   BalloonLimits.CapTransTimeBase = 1.0;
   BalloonLimits.CapTransTimeMinMax = 0.5;
   BalloonLimits.CpbRatioBase = 0.01;
   BalloonLimits.CpbRatioMinMax = 0.005;
   BalloonLimits.VolRatioBase = 75;
```

```
BalloonLimits.VolRatioMinMax = 25;
BalloonLimits.MetabolicBase = 0.1;
BalloonLimits.MetabolicMinMax = 0.05;
BalloonLimits.StiffnessBase = 0.3;
BalloonLimits.StiffnessMinMax = 0.1;
BalloonLimits.TissueOxConcBase = 0.1;
BalloonLimits.TissueOxConcMinMax = 0.05;
BalloonLimits.TOScale = 5;
BalloonLimits.RestExtractBase = 0.4;
BalloonLimits.RestExtractMinMax = 0.1;

% In case default values are appropriate use the following instead
% BalloonLimits = GetDefBalloon();

% Setup the paradigm. The model basis function can also be set to 'Gamma
% diff' in which case the structure GammaDiffLimits should replace
% BalloonLimits below.
ParadigmDesign.ModelBasisFunction = 'Balloon';
ParadigmDesign.Name = 'Test paradigm';
ParadigmDesign.TotalSequenceTime = 360;
ParadigmDesign.NumberOfRepetitions = 1;
ParadigmDesign.SamplingInterval = 2.7;
Onsets = [40 120 200 280];
Durations = [40 40 40 40];
ParadigmDesign.RelaxStimEvents = Compatibility('OnsetDuration2EventTime', Onsets, Durations);
HemoDynRespModel = scrCreateHemodynModel(ParadigmDesign, BalloonLimits);


% LOAD IMAGE DATA SET

% Check that the number of files is equal to the number of expected data
% points in the paradigm
NumberOfFiles = size(spm_Headers,1);
SampleCount = size(HemoDynRespModel,1);
if (NumberOfFiles > SampleCount)

   % Adjust the number of images
   disp ('Clipping trailing image files');
   spm_Headers = spm_Headers(1:SampleCount);
elseif (NumberOfFiles < SampleCount);

   % Not enough image files
   disp ('Not enough image files');
```

```
   return;
end;

% Init image storage
disp ('Initiating image storage...');
MemFileName = 'c:\temp\memfile.dat';
ImageStorage = InitImageFileStorage(MemFileName, spm_Headers);

% Load images into memory mapped file
disp ('Loading images...');
LoadImageFiles(spm_Headers, ImageStorage, 0);


% GET MEAN IMAGE AND BRAIN SEGMENTATION MASK OF ORIGINAL IMAGE VOLUMES
disp('Calculate mean image...');

% Calculate the mean image volume
[MeanImageVolume SegMask] = scrCalcMeanImage(spm_Headers);


% SETUP FILTERS
disp('Setting up filters...');
FilterSettings.FilterSize = 7;
FilterSettings.FWHMLowPass = 5;
FilterSettings.FWHMFilter = 2;
FilterSettings.IsoFilterWeight = 0.3;
FilterSettings.Filter3D = false;


% PROCESS THE IMAGE DATA
disp('Processing images...');
CorrelationMap = scrRCCA(HemoDynRespModel, FilterSettings, ImageStorage, SegMask);


% FINISH UP
disp ('Saving results...');

% Put together the results. Warning is disabled to prevent numerous
% "Warning: Cant get default Analyze orientation - assuming
% flipped" -messages in case SPM isn't currently running.
warning off;
Results = GenerateResults(ParadigmDesign, spm_Headers, CorrelationMap, MeanImageVolume, SegMask);
warning on;
```

```
% Save the results to a file that is readable by the CCA-fMRI toolbox's
% Results window.
ResultsFileName = 'c:\temp\Results2D.mat';
save (ResultsFileName, 'Results');

% Done
disp('Done!!!');
```

Script for 3-dimensional analysis

The script `Sample3DAnalysis.m` uses analysis wrapper functions, also provided in the toolbox, for the actual processing. Their usage and calling parameters are described in *Appendix A – Scripting wrapper functions*. The wrapper functions 3-dimensionala analysis are:

3. `scrCreateHemydynModel()`
4. `scrFilterVolumes()`
5. `scrCalcMeanImage()`
6. `scrRCCAPass1()`
7. `scrRCCAPass2()`

