GIFT v1.3c Functions

Srinivas Rachakonda, Eric Egolf and Vince Calhoun

February 20, 2007

Contents

1 Introduction

This document explains some of the main functions used in giftv1.3c and is divided into four sections like pre-analysis, analysis, miscellaneous and display functions.

2 Pre-Analysis Functions

2.1 icatb_defaults

Function containing the global variables that are required for the analysis or display.

2.2 icatb_enterParametersGUI

Function used to get the parameters required for the analysis. The input parameters for this function are optional and they are as follows:

- sub_file - Subject MAT file. If you have already selected the data, a Subject MAT file will be created.
- inputFile - Input M file for batch analysis.

Usage:

- GUI:

  `Type icatb_enterParametersGUI at the command prompt.`

- Batch:

  `Type icatb_enterParametersGUI([], inputFile) at the command prompt.`

  Where inputFile is the full file path of the input file. Example input files like `Input_data_subjects_1.m` and `Input_data_subjects_2.m` are located in directory `icatb/icatb_batch_files`.

Result: ICA parameter file with the suffix `ica_parameter_info.mat` will be created in the selected output analysis directory.

2.3 icatb_estimate_dimension

Function used for dimensionality estimation. The input to this function is data files information.

Usage:

`Type [numComp, mdlVal, aicVal] = icatb_estimate_dimension(files); at the command prompt.`

Where files is a character array of file names. If you don't give the input file names this will let you select the files using a file selection window. If you have already selected the data files, a parameter file will be created. You can use this parameter file information to do the estimation. The following are the steps:

`Load parameter file. This will create sesInfo variable.`

`Type [estCompVec, estimateComp, mdlVal, aicVal] = icatb_estimateCompCallback([], [], sesInfo ); at the command prompt.`

Result:

- numComp - Number of estimated components.
- mdlVal - Vector of MDL values.
- aicVal - Vector of AIC values.

3 Analysis Functions

3.1 icatb_runAnalysis

Function used for running the analysis using the parameter file information.

Usage:

- GUI:

  `Type icatb_runAnalysis at the command prompt and this will let you select the parameter file.`

- Batch:

  `Load the parameter file and this will create sesInfo variable. Type icatb_runAnalysis(sesInfo, 1); at the command prompt and this will run all the analysis steps.`

Result: ICA parameter file with the suffix `ica_parameter_info.mat` will be created in the selected output directory. The parameter file now contains the information about the output files which will be used while displaying the results. You can also run each step from the command prompt. The following are the steps involved:

- Parameter Initialization:

  `Type icatb_parameterInitialization(sesInfo, []); or icatb_runAnalysis(sesInfo, 2); at the command prompt to do parameter initialization.`

- Data Reduction:

  `Type icatb_dataReduction(sesInfo, []); or icatb_runAnalysis(sesInfo, 3); at the command prompt to do data reduction.`

- Calculate ICA:

  `Type icatb_calculateICA(sesInfo, []); or icatb_runAnalysis(sesInfo, 4); at the command prompt to calculate ICA.`

- Back Reconstruction:

  `Type icatb_backReconstruct(sesInfo, []); or icatb_runAnalysis(sesInfo, 5); at the command prompt to do back reconstruction.`

- Calibrate Components:

  `Type icatb_calibrateComponents(sesInfo, []); or icatb_runAnalysis(sesInfo, 6); at the command prompt to calibrate components.`

- Group Stats:

  `Type icatb_groupStats(sesInfo, []); or icatb_runAnalysis(sesInfo, 7); at the command prompt to do statistics.`

3.2 icatb_calculate_pca

Function used for doing PCA and whitening.

Usage:

```
[pcasig, dewhiteM, Lambda, V] = icatb_calculate_pca(data, numpc,
mask_ind, '', imageDim);
```

Input:

- data - 2D matrix whose dimensions are number of voxels (regions in the mask) by time points.
- numpc - Number of principal components to be extracted from the data.
- mask_ind - Vector containing the regions in the mask.
- imageDim - Structure containing the fields 'xdim' (X dimension), 'ydim', (Y dimension), 'zdim' (Z dimension), 'tdim' (Time dimension). Example is given below:
  - imageDim = struct('xdim', 53, 'ydim', 63, 'zdim', 34, 'tdim', 220);

Output:

- pcasig - Reduced data whose dimensions are number of components by voxels.
- dewhiteM - de-whitening matrix.
- Lambda - Eigen values matrix.
- V - Eigen vectors.

