fMRI Signal Processing Toolbox v1.0 Documentation

Ameera X. Patel

June 29, 2012

Acknowledgements

First, a special thank you to Prantik Kundu for all his helpful advice and for providing speedypp.py (and the original DVARS algorithm), without which this toolbox would be much slower!

Second, a thank you to Petra Vértes for all her helpful advice throughtout the development of this toolbox, and for testing the early editions.

Next, thanks to all the original members of the 'Motion task force' (sepia tinted photographs available) for helpful feedback at Tuesday morning meetings.

Finally, a thank you to The Wellcome Trust (TMAT Programme) for funding my work.

Contents

1 Introduction

Welcome to the fMRI Signal Processing Toolbox (v.1.0, beta release). This toolbox contains a number of programs to guide you from raw data through to fully processed fMRI signal ready for analysis in a matter of minutes.

1.1 Document Usage

This document contains detailed descrptions of all functions included in the toolbox along with example usage. If you want to dive straight in and start using the toolbox, please read through the Introduction, section 2 on Installation and Setup and then have a quick look through the "Example Usage" subsections for the functions you intend to use.

1.2 Toolbox Usage

All programs in this toolbox (with the exception of speedypp.py which executes directly through python) have been written with bash script wrappers. All programs are therefore executed through the terminal. Each program has an in-built help file which is displayed if any of the options are entered incorrectly, or if you request it using the -h flag.

This toolbox was designed with two usages in mind:

1. As an end-to-end product where the user starts with raw (unprocessed) fMRI data and uses the functions in this toolbox sequentially.
2. As a set of modular functions for diagnosing pre-processing-related effects in existing pre-processed and/or parcellated data.

For use with raw fMRI data: This is the easiest way of interacting with this toolbox. Simply execute each program with one line from the terminal (see below for detailed explanations on each of the functions individually). The end-to-end process for one dataset will take less than 8-10 minutes (often less than 5 minutes)! Figure 1 below shows an overview of how the main functions in this toolbox interact. You need not worry about the data format and alternate algorithm warnings downstream of pre-processing and parcellation if you are using the functions in this toolbox sequentially. These only apply if specific toolbox modules are being selected for use with existing pre-processed data.

For use with existing pre-processed or parcellated data: The individual modules are written to allow compatability with a wide range of pre-processing methods. However, there are a few important points to note if you are planning to use the toolbox in this way:

1. If you wish to replicate the Power *et al.*, 2012 method, this relies on a whole-brain mode or median normalisation step during pre-processing. If you have not done this during pre-processing, then you will need to specify the "alternate algorithm" flag, "-a" for the relevant functions (principally dvars.sh and dbold.sh). The alterate algorithm relies on time series data that have NOT been de-meaned (either manually or indirectly through a band-pass).
2. File formats. Specific text file formats are required for these programs to work, as the usage of python in these scripts cannot deal with formats such as csv. If you have not used speedypp.py or parcellate.sh in this toolbox to pre-process and parcellate your data, then please pass all your text files through the program reformat.sh before using them in in the rest of this toolbox. This program will re-write the files with tab-separation, and detect the orientation of the matrix, transposing it if necessary. If you do not first use reformat.sh, the other functions may not work. Future releases of this toolbox will have built-in functions that will automatically detect alterate file formats and correct them.