```
function [CorrelationMap MeanImageVolume] = Sample3DAnalysis()

   % GET IMAGE VOLUME FILES TO PROCESS
   disp('Get image files...');

   % Query the user for the files in the data set. The is not the usual way
   % of specifying what files to analyze in a script.
   warning off;
   ImageFiles = spm_get([],'.img','Select image files',pwd);
   spm_Headers = spm_vol(ImageFiles);
   warning on;


   % CREATE HEMODYNAMIC RESPONSE MODEL
   disp('Create hemodyn model...');

   % Setup parameters for balloon model
   BalloonLimits.NeuronEffBase = 0.5;
   BalloonLimits.NeuronEffMinMax = 0.15;
   BalloonLimits.SigDecayBase = 1.2;
   BalloonLimits.SigDecayMinMax = 0.3;
   BalloonLimits.AutoRegBase = 2.4;
   BalloonLimits.AutoRegMinMax = 0.5;
   BalloonLimits.TransTimeBase = 1;
   BalloonLimits.TransTimeMinMax = 0.5;
   BalloonLimits.CapTransTimeBase = 1.0;
   BalloonLimits.CapTransTimeMinMax = 0.5;
```

```
BalloonLimits.CpbRatioBase = 0.01;
BalloonLimits.CpbRatioMinMax = 0.005;
BalloonLimits.VolRatioBase = 75;
BalloonLimits.VolRatioMinMax = 25;
BalloonLimits.MetabolicBase = 0.1;
BalloonLimits.MetabolicMinMax = 0.05;
BalloonLimits.StiffnessBase = 0.3;
BalloonLimits.StiffnessMinMax = 0.1;
BalloonLimits.TissueOxConcBase = 0.1;
BalloonLimits.TissueOxConcMinMax = 0.05;
BalloonLimits.TOScale = 5;
BalloonLimits.RestExtractBase = 0.4;
BalloonLimits.RestExtractMinMax = 0.1;

% In case default values are appropriate use the following instead
% BalloonLimits = GetDefBalloon();

% Setup the paradigm
ParadigmDesign.TotalSequenceTime = 360;
Onsets = [40 120 200 280];
Durations = [40 40 40 40];
ParadigmDesign.RelaxStimEvents = Compatibility('OnsetDuration2EventTime', Onsets, Durations);
ParadigmDesign.NumberOfRepetitions = 1;
ParadigmDesign.SamplingInterval = 2.7;
HemoDynRespModel = scrCreateHemodynModel(ParadigmDesign, BalloonLimits);


% SETUP FILTERS
disp('Setup filters...');
FilterSettings.FilterSize = 7;
FilterSettings.FWHMLowPass = 5;
FilterSettings.FWHMFilter = 2;
FilterSettings.IsoFilterWeight = 0.3;
FilterSettings.Filter3D = true;

% Create the 3D-filters
[IsoFilt AnisoFilt_1 AnisoFilt_2 AnisoFilt_3 AnisoFilt_4 AnisoFilt_5 AnisoFilt_6] = GetFilters3D(FilterSettings);
Filters3D = {IsoFilt AnisoFilt_1 AnisoFilt_2 AnisoFilt_3 AnisoFilt_4 AnisoFilt_5 AnisoFilt_6};


% SETUP TEMPORARY STORAGE
disp('Setup temporary storage...');
```

```
% Specify names of temporary storage used during the processing
FilteredFileName = 'C:\temp\memmapfile1.dat';
RCCAFileName = 'C:\temp\memmapfile2.dat';

% Setup the temporary files (memory mapped files)
VolumeSize = spm_Headers(1).dim(1:3);
VolumeCount = size(spm_Headers,1);
[FilteredStorage RCCAStorage] = InitImageFileStorage3D(VolumeSize, VolumeCount, FilteredFileName, RCCAFileName);


% GET MEAN IMAGE AND BRAIN SEGMENTATION MASK OF ORIGINAL IMAGE VOLUMES
disp('Calculate mean image...');

% Calculate the mean image volume
[MeanImageVolume SegMask] = scrCalcMeanImage(spm_Headers);


% FILTER VOLUMES
disp('Filter image volumes...');

% Filter all volumes with all filters
scrFilterVolumes(spm_Headers, Filters3D, FilteredStorage, RCCAStorage);