3.3 icatb_detrend

Remove trend from the data.

Usage:

```
[y] = icatb_detrend(y, 1, size(y, 1), detrendNumber);
```

Input:

- y - 2D data matrix where rows correspond to observations.
- detrendNumber - Options are 0, 1, 2 or 3.
  - 0 - Mean is removed.
  - 1 - Mean and linear trend is removed.
  - 2 - Sine one cycle, cosine one cycle, mean and linear trend is removed.
  - 3 - Sine two cycles, cosine two cycles, sine one cycle, cosine one cycle, mean and linear trend is removed.

Output: y - 2D data matrix obtained after removing the trend.

3.4 icatb_gaussSmooth1D

Data smoothing is done using a Gaussian kernel.

Usage:

```
data = icatb_gauss_smooth1D(data, fwhm);
```

Input:

- data - 2D data matrix where rows correspond to observations.
- fwhm - Smoothing factor.

Output: data - smoothed data

3.5 icatb_new_icaOptions

Function used to return the ICA options.

Usage:

```
[ICA_Options] = icatb_new_icaOptions(dataSize, algorithm_index,
handle_visibility);
```

Input:

- dataSize - Vector containing data dimensions like [20, 5826] where the first value contains the number of components and the second value contains the number of voxels.
- algorithm_index - ICA algorithm number. See icatb_icaAlgorithm for the corresponding ICA algorithm number.
- handle_visibility - Options are 'on' or 'off'.If handle_visibility is set to 'on' GUI will open showing the options for the ICA algorithm.

Output: ICA_Options - Cell array containing the answers.

3.6 icatb_icaAlgorithm

Function used to calculate ICA based on the selected ICA algorithm.

Usage:

```
[icaAlgo, W, A, icasig_tmp] = icatb_icaAlgorithm(algorithm_index,
data, ICA_Options);
```

Input:

- algorithm_index - ICA algorithm number.
- data - 2D matrix whose dimensions are number of components by voxels.
- ICA_Options - Cell array containing the ICA options.

Output:

- icaAlgo - ICA algorithms available.
- W - Un-mixing matrix.
- A - Mixing matrix.
- icasig_tmp - Source signals.

3.7 icatb_kurtosis

Function used to calculate the kurtosis of the data.

Usage:

```
[kurtosisValue, detrended_data] = icatb_kurtosis(data, 1, size(data,
1), detrendNumber);
```

Input:

- data - 2D data matrix where rows correspond to observations.
- detrendNumber - Detrend number. Options are 0, 1, 2 and 3.

Output:

- kurtosisValue - Kurtosis value.
- detrended_data - Data obtained after removing the trend.

3.8 icatb_multipleRegression

Function used for doing multiple regression.

Usage:

```
[rSquare_stat, b, ModelIndices, otherIndices, linearRegress,
removeTrend, ica, modelX, subject_partial_corr, partialCorrSlopes] =
icatb_multipleRegression(model, ica, size(model, 2), 1, size(model,
1), detrendNumber);
```

Input:

- model - Model matrix whose dimensions are number of points by number of regressors.
- ica - Observation vector.
- detrendNumber - Detrend number. Options are 0, 1, 2 and 3.

Output:

- rSquare_stat - R-square statistic value.
- b - Slopes of the regressors.
- ModelIndices - Model indices.
- otherIndices - Nuisance indices.
- linearRegress - Line fit.
- removeTrend - Matrix containing the trend to be removed.
- ica - Detrended data.
- modelX - model matrix.
- subject_partial_corr - Partial correlation values.
- partialCorrSlopes - Slopes of the regressors obtained after calculating partial correlation.

3.9 icatb_percentSignalChange

Function used for calculating percent signal change.

