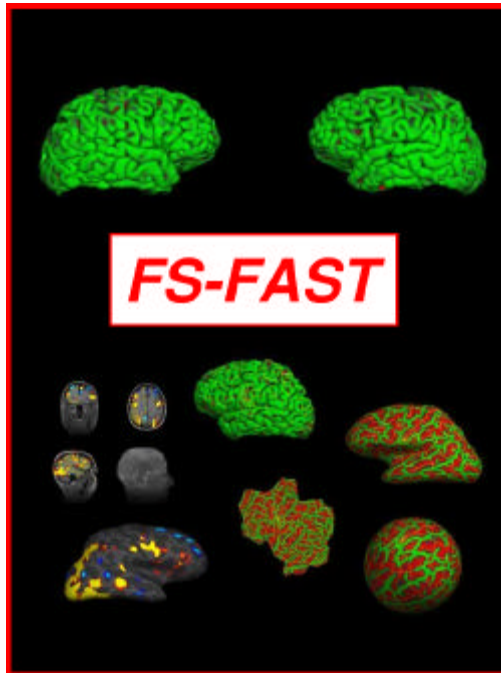


# FS-FAST Tutorial

This is a tutorial for using **FS-FAST**: the **FreeSurfer Functional Analysis Stream**.



FS-FAST helps you analyze functional MR data by creating statistical maps that can be overlaid directly onto the structural MR data you generate using FreeSurfer. This tutorial will help you process the distributed functional MR data set of a single subject we call Bert, using the program's most basic steps. For advanced steps and options for multi-subject analysis, consult the FS-FAST Guide. All FS-FAST and FreeSurfer software and documents, along with the sample data sets to use with their tutorials, are available for download at:

<http://surfer.nmr.mgh.harvard.edu/download.html>  
<http://www.cortechs.net>

The interface between FS-FAST and FreeSurfer supports a wide range of applications, including automatic Talairach transforming, morphometry, spherical morphing and inter-subject averaging, as well as cortical inflation and flattening based on Dale, et al, 1999, and Fischl, et al, 1999. As a result, you can view the statistical maps on same-slice anatomicals, high-resolution anatomicals, Talairach anatomicals, and on the cortical surface (folded, inflated, or flattened).

**FS-FAST requires a Linux computer with MATLAB® installed on it in order to work, with approximately 300-400 MB disk space per subject, depending on the number of runs.**

**Author of FS-FAST:** Doug Greve, Ph.D.

This manuscript prepared by Evelina Busa.

Bug Reports, Questions, Comments:

Limited support is offered for this software. Any bug reports should include complete information so that the problem can be recreated. As a minimum, give the following:

1. Specify which version date of the Tutorial you are using.
2. Specify which machine/operating system you were on when the problem occurred.
3. Specify which program (along with command-line options) you were running when the problem occurred. Note: "program" means a program from the FS-FAST package, not one that you have created that calls FS-FAST programs.
4. Specify the directory you were in when you ran the program.
5. Include any text that the program printed out to the screen.
6. Include the log file (if it exists).

Send your bug-reports, questions, and comments to:

[analysis-bugs@nmr.mgh.harvard.edu](mailto:analysis-bugs@nmr.mgh.harvard.edu)

## TABLE OF CONTENTS

<b>FS-FAST TUTORIAL AT A GLANCE</b> .....	<b>4</b>
<b>USING FS-FAST: A STEP BY STEP GUIDE</b> .....	<b>5</b>
1. Download the sample subject's data .....	5
2. Get to know the Sessions Format .....	5
3. Make a directory for your Study.....	7
4. Make paradigm files for your experiment.....	8
5. Motion correct the data using "mc-sess".....	9
6. Normalize signal intensity using "inorm-sess".....	12
7. Set up session-level analysis using "mkanalysis-sess.new" .....	13
8. Average session-level data by condition using "selxavg-sess" .....	14
9. Define an omnibus contrast using "mkcontrast-sess" .....	15
10. Compute statistical maps of the omnibus contrast using "stxgrinder-sess" .....	16
11. Run functional/structural registration using "autoreg-sess" .....	18
INTERFACING WITH FREESURFER.....	18
Functional/Anatomical Registration .....	19
12. Visualization.....	23
Visualization: slice-based using "sliceview-sess".....	23
Viewing Bert's hemodynamic responses .....	26
Visualization: surface-based using "paint-sess" and "surf-sess" .....	31

## **FS-FAST TUTORIAL AT A GLANCE**

**DOWNLOAD SAMPLE DATA SETS FOR SUBJECT “BERT”**  
(Linux, SGI, Sun)

**GET TO KNOW THE SESSIONS FORMAT**  
(Linux, SGI, Sun)

**MAKE APPROPRIATE DIRECTORIES FOR THE STUDY**  
(Linux, SGI, Sun)

**MAKE PARADIGM FILES**  
(Linux, SGI, Sun)

**MOTION CORRECT**  
“mc-sess”  
(Linux, SGI, Sun)

**NORMALIZE SIGNAL INTENSITY**  
“inorm-sess”  
(Linux, SGI, Sun: matlab)

**SET UP THE ANALYSIS**  
“mkanalysis-sess.new”

**AVERAGE WITHIN-SESSION**  
“selxavg-sess”  
(Linux, SGI, Sun: matlab)

**DEFINE OMNIBUS CONTRAST**  
“mkcontrast-sess”  
(Linux, SGI, Sun: matlab)

**COMPUTE STATISTICAL MAP**  
“stxgrinder-sess”  
(Linux, SGI, Sun: matlab)

**RUN FUNCTIONAL/STRUCTURAL REGISTRATION**  
“autoreg-sess”  
(Linux, SGI, SUN: matlab)

**VISUALIZE**  
slice based: “sliceview-sess”  
surface based: “paint-sess” / “surf-sess”  
(Linux, SGI)

## USING FS-FAST: A STEP BY STEP GUIDE

### 1. DOWNLOAD THE SAMPLE SUBJECT'S DATA

Functional and anatomical data sets for the subject “Bert” are available for download at:

<http://surfer.nmr.mgh.harvard.edu/download.html>  
<http://www.cortechs.net>

#### a) FUNCTIONAL DATA:

Select the functional data set named **bert-functional**.

These data will be in a compressed file called: bert.func.tar.gz. To unpack these data, go to the directory where you will want the functional data to be stored. This directory will be called the Session Parent. Once, there, run the following:

```
tar xvfz <fsdist>/bert.func.tar.gz
```

This will create a directory called bert-functional, which will have four subdirectories.

#### b) ANATOMICAL DATA:

Select the anatomical data set named **bert-anatomical (reconstructed)**. These data will also be in a compressed file, called: bert.recon.tar.gz, and must be similarly unpacked, but into a directory other than your Session Parent. This compressed file will unpack into several subdirectories. The “Interfacing With FreeSurfer” section provides further details.

You will notice that Bert’s anatomical data are available in two formats: raw and reconstructed. Since you need to overlay the functional data on your subject’s reconstructed surfaces in subsequent FS-FAST steps, you should download the reconstructed data set, which already has all the volumes and surfaces you will need. The raw anatomical data are intended for use with the FreeSurfer Tutorial, a guide to cortical reconstruction of raw MR data.

**IMPORTANT:** Since cortical reconstruction does require several hours, always arrange to have your subjects’ anatomical data sets reconstructed in advance of FS-FAST processing so that they will be ready for functional overlay.

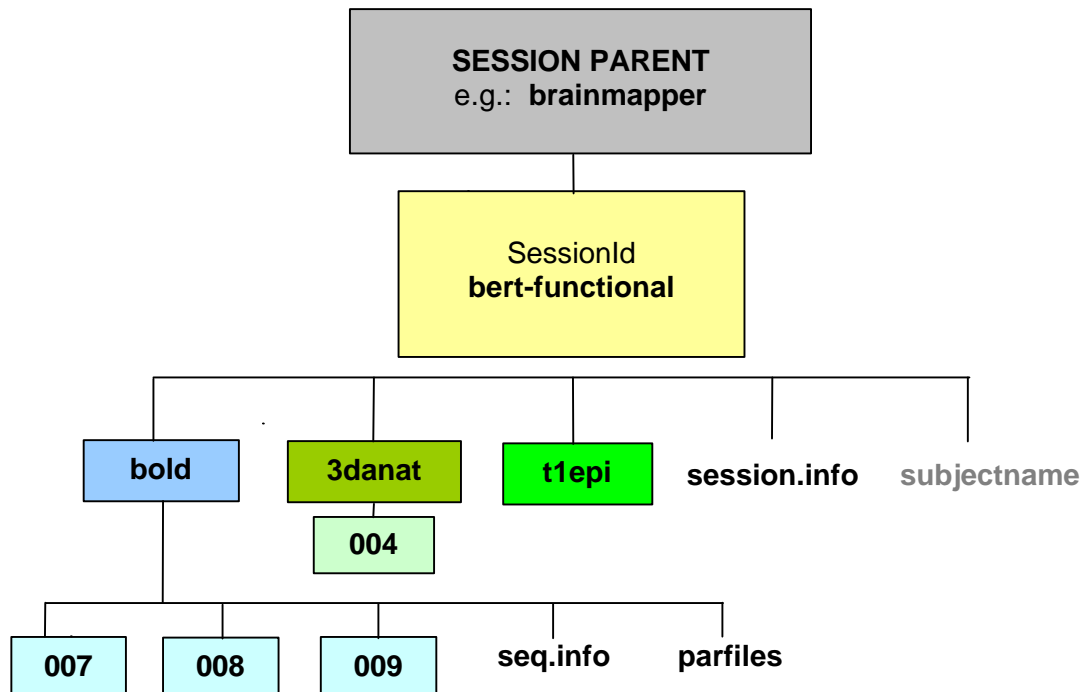
### 2. GET TO KNOW THE SESSIONS FORMAT

You will notice several subdirectories and a text file in Bert’s directory, and that they are already in a structure called the Sessions Format. If you unpack subject data directly from your facility’s scanner, the automatically created subdirectories and files might not conform to the format required by FS-FAST, i.e. the Sessions Format. You may need to manually arrange and/or rename them, to ensure that they conform to the Sessions Format. Further details are provided in the next step. For now, it is a good idea to familiarize yourself with the FS-FAST’s Sessions Format.

## THE SESSIONS FORMAT

The Sessions Format is a structure that demands a certain file and directory naming convention for the scanning acquisitions (sessions). There is some flexibility in how you name the individual files and directories in this Format, but the naming of the sessions must conform exactly to the convention outlined here in order for FS-FAST to work. New files and directories will be automatically added to this structure during each stage of FS-FAST processing.

Here is a look at Bert's Sessions Format:



**SESSION PARENT:** This is the directory to where you unpack subjects' individual session directories, e.g. "bert-functional". You can call it whatever you like, e.g. "brainmapper".

**SessionId:** This directory name identifies the session for a particular subject. It contains the MRI or fMRI scan, or series of scans, carried out on that particular subject on a particular occasion. The SessionID is also known simply as the subject's "session directory". For this tutorial you will have only one SessionID (Bert's) in your Session Parent.

**bold:** This is the *functional subdirectory*. It contains the individual run(s) of functional MRI data; each of which must be named by three-digit, zero-padded numbers (e.g. 007, 008, 009) in order for FS-FAST to recognize them as functional data directories. (Although not applicable in this case, if this directory were named something other than "bold", FS-FAST would require modifications to further steps in the processing stream. These variations are described in the FS-FAST Guide.)

**007/008/009:** Under the bold directory, these are the subdirectories where each functional volume (or "run") is stored, in bfile format (i.e. bshorts) with an "f" stem.

**3danat:** This directory contains any raw structural MR data collected during the session, and stores it in COR format. Each volume/structural run should be in an individual subdirectory.

**004:** Under 3danat, this subdirectory contains only anatomical data from the fast structural scan (i.e. 004), acquired on the same date as the functional session. The anatomical data from this fast scan will later be used for registration of the functional data with Bert's reconstructed anatomical data. Bert's anatomical data (acquired with a different scan protocol) that were used for the anatomical reconstruction would initially appear here also, but now have a distinctive directory structure of their own, one that is automatically created in the FreeSurfer reconstruction.

**t1epi:** This directory contains the t1-weighted EPI image data collected during the session, whose slice prescription must be identical to that of the functional data. Bert's actual t1epi volume is stored in a subdirectory called 004 in bshort format, with a stem called "f".