% Clean up
clear Filters3D;
clear FilteredStorage;


% RUN 1ST RCCA PASS
disp('1st RCCA pass...');

% Perform 1st RCCA pass reusing the temporary file used during the
% filtering step as output file
SizeRCCAArray = scrRCCAPass1(HemoDynRespModel, FilterSettings.IsoFilterWeight, ...
   RCCAStorage, FilteredFileName, VolumeCount, VolumeSize, SegMask);
clear RCCAStorage;


% RUN 2ND RCCA PASS
disp('2nd RCCA pass...');

% Perform 2nd RCCA pass using the input file from the 1st pass
CorrelationMap = scrRCCAPass2(HemoDynRespModel, FilteredFileName, SizeRCCAArray, VolumeSize, SegMask);
```

```
% FINISH UP
disp ('Saving results...');

% Put together the results. Warning is disabled to prevent numerous
% "Warning: Cant get default Analyze orientation - assuming
% flipped"-messages in case SPM isn't currently running
warning off;
Results = GenerateResults(ParadigmDesign, spm_Headers, CorrelationMap, MeanImageVolume, SegMask);
warning on;

% Save the results to a file that is readable by the CCA-fMRI toolbox's
% Results window.
ResultsFileName = 'c:\temp\scrResults3D.mat';
save (ResultsFileName, 'Results');

% Done
disp('Done!!!');
```

12. Appendix A – Scripting wrapper functions

scrCreateHemodynModel()

Definition

```
[HemoDynRespModel] = scrCreateHemodynModel(ParadigmDesign,
BalloonLimits)
```

Description

Creates a hemodynamic response model based on the settings of `ParadigmDesign`. The model is returned in `HemoDynRespModel`. The `ParadigmDesign` parameter has the following fields:

```
ParadigmDesign.TotalSequenceTime;
ParadigmDesign.RelaxStimEvents;
ParadigmDesign.NumberOfRepetitions;
ParadigmDesign.SamplingInterval;
```

The hemodynamic model created is based on the balloon model using the parameters specified in the `BalloonLimits` structure having the following fields:

```
BalloonLimits.NeuronEffBase;
BalloonLimits.NeuronEffMinMax;
BalloonLimits.SigDecayBase;
BalloonLimits.SigDecayMinMax;
BalloonLimits.AutoRegBase;
BalloonLimits.AutoRegMinMax;
BalloonLimits.TransTimeBase;
BalloonLimits.TransTimeMinMax;
BalloonLimits.CapTransTimeBase;
BalloonLimits.CapTransTimeMinMax;
BalloonLimits.CpbRatioBase;
BalloonLimits.CpbRatioMinMax;
BalloonLimits.VolRatioBase;
BalloonLimits.VolRatioMinMax;
BalloonLimits.MetabolicBase;
BalloonLimits.MetabolicMinMax;
BalloonLimits.StiffnessBase;
BalloonLimits.StiffnessMinMax;
BalloonLimits.TissueOxConcBase;
BalloonLimits.TissueOxConcMinMax;
BalloonLimits.TOScale;
BalloonLimits.RestExtractBase;
BalloonLimits.RestExtractMinMax
```

See Also

```
ValidateDesign.m
ValidateBalloonLimits.m
```

scrFilterVolumes()

Definition

```
scrFilterVolumes(spm_Headers, Filters3D, FilteredStorage,
RCCAStorage)
```

Description

Applies the 3-dimensional filter kernels `Filters3D` to the image volumes defined by `spm_Headers`. `scrFilterVolumes`() uses the memory mapped file `FilteredStorage` for intermediate storage. After the filtering has been performed the final results are rearranged and copied to the memory mapped file `RCCAStorage`, which is used by the subsequent canonical correlation analysis.

`spm_Headers` is a standard SPM image file header vector. `Filters3D` is a cell array holding the different filter kernels that should be applied to the image volumes. `FilteredStorage` and `RCCAStorage` are structures of the type `ImageStorage` having the following fields:

```
ImageStorage.TotalSize
ImageStorage.TotNumOfObjects
ImageStorage.ObjectSize
ImageStorage.CurrentObjectIndex
ImageStorage.RepeatValue
ImageStorage.MemFileHandle
ImageStorage.MemFileHandle.data[].Object
ImageStorage.MemFileHandle.format
ImageStorage.MemFileHandle.offset
ImageStorage.MemFileHandle.repeat
ImageStorage.MemFileHandle.writable
```

See Also