Usage:

```
[icasig, A] = icatb_percentSignalChange(origData, icasig, A);
```

Input:

- origData - 2D data matrix whose dimensions are number of voxels by time points.
- icasig - 2D sources matrix whose dimension are number of components by voxels.
- A - Time course matrix whose dimensions are number of time points by components.

Output:

- icasig - Component sources are scaled to represent percent signal change.
- A - Time courses are scaled to represent percent signal change.

3.10 icatb_convertImagesToZscores

Function used for converting spatial maps to z-scores.

Usage:

```
[im] = icatb_convertImageToZScores(icasig);
```

Input: icasig - Spatial maps whose dimensions are number of components by voxels.

Output: im - Spatial maps converted to z-scores.

3.11 icatb_convertToZScores

Function used for converting data to z-scores.

Usage:

```
[y] = icatb_convertToZScores(y);
```

Input: y - Data matrix where columns corresponds to observations.

Output: y - Data converted to z-scores.

4 Miscellaneous Functions

4.1 icatb_batch_file_run

Function used for running the group ICA in batch mode.

Usage:

```
icatb_batch_file_run(inputFile);
```

Input:

- inputFile - Input file containing the information about the parameters required for group ICA. Example batch files like `Input_data_subjects_1.m` and `Input_data_subjects_2.m` are located in directory `icatb/icatb_batch_files`.

4.2 SPM Functions

SPM volume functions are used for reading and writing image data. SPM2 and SPM5 functions are located in directory `icatb/icatb_spm2_files` and `icatb/icatb_spm5_files`.

4.3 icatb_loadData

Function used for reading image (3D analyze, 4D analyze and 4D Nifti) data.

Usage:

```
[data, HInfo] = icatb_loadData(files);
```

Input: files - Character array of file names.

Output:

- Data - 4D image data.
- HInfo - Header information structure containing fields like 'V', 'VOX' and 'DIM'. The explanation of each field is as follows:
  - V - Volume information of files.
  - VOX - Voxel size.
  - DIM - X, Y and Z dimensions.

4.4 icatb_write_vol

Function used for writing the image data.

Usage:

```
icatb_write_vol(V, data);
```

Input:

- V - Volume information of the file.
- data - 3D image data.

4.5 icatb_calculate_eventAvg

Function used for calculating event average.

Usage:

```
eventAvg = icatb_calculate_eventAvg(time_course, interpFactor,
TR, windowSize, selectedOnset);
```

Input:

- time_course - Time course vector.
- interpFactor - Interpolation factor. Default is 5.
- TR - TR of the experiment.
- windowSize - Window size in seconds. Default is 30 seconds.
- selectedOnset - Onset timings.

4.6 icatb_removeArtifact

Function used for removing component/components from the data.

Usage:

```
icatb_removeArtifact;
```

4.7 icatb_selectEntry

Function used for selecting files or directory.

Usage:

- Directory:

```
selected_directory = icatb_selectEntry('typeEntity', 'directory',
'typeSelection', 'single' 'title', 'Select a directory');
```

- Files:

```
selected_files = icatb_selectEntry('typeEntity', 'file',
'typeSelection', 'multiple', 'title', 'Select files ', 'filter',
'*.img');
```

4.8 icatb_inputDialog

Input dialog box.

Usage:

```
answer = icatb_inputDialog('inputText', inputText,
'handle_visibility', handle_visibility, 'title', title_string);
```

Input: Inputs must be in pairs. The following are the variables involved:

- inputText - Array of structures containing fields like 'promptString', 'uiType', 'answerString', 'dataType', 'tag' and 'enable'. Each option is explained below:
  - promptString - Prompt text for the input control.
  - uiType - Type of uicontrol like 'edit' or 'popup'.
  - answerString - Cell array containing answers.
  - dataType - Options are 'string' or 'numeric'.
  - tag - Name for the control.
  - enable - Options are 'on' or 'off'.
- handle_visibility - Options are 'on' or 'off'.
- title_string - Title for the input dialog box.

Output: answer - Cell array containing the answers.