**session.info:** This text file contains general information about the subject (e.g. DOB, gender, console name, population, study date, etc.).

**subjectname:** This text file contains the subject identifier string (i.e. the name of the directory containing the subject's anatomical directory—e.g. "bert-anatomical"), and nothing else. *You will create this text file in a subsequent step.*

**seq.info:** This text file contains details such as the scanning protocol used during the scan, date of scan, etc.

**parfiles:** These paradigm files give information about the stimulus schedule for each run, that is, details about which stimulus was presented at which time during the functional run. You should have one paradigm file for each functional run. The parfiles are created by the FS-FAST program **optseq**.

### **3. MAKE A DIRECTORY FOR YOUR STUDY**

Before proceeding to analyze the subject data for any particular study, you must create a working directory for the study analyses. This is called the Study Directory and can be thought of as a scratch space where various files are written when the processing commands are run. Your Study Directory can contain any number of the Session Directories, and all the programs are run from here, regardless of which Session is being processed. For this reason, and to avoid confusion, the Study Directory *should be separate from the SESSION PARENT directory and from the Anatomical Database where you unpacked the reconstructed data set.* To make a Study Directory, cd to another location—an appropriate directory that is separate from the SESSION PARENT directory—and type:

```
mkdir MYSTUDY (this will be referred to as your "Study Directory")  
cd MYSTUDY/
```

Next, you would specify which subjects you want to process in your study. In this tutorial, you will only process Bert, but the standard stream processes all the subjects in your study. You have to specify the list of each of your study subjects' Session Directories in order for them to be processed.

Each subject's Session directory should be thought of as consisting of two parts:

- a) the **session parent** and
- b) the **session identifier**.

The **session parent** is the path to the subjects' Session directories (which should end with the name of your SESSION PARENT), and the **session identifier** corresponds to the subject name that you used to name a particular subject's Session, e.g.:

In Bert's Session Directory: **/space/someday/users/brainmapper/bert**

the **session parent** is: **/space/someday/users/brainmapper/**  
and the **session identifier** is: **bert**

Bert's session parent must be entered into a textfile (**sesspar**). Bert's session identifier must be entered into another text file (**sessid**).

Doing an "ls" to see what's in your Study Directory, then, should show this:

```
sessid  sesspar
```

Your **sesspar** text file would look like this: **/space/someday/users/brainmapper**

Your **sessid** text file would look like this: **bert**

#### **4. MAKE PARADIGM FILES FOR YOUR EXPERIMENT**

Paradigm files are already distributed with Bert's data, but you normally would have to create your own, using any text editor. The paradigm files simply describe what was going on during your study experiment with respect to time of presentation and stimulus type. The following will explain how Bert's paradigm files were created. The methodology of the experiment is described in detail in Wagner AD, Pare-Blagoev J, Clark J, and Poldrack RA. **Recovering Meaning: Left Prefrontal Cortex Guides Controlled Semantic Retrieval**. Neuron 31: 329-338. August 2, 2001.

(Note that Bert's data were not part of Wagner et al's study, neither was that study processed with FS-FAST. Russ Poldrack of the MGH-NMR Center generously reran the experiment using a subject who agreed to provide data, specifically for this tutorial.)

The experiment involves an associative memory task. The subject is shown one of four stimulus types. Each stimulus has five words in the following pattern:

```
targetword
testword1 testword3
testword2 testword4
```

The subject is asked to choose which test word is most related to the target word.

By design, the target word is either highly related or loosely related to one of the test words.



Sometimes there are four test words presented, and sometimes there are two test words presented along with two X's.

This makes four conditions:

1. H2 - highly related, two words/two X's (used in run 006)
2. H4 - highly related, four words (used in run 007)
3. L2 - loosely related, two words/two X's (used in run 008)
4. L4 - loosely related, four words (used in run 009)

This number coding is used in the paradigm files to identify which stimulus was presented when. The contrast (L2+L4) - (H2-H4) should show activity in the left frontal region.

Based on your study description, you would create a paradigm file for each functional run, using any text editor. For this tutorial, the paradigm files are already provided in the subdirectory called parfiles:

- sem\_assoc-s001-r001.dat (for functional run 007)
- sem\_assoc-s001-r002.dat (for functional run 008)
- sem\_assoc-s001-r003.dat (for functional run 009)

**IMPORTANT:** For any subject whose functional data you process, a copy of each run's paradigm file should be placed into the directory containing that functional run's bshort files. All paradigm files for all runs averaged together should have the same name, even though their contents may differ. In Bert's case, we will call the copied paradigm files "sem\_assoc.par".

To copy Bert's paradigm files into the three functional run directories, cd to bold/ and type:

```
cp parfiles/sem_assoc-s001-r001.dat 007/sem_assoc.par
cp parfiles/sem_assoc-s001-r002.dat 008/sem_assoc.par
cp parfiles/sem_assoc-s001-r003.dat 009/sem_assoc.par
```

## **5. MOTION CORRECT THE DATA USING "MC-SESS"**

You will next motion correct Bert's functional data. Motion correction is optional, however if you *do* motion correct the data, you will still be able to choose whether to use the raw or motion corrected data later in the analysis stream. Motion correction is recommended since it allows you to determine how much motion there was during the scan.

The "mc-sess" script uses the AFNI motion correction algorithm to align all the images in a session to the first time image of the first run. This command will motion correct any sessions specified in the sessid and sesspar files of your Study Directory, so in this case, only Bert will be motion corrected.

To motion correct, you would go to your Study Directory and type:

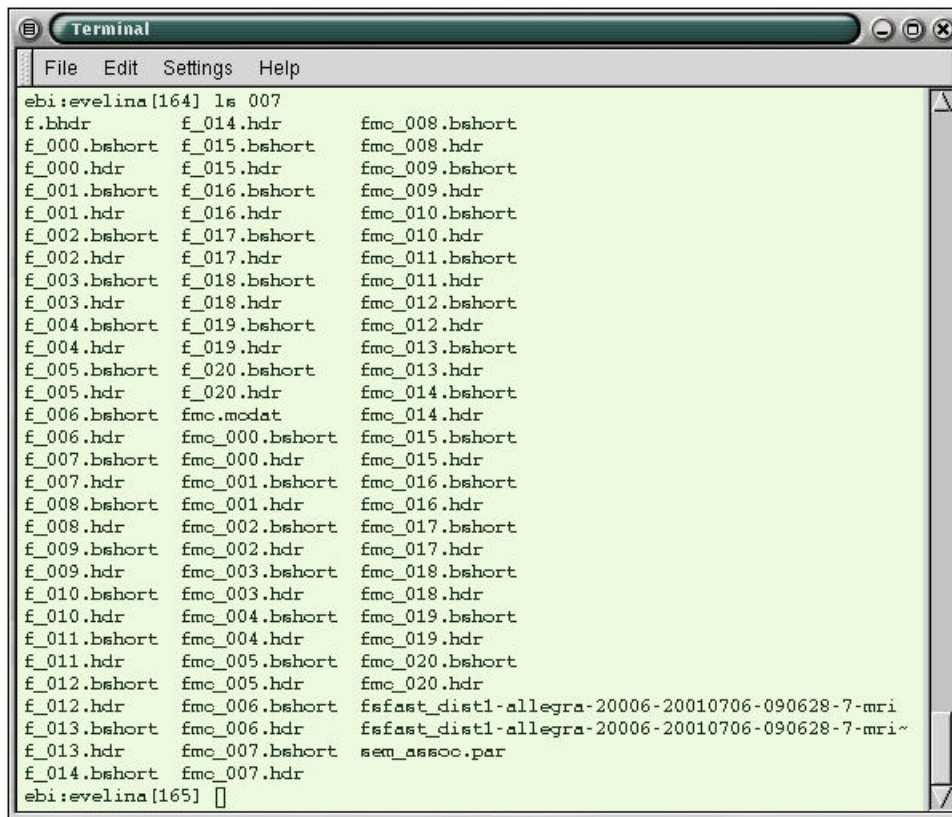
```
mc-sess -sf sessid -df sesspar
```

The motion correction takes approximately 30 minutes.

**TIP:**  
**All the options for any FS-FAST command can be displayed by typing the command by itself (e.g. “mc-sess”) at the prompt.**

**OUTPUT:** The mc-sess command creates a new functional volume in each of Bert’s functional runs. This new volume will have the same size as the raw volume, but will have stem “fmc” instead of “f”.

Here is what you should now see if you do an “ls” for Bert’s functional run 007:



```
Terminal
File Edit Settings Help
ebi:evelina[164] ls 007
f.bhdr      f_014.hdr      fmc_008.bshort
f_000.bshort f_015.bshort   fmc_008.hdr
f_000.hdr   f_015.hdr     fmc_009.bshort
f_001.bshort f_016.bshort   fmc_009.hdr
f_001.hdr   f_016.hdr     fmc_010.bshort
f_002.bshort f_017.bshort   fmc_010.hdr
f_002.hdr   f_017.hdr     fmc_011.bshort
f_003.bshort f_018.bshort   fmc_011.hdr
f_003.hdr   f_018.hdr     fmc_012.bshort
f_004.bshort f_019.bshort   fmc_012.hdr
f_004.hdr   f_019.hdr     fmc_013.bshort
f_005.bshort f_020.bshort   fmc_013.hdr
f_005.hdr   f_020.hdr     fmc_014.bshort
f_006.bshort fmc.mcdat      fmc_014.hdr
f_006.hdr   fmc_000.bshort fmc_015.bshort
f_007.bshort fmc_000.hdr    fmc_015.hdr
f_007.hdr   fmc_001.bshort fmc_016.bshort
f_008.bshort fmc_001.hdr    fmc_016.hdr
f_008.hdr   fmc_002.bshort fmc_017.bshort
f_009.bshort fmc_002.hdr    fmc_017.hdr
f_009.hdr   fmc_003.bshort fmc_018.bshort
f_010.bshort fmc_003.hdr    fmc_018.hdr
f_010.hdr   fmc_004.bshort fmc_019.bshort
f_011.bshort fmc_004.hdr    fmc_019.hdr
f_011.hdr   fmc_005.bshort fmc_020.bshort
f_012.bshort fmc_005.hdr    fmc_020.hdr
f_012.hdr   fmc_006.bshort fsfast_dist1-allegra-20006-20010706-090628-7-mri
f_013.bshort fmc_006.hdr    fsfast_dist1-allegra-20006-20010706-090628-7-mri
f_013.hdr   fmc_007.bshort seq_assoc.par
f_014.bshort fmc_007.hdr
ebi:evelina[165] 
```

### The fmc.mcdat text file

A text file called fmc.mcdat will also be showing in each run subdirectory. The motion correction parameters are stored in this file.

For example, on the next page we show what the fmc.mcdat file should look like for Bert’s run 007.