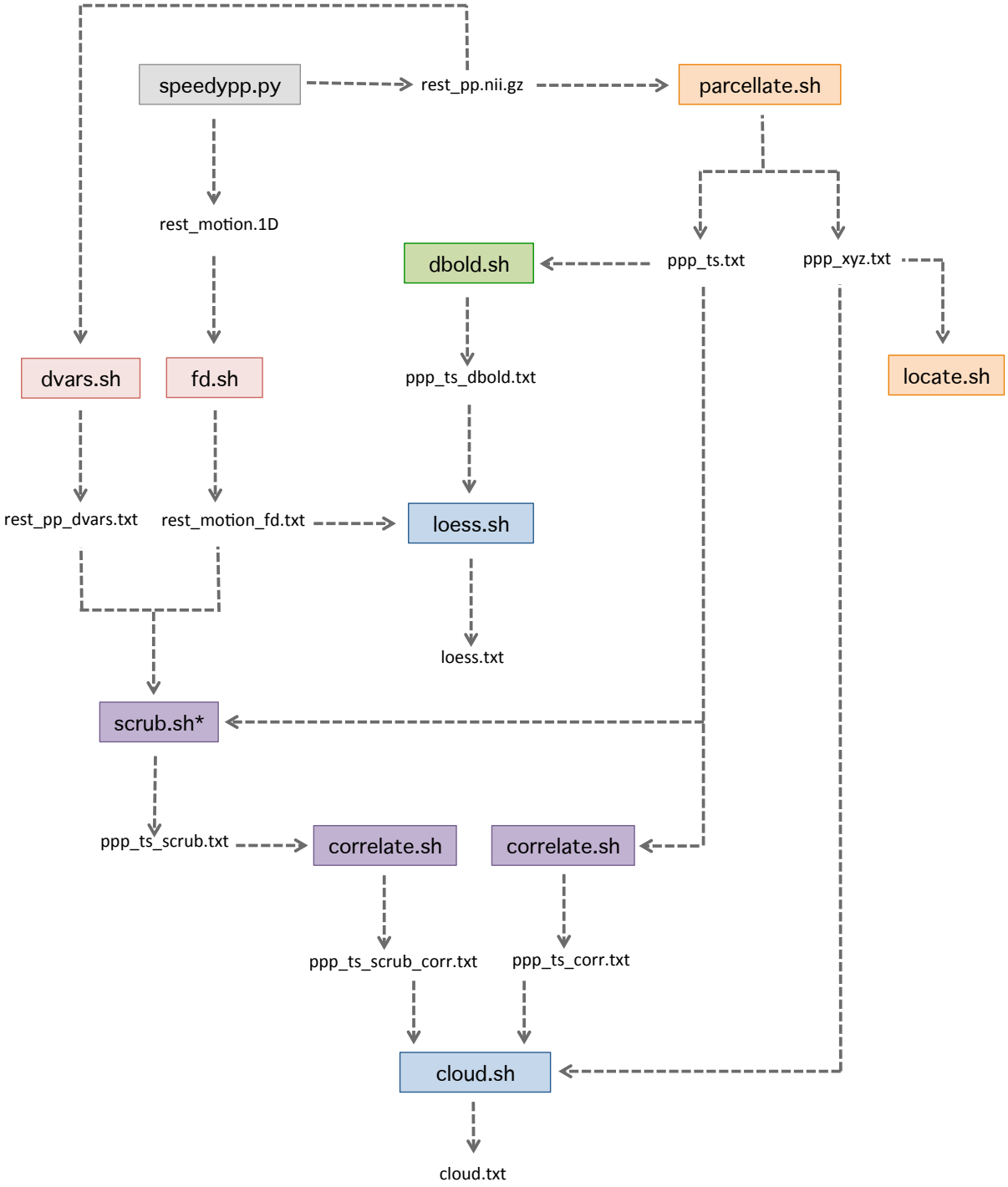

Figure 1: fMRI Signal Processing Toolbox Flowchart. Please note, some programs are not included in this diagram, however, all functions are detailed in individual sections below. *scrub.sh also has a sister function called randscrub.sh which may be of interest (see section 9 below)

2 Installation and Setup

This short guide will take you through the steps required to run this toolbox on your computer.

2.1 Setting up this Toolbox

You have already completed the first step by downloading the toolbox! All you need to do is to copy this folder into your HOME directory. All scripts in this toolbox are configured to search for this folder in your HOME directory, so if this toolbox is located elsewhere, the programs will complain.

2.2 Installing AFNI/SUMA Software

AFNI/SUMA Sofware is required for fMRI pre-processing and parcellation. Setting up AFNI is very simple. Download the latest version of the AFNI software from the website below. Create a directory called "abin" in your home directory and copy the contents of the downloaded folder into this new directory $HOME/abin/. Please note that AFNI is not available for Windows.

http://afni.nimh.nih.gov/afni/download/afni/releases/latest

Next you will need to set the paths. If you are using UNIX or Linux (with standard shells e.g. bash, sh, ksh etc), copy the following command into your terminal. This will set the AFNI path directory so the commands can be called directly from the fmri_spt directory:

export PATH=$PATH:/$HOME/abin

If you are using csh and tcsh shells, then use setenv instead of export.

To view a NIfTI file or to view one of the templates/atlases in this toolbox, using the afni viewer, cd into the relevant directory and type the prompt "afni" into the terminal. This will load the afni viewer.

2.3 Installing Python

You will need a working version of Python plus a number of its packages (NumPy, matplotlib, NiBabel, SciPy). The Enthought distribution of python comes with a range of packages, including NumPy, SciPy and matplotlib. This is available free for academic download.

http://www.enthought.com/products/edudownload.php

NiBabel (the neuroimaging package for python) can be downloaded from the following link.

http://nipy.sourceforge.net/nibabel/installation.html#installation

To test whether python is running, type "python" into your terminal and hit enter. If installation has been succesful, then you will enter the python interactive environment and will see a new command prompt ">>>". To check whether the packages have installed, type the following commands into your python prompt:

```
>>> import numpy
>>> import matplotlib
>>> import nibabel
```

If you are returned with an "ImportError" for any of these commands, the packages have not been successfully installed. Check compatibility of the package versions you have installed with the version of python you are running and re-install. [Note the exit command for python is ctrl D].

2.4 Installing and configuring MATLAB

MATLAB is available free from the University Computing Service. If you have MATLAB installed, but have never run matlab from your terminal, you will need to add MATLAB to your .bash_profile. To check whether

MATLAB has already been added to your .bash_profile, type "matlab" into your terminal. If MATLAB does not open, then type:

export PATH=$PATH:/Applications/MATLAB_R2012a.app/bin
[You will need to amend that path according to your MATLAB version and where the MATLAB application is located on your computer].

If this does not set the path, then please edit your .bash_profile using a text editor (e.g. open -e .bash_profile).

In order for matlab to recognise the toolbox, you'll need to set the path to the toolbox within MATLAB. You can do this by opening MATLAB and going to File >"Set Path". Find the toolbox folder and add the folder plus subfolders.

You are now ready to start using the fMRI Signal Processing Toolbox v.1.0. The rest of this document contains tutorials for all of the functions included in this toolbox.

ENJOY!

3 fMRI Pre-processing (speedypp.py)

3.1 Program Description

This program pre-processes anatomical and functional MRI data, ready for analysis, in under 5 minutes.

The basic steps involved are as follows:

1. Anatomical dataset pre-processing:
   - Deoblique (if dataset is oblique)
   - Skullstrip (if specified at input) [Recommended]
2. Functional data pre-processing:
   - Slice-time correction (acquisition pattern specified at input).
   - Deoblique (if dataset is oblique)
   - Zeropad (if specified at input) [Recommended]
   - Volume registation. This is a rigid body alignment
   - Co-registration of functional and anatomical images. This is an affine transformation
   - Spatial blur (radius specified at input)
   - Whole-brain intensity normalisation (to median = 1000)
   - Build regressor matrix. This first involves segmentation of the downsampled anatomical dataset to obtain csf and white matter signals. The matrix is built with the regressors specified at input. The –rall option will regress motion parameters, motion parameter derivatives and CSF signal only.
   - Band-pass (frequency bands defined at input), regress and despike (if options specified at input) in one step.

This script is executed directly from the terminal in python. speedypp.py compiles a bash script based on the options specified and outputs this to the working directory, along with a folder containing the intermediate files.

There are four important output files which you should be aware of:

1. The pre-processed anatomical image. This NIfTI file will have the output suffix _ns appended to the input anatomical file name (e.g. mprage_ns.nii.gz)
2. The motion parameters. This file is output from the volume registration. The suffix _motion is appended to the functional input file name (e.g. rest_motion.1D)
3. The fully pre-processed functional image. This NIfTI file will have the suffix _pp appended to the input functional dataset file name (e.g. rest_pp.nii.gz)
4. An intermediate pre-processed functional image. This NIfTI file is generated before the last step of band-pass/regress/despike, and has the suffix _in appended to the input file name (e.g. rest_in.nii.gz).