Example:

```
numParameters = 1;

% define all the input parameters in a structure
inputText(numParameters).promptString = 'Select colormap';
inputText(numParameters).uiType = 'popup';
inputText(numParameters).answerString = {'hsv', 'cool', 'winter'};
inputText(numParameters).dataType = 'string';
inputText(numParameters).tag = 'colormap';
inputText(numParameters).enable = 'on';

numParameters = numParameters + 1;

% define all the input parameters in a structure
inputText(numParameters).promptString = 'Number of images';
inputText(numParameters).uiType = 'edit';
inputText(numParameters).answerString = '120';
inputText(numParameters).dataType = 'numeric';
inputText(numParameters).tag = 'num_images';
inputText(numParameters).enable = 'on';

answer = icatb_inputDialog('inputText', inputText,
'handle_visibility', 'on', 'title', 'Select answers ...');
```

4.9 icatb_listdlg

List dialog box.

Usage:

```
[getIndex] = icatb_listdlg('PromptString', prompt, 'SelectionMode',
'multiple', 'ListString', sel_refnames, 'movegui', 'center',
'windowStyle', 'modal', 'title_fig', title_fig);
```

Input:

- prompt - Text displayed above listbox.
- sel_refnames - Cell array containing the list.
- title_fig - Title for the figure.

Output: getIndex - Selected indices.

Example:

```
[getIndex, name_button] = icatb_listdlg('PromptString', 'Select
colormaps', 'SelectionMode', 'multiple', 'ListString', {'hsv',
'winter', 'grey', 'cool', 'summer'}, 'movegui', 'center',
'windowStyle', 'modal', 'title_fig', 'List Dialog Box');
```

5 Display functions

5.1 icatb_displayGUI

icatb_displayGUI function is used to open display GUI.

5.2 icatb_componentExplore

icatb_componentExplore function is used to display the components of a particular viewing set.

5.3 icatb_orthoViewer

icatb_orthoViewer displays axial, saggital and coronal views of a specific component of a particular viewing set.

5.4 icatb_compositeViewer

icatb_compositeViewer is used to overlay multiple components of a particular viewing set.

5.5 icatb_batch_display

Function used to display components using a batch file. Display methods supported are Component Explorer, Composite Viewer and Orthogonal Viewer. Example input file `Input_data_display_1.m` is in directory `icatb/icatb_display_functions`.

Usage:

```
icatb_batch_display(inputFile);
```

5.6 icatb_resizeImage

Function used for resizing images.

Usage:

```
[images] = icatb_resizeImage(structVol, compFile, 'axial', [],
file_numbers);
```

Input:

- structVol - Volume of structural file.
- compFile - Character array containing file names.
- file_numbers - Vector containing file numbers that need to be resized.

Output: images - 4D image data where the first file represents data of structural file.

Example:

```
compFiles = str2mat('Visuomotor_mean_component_ica_s_all_004.img',
'Visuomotor_mean_component_ica_s_all_008.img');

file_numbers = [1 2];

structVol = icatb_spm_vol_nifti(which('nsingle_subj_T1_2_2_5.img'),
1);

[images] = icatb_resizeImage(structVol, compFiles, 'axial', [],
file_numbers);
```

```
structuralData = images(1, :, :, :); % First image is structural data

compData = images(2:end, :, :, :); % Resized images
```

5.7 icatb_showTimeCourses

Function used for showing multiple plots using sliders.

Example:

```
data = [ones(100, 1), sin(rand(100, 1)), cos(rand(100, 1)),
tan(rand(100, 1)), rand(100, 1)];

numPlots = size(data, 2);

for nPlots = 1:numPlots
    timeCourseStruct.sub(nPlots).sess(1).tc = data(:, nPlots); % Y Axis
    xAxis.sub(nPlots).sess(1).tc = (1:length(data(:, nPlots))); % X Axis
end

numSubjects = numPlots;

numSessions = 1;

% Show Time courses using slider
icatb_showTimeCourses('TC', timeCourseStruct, 'numsubjects',
numSubjects, 'numsessions', numSessions, 'titlefig', 'Showing
Plots', 'meandisplay', 'no', 'SEM', 'no', 'timecourse_color', 'm');
```