Here is the beginning of Bert's fmc.mcdat file for bold/007:

```

0 0.000 -0.000 0.000 -0.000 0.000 -0.000 0 0 0.000
1 0.018 -0.002 0.012 -0.023 0.012 0.031 9.813 9.792 0.040
2 0.010 0.004 0.004 0.026 0.015 -0.027 9.594 9.559 0.040
3 0.010 0.010 0.010 0.011 0.012 0.004 8.54 8.526 0.017
4 0.032 0.009 0.015 0.005 0.031 -0.012 9.114 9.059 0.034
5 0.031 0.005 0.018 0.030 0.029 0.011 10.18 10.12 0.043
6 0.046 0.001 0.033 0.006 0.045 -0.001 9.09 8.978 0.045
7 0.051 0.017 0.033 0.017 0.043 -0.019 9.852 9.736 0.050
8 0.046 0.008 0.030 0.047 0.052 -0.039 10.29 10.12 0.080
9 0.051 0.010 0.037 0.034 0.054 -0.016 9.104 8.919 0.066
10 0.079 0.014 0.046 0.016 0.058 -0.057 10.42 10.18 0.083
11 0.075 0.016 0.048 0.034 0.070 -0.065 10.42 10.09 0.101
12 0.074 0.010 0.048 0.010 0.065 -0.020 9.809 9.573 0.069
13 0.089 0.017 0.047 0.025 0.066 -0.048 11.07 10.79 0.085
14 0.065 0.001 0.042 0.043 0.057 -0.028 8.836 8.584 0.077
15 0.087 0.008 0.054 0.019 0.067 -0.007 10.69 10.44 0.070
16 0.084 0.004 0.048 0.043 0.066 -0.046 9.76 9.433 0.091
17 0.081 -0.016 0.052 0.046 0.065 -0.002 10.34 10.06 0.080
18 0.091 -0.006 0.054 0.036 0.072 -0.070 10.46 10.07 0.107
19 0.088 -0.015 0.073 0.070 0.083 -0.060 10.44 9.938 0.124
20 0.097 -0.011 0.072 0.054 0.093 -0.024 11.21 10.76 0.110
21 0.089 0.025 0.078 0.064 0.094 -0.090 10.66 10.07 0.145
22 0.134 0.089 0.151 -0.193 0.124 -0.482 18.26 15.59 0.534
23 0.163 0.063 0.107 -0.151 0.101 -0.303 14.33 12.53 0.353
24 0.180 0.057 0.081 -0.092 0.081 -0.311 14.17 12.67 0.334
25 0.180 0.062 0.076 -0.106 0.082 -0.233 13.72 12.46 0.269
26 0.190 0.043 0.065 -0.097 0.069 -0.247 12.76 11.48 0.274
27 0.184 0.018 0.059 -0.051 0.061 -0.184 12 11.11 0.200
28 0.193 0.012 0.063 -0.068 0.074 -0.190 13.19 12.26 0.215
29 0.204 0.017 0.059 -0.042 0.068 -0.207 13.27 12.35 0.222
30 0.210 0.011 0.057 -0.055 0.073 -0.189 12.1 11.04 0.210
31 0.214 -0.001 0.049 -0.031 0.055 -0.225 12.71 11.65 0.234
32 0.195 -0.004 0.043 -0.030 0.044 -0.201 11.8 10.92 0.208

```

Each row of the fmc.mcdat text file corresponds to a different time point. There are ten columns in any fmc.mcdat file--

(1) (2) (3) (4) (5) (6) (7) (8) (9) (10)

--corresponding to various parameters:

- (1) time point number
- (2) roll rotation (in °)
- (3) pitch rotation (in °)
- (4) yaw rotation (in °)
- (5) between-slice motion or translation (in mm)
- (6) within slice up-down motion or translation (in mm)
- (7) within slice left-right motion or translation (in mm)
- (8) RMS error before correction
- (9) RMS error after correction
- (10) total vector motion or translation (in mm)

Those who use this program should cite Cox RW and Jesmanowicz A. **Real-time 3D image registration for functional MRI**. Magnetic Resonance in Medicine, 42: 1014-1018; 1999. This data file can be plotted using gnuplot or another plotter (e.g. by exporting and plotting in Excel).

## 6. NORMALIZE SIGNAL INTENSITY USING “INORM-SESS”

The “inorm-sess” command generates a file containing mean intensity data for the entire functional volume for each run, whether raw or motion corrected. Intensity normalization is a slight misnomer of this process since the global mean of the fMRI signal inside the tissue is calculated by segmentation of tissue from air. The number is later used to rescale the data so that when intersubject averaging is done, all subjects have the same global mean. This step is also optional, but it is recommended if you are averaging data across subjects. Run “inorm-sess” on Bert, even though you won’t be averaging data across subjects.

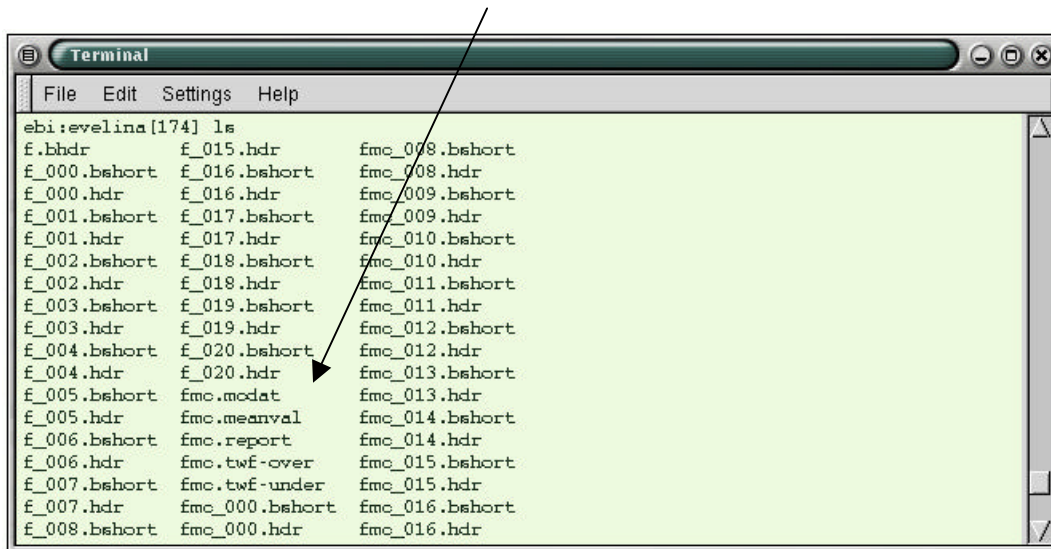
To normalize the intensity of Bert’s motion corrected data, cd to the Study Directory and type:

```
inorm-sess -sf sessid -df sesspar -motioncor
```

Normalization takes about ten minutes to complete.

**OUTPUT:** After running the signal intensity command separately for each functional run subdirectory, you should see four new files in each run subdirectory called:

fmc.meanval; fmc.report; fmc.twf-over; and fmc.twf-under.

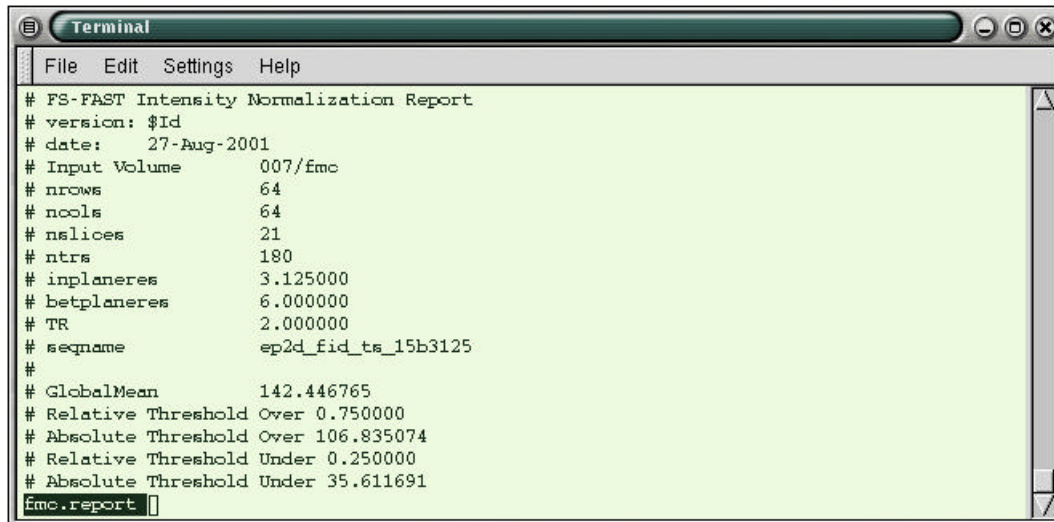


```
Terminal
File Edit Settings Help
ebi:revelina[174] ls
f.bbhdr      f_015.hdr      fmc_008.bshort
f_000.bshort f_016.bshort   fmc_008.hdr
f_000.hdr    f_016.hdr     fmc_009.bshort
f_001.bshort f_017.bshort   fmc_009.hdr
f_001.hdr    f_017.hdr     fmc_010.bshort
f_002.bshort f_018.bshort   fmc_010.hdr
f_002.hdr    f_018.hdr     fmc_011.bshort
f_003.bshort f_019.bshort   fmc_011.hdr
f_003.hdr    f_019.hdr     fmc_012.bshort
f_004.bshort f_020.bshort   fmc_012.hdr
f_004.hdr    f_020.hdr     fmc_013.bshort
f_005.bshort fmc.modat      fmc_013.hdr
f_005.hdr    fmc.meanval   fmc_014.bshort
f_006.bshort fmc.report     fmc_014.hdr
f_006.hdr    fmc.twf-over  fmc_015.bshort
f_007.bshort fmc.twf-under  fmc_015.hdr
f_007.hdr    fmc_000.bshort fmc_016.bshort
f_008.bshort fmc_000.hdr    fmc_016.hdr
```

1) In the text file called “fmc.meanval” in 007 you should see the number: 402.142299, which is the global mean.

2) The fmc.report file contains many statistics relating to the “cleanliness” of your data, the most important of which are: **OV Zmax** and **OU Mean**. The Zmax statistic is an indication of spiking or instability. The OU Mean statistic is a measure of distortion. These are meant to be indicators of a possible problem during scanning. However, it is difficult to assign hard thresholds below which the user doesn’t need to investigate the integrity of the data. The user should examine these statistics for all the sessions in a study in order to determine appropriate levels. As a rule of thumb, a Zmax over 3.5 or an OU Mean under 25 should trigger a closer look at that run. For backwards compatibility, inorm also reports the same statistics produced by an older program called “stackfix”.

The beginning of Bert's `fmc.report` looks like this:



```
Terminal
File Edit Settings Help
# FS-FAST Intensity Normalization Report
# version: $Id
# date: 27-Aug-2001
# Input Volume 007/fmc
# nrows 64
# ncols 64
# nslices 21
# ntrs 180
# inplaneres 3.125000
# betplaneres 6.000000
# TR 2.000000
# seqname ep2d_fid_ts_15b3125
#
# GlobalMean 142.446765
# Relative Threshold Over 0.750000
# Absolute Threshold Over 106.835074
# Relative Threshold Under 0.250000
# Absolute Threshold Under 35.611691
fmc.report
```

## **7. SET UP SESSION-LEVEL ANALYSIS USING “MKANALYSIS-SESS.NEW”**

The “analysis” is the collection of parameters you will use to analyze each session. It includes the signal model, the stimulus schedule, and any preprocessing options such as smoothing and motion correction that you might have run. Data are not actually analyzed at this stage, but with this program you define how you want the raw or pre-processed data for each session to be averaged together later.

You may want to analyze your study in more than one way, e.g. with and without motion correction, or you may wish to define your events based on subject response in one analysis, and on stimulus type in another. The key here is to realize that you have to give a name to all these parameters, and then refer to that name in later processing stages. Each analysis you set up defines a specific set of conditions and the same paradigm file name.

**IMPORTANT:** Setting up your analysis is only done ONCE, and then it is applied to any and all sessions. If you decide you want to change your analysis parameters, you should either create a new analysis (with a new “analysisname”), or create a new analysis from scratch after deleting the original, which can then be renamed with the original name.

Each analysis name will be used in subsequent processing steps, and for multiple analyses, you have to run `mkanalysis-sess` separately, once for each analysis, giving each a different analysis name.

For this tutorial, you will create an analysis named “`sem_assoc`” for Bert, including all the following parameters:

- |                               |   |
|-------------------------------|---|
| <b>-analysis analysisname</b> | name of analysis                            |
| <b>-TR #</b>                  | TR (in seconds, e.g. 2)                     |
| <b>-paradigm parname</b>      | name of paradigm file                       |
| <b>-designtype design</b>     | design type (e.g. event-related)            |
| <b>-funcstem fmc</b>          | use motion corrected (rather than raw) data |



**-inorm** use intensity normalized data  
**-fwhm #** use in-plane spatial smoothing fwhm=full-width/half-max (in mm)  
**-nconditions #** the number of conditions in the paradigm (exclude null/fixation)  
**-timewindow #** the time range in an FIR signal model (in seconds)  
**-tprestim #** begin the FIR time window (in seconds) before stimulus onset

To make this analysis, cd to your Study Directory and type:

**mkanalysis-sess.new -analysis sem\_assoc -TR 2 -paradigm sem\_assoc.par -designtype event-related -funcstem fmc -inorm -fwhm 4 -nconditions 4 -timewindow 26 -tprestim 4**

**OUTPUT:** This quick step creates a subdirectory called “analysisname” in your Study Directory and writes several files there as well, including “analysis.cfg” and “analysis.info”. The program also checks that all subject data are accessible and creates default scripts for the analysis. This is what the contents of your Study Directory should now look like:

```

Terminal
File Edit Settings Help
ebi [MYSTUDY] ls
007/ log/ sem_assoc/ sessid sesspar
ebi [MYSTUDY]
  
```

and here are the contents of the sem\_assoc (which is your analysisname) subdirectory:

```

Terminal
File Edit Settings Help
ebi [MYSTUDY] ls sem_assoc
analysis.cfg analysis.info
ebi [MYSTUDY]
  
```

## **8. AVERAGE SESSION-LEVEL DATA BY CONDITION USING “SELXAVG-SESS”**

In step 7 (mkanalysis-sess.new) you created the analysis by specifying all the parameters needed to analyze the data. Now, using selxavg-sess, the actual analysis takes place.

This program separately analyzes the data in each session indicated in the sessid file, then computes the average signal intensity maps for each condition. This average data can be further processed on an individual basis and/or can be used for group analyses.

Note: The -fsd flag specifies the functional subdirectory when you make the analysis, and is not needed in this step.

In your Study Directory, run “selxavg-sess” with all the following options:

**-analysis analysisname** from Step 7  
**-sf sessid** from Step 3  
**-df sesspar** from Step 3

**selxavg-sess -sf sessid -df sesspar -analysis sem\_assoc**

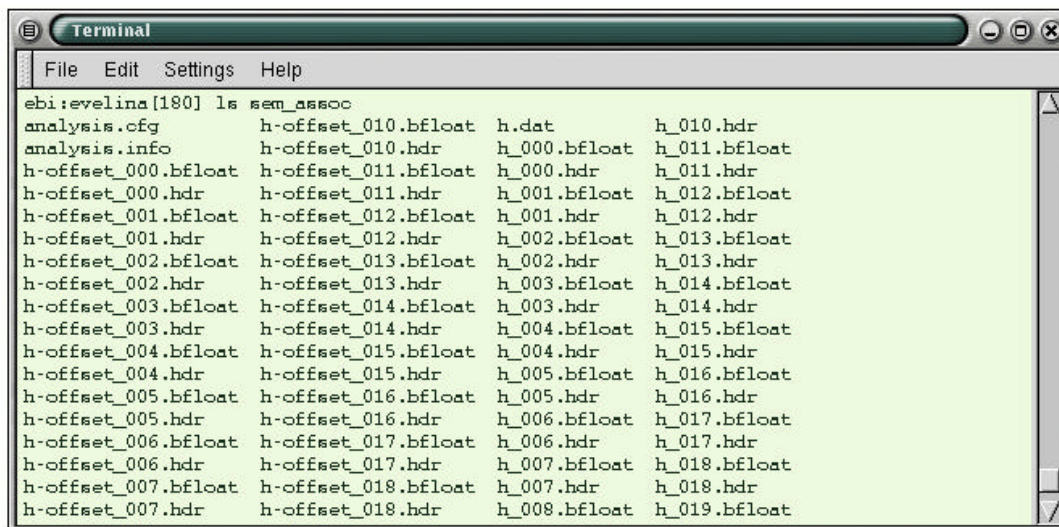
**OUTPUT:** This will create a subdirectory with the same name as your “analysisname” (e.g. sem\_assoc) under the bold directory in Bert’s session directory. There will be two volumes in this directory, one with stem “h” and one with stem “h-offset”. These are where the hemodynamic averages are stored, i.e.:

**h:** contains the estimated hemodynamic response at each voxel, along with the standard deviation of the residual error.

**h-offset:** contains a map of the mean value at each voxel.

There is also a text file called h.dat with information about the parameters used during the analysis. Future processing of this analysis will create other subdirectories under “analysisname”.

Here is what the beginning of Bert’s sem\_assoc directory looks like after running this step, which takes about 15 minutes to complete:



```
Terminal
File Edit Settings Help
ebi:evelina [180] ls sem_assoc
analysis.cfg          h-offset_010.bfloat h.dat          h_010.hdr
analysis.info        h-offset_010.hdr   h_000.bfloat  h_011.bfloat
h-offset_000.bfloat  h-offset_011.bfloat h_000.hdr     h_011.hdr
h-offset_000.hdr     h-offset_011.hdr   h_001.bfloat  h_012.bfloat
h-offset_001.bfloat  h-offset_012.bfloat h_001.hdr     h_012.hdr
h-offset_001.hdr     h-offset_012.hdr   h_002.bfloat  h_013.bfloat
h-offset_002.bfloat  h-offset_013.bfloat h_002.hdr     h_013.hdr
h-offset_002.hdr     h-offset_013.hdr   h_003.bfloat  h_014.bfloat
h-offset_003.bfloat  h-offset_014.bfloat h_003.hdr     h_014.hdr
h-offset_003.hdr     h-offset_014.hdr   h_004.bfloat  h_015.bfloat
h-offset_004.bfloat  h-offset_015.bfloat h_004.hdr     h_015.hdr
h-offset_004.hdr     h-offset_015.hdr   h_005.bfloat  h_016.bfloat
h-offset_005.bfloat  h-offset_016.bfloat h_005.hdr     h_016.hdr
h-offset_005.hdr     h-offset_016.hdr   h_006.bfloat  h_017.bfloat
h-offset_006.bfloat  h-offset_017.bfloat h_006.hdr     h_017.hdr
h-offset_006.hdr     h-offset_017.hdr   h_007.bfloat  h_018.bfloat
h-offset_007.bfloat  h-offset_018.bfloat h_007.hdr     h_018.hdr
h-offset_007.hdr     h-offset_018.hdr   h_008.bfloat  h_019.bfloat
```

## **9. DEFINE AN OMNIBUS CONTRAST USING “MKCONTRAST-SESS”**

A contrast is an instantiation of a hypothesis. The omnibus contrast is a test for any task-related activity against the baseline. Here, the “baseline” means the variance left unexplained after fitting the time course at each voxel for the mean, linear trend, and task-related activity. Mathematically, the mean of the baseline is forced to zero. The omnibus is usually tested first to ensure that the data have been processed properly and that the subject was responding. If there is no omnibus activity, then there is no use in continuing the analysis.

As with mkanalysis-sess, this step merely prepares the required information for the computation, which actually occurs later. A contrast only needs to be made ONCE. If you change the analysis (i.e. the time window, the prestimulus window, or the number of conditions), then you have to recreate the contrast.

You will supply a name for this comparison, which will be required in subsequent processing steps. For multiple comparisons, you would run mkcontrast-sess multiple times, defining a different contrast name each time. The contrast is defined with respect to a particular analysis by using the command options.

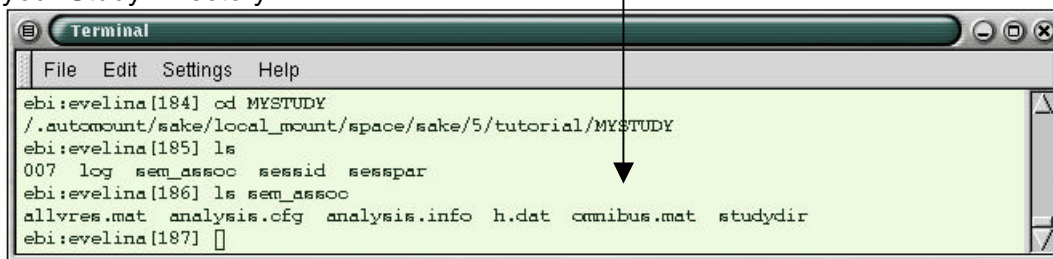
In your Study Directory, you would run “mkcontrast-sess” with the all the following options:

<b>-analysis analysisname</b>	name of analysis
<b>-contrast contrastname</b>	name of contrast (you have to name it, e.g. “omnibus”)
<b>-a active_condition_code (#)</b>	to specify >1 active conditions use multiple –a options
<b>-c control_condition_code (#)</b>	to specify >1 active conditions use multiple –c options
<b>-nosumconds</b>	specifies that the conditions be tested separately

For Bert, type:

**mkcontrast-sess –analysis sem\_assoc –contrast omnibus –a 1 –a 2 –a 3 –a 4 –c 0 -nosumconds**

**OUTPUT:** This will create a file called “omnibus.mat” under the “analysisname” directory in your Study Directory:



Condition codes are the same as those used in the paradigm file specified for this analysis in mkanalysis-sess.new, in Step 7.

## **10. COMPUTE STATISTICAL MAPS OF THE OMNIBUS CONTRAST USING “STXGRINDER-SESS”**

This program computes the actual contrast. Statistical, significance, and parametric maps for all of your subjects on an individual basis can be computed. (Note that the same program with different options is used for the group analysis—see the FS-FAST Guide for more details.) The map is computed from a given analysis and contrast. Significance results are stored as  $\log_{10}$  unless you specify otherwise using the –pxform option.

In your Study Directory, you would run “stxgrinder-sess” with the following options:

<b>-analysis analysisname</b>	name of analysis from Step 7
<b>-contrast contrastname</b>	name of contrast from Step 9
<b>-sf sessid</b>	from Step 3
<b>-df sesspar</b>	from Step 3

For Bert, type the following:

**stxgrinder-sess –contrast omnibus –analysis sem\_assoc -sf sessid –df sesspar**

---

<sup>1</sup> Note to xds users: xds sets the activation color scale assuming that the significance values have been transformed by the natural log.



This step takes about 2 minutes to complete.

**OUTPUT:** This will create a subdirectory called “omnibus” under the bold/analysisname directory for each session. Within this directory, you will find six new volumes with stems “f”, “fsig”, “t”, “sig”, “minsig”, and “iminsig”.

**f** – value of the F ratio

**fsig** – significance of the F-test

**t** – stores the variate-by-variate t-statistic (value of the t ratio)

**sig** – significance of the t-test, which will be determined by default (possibly multiple maps for multiple post-stimulus delays)

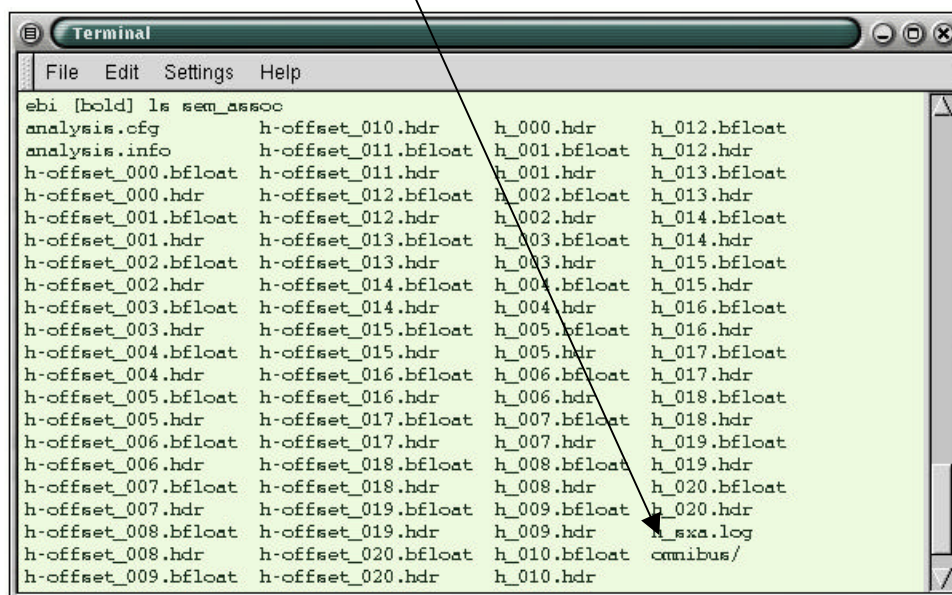
**minsig** – looks for best significance, i.e. time at which best significance occurred for each voxel, using the Bonferroni corrected t-test

**iminsig** – tells you in which frame the best significance was found; the index of the best t-test significance at each voxel

Each map contains the frames of the points on graph (see Appendix A). The number of frames in each map is determined by the values you set for `-timewindow` and `-prestim` commands in `mkanalysis` (i.e. 1 image per TR in time window). There are 12 different significance maps.

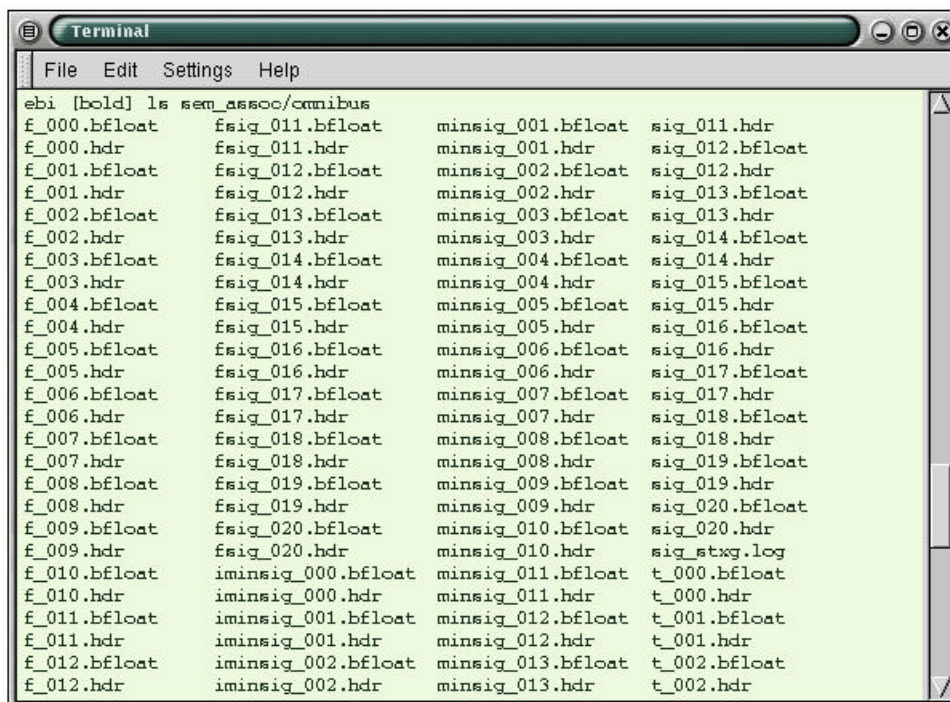
The variate-by-variate t-statistic is stored in the “t” volume and its significance is stored in the “sig” volume. The variate is a time-point (or ideal hemodynamic curve). Multiple variates only apply to event-related design. In addition, the best variate (as measured by the minimum significance) at each voxel is also determined and stored as a map (after Bonferroni correction) in the “minsig” volume. The index of the best variate is stored in the “iminsig” volume. In subsequent stages, you will be required to enter a *map*—this refers to one of these volumes.

Here we see the new omnibus subdirectory created in Bert’s bold/sem\_assoc directory:



```
Terminal
File Edit Settings Help
ebi [bold] ls sem_assoc
analysis.cfg      h-offset_010.hdr  h_000.hdr      h_012.bfloat
analysis.info    h-offset_011.bfloat h_001.bfloat  h_012.hdr
h-offset_000.bfloat h-offset_011.hdr  h_001.hdr      h_013.bfloat
h-offset_000.hdr h-offset_012.bfloat h_002.bfloat  h_013.hdr
h-offset_001.bfloat h-offset_012.hdr  h_002.hdr      h_014.bfloat
h-offset_001.hdr h-offset_013.bfloat h_003.bfloat  h_014.hdr
h-offset_002.bfloat h-offset_013.hdr  h_003.hdr      h_015.bfloat
h-offset_002.hdr h-offset_014.bfloat h_004.bfloat  h_015.hdr
h-offset_003.bfloat h-offset_014.hdr  h_004.hdr      h_016.bfloat
h-offset_003.hdr h-offset_015.bfloat h_005.bfloat  h_016.hdr
h-offset_004.bfloat h-offset_015.hdr  h_005.hdr      h_017.bfloat
h-offset_004.hdr h-offset_016.bfloat h_006.bfloat  h_017.hdr
h-offset_005.bfloat h-offset_016.hdr  h_006.hdr      h_018.bfloat
h-offset_005.hdr h-offset_017.bfloat h_007.bfloat  h_018.hdr
h-offset_006.bfloat h-offset_017.hdr  h_007.hdr      h_019.bfloat
h-offset_006.hdr h-offset_018.bfloat h_008.bfloat  h_019.hdr
h-offset_007.bfloat h-offset_018.hdr  h_008.hdr      h_020.bfloat
h-offset_007.hdr h-offset_019.bfloat h_009.bfloat  h_020.hdr
h-offset_008.bfloat h-offset_019.hdr  h_009.hdr      h_sxa.log
h-offset_008.hdr h-offset_020.bfloat h_010.bfloat  omnibus/
h-offset_009.bfloat h-offset_020.hdr  h_010.hdr
```

And next we see the contents of the omnibus subdirectory, with their “f”, “fsig”, “t”, “sig”, “minsig”, and “iminsig” stems:



```
Terminal
File Edit Settings Help
ebi [bold] ls /$SUBJECTS_DIR/omnibus
f_000.bfloat      fsig_011.bfloat      minsig_001.bfloat    sig_011.hdr
f_000.hdr         fsig_011.hdr         minsig_001.hdr       sig_012.bfloat
f_001.bfloat      fsig_012.bfloat      minsig_002.bfloat    sig_012.hdr
f_001.hdr         fsig_012.hdr         minsig_002.hdr       sig_013.bfloat
f_002.bfloat      fsig_013.bfloat      minsig_003.bfloat    sig_013.hdr
f_002.hdr         fsig_013.hdr         minsig_003.hdr       sig_014.bfloat
f_003.bfloat      fsig_014.bfloat      minsig_004.bfloat    sig_014.hdr
f_003.hdr         fsig_014.hdr         minsig_004.hdr       sig_015.bfloat
f_004.bfloat      fsig_015.bfloat      minsig_005.bfloat    sig_015.hdr
f_004.hdr         fsig_015.hdr         minsig_005.hdr       sig_016.bfloat
f_005.bfloat      fsig_016.bfloat      minsig_006.bfloat    sig_016.hdr
f_005.hdr         fsig_016.hdr         minsig_006.hdr       sig_017.bfloat
f_006.bfloat      fsig_017.bfloat      minsig_007.bfloat    sig_017.hdr
f_006.hdr         fsig_017.hdr         minsig_007.hdr       sig_018.bfloat
f_007.bfloat      fsig_018.bfloat      minsig_008.bfloat    sig_018.hdr
f_007.hdr         fsig_018.hdr         minsig_008.hdr       sig_019.bfloat
f_008.bfloat      fsig_019.bfloat      minsig_009.bfloat    sig_019.hdr
f_008.hdr         fsig_019.hdr         minsig_009.hdr       sig_020.bfloat
f_009.bfloat      fsig_020.bfloat      minsig_010.bfloat    sig_020.hdr
f_009.hdr         fsig_020.hdr         minsig_010.hdr       sig_stxg.log
f_010.bfloat      iminsig_000.bfloat   minsig_011.bfloat    t_000.bfloat
f_010.hdr         iminsig_000.hdr      minsig_011.hdr       t_000.hdr
f_011.bfloat      iminsig_001.bfloat   minsig_012.bfloat    t_001.bfloat
f_011.hdr         iminsig_001.hdr      minsig_012.hdr       t_001.hdr
f_012.bfloat      iminsig_002.bfloat   minsig_013.bfloat    t_002.bfloat
f_012.hdr         iminsig_002.hdr      minsig_013.hdr       t_002.hdr
```

## **11. RUN FUNCTIONAL/STRUCTURAL REGISTRATION USING “AUTOREG-SESS”**

The functional volumes for each subject need to be registered with the reconstructed anatomical volume in the Anatomical Database – i.e. the directory where you keep the subjects whose anatomical data have been reconstructed with FreeSurfer..

### **INTERFACING WITH FREESURFER**

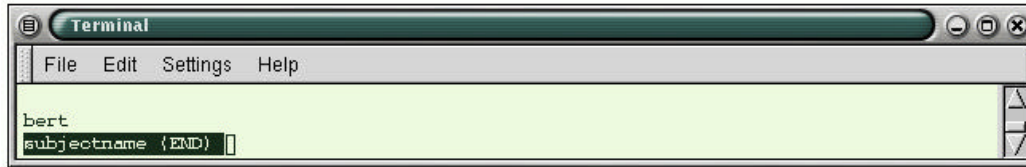
Linking into Freesurfer allows the user to view the functional results on high-resolution 3D anatomical images as well as on the cortical surface. It also allows the user to resample into talairach, spherical, or region-of-interest (ROI) spaces in order to perform intersubject averaging. Of course, to link to Freesurfer, the subject must have been processed through the Freesurfer anatomical reconstruction stream. For more information on FreeSurfer, see:

<http://surfer.nmr.mgh.harvard.edu/download.html>

#### **Creating The Subjectname File**

Before you proceed, ensure that a file called "subjectname" appears in the functional session. You must create this file. One of the first steps in the anatomical reconstruction is to create a subject directory where “subjectname” is a unique identifier for each subject that will be analyzed. This is also known as the Subject Identifier String. The subject’s anatomical reconstruction directory must be visible from the /\$SUBJECTS\_DIR, i.e. you should see something when running the unix command {em /\$SUBJECTS\_DIR/subjectname}. To interface FS-FAST to FreeSurfer, you must create a file called subjectname in the functional

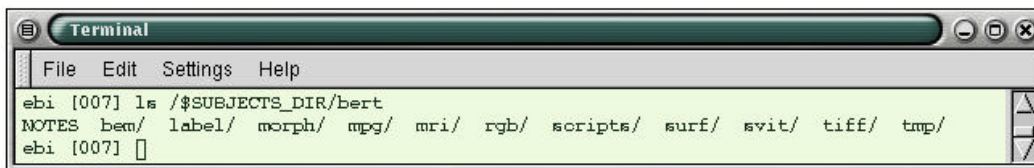
session directory, the contents of which should be the name of the subject. Note that the contents hold the subject identifier string, whereas the file itself should actually be called “subjectname”. For example, for Bert you would cd to bert-functional and create a file called subjectname with contents “bert” (no quotes), e.g.:



Then, to ensure that Bert’s reconstructed anatomical data set (the one you downloaded in the first step) exists in the right place on your system, type:

**ls /\$SUBJECTS\_DIR/bert**

You should now see the unique FreeSurfer anatomical directory structure for bert:



## **Functional/Anatomical Registration**

Registration can be challenging because the two volumes have different resolutions, contrast weightings and fields of view. In addition, they may have been collected on different sessions, sometimes years apart and on different scanners. Yet, we still need to assign voxels in the anatomical space to voxels in the functional space in order to view the functional data on the anatomical data. This is done in three stages, two of which are automatic. The automatic is used to get the registration close, and the manual is used to fine-tune the registration.

### **Automatic Registration**

The automatic registration can be broken into two conceptual steps, though both are accomplished with one command.

The automatic registration can be used only if a high-resolution T1-weighted anatomical data set was acquired during the same scanning session. These may have been collected in order to process the subject with FreeSurfer, in which case there will probably be two or three very high-quality scans. Or, if a subject has already been reconstructed, there may only be one low-quality anatomical acquired during the functional session. The latter is the case for bert. One run of low-quality anatomicals can be found in bert-functional/3danat. These are referred to as the “same-session anatomicals”, whereas those in \$SUBJECTS\_DIR/bert are referred to as the “reconstructed anatomicals” and are high quality.

**The first step is the *Header Registration* in which the functional volumes are registered with the anatomicals that were collected during the same session. No MRI data are used in the header**

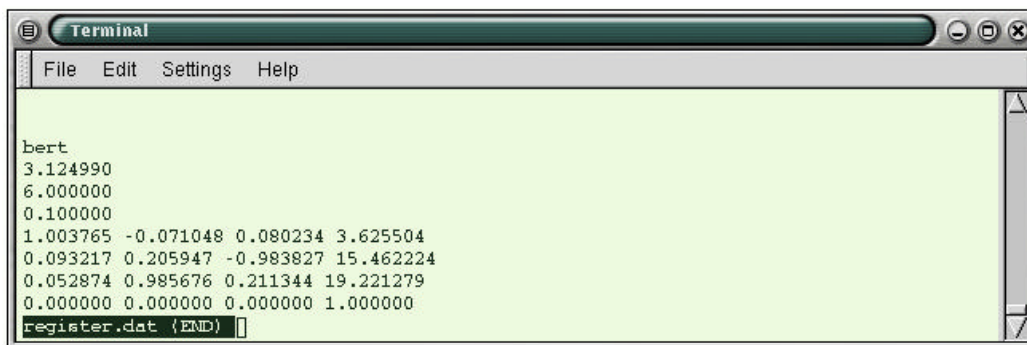
registration. It relies on the assumption that the head has not moved so that registration can be accomplished simply by registering the fields-of-view.

In the second step, the same-session anatomical is registered with the Database anatomicals. This is accomplished by translating/rotating the volumes based on a voxel-by-voxel comparison until they match best. This can be done because both volumes have T1 contrast.

Concatenating these two transforms yields the transform between the functionals and the reconstructed anatomicals. Those using this procedure should cite Collins et al (1994). The registration for Bert can be accomplished by typing the following in your Study Directory:

```
autoreg-sess -sf sessid -df sesspar
```

This will create a file called register.dat in bert-functional/bold. This file will have eight lines:

A terminal window titled "Terminal" with a menu bar (File, Edit, Settings, Help) and a green background. The text displayed is:

```
bert
3.124990
6.000000
0.100000
1.003765 -0.071048 0.080234 3.625504
0.093217 0.205947 -0.983827 15.462224
0.052874 0.985676 0.211344 19.221279
0.000000 0.000000 0.000000 1.000000
register.dat {END}
```

The first four lines are: subjectname, in-plane resolution, between-plane resolution, and intensity value. The between-plane resolution is the slice thickness if there is no skip. The intensity value does not actually have anything to do with the registration itself; it is used when displaying the functional volume during the manual registration phase. The last four lines of register.dat have the values of the registration matrix that converts a location in anatomical space to a location in functional space.

### Manual Registration

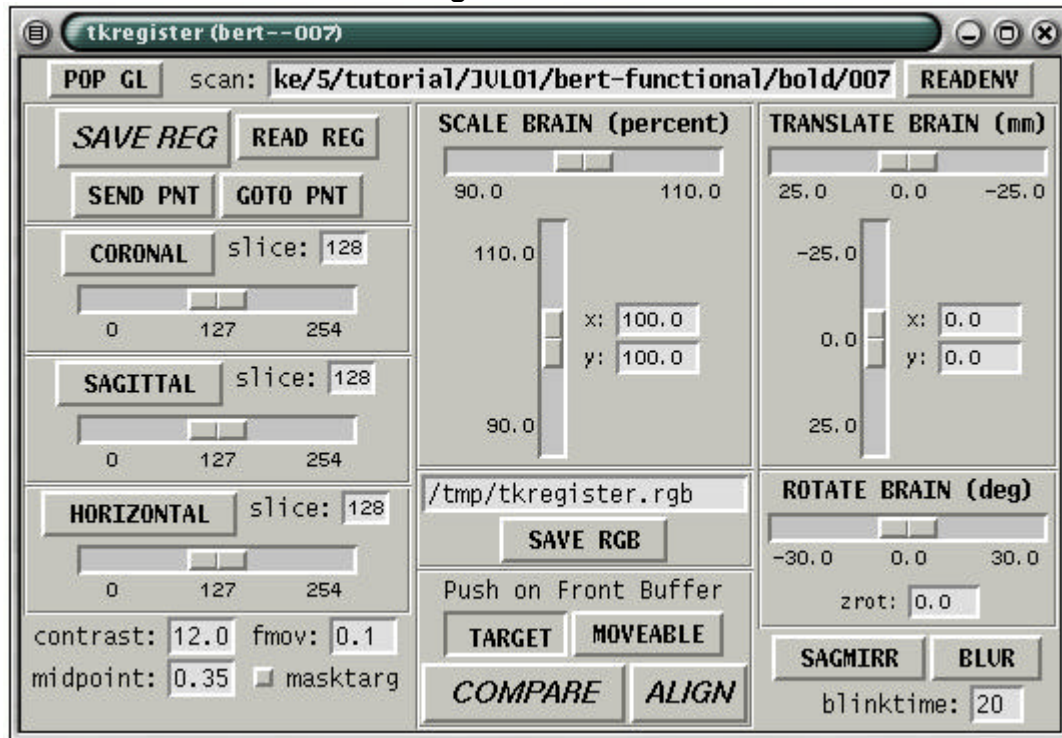
Because of movement and EPI distortion, the automatic stage can only get the registration close. Manual registration is used to check and fine-tune the automatic registration, or to perform the entire registration in the case that there are no same-session anatomicals or if the automatic registration failed. Even if the automatic registration succeeded, always check the registration manually by using the following commands:

```
tkregister-sess -sf sessid -df sesspar
```

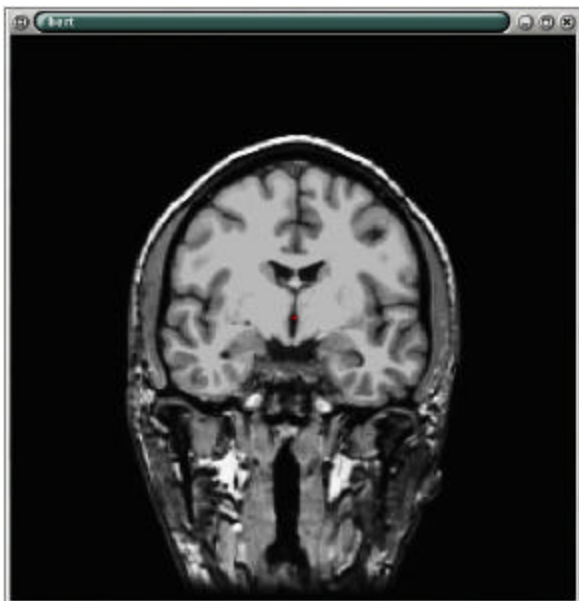
This will read the register.dat created by the automatic registration and bring up two windows, one with a brain, the other a control window.



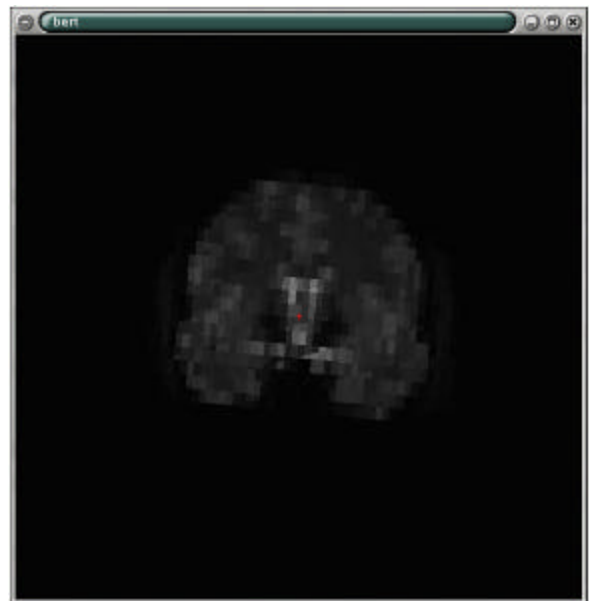
## The TkRegister Control Window



Bert's Anatomical Volume



Bert's Functional Volume



Hitting the COMPARE button in the control window allows toggling between the functional and the anatomical volumes. The image window will initially have a coronal view of the anatomical volume, but you can change the view by hitting the “coronal” or “sagittal” or horizontal” buttons. The functional brain can be translated left or right by adjusting the horizontal scrollbar under “Translate Brain”. It can be translated up and down by using the vertical scrollbar. The brain can be rotated about the red cursor by using the “Rotate Brain” scrollbar. With these operations, the user can navigate through the volume and check the registration.

## Fine Tuning the Registration

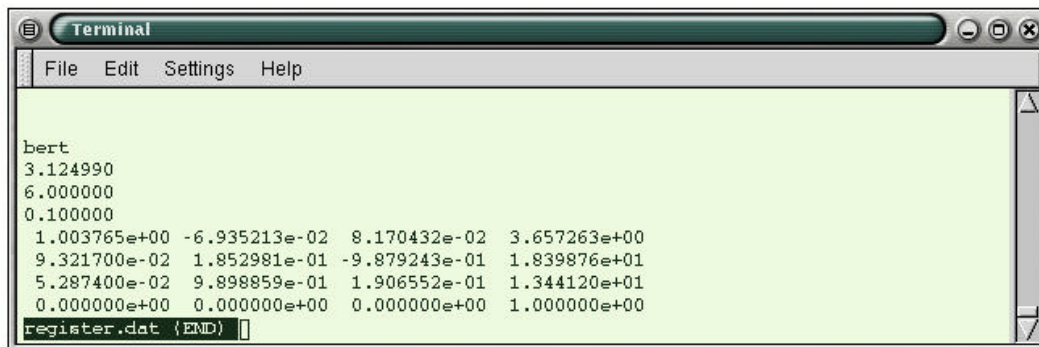
If the registration is off, then the user has three options: translation, rotation, and scaling, each of which has its own frame inside the control window. Within the Translation frame, there are two sliders. The vertical slider translates the functional brain in the vertical direction, where “vertical” is with respect to the current view in the image window. The horizontal slider translates the functional brain horizontally. The Rotation frame has only one slider. Moving the slider will rotate the functional brain around the current cursor location about the axis perpendicular to the current image view. The Scale frame has two sliders. The vertical slider will stretch/compress the functional brain in the current vertical direction. The horizontal slider does the same for the current horizontal direction. When finished, hit the SAVE REG button (it may prompt you to overwrite the current registration; if so, do so). To quit, hit the button on the upper left hand corner to bring down a menu and choose the Close option. This will update the register.dat. Note that rerunning autoreg-sess will overwrite your results from the manual registration. Also note that any changes in registration will require that all downstream operations (e.g., resampling, inter-subject averaging) be rerun.

When the volumes are properly registered, the anatomical features in the functional slice should perfectly overlay those in the anatomical slice. After adjusting the registration this way, hit the “SAVE REG” button (answer **yes** to overwrite).

**OUTPUT:** This step creates a file called *register.dat* in each functional subdirectory. This represents the registration between the functional volume and the Database anatomical volume.

### **Registration Used in the Tutorial**

Here are the new contents of register.dat used for Bert’s registration in this tutorial:



```
Terminal
File Edit Settings Help

bert
3.124990
6.000000
0.100000
1.003765e+00 -6.935213e-02 8.170432e-02 3.657263e+00
9.321700e-02 1.852981e-01 -9.879243e-01 1.839876e+01
5.287400e-02 9.898859e-01 1.906552e-01 1.344120e+01
0.000000e+00 0.000000e+00 0.000000e+00 1.000000e+00
register.dat {END}
```

### **Tips for Manual Registration:**

Manual registration can be a tedious process. It is recommended that the user first translate, then rotate, and, as a last resort, scale. Keep in mind that a registration improvement in one part of the volume can degrade that in another part. It is best to make small adjustments at several locations around the brain, slowly converging on the optimum registration.

## **12. VISUALIZATION**

### **Visualization: slice-based using “sliceview-sess”**

In Slice-based visualization, the results are displayed on 2D slices of the brain—either individual slices or mosaics of slices.

To see the individual results for any subjects in your study for a single slice, you would run “sliceview-sess” from your Study Directory with the following options:

<b>-slice slicenumber</b>	number of the slice to view
<b>-analysis analysisname</b>	name of analysis from mkanalysis-sess.new (Step 7)
<b>-contrast contrastname</b>	name of contrast from mkcontrast-sess (Step 9)
<b>-map mapname</b>	name of map made in Step 10 (sig, t, minsig, iminsig)
<b>-sf sessid</b>	from Step 3
<b>-df sesspar</b>	from Step 3

To view Bert’s slices as a mosaic, go to your Study Directory and type:

```
sliceview-sess –contrast omnibus –analysis sem_assoc –sf sessid –df sesspar –map fsig –slice mos –nohdr
```

(If you were running this on multiple subjects, the results for each subject will be displayed sequentially, i.e., you must close the window for one subject before you can see the next).

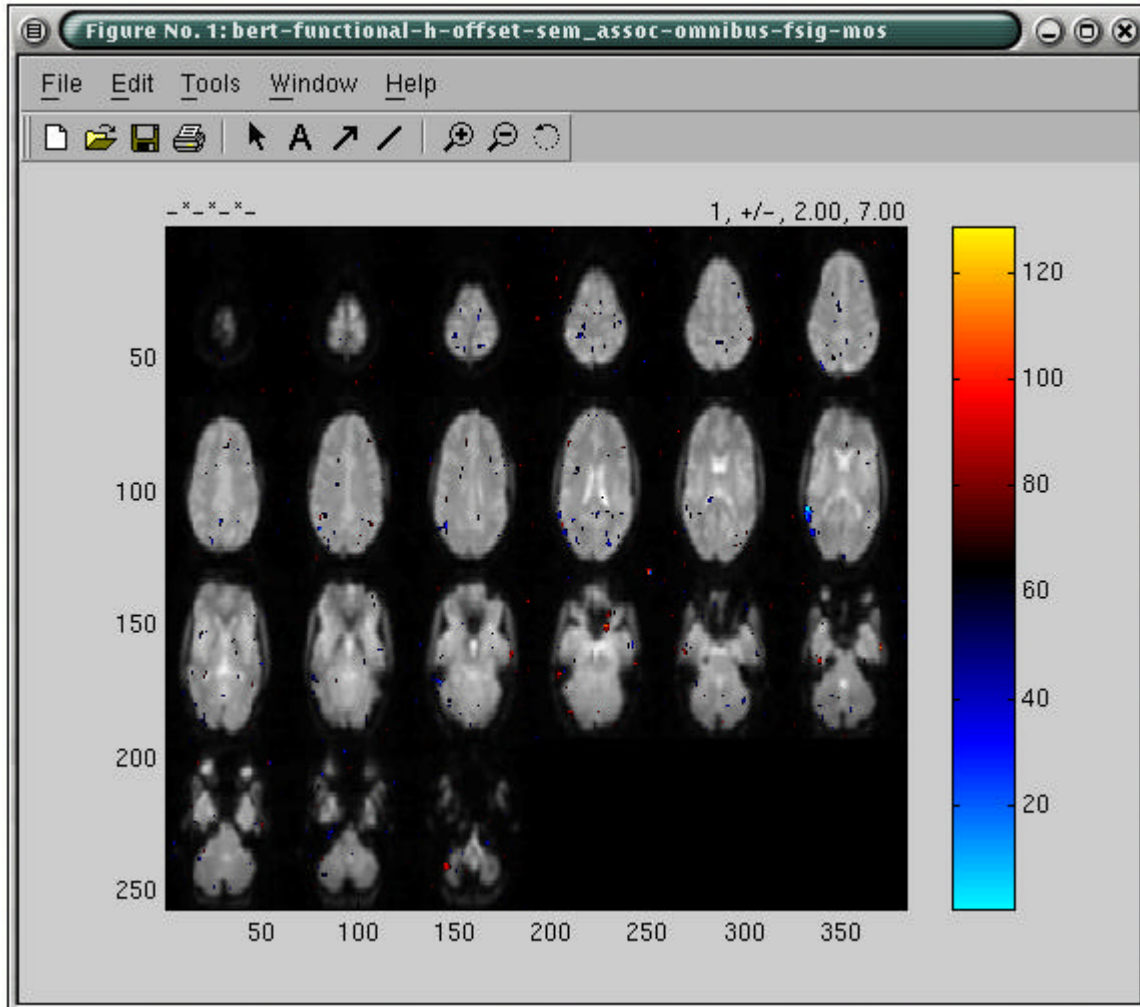
### **Sliceview Operation**

Sliceview is one way of visualizing the statistical maps you generated with FS-FAST. Actually a front-end for a program called yakview, it can display maps laid over T1 or T2 structural images (slices or mosaics of slices), and can also display the average hemodynamic changes in response to different event types.

When sliceview is run, it immediately brings up an image window displaying the structural and overlay slices. The base image (or underlay) is the average functional image. The overlay is the significance of the F-test. Details on how to operate the viewer follow in Sliceview Operation.

Note: If you wanted to view motion corrected data, you would add the flag “-motioncor”. If the functional subdirectory were not named “bold” then you would use the –fsd flag to specify the name. The –raw and –fsd flags can be used at the same time with –analysis, -contrast, and –map flags to simultaneously view raw time courses, hemodynamic responses, and statistical maps.

**OUTPUT:** On the next page is Sliceview showing the initial mosaic we get for Bert:



Color indicates significances in the range of 2 (i.e.  $10^{-2}$ ) to 7 (i.e.  $10^{-7}$ ). The **red/yellow** scale indicates activations **above** baseline. The **blue** scale indicates activations **below** baseline.

Other maps (e.g. f, sig, minsig, iminsig) can be viewed by changing the `-map` option. For the omnibus test, only the fsig is relevant. If you want to view only a single slice, use `-slice SLICENO`, where SLICENO is the zero-based slice number. For event-related studies the hemodynamic response for each condition can be viewed by clicking on a particular voxel in the slice. Alternatively, the `-nohdr` option tells sliceview not to load the hemodynamic response. For more information, see the upcoming section 'Viewing the Hemodynamic Response'.

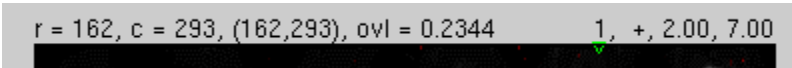
No information about the volume is available. The slices are defined as those in the native functional space. Typically, a significance map from a particular contrast is brought to a specified threshold and laid over a structural slice.



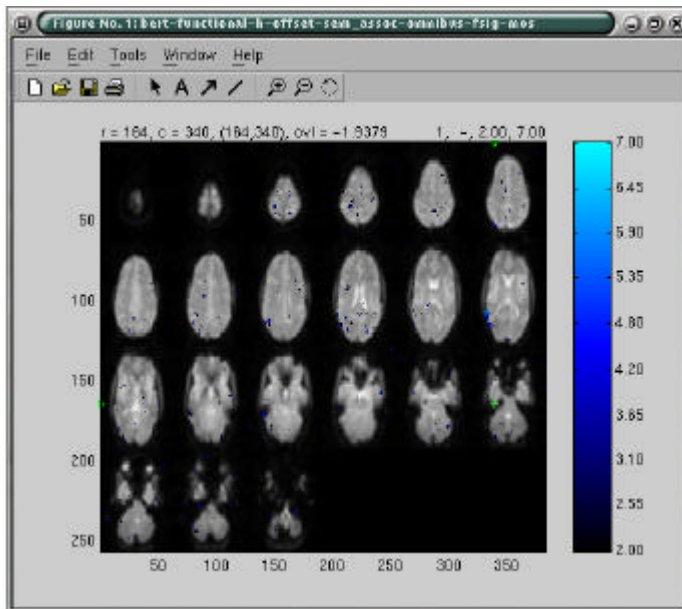
**Viewing:** The minimum and maximum thresholds are set when invoking sliceview by using the `-themin` and `-thmax` options. Clicking with the mouse on a voxel will cause the voxel location and overlay value to be printed above the image. The voxel selected by the mouse is referred to as the current voxel. The various functions of sliceview are controlled by hitting keys when the sliceview window is open. Hitting “h” brings up a help menu of control keys:



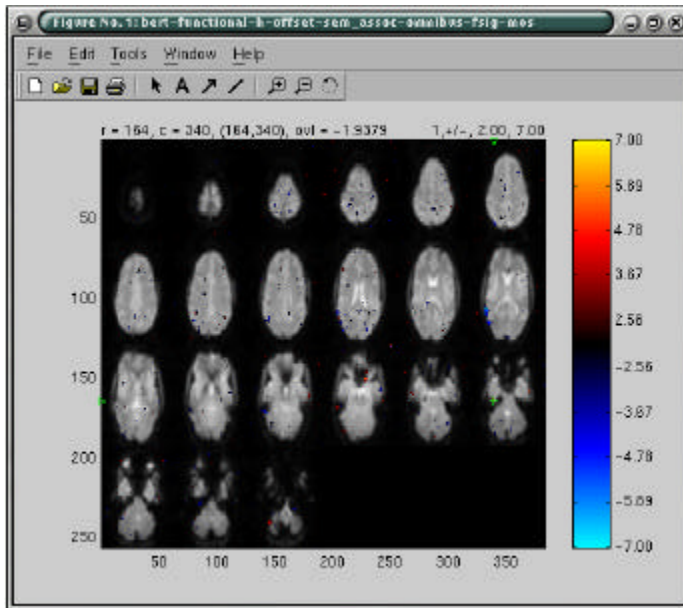
**Image Status Bar:** The status bar is just above the image (this may not be visible during zoom). The first set of numbers gives the index of the current voxel in the structural image. The second set gives the index in the map. The third number is the value of the map at the current voxel. On the right side of the image, the fields show the current plane, sign of the tail, and the range of values for the color scale:



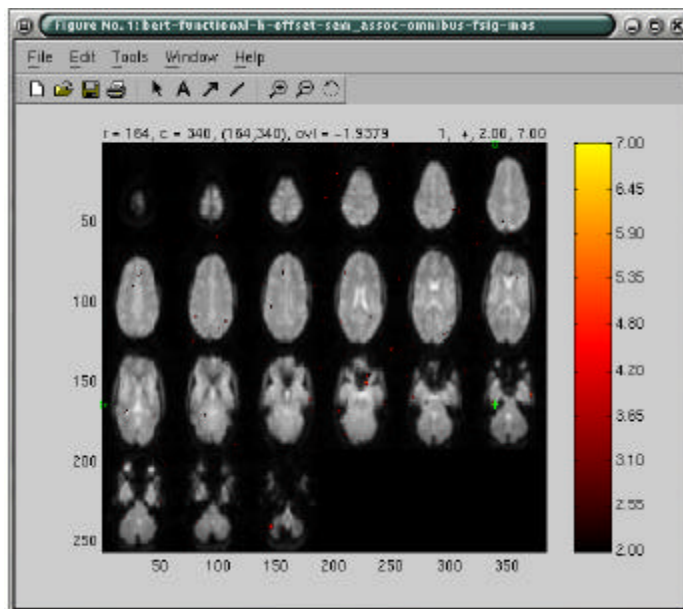
**Image Control Keys:** Initially, only the positive tails will be displayed in a red-yellow color scale. Hitting “t” will shift to the negative tails in a blue color scale:



Hitting “t” again will show both positive and negative tails in a red-yellow scale:



Hitting “t” again brings it back to positive only:



### Viewing Bert's hemodynamic responses

Bert's hemodynamic responses could have been viewed with the omnibus contrast, but it is more instructive to view them using an all-versus-baseline overlay. First, create the contrast by running the following in your Study Directory:

```
mkcontrast-sess -analysis sem_assoc -contrast allvbase -a 1 -a 2 -a 3 -a 4 -c 0
```

Note that this is similar to the omnibus created in Step 9, except that it does not include the `-nosumconds` flag. This means that the contrast will sum the conditions together, whereas the omnibus tests each condition separately. The omnibus and all-v-baseline generally paint the same picture, but all-v-baseline allows for some more sophisticated display options.

To compute the contrast maps, type:

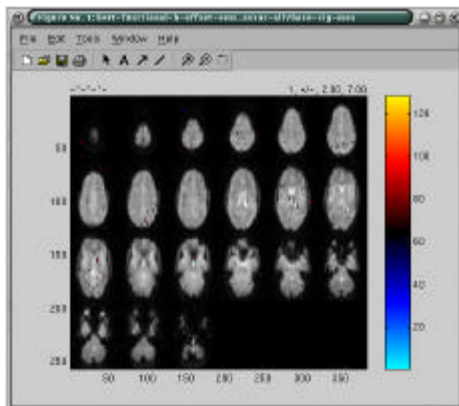
```
stxgrinder-sess -contrast allvbase -analysis sem_assoc -sf sessid -df sesspar
```

To view the contrast maps, type:

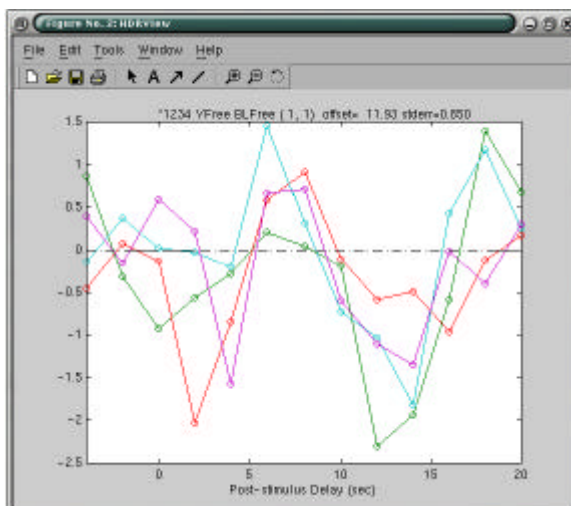
```
sliceview-sess -contrast allvbase -analysis sem_assoc -sf sessid -df sesspar -map sig -slice mos
```

Note that the `-nohdr` flag has been removed. This forces the hemodynamic averages to be displayed (the loading can take a while). Also, the map is now `sig` instead of `fsg`, meaning that you are about to view the par-poststimulus-delay t-significance maps.

When the slices are displayed in the **image window**, you will not see much activation, which is fine, because you are looking at the map of the activation 4 seconds *before* stimulus onset:

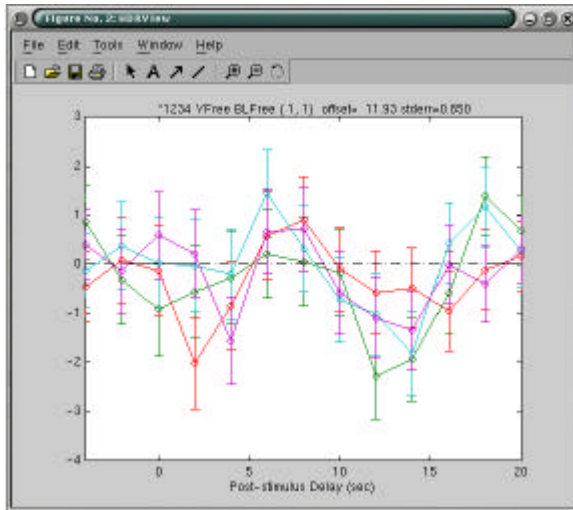


Click in the window and press `<g>`. This will bring up another window with four plots:

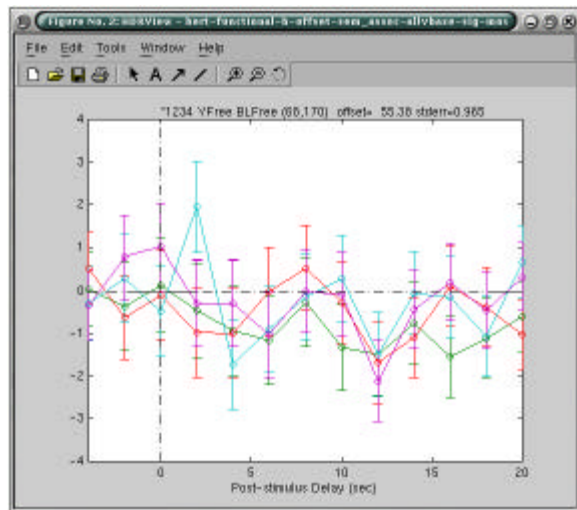
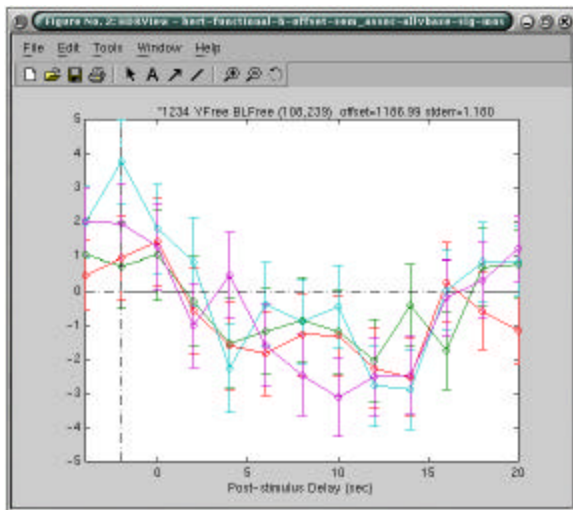


Each plot represents the unbiased hemodynamic response to each of the four conditions at the voxel selected in the image window. As you click in the image window, you will change the time courses in this plot window.

Click in the **plot window** and hit <e>; this will display the standard error bars for each condition at each time point:



Click in the image window and hit <+>. This will advance the map from that of 4 seconds before stimulus onset to 2 seconds before stimulus onset. You will also see a vertical line in the post-stimulus delay you are currently viewing. Hitting <+> again will advance to the next frame thus allowing you to view the activation like a movie:



### HDR Status Bar

There is a series of text fields just above the HDR plot that indicates the current status of the window:

- The first field contains a series of numbers (and possible asterisks) that indicate which conditions are currently being displayed.
- The next field is either ‘Yfree’ or ‘Yhold’. Yfree: indicates the y-axis will scale the data. The field is either ‘BLFree or BLZero indicating whether the pre-stimulus baseline has been zeroed.
- The next *two* fields indicate the index of the current voxel.
- The next field will be either blank or “%” to indicate whether percent baseline is being viewed.
- Finally, the last field gives the baseline (or offset) value at the current voxel.

## HDR Control Keys

Pressing “h” when the HDR window is active will bring up a HELP menu with the available control keys and a short description of what they do.

Pressing a number key will toggle the display of that condition (unviewed conditions have an asterisk in their status field).

Pressing “e” will toggle the display of standard error bars.

Pressing “p” will toggle display of percent signal change.

Pressing “y” will toggle whether the y-axis will change with the data (Yfree) or stay fixed at the current range (Yhold).

Pressing “v” will save the plot values to an ascii file.

## A Contrast-of-Interest

Up to this point, you have only been looking at task-vs-nothing contrasts which, while useful, are not particularly interesting. The contrast-of-interest will compare the response to two tasks, loosely related with four words (L4) versus highly related with two words (H2). Here is the command to use for Bert:

```
mkcontrast-sess –analysis sem_assoc –contrast L4vH2 –a 4 –c 1 –sumdelays
```

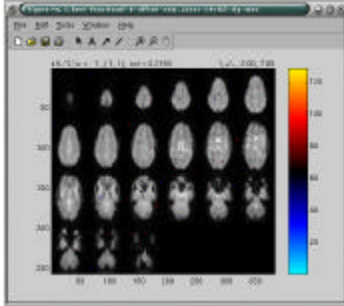
Using “-a 4 –c 1” means that positive values indicate that L4 is greater than H2. Normally, a separate t-test is done at each poststimulus delay. However, the –sumdelays flag indicates that the statistical test should be done after summing the hemodynamic responses across poststimulus delays. This can help bring out activation when the responses to two conditions are similar but one is slightly and consistently larger than the other over time. To compute Bert’s contrast, type the following in your Study Directory:

```
stxgrinder-sess –contrast L4vH2 –sf sessid –df sesspar
```

And to view the contrast, type:

```
sliceview-sess –contrast L4vH2 –analysis sem_assoc –sf sessid –df sesspar –map sig –slice mos
```

In the ninth slice, you should see a small patch of activation in the posterior portion of the left inferior prefrontal cortex (LIPC). In the thirteenth slice, you should see a small patch of activation in the anterior portion of the LIPC. A snapshot appears on the next page.



**Note that the images are in radiological convention, so the activity of the left hemisphere will appear on the right side of the image.**

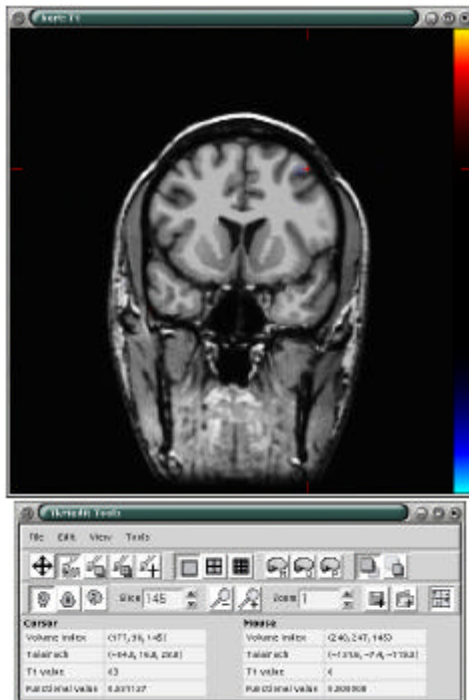
Also note that it is not possible to scroll through the different post-stimulus delays because they have all been collapsed into one image.

### Viewing the Functional Results on the Anatomical Volume

In the section “sliceview.sec”, we showed how to view the functional results overlaid on the original functional slices. In this section we show how to view the functional results on the high-resolution T1 anatomical volume. Here is the command to type from your Study Directory:

**tkmedit-sess -sf sessid -df sesspar -a sem\_assoc -c omnibus -map fsig**

You should see activation in the posterior LIPC at approximately talairach coordinates (-54, 16, 28). You should see activation in the anterior LIPC at approximately talairach coordinates (-58, 12, 2). There should also be activation in the visual and motor cortices.



## Visualization: surface-based using “paint-sess” and “surf-sess”

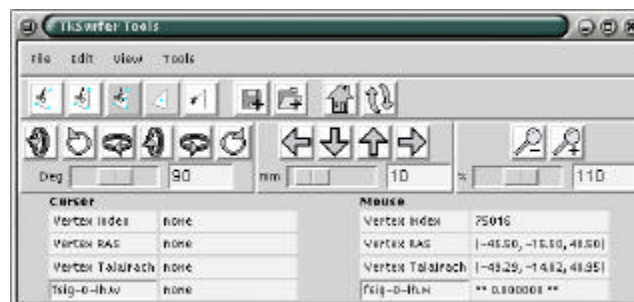
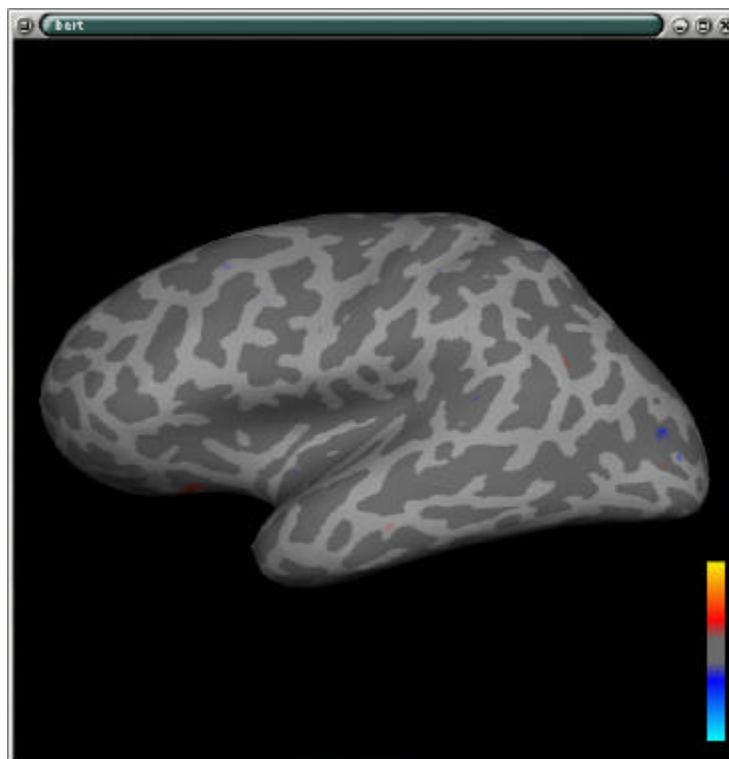
Surface-based visualization allows you to view the results of the functional analysis on the cortical surface (from the reconstructed data set that you downloaded in step 1). This surface can be folded, inflated, or flattened. The visualization requires two stages: “painting:” and “surfing”. Painting is the process of converting from native (Cartesian) space to cortical surface space. Surfing is the process of viewing the results on the surface. Note that these same programs (with different options) are used to view group-averaged results. To run each program, you must be in your Study Directory.

First, to paint, type the following:

```
paint-sess -sf sessid -df sesspar -a sem_assoc -c omnibus -map fsig
```

Then, to view Bert's functional results on the inflated surface, type:

```
surf-sess -sf sessid -df sesspar -a sem_assoc -c omnibus -map fsig
```



## Bibliography

Collins DL, Needlin P, Peters TM and Evans AC. **Automatic 3D Inter-Subject Registration of MR Volumetric Data in Standardized Talairach Space.** Journal of Computer Assisted Tomography, 18(2) 192-205; 1994.

Cox RW. AFNI: **Software for analysis and visualization of functional magnetic resonance neuroimages.** Computers and Biomedical Research, 29: 162-173; 1996.

Cox RW and Jesmanowicz A. **Real-time 3D image registration for functional MRI.** Magnetic Resonance in Medicine, 42: 1014-1018; 1999.

Wagner AD, Pare-Blagoev J, Clark J, and Poldrack RA. **Recovering Meaning: Left Prefrontal Cortex Guides Controlled Semantic Retrieval.** Neuron 31: 329-338. August 2, 2001.