3.2 Example Usage

The following options are available

OPTIONS:

-h,	–help	Show the help message
-d,	–DSINPUT	Functional dataset, e.g. rest.nii
-f,	–FWHM	Spatial blur - target smoothness e.g. -f 8mm [DEFAULT=5mm]
-a,	–ANAT	Anatomical dataset e.g. mprage.nii
-o		Oblique acquitision (Flag)

Denoising options

–rall	Regress all (except white matter)
–rmot	Regress motion parameters [Recommended]
–rmotd	Regress motion derivatives [Recommended]
–rcsf	Regress CSF signal
–rwm	Regress white matter signal [Not recommended]
–lowpass=LOWPASS	Low pass frequency in Hz e.g. 0.1 [DEFAULT=None]

Extended Processing options:

–basetime=BASETIME	Time to steady-state equilibration in seconds [DEFAULT=0]
–betmask	More lenient functional masking (using BET)
–skullstrip	Skullstrip anatomical image [Recommended]
–despike	Despike data. Good for datasets with spikey motion artefact
–TR=TR	The TR. [DEFAULT=Read from input dataset header]
–tpattern=TPATTERN	Slice acquisition pattern (e.g. alt+z, see 3dTshift –help) [DEFAULT= Read from header. Correction skipped if not found]
–zeropad=ZEROPAD	Zeropadding option. -z N means add N slabs in all directions [DEFAULT=15]
–highpass=HIGHPASS	Highpass filter in Hz [DEFAULT=0.02] lll
–align_base=ALIGN_BASE	Base EPI for allineation
–align_interp=ALIGN_INTERP	Interpolation method for allineation
–align_args=ALIGN_ARGS	Additional arguments for 3dAllineate functional to anatomical alignment.
–label=LABEL	Extra label to tag the SpeedyPP folder
–keep_int	Keep pre-processing intermediates [DEFAULT=delete intermediates to save space]
–OVERWRITE	If spp.xyz directory exists, overwrite
–CLUSTER	Cluster mode (disables greedy multithreading)
–exit	Generate script and exit

E.g. python ∼/fmri_spt/speedypp.py -d rest.nii -a mprage.nii -f 6mm -o –skullstrip –zeropad=50 –despike –tpattern=seq+z –lowpass=0.1 –rall

would produce a bash script and directory of intermediate files in the working directoy, along with the imporant output files mprage_ns.nii.gz, rest_motion.1D, rest_in.nii.gz and rest_pp.nii.gz

4 Parcellation (parcellate.sh)

Runtime: <1 minute

parcellate.sh is designed to take functional and anatomical datasets in NATIVE SPACE and warp these to TLRC (standard) space so that the functional image can be divided into regions. The parcellation templates included in this package are based on the Zalesky algorithm which randomly sub-parcellates an anatomical atlas into roughly evenly sized regions.

4.1 Standard Space Transformation

Both anatomical and functional datasets are required for this program. In general, using just a functional image will produce sub-optimal results. The script first warps the anatomical dataset to a standard space brain (specified at input) and then computes the transformation of the functional dataset to the anatomical.

Please note that the input functional and anatomical datasets MUST be in NATIVE SPACE. If they are not, then this program will most likely not work.

There are 4 standard space brains to chose from:

1. MNI - The Montreal Neurological Institute template (average of 152 brains), T1 weighted. [DEFAULT].
2. N27 - Based on the "Colin Brain" scanned 27 times.
3. ICBM - The International Consortium for Brain Mapping template (average of 452 brains).
4. EPI - The EPI template from SPM.

The templates all have different resolutions, depending on the number of brains that have been averaged. The MNI template produces the best results overall so is the default. If warping does not converge with one of the templates, and you find some unexpected intermediate files in your working directoy, then please try a different template. A number of features relating to the way the data has been acquired can cause standard space registration to fail (e.g. distance from standard space template, or image clipping).

It is strongly recommended that you check the functional and anatomical registration to standard space after running this code to ensure that the results with your datasets are good.

To view the standard space templates, go to the following directory: ~/fmri_spt/templates/standard/

4.2 Parcellation

The parcellation templates in this toolbox are based on the AFNI TT_N27_EZ_ML atlas. The templates were generated using the Zalesky algorithm which randomly subparcellates the anatomical template. There are four parcellation templates available in v.1.0 of this toolbox:

1. AT116.nii - This is an anatomical parcellation with 116 regions
2. AT150.nii - Template with 150 parcels
3. AT230.nii - Template with 230 parcels
4. AT325.nii - Template with 325 parcels.

To view these templates, go to the following directory: ~/fmri_spt/templates/parcel_temps/

All four parcellation template options will work with all standard space template options. However, the MNI template produces the best results overall, so has been set as the default.

All templates use the talairach (TLRC) coordinate system. For the non-talairach standard space brains (e.g. MNI), the coordinates for these templates have been transformed to the TLRC coordinate system, though the brain remains the same. The transformations used are described in the next section. This was done to enable a range of template brains (traditionally described in different spaces) to be used whilst maintaining consistency of coordinate systems. The coordinates output are thus RAI. To switch from RAI to LPI (a widely used axis

orientation system) change the sign of the x and y coordinates. Please note that the coordinate systems (TLRC, MNI, MNI_ANAT) are separate from the coordinate orientation (RAI, LPI). It just so happens that certain coordinate systems have come to be associated with certain axis orientations. It is easy to switch between orientations, but switching between coordinate systems requires transformations.

4.3 Extra Features

If any zero parcels are detected, this program will warn the user and output a text file to the working directory called "par_error.txt". This contains a list of parcels which contain no time series information after parcellation. If this file is output, please DO NOT use this dataset for further motion diganosis until you have corrected the error.

Zero parcel errors can occur for a number of reasons. First is that the autotalairach procedure has failed and the brain has been warped to standard space incorrectly. If this has occurred, it will be obvious from visual inspection of the brains. The second is that part of the brain has been clipped. This can occur because of poor acquisition or because of insufficient zero-padding in the pre-processing stages. If the latter is the case, please re-pre-process your functional image, otherwise exclude the dataset for motion diagnosis using subsequent functions in this toolbox.

4.4 Example usage

The following bash options are available:

OPTIONS:

-h Show help message (Flag)
-i Pre-processed functional dataset (in native space) [REQUIRED]
-a Pre-processed anatomical dataset (in native space) [REQUIRED]
-t Standard space template options (MNI, N27, ICBM, EPI) [DEFAULT=MNI]
-p Parcellation template (AT116.nii, AT150.nii, AT230.nii, AT325.nii) [DEFAULT=AT116.nii]
-o Output prefix [DEFAULT=ppp]
-v Verbose (Flag)

The only required inputs are -i and -a. If the other inputs are not specified, they will be set to the DEFAULT options.

E.g., if you want to parcellate input datasets "mprage_ns.nii.gz" and "rest_pp.nii.gz" into 230 regions using the MNI template, you would type the following into your terminal

```
bash ~/fmri_spt/parcellate.sh -a mprage_ns.nii.gz -i rest_pp.nii.gz -p AT230.nii
```

This will output the following:

- Anatomical dataset in TLRC space "mprage_ns_at.nii"
- Functional dataset in TLRC space "rest_pp_at+tlrc" (Note, this is an AFNI-specific output equivalent to a NIfTI file. It can be read by the AFNI viewer in the same way as a NIfTI file.
- "tlrc_log" directory, containing talairach log files.
- Average time series for each parcel "ppp_ts.txt" (columns = parcel number, rows = frame number)
- Number of voxels in each parcel "ppp_n.txt" (columns = parcel number, rows = number of voxels in each parcel)
- Centroid coordinates for each parcel "ppp_xyz.txt" (columns= parcel number, row 1 = x coordinates, row 2 = y coordinates, row 3 = z coordinates

If there are zero parcels present, then "par_error.txt" will also be output.

5 Coordinate Location (locate.sh)

Runtime: <5 seconds

5.1 Program description

This program allows the user to input coordinates in any space and obtain the brain locations at the focus point and within a 5 mm radius. This function is based on AFNI's whereami and is designed to be used in conjunction with parcellate.sh.

Coordinate systems and transformation: The first output of the program is a summary of the input coordinates transformed to TLRC, MNI and MNI_ANAT coordinate systems. Please note that though the input coordinates may be in RAI orientation (if you used parcellate.sh or are inputting TLRC coordinates), the program will express these in LPI orientation as well, as LPI is another widely recognised format. The only difference will be the signs of the x and y coordinates.

Transformation from TLRC to MNI coordinate systems involves an approximate function (Brett transform). The conversion from TLRC to MNI_ANAT coordinate systems involes a 12 piece-wise linear transform. This allows the program to use atlases that originate in different spaces / coordinate systems.

Atlases: To reduce errors in identifying regions based on one atlas alone, this function will simultaneously output the brain location at [x,y,z] as identified using a range of atlases. Some of these atlases contain macro-labels, but others are more fine-grain. Please use the outputs with caution. The brain regions identified are approximations.

Inputs: If you have used parcellate.sh to compute time series from your pre-processed functional dataset, then the only inputs required are x, y and z coordinates. If you wish to use this function more generally, and would like to input MNI or MNI_ANAT coordinates, please specify the space using the -s flag. The default space is TLRC.

If you wish to redirect the output into a text file, please specify the output flag -o. The program will save the file as x_y_z, where x, y and z are the input coordinates.

5.2 Example Usage

The following bash options are available:

OPTIONS:

-h Show help message (Flag)
-x x coordinate [REQUIRED]
-y y coordinate [REQUIRED]
-z z coordinate[REQUIRED]
-s coordinate system. Please specify either TLRC, MNI or MNI_ANAT [DEFAULT=TLRC]
-o Redirect screen output to text file (Flag)
-v Verbose (Flag)

E.g. if you wish to identify the brain regions at the coordinates [12,12,12], as output by parcellate.sh, and save the output to the working directory, then you would type the following into the terminal:

bash ~/fmri_spt/locate.sh -x 12 -y 12 -z 12 -o

The program will output a file named "12_12_12.txt" into the working directory. See Figure 2 below for an example output.

```
++ Input coordinates orientation set by default rules to RAI
+++++++ nearby Atlas structures +++++++

Original input data coordinates in TLRC space

Focus point (LPI)=
    -12 mm [L],  -12 mm [P],   12 mm [S] {TLRC}
    -12 mm [L],  -13 mm [P],   12 mm [S] {MNI}
    -12 mm [L],  -17 mm [P],   17 mm [S] {MNI_ANAT}

Atlas TT_Daemon: Talairach-Tournoux Atlas
   Focus point: Left  Thalamus
          -AND- Left  Ventral Lateral Nucleus
   Within 3 mm: Left  Medial Dorsal Nucleus
          -AND- Left  Anterior Nucleus
   Within 4 mm: Left  Ventral Anterior Nucleus
          -AND- Left  Ventral Posterior Lateral Nucleus

Atlas CA_ML_18_MNIA: Macro Labels (N27)
   Focus point: Left Thalamus

Atlas CA_MPM_18_MNIA: Cytoarch. Max. Prob. Maps
   Focus point: Th-Prefrontal
   Within 2 mm: Th-Temporal

Atlas CA_PM_18_MNIA: Cytoarch. Probabilistic Maps
   Focus point: Th-Parietal   (p = 0.07)
          -AND- Th-Temporal   (p = 0.24)
          -AND- Th-Prefrontal   (p = 0.57)
          -AND- Th-Visual   (p = 0.18)

Atlas CA_LR_18_MNIA: Left/Right (N27)
   Focus point: Left Brain

Atlas CA_GW_18_MNIA: Cytoarch. Prob. Maps for gray/white matter
   Focus point: grey   (p = 0.33)
          -AND- white   (p = 0.68)

Atlas CA_N27_LR: Left/Right (N27)
   Focus point: Left Brain

Atlas DD_Desai_MPM: Contains the maximum probability maps of Desai DD and FS maps
   Focus point: Left-Thalamus-Proper
   Within 3 mm: Left-Cerebral-White-Matter

Atlas DKD_Desai_MPM: Contains the maximum probability maps of Desai DKD and FS maps
   Focus point: Left-Thalamus-Proper
   Within 3 mm: Left-Cerebral-White-Matter


******** Please use results with caution! ********
******** Brain anatomy is quite variable! ********
******** The database may contain errors! ********
```

Figure 2: Example output of locate.sh with input coordinates [12,12,12] in TLRC space

6 Framewise Displacement (fd.sh)

Runtime: <5 seconds

6.1 Program Description

This program computes the framewise displacement (FD) of your functional dataset, as described in Power *et al.* (2012). This is the sum of the derivatives of the 6 motion parameters output by volume registration. The suffix _fd is appended to the input file name and the output is saved as a text file to the working directory.

If you have not used speedypp.py and parcellate.sh to pre-process your fMRI data, please note that the roll, pitch and yaw movements should be in degrees. In addition, please ensure that the input matrix has the 6 motion parameters represented in columns. reformat.sh can be used to automatically re-format your text files to a compatible format.

This program will also produce a plot of the FD against frame number, if the plot flag -p is specified. The time point for each frame is simply the frame number multiplied by the TR.

6.2 Example Usage

The following options are available:

OPTIONS:

-h Show help message (Flag)
-i Text file containing the motion parameters (reformatted using reformat.sh if speedypp.py has not been used for pre-processing) [REQUIRED]
-v Verbose (Flag)

Extra options:

-p Plot data (Flag)
-c RBG colour of output figure [DEFAULT=black]
Colour options. Please choose from one of the following:
b = blue, g = green, r = red, c = cyan, m = magenta, y = yellow, k = black.

E.g. bash ~/fmri_spt/fd.sh -i rest_motion.1D -p -c b

will output a file called "rest_motion_fd.txt" to the working directory and produce a plot on screen (see Figure 3).

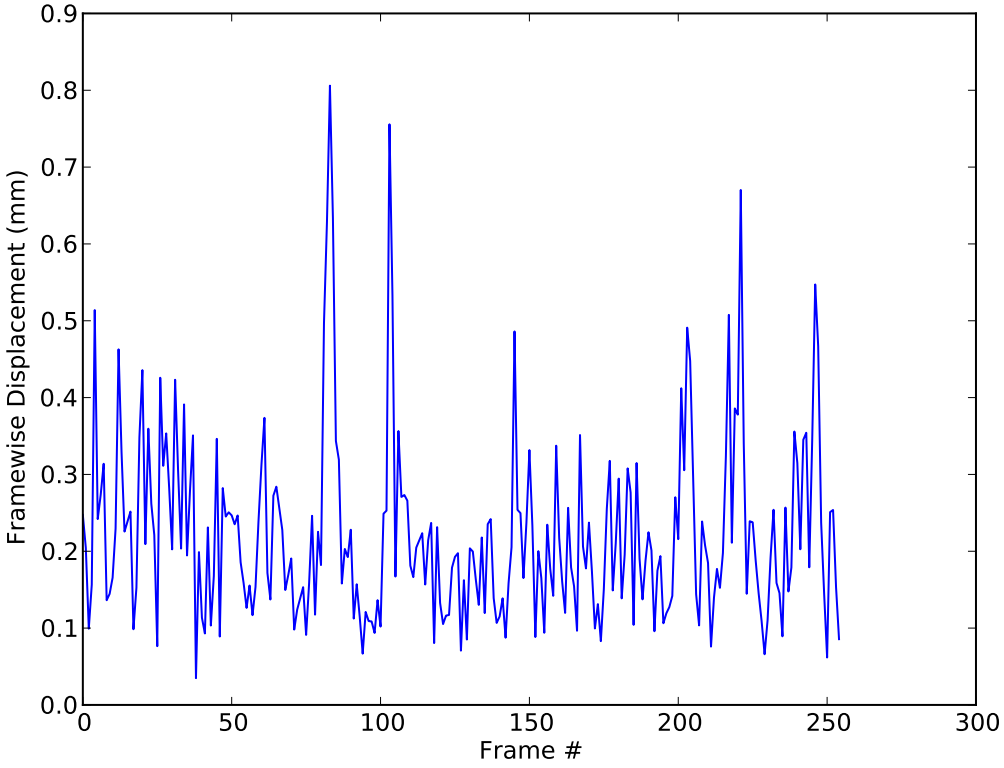

Figure 3: Example output of fd.sh using the -p flag with -c b

7 DVARS (dvars.sh)

Runtime <5 seconds

7.1 Program Description

This program computes the DVARS on a 3D+time NIfTI file, as described in Power *et al.*, 2012. For each voxel, the derivative of percent signal change in voxel intensity is computed, and the root mean square (RMS) of all voxel Δ%BOLD time series is calculated to produce a single representative value (DVARS) per frame.

The units of DVARS here are % signal change x10 as reported by Power *et al.*, 2012. This metric can be computed at various stages of fMRI pre-processing (e.g. before regression, after regression etc.).

The vector representing DVARS is output in a text file to the working directory with the suffix _dvars appended to the input file name. This program will also produce a plot of DVARS against frame number, if the plot flag -p is specified.

Algorithms: If you have not used speedypp.py, and wish to replicated the Power *et al.*, 2012 method of computing DVARS, then please ensure you have mode/median 1000 normalised during pre-processing. If your pre-processing does not contain this step, please specify the "alternate algorithm" flag -a. This will compute DVARS using a different algorithm, but will produce very similar results to the default method that relies on mode/median normalisation. The alternate algorithm relies on voxel time series still containing the means. The alternate algorithm will either crash or produce non-sensical results if the time series have been de-meaned (either manually or indirectly via a band-pass).

7.2 Example Usage

The following options are available:

OPTIONS:

-h Show help message (Flag)
-i NIfTI file of pre-processed functional MRI data [REQUIRED]
-a Alternate Algorithm. Please use this flag if you HAVE NOT mode/median normalised your data during pre-processing. Note, if you used speedypp.py, then you do not need to use this flag.
-v Verbose (Flag)

Extra options:

-p Plot data (Flag)
-c RBG colour of output figure [DEFAULT=black]
Colour options. Please choose from one of the following:
b = blue, g = green, r = red, c = cyan, m = magenta, y = yellow, k = black.

E.g. bash ~/fmri_spt/dvars.sh -i rest_pp.nii.gz -p -c r

will output a file called "rest_pp_dvars.txt" to the working directory and produce the a plot on screen (see Figure 4).

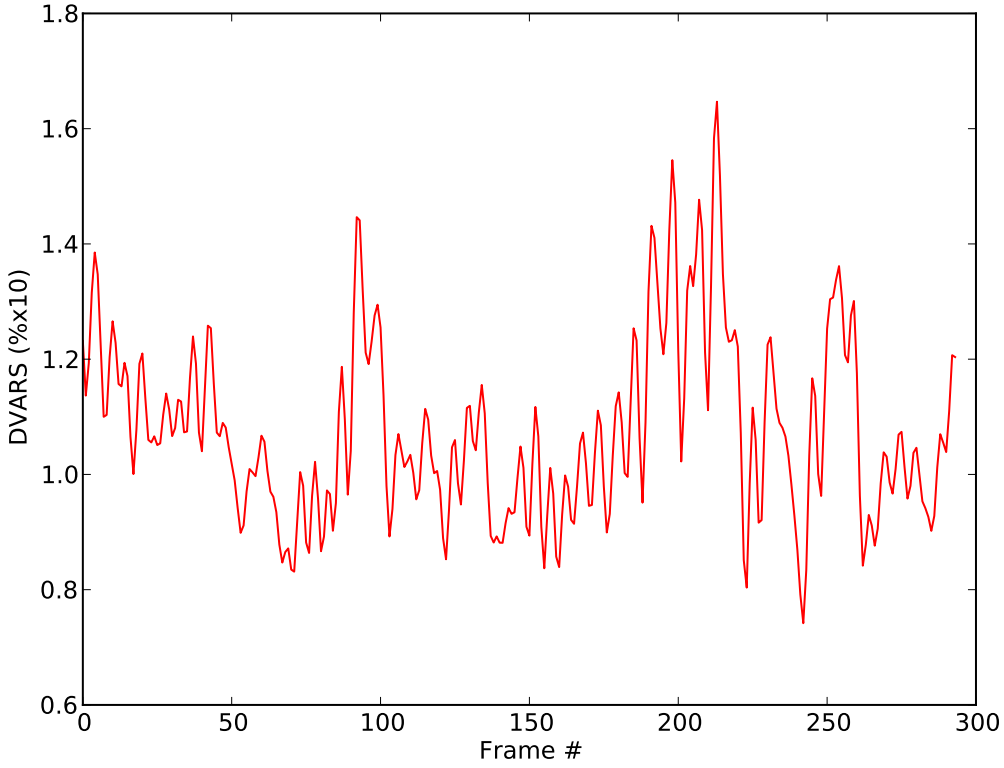

Figure 4: Example output of dvars.sh using the -p flag with -c r

8 Frame censoring (scrub.sh)

Runtime <5 seconds

8.1 Program Description

This program identifies suspect frames of data, based on the parameters of DVARS and FD (as described by Power *et al.*, 2012), and "scrubs" these frames from the regional time series.

Thresholds: The Power method excludes all frames of data where the FD is >0.5 mm OR where the DVARS is >0.5 % signal change (or 5 %x10 as quoted in the paper). Given that their threhold of excluding frames based on DVARS >5 %x10 relies heavily on their method of band-pass followed by regression, it is impossible to use a fixed threshold of 5 %x10 to be applied to all datasets, given that different pre-processing methods may have been used. Instead, the threshold has been defined relative to the baseline of the DVARS trace. On average, the Power *et al.* datasets have a DVARS baseline of $\sim$ 1 %x10. Thus, the threshold in this program has been defined as 4 %x10 above baseline. Given that DVARS computed before and after regression may have consideraly different baselines, this method of varying the threshold according to the baseline ensures that the same frames would be removed regardless of where in the pre-processing pipeline DVARS is computed. This would not be the case if a fixed threshold of 5 %x10 were used.

Dataset Exclusion: The percentage of suspect frames in the dataset, as identified by this method, is a good way of identifying datasets likely to have a high degree of head movement affecting signal intensity. For this reason, in addition to truncating time series, this program will also compute the number of suspect frames of data, and output onto the screen whether the dataset should be excluded or not. The default exclusion criterion is $\geq$ 50% of frames are suspect (expressed as a threshold of 0.5). However, there is an option of changing this to a threshold of your choice (see Example Usage below).

File formats: If parcellate.sh was not used to produce the regional time series, then please ensure the time series matrix has time series represented in columns. reformat.sh can be used to automatically re-format your text files to a compatible format. This program will output a single text file to the working directory with the suffix _scrub appended to the input file name.

8.2 Example Usage

The following options are available:

OPTIONS:

-h Show help message (Flag)
-i Matrix of regional time series (reformatted using reformat.sh if parcellate.sh was not used to process your data) [REQUIRED]
-f Framewise displacement (as output by fd.sh). Note, if you did not use fd.sh, this should be a column vector [REQUIRED]
-d DVARS (as output by dvars.sh). Note, if you did not use dvars.sh, this should be a column vector [REQUIRED]
-r Rejection fraction - please enter a number between 0 and 1. This is the upper bound of suspect frames allowed, to keep dataset [DEFAULT=0.5]
-v Verbose (Flag)

E.g. bash ~/fmri_spt/scrub.sh -i ppp_ts.txt -f rest_motion_fd.txt -d rest_pp_dvars.txt -r 0.4

will output the file ppp_ts_scrub.txt to the working directory and output on the screen whether the dataset should be discarded or not, based on the criteria input.

9 Random Frame Censoring (randscrub.sh)

Runtime <5 seconds

9.1 Program Description

This program randomly removes a specified number of frames from the input time series (as implemented by Power *et al.*, 2012). This program is auxillary to the main functions of this toolbox, and was designed to enable comparison of "scrubbed" time series (as output by scrub.sh) to time series with the same number of frames randomly removed (by this program).

Please note that this program (like scrub.sh) requires regional time series in a matrix where time series are represented in columns. reformat.sh can be used to automatically reformat your text file to a compatible format.

This program will append the suffix _randscrub to the input file name.

9.2 Example Usage

The following options are available:

OPTIONS:

-i Matrix of regional time series (reformatted using reformat.sh if parcellate.sh was not used to process your data) [REQUIRED]
-n Number of frames of data to remove [REQUIRED]

E.g.bash ∼/fmri_spt/randscrub.sh -i ppp_ts.txt -n 50

will output the file ppp_ts_randscrub.txt to the working directory.

10 Time series correlation (correlate.sh)

Runtime <5 seconds

10.1 Program Description

This is a very simple program that takes a matrix of time series, computes the Pearson correlation coeffients between all time series and outputs a symmetric correlation matrix. The matrix is saved to the working directory in a text file with the suffix _corr appended to the input file name.

This step was originally part of cloud.sh (next section), but was excised to allow for alternatives to Pearson correlation to be tested and input into cloud.sh.

If parcellate.sh was not used to produce regional time series, please ensure that the input matrix of time series has time series represented in columns. Alternatively you can use reformat.sh.

10.2 Example Usage

The following options are available:

OPTIONS:

- -h Show help message (Flag)
- -i Matrix of regional time series (reformatted using reformat.sh if parcellate.sh was not used to process your data) [REQUIRED]
- -v Verbose (Flag)

E.g. bash ~/fmri_spt/correlate.sh -i ppp_ts.txt

will output the file ppp_ts_corr.txt to the working directory.

11 The Cloud (cloud.sh)

Runtime <5 seconds

11.1 Program Description

This program creates a scatter plot of ΔR against Euclidean distance (the "cloud plot"), as described in Figure 5 of the Power *et al.*, 2012 paper.

This program requires both "scrubbed" and "unscrubbed" correlation matrices. The scrubbed time series output by scrub.sh and the original time series output by parcellate.sh both need to be passed through correlate.sh (or a alternative algorithm).

The Euclidean distance is calculated from the centroid coordinates of each region or parcel. If you have not used parcellate.sh to define regional time series, please note that regional x, y and z coordinates MUST be represented in rows of the input text file (i.e. row 1 = x, row 2 = y, row 3 = z). reformat.sh can be used to format your text file of coordinates for use with this program.

There are two file outputs from this program:

1. A text file of Euclidean distance and ΔR values. The former is contained in column 1, and the latter in column 2. ΔR values are calculated as described in Power *et al.*, 2012, from the "scrubbed" correlation matrix minus the "unscrubbed" correlation matrix.
2. A text file containing values for the fit through the scatter points. The fit is created by sorting the full array of Euclidean distance vs. ΔR values, according to Euclidean distance, binning at every 100 points and averaging within the bins. The suffix _fit is appended to the output prefix specified.

In addition, there is an option to plot the cloud plot and corresponding fit by specifying the -p flag.

11.2 Example Usage

The following options are available:

OPTIONS:

-h Show help message (Flag)
-i Unscrubbed correlation matrix (e.g. as output by correlate.sh) [REQUIRED]
-s Scrubbed correlation matrix (e.g. as output by correlate.sh) [REQUIRED]
-c Matrix of coordinates (e.g. as output by parcellate.sh). If you have not used parcellate.sh, please ensure that the the text file contains three rows (row 1 = x, row 2 = y and row 3 = z) and n columns, where n is the number of parcels [REQUIRED]
-o Output file name prefix [DEFAULT='cloud']
-v Verbose (Flag)

Extra options:

-p Plot data (Flag)
-d RGB colour of the scatter points/dots (please see below for colour options) [DEFAULT=blue]
-l RGB colour of best fit line (please see below for colour options) [DEFAULT=red]
Colour options. Please choose from one of the following:
b = blue, g = green, r = red, c = cyan, m = magenta, y = yellow, k = black.

E.g. bash ~/fmri_spt/cloud.sh -i ppp_ts_corr.txt -s ppp_ts_scrub_corr.txt -c ppp_xyz.txt -o 001_cloud -p

will output files called "001_cloud.txt", "001_cloud_fit.txt" to the working directory and produce a plot on screen (see Figure 5).

This program produces "cloud plots" for individual datasets. To produce a group level cloud plot (as in Power *et al.*), simply average the ΔR values for all subjects/datasets you wish to include and plot the group ΔR against the Euclidean distance. If the same parcellation template has been used for all subjects, the Euclidean distance vector (output for subjects individually by cloud.sh) will be identical.

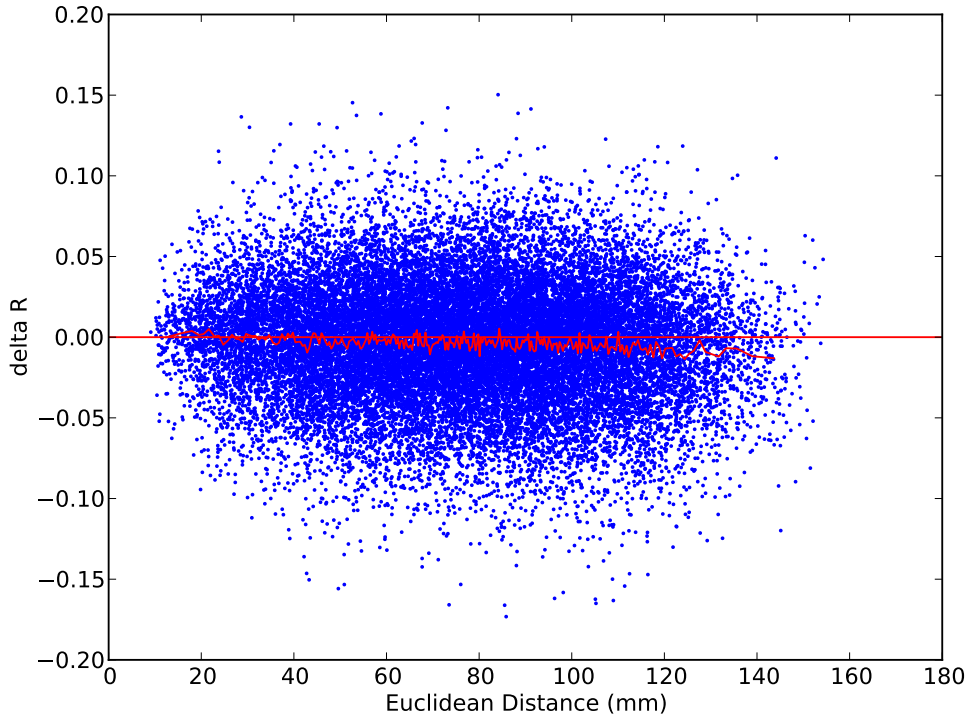

Figure 5: Example output from cloud.sh using the -p flag

12 Δ % BOLD (dbold.sh)

Runtime <5 seconds

12.1 Program Description

dbold.sh computes the derivative of percent signal change on time series that have been parcellated. The output of this program is designed to be used with the program described in the next section (loess.sh).

The only input for this program is a text file containing a matrix of regional time series values (e.g. as output by parcellate.sh). Please note, if you have not used parcellate.sh to generate these time series, please ensure that the time series are represented in columns. reformat.sh can be used to automatically format yout text file to a format that is compatible with this program.

This program will output a text file containing a matrix of Δ%BOLD for each time series, where columns represent regions/parcels, as in the input file.

Algorithms: If you have not used speedypp.py, and wish to replicated the Power *et al.*, 2012 method of computing Δ%BOLD, then please ensure you have mode/median 1000 normalised during pre-processing. If your pre-processing does not contain this step, please specify the "alternate algorithm" flag -a. This will compute Δ%BOLD on the regional time series using a different algorithm, but will produce very similar results to the default method that relies on mode/median normalisation. The alternate algorithm relies on regional time series still containing the means. The alternate algorithm will either crash or produce non-sensical results if the time series have been de-meaned (either manually or indirectly via a band-pass).

12.2 Example Usage

The following options are available:

OPTIONS:

- -h Show help message (Flag)
- -i Matrix of regional time series (reformatted using reformat.sh if parcellate.sh was not used to process your data) [REQUIRED]
- -a Alternate Algorithm. Please use this flag if you HAVE NOT mode/median normalised your data during pre-processing. Note, if you used speedypp.py, then you do not need to use this flag.
- -v Verbose (Flag)

E.g. bash ~/fmri_spt/dbold.sh -i ppp_ts.txt

will output the file ppp_ts_dbold.txt to the working directory.

13 Loess Curve (loess.sh)

Runtime <5 seconds

13.1 Program Description

This program computes a locally-weighted estimate (loess) curve of framewise displacement (x-axis) versus regional Δ%BOLD, as described in Power *et al.*, 2012. Δ%BOLD is the derivative of percent signal change for each time series. As in the Power *et al.* paper, the units of Δ%BOLD are expressed as Δ%BOLDx10.

The loess curve allows you to identify whether head movement has an impact on percent signal change in your data.

This program requires two file inputs. First, is a text file containing values for the framewise displacement. If fd.sh was not used to compute the framewise displacement, please ensure the input file is a column vector. Alternatively, reformat.sh will reformat your file automatically. The second input is Δ%BOLDx10 computed at regional (parcellated) time series (e.g. as output by dbold.sh). If dbold.sh was not used, then please ensure that Δ%BOLDx10 time series for each parcel are represented in columns. reformat.sh can be used to automatically format files to a format that is compatible with this program.

This program will also produce a scatter plot of framewise displacement (x-axis) plotted against Δ%BOLDx10 (y-axis) with the loess curve superimposed, if the plot flag -p is specified. The loess curve values will be output to the working directory (column 1 containing framewise displacement values, column 2 containing loess fit values) with the output prefix specified.

13.2 Example Usage

The following options are available:

OPTIONS:

- -h Show help message (Flag)
- -f Framewise displacement (as output by fd.sh). Note, if you did not use fd.sh, this should be a column vector [REQUIRED]
- -d Δ%BOLDx10 (e.g. as output by dbold.sh). Note, if you did not use dbold.sh, Δ%BOLDx10 for each parcel should be represented in columns [REQUIRED]
- -o Loess curve output prefix [DEFAULT='loess']
- -v Verbose (Flag)

Extra options:

- -p Plot data (Flag)
- -s RGB colour of the scatter points (please see below for colour options) [DEFAULT=blue]
- -l RGB colour of loess curve (please see below for colour options) [DEFAULT=red]
  Colour options. Please choose from one of the following:
  b = blue, g = green, r = red, c = cyan, m = magenta, y = yellow, k = black.

E.g. bash ~/fmri_spt/loess.sh -f rest_motion_fd.txt -d ppp_ts_dbold.txt -p

will output the file "loess.txt" to the working directory and output a scatter plot of framewise displacement against Δ%BOLDx10, with the loess curve overlayed (see Figure 6).

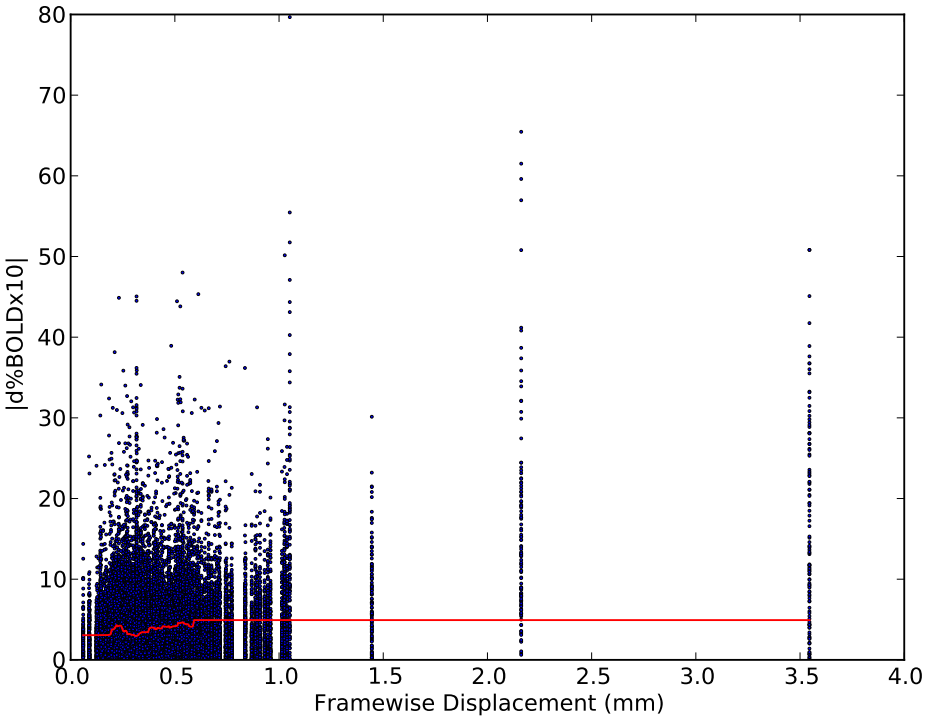

Figure 6: Example output from loess.sh using the -p flag

14 File reformatting (reformat.sh)

Runtime <5 seconds

14.1 Program Description

This program allows you to reformat your text files to a format that is compatible with the rest of this toolbox. This only applies if you are using the modules in isolation and are not running speedypp.py or parcellate.sh. The usage of python in this toolbox will not be able to deal with csv files.

You will need to ensure that the following files are in the correct format if you have not used the relevant functions in this toolbox to produce them:

1. Matrix of parcellated regional time series
2. Matrix of centroid parcel coordinates
3. Vectors containing DVARS and FD values

reformat.sh will take all of these files and transform the vectors/matrices if necessary, and then output the new text files with tab separation, and with the file extension .txt.

14.2 Example Usage

The following options are available:

OPTIONS:

-h Show help message (Flag)
-i Text file to be reformatted [REQUIRED]
-n Number of frames of data in your dataset [REQUIRED]
-v Verbose (Flag)

E.g. bash /fmri_spt/reformat.sh -i time_series.1D -n 265

will output a text file to the working directory called "time_series.txt" for use with the rest of this toolbox.