```
InitImageFileStorage3D.m
GetFilters3D.m
Filter3D.m
spm_get()
spm_vol()
```

scrCalcMeanImage()

Definition

```
[MeanImageVol SegMask] = scrCalcMeanImage(spm_Headers)
```

Description

Calculates the mean image volume and the brain segmentation mask for the image files specified by `spm_Headers`. The resulting mean image is returned as a standard Matlab® matrix in `MeanImageVol` and the segmentation mask is returned as a logical matrix in `SegMask`. The `spm_Headers` is the same headers as returned by e.g. `spm_vol()`.

See Also

scrRCCA()

Definition

```
[CorrelationMap MeanImageVolume] = scrRCCA(HemoDynRespModel,
FilterSettings, ImageStorage, Segmask)
```

Description

Performs a 2-dimensional correlation analysis and returns the correlation map in `CorrelationMap` and the mean image in `MeanImageVolume`.

`HemoDynRespModel` is the hemodynamic response model to which the MRI-data should be compared. `FilterSettings` specifies the size and shape of the steerable filters and have the following fields:

```
FilterSettings.FilterSize;
FilterSettings.FWHMLowPass;
FilterSettings.FWHMFilter;
FilterSettings.IsoFilterWeight;
FilterSettings.Filter3D;
```

`ImageStorage` specifies the memory mapped file into which the images to analyzed have been loaded and `SegMask` is the brain segmentation mask as returned by `scrCalcMeanImage()`.

See Also

```
ValidateSteerableFilter.m
InitImageFileStorage.m
LoadImageFiles.m
```

scrRCCAPass1()

Definition

```
[SizeRCCAArray] = scrRCCAPass1(HemoDynRespModel,
IsoFilterWeight, RCCAStorage, OutputFileName, VolumeCount,
ValidVolumeSize, SegMask)
```

Description

Performs the first phase of the 3-dimensional canonical correlation analysis and returns the size of the output data from phase 1 in `SizeRCCAArray`.

`HemoDynRespModel` is the hemodynamic response model to which the MRI-data should be compared. `IsoFilterWeight` specifies the amount of isotropic filtering to add to the anisotropic filtered images (see headline *Steerable Filters…* for more information about this parameters). `RCCAStorage` is the very same memory mapped file storage used by `scrFilterVolumes`() to store filtered image volumes. `OutputFileName` is the name of a temporary file in which `scrRCCAPass1()` can store intermediate results that subsequently will be processed by the second phase of the canonical correlation analysis. If the file doesn't exists it will be created during the first phase. `VolumeCount` specifies the number of image volumes/files in the original SPM data file set, i.e. `size(spm_Headers,1)`. `ValidVolumeSize` specifies the size of the volumes that are valid after filtering. The valid volume size can be calculated the following way using the first image data file in the set to get the original image size:

```
SkipVoxels = fix(FilterSettings.FilterSize/2);
ValidVolumeSize = spm_Headers(1).dim(1:3) - SkipVoxels * 2;
```

`SegMask` is the brain segmentation mask as returned by `scrCalcMeanImage()`.

See Also

`spm_get()`
`spm_vol()`
`Filter3D.m`
`RestrictedCCA.m`

scrRCCAPass2()

Definition

```
[CorrelationMap] = scrRCCAPass2(HemoDynRespModel,
InputFileName, SizeRCCAArray, ValidVolumeSize, SegMask)
```

Description

Performs the second and final step of the 3-dimensional canonical correlation analysis and returns the correlation map in `CorrelationMap`. The correlation map has the same size as an image volume in the original SPM data set analyzed. `HemoDynRespModel` is the hemodynamic response model earlier created by `scrCreateHemodynModel`(). `InputFileName` is the name of the temporary file used in phase 1 as output file (`OutputFileName`). `SizeRCCAArray` is the size of the input data object as returned by `scrRCCAPass1`(). `ValidVolumeSize` is the same valid volume size used by `scrRCCAPass1`(). `SegMask` is the brain segmentation mask as returned by `scrCalcMeanImage()`.

See Also

`RestrictedCCA.m`

13. Appendix B - Software License

The GNU General Public License (GPL)

Version 2, June 1991

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free

program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is

no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a

special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS