CANLab Repository Guide

Mission: To provide a guide that holds your hand through all the steps of CANLab data analyses, and provides a brief theoretical motivation for each step.

This document is maintained by Marianne Reddan.

Contact: marianne.reddan@colorado.edu

For more information on this software check out http://canlabcore.readthedocs.org/

INTERNAL PROCEDURES

GETTING STARTED WITH BLANCA

What To Know More? Check the wiki

- **Log in**
  - Register for an account with research computing: https://www.rc.colorado.edu/accountrequest
  - If you cannot access /work/ics you need to email rc-help@colorado.edu and cc Tor and ask:
    - Could you give me permissions to towa7957grp and access to /work/ics?
  - Do you have one of those cool key fobs? You'll need it. Contact OIT if not.
  - ssh YOURIDENTIKEY@blogin01.rc.colorado.edu
  - Password is your 4 digit pin followed by whatever series of numbers appears when you click that button
  - save blanca as an alias so you type less in the future:
  - emacs .bash_profile
  - at the bottom of that add:
  - alias blancalogin='ssh mare8532@blogin01.rc.colorado.edu'
    - CNTRL+ XS to save
    - CNTRL+ XC to quit emacs editor
  - now you can just type in **blancalogin** to get in in the future
- **How to talk to Blanca**
  - Start terminal directly in the eye, and type something in:
    - ~~torque~~
    - ~~Moab, or~~
    - Slurm
      - slurm is now the queue submision system on blanca.
- **Configuring your Blanca**
  - this is on the wiki
  - edit your bash profile
  - emacs my.bashrc
  - paste all of this stuff in:
- **NOTE: R module removed because Blanca can't locate it.**
  - export MODULEPATH=$MODULEPATH:/projects/ics/modules
  - module add torque/torque-default-blanca
  - module add moab/moab-7.1.3
  - module add matlab/matlab-2014a
  - module add fsl/5.0.2.2
  - module add spm/spm8_r5236
  - module add afni
  - module add caret/5.65
  - module add mricron
  - #adding the spirit of luka
  - PATH=/home/ruzic/scripts:${PATH}
  - #adding dicom scripts
  - module add dcmtk
  - export DCMDICTPATH="$(dirname $(which dcmdump))/../share/dcmtk/dicom.dic"
  - #adding VNCPORT
  - export VNCPORT=`echo $DISPLAY|cut -d':' -f2`
  - alias qrsh='qsub -q blanca-ics -I -vDISPLAY=blogin01:$VNCPORT.0 -lwalltime=43200'

Note that there are newer versions for most of the software listed above that you should consider using.
for slurm module, add module load slurm/blanca to bash.rc

NOTE: when I use qrsh set up like this my job only lasts 24 hours for some reason. But if I type the same thing in the terminal, changing to VNCPORT to my vnc port my job lasts 7 days)

No longer current with slurm! The only way I have been able to get an interactive job to work so far is with this command: "sinteractive -p blanca-ics -A UCB00000358". note that the account number is for the blanca-ics allocation. Does not work when mapped to qrsh alias for some reason

--> interactive jobs should not run for more the 12/24hours

- configure MATLAB for SPM too

To edit this in the future, go back into the emacs editor, make the changes and then type:

- source my.bashrc

- **Start VNC Server**
  - You only need to do this once!
  - vncserver
  - in another terminal, on your home directory, edit bash_profile
  - emacs .bash_profile
  - add
  - alias blancavnc='ssh -L 5933:localhost:5933 [IDENTIKEY]@blogin01.rc.colorado.edu'
  - No one else can use your VNC port number (5933 is Marianne). You find this by, inside of blanca:
  - ps -ef | grep [IDENTIKEY] | grep vnc
  - In the print out, look for:
  - -rfbport 5933

    (and find your port number)
  - you can add this port number, your password, & localhost to chicken to vnc like that
  - now make sure you source this bash_profile:
  - source .bash_profile

- **Restarting your VNC Server**
  - ssh into Blanca
  - then terminate your vnc session:
  - vncserver -kill :[YOUR PORT NUMBER] (only last two digits)
  - NOTE: Your port number probably changed when this happened so scroll back up to the Start VNC Server instructions, and type this again to find out your new port number:
  - *ps -ef | grep [IDENTIKEY] | grep vnc*
  - If it changed, update your vnc alias (edit your bash_profile) like it describes above
  - Then start a new vnc session using the correct port number:
  - vncserver :[YOUR PORT NUMBER] (only the last two digits)

DO NOT TRY TO RESTART OR LOG OUT WHILE VNC-ING IN THE REDHAT ENVIRONMENT, YOU WILL HAVE A GIANT HEADACHE THAT RC-HELP CAN'T SOLVE. USE TERMINAL (with the instructions right above this rant).

Limiting OTP Use with SSH Sockets and Screen

Add the following to your .ssh/config file on your local computer. This will allow you to login with OTP only once and use sockets for additional terminals and rsync etc. I find this pretty helpful

- Host blogin01.rc*
- ControlMaster auto
- ControlPath ~/.ssh/sockets/%r@%h:%p

Will add note about Screen later

- **Navigating Blanca**
  - our data are stored in the archive folder
  - archive/wagerlab/

- **Moving your data**

Use rsync to move data. The first filepath is where you want to move data FROM. The second filepath is where you want to move data TO.

- If you need to move files TO BLANCA from another computer, use the following rsync command:
- cd [folder you want to move]
- rsync -auvR . [your_username]@blogin01.rc.colorado.edu:/work/ics/data/projects/wagerlab/labdata/projects/Imagination/
  - Your username here is your identikey
  - Password prompt will be your *blanca* password with the key fob
- If you need to move files FROM BLANCA, use the following rsync command:
- rsync -auvR . [your_username]@10.224.20.7:/Volumes/engram/labdata/data/
  - You need to put the IP address of Claustrum or Calor, not the calor.colorado.edu path.
    - Claustrum: 10.224.20.6

        - Calor: 10.224.20.7
    - Your username here is your name on Calor or Claustrum (i.e., marianne)
    - PW prompt will be for your *engram* password
    - For more information on engram and its mysteries, check the wiki
- If you followed the new file structure, chances are Zeb has already rsync-ed your data to Blanca. You need to un-tar it.

- **How to Un-Tar Your Data**

- *tar -xvf /archive/wagerlab/Imagination.tar -C /work/ics/data/projects/wagerlab/labdata/fmri_data*
    In English:
    - x = extract (untar)
    - v = verbose
    - f = specify what file
    - example file is called Imagination
    - C = specifies destination to untar to (otherwise it will happen in the folder it is in)

Backing up your data

You should tar and back up all of your data from engram onto blanca. This is important in case a drive fails.

To do this, tar your data on engram. Then rsync to this folder on blanca:

- /archive/wagerlab/

- **Our Labdata filepath:**

- /work/ics/data/projects/wagerlab/labdata

- **Running MATLAB jobs on Blanca**

INTERACTIVE JOBS

1. Open up terminal, type:

- matlab

This will launch the matlab environment, and you can mess around with things as normal.

- **BACKGROUND JOBS**

1. How to run matlab scripts from the terminal:
    1. You first make a shell script that will call up matlab and run your matlab script. This is how you configure this script:
    If you're running through the terminal you will want to suppress the start up screen, the GUI, and the production of figures. This will save you time and computational resources, so it will make it easier for your stuff to run and your job will less likely be aborted or kicked off the node for being a memory hog.

- matlab -nosplash -nodesktop -noFigureWindows -r SCRIPTNAME
    In English:
    - -nosplash = No start up screen
    - -nodesktop = not MATLAB environment
    - -noFigureWindows = no pop ups of graphs, etc, but they should be saved
        - **NOTE**: -noFigureWindows is not currently working on unix....
            - it works fine on Blanca...
    - **NOTE**:
        - -nojvm HAS BEEN DEPRICATED

another example, where you can talk to matlab in the way you are used to by using <<EOF

- #!/bin/bash
- echo "starting..."
- matlab -nosplash -nodesktop <<EOF
- cd /your/path/to/
- x=1+1
- scriptname
- exit
- EOF

Here we do x=1+1 to demonstrate that you can use the matlab command line in the EOF environment

Put your script or function in there

Note: thanks to Hedwig for this!!

For more help with using MATLAB in bash:

http://www.mathworks.com/help/matlab/ref/matlabunix.html?refresh=true

1. Adding paths
    1. Blanca isn't going to commit your paths to memory, so make a script with all the paths you need and run it before each job. For example, you'll need:

- addpath(genpath('work/ics/data/projects/wagerlab/Respository'))

1. Now you want to send this job to Blanca. Say you saved this bash script as run_mymatlabjob.sh In terminal:

- qsub -q blanca-ics run_mymatlabjob.sh

- It will spit out the number of your job.

- To check on the status of your job:
- qstat -u [USERNAME]

- **On Walltime**

Walltime is the amount of time you can run a job on the cluster.

- The default is 24 hours
- Max is 7 days aka 168 hours
- you must specify your walltime if your jobs takes a long ass time to run, or it will stop early on you without warning

- #PBS -l nodes=1:ppn=1,walltime=168:00:00

Q: can insert that into a bash script? anywhere? just before the script call? or this has to go into qsub?

aren't hashtags comments in bash?

A: You can either specify the PBS directive in your script (as with "#PBS -m ae") OR include it in the qsub command.

Blanca Problems

Common oddities and how to solve them.

Q: I'm getting a weird 'Authenticat' applet popping up, wanting a password for root in order to download some package. I didn't try to download anything and it won't accept my password anyway. Make it go away?

A: You can disable this popup window by continuing your login, then running

- gnome-session-properties

at the command line and un-checking "PackageKit Update Applet" in the resulting window. Then next time you start a vnc server the message should not appear.

Q: I want to write a new script in emacs but the buffer is read-only.

A: Toggle read-only permissions with CNTRL-c CNTRL-q. Don't forget to save with CNTRL-x CNTRL-s

Parallel Jobs on Blanca

Motivation: reducing processing time by submitting jobs in parallel, e.g. for each subject

There are several possibilities to do that, here is one option sing bash scripts:

1. have your matlab function /script ready working on blanca

This kind of loop will use a lot of resources at once - one core and lots of mem per job/subject. it may be more fair to bundle subjects and have each job run several subjects serially. Right now I only have a workaround where I hardcode the number of subjects per job and have all subjects for one job in a row in subs.txt. I found that first PBS command is not necessary.
According to RC 4GB per core should not be exceeded in order to not crash a node. setting the SPM.stats.maxmem=2^33 allows for 8GB of RAM and the job may actually use more than 10GB on blanca. You want to then request 3-4 cores for your job to not use up the memory of other people.

- #!/bin/bash
- while read sub # this loops through the variable sub (input for matlab function)
- do
- #PBS -l nodes=1:ppn=1,walltime=24:00:00 # set the walltime, here 24hrs
- JOB=`qsub -q blanca-ics -m abe -N Big_run - <<EOJ # outer job description
- #!/bin/bash
- #PBS -l nodes=1:ppn=1,walltime=24:00:00 # set the walltime, here 24hrs
- #PBS N ANSpredict # Job submission name
- #PBS o ANSpredict.out # Output file name
- #PBS -A <account> ! !# Allocation
- #PBS -m bea # Send Email on beginning completion and abortion
- #PBS M <hedwig.eisenbarth@colorado.edu> # Email address
- cd /data/projects/wagerlab/labdata/projects/ANSpredict/Scripts
- matlab << M_PROG
- run_run_ans_predict(${sub}); %my matlab function
- exit
- M_PROG
- EOJ
- `
- echo "JobID = ${JOB} for parameters ${sub} submitted"
- done <subs.txt
- exit
-

for slurm:

- while read sub
- do
- #SBATCH -N 1
- #SBATCH --time=00:10:00
- echo "got to line before JOB"
- #SBATCH --qos=blanca-ics
- #SBATCH -J
- JOB=`sbatch <<EOF
- #!/bin/bash
- echo "got into the bash part"

- #SBATCH -N 1
- #SBATCH -c 2
- #SBATCH --time=48:00:00
- #SBATCH --qos=blanca-ics
- #SBATCH -J ANSpred_run_rolling
- #SBATCH --output=ANSpred_run_%j.out
-
- module load matlab/matlab-2014a
- cd /data/projects/wagerlab/labdata/projects/ANSpredict/Scripts
- matlab << M_PROG
- run_run_ans_predict_ebANSzBRz(${sub});
- exit
- M_PROG
- EOF
- `
- echo "JobID = ${JOB} for parameters ${sub} submitted"
- done <subs.txt
- exit

3. you start the just by submitting the bashscript:

- >./bashscriptname.sh

4. you can check for the status of your jobs using qstat:

- > qstat -u yourusername
- > squeue -u yourusername

and with this command you can check on how much memory you use:

- > qstat -f JOBID | grep used.mem

if you use more then 4GB, use more processors:

- #PBS -l nodes=1:ppn=2,walltime=24:00:00 # set ppn to 2 for 2 processors (max. 8GB total)

More comments on submitting jobs on blanca:

- use -vDISPLAY=blogin01:$PORT.0 if you use an interactive job (-I), this seems to speed up the display

---

FROM THE SCANNER TO YOUR OFFICE

**** NOTE: if you use MRN's auto-analysis, it does all of the below for you! ****

Motivation: to understand the components of the data you collected and how to prepare them for preprocessing

- **Scripts & Files You Will Need For This Section:**

  - dicomsdicomsdicoms.sh
  - dcmdump.sh
    - called by dicomsdicomsdicoms
  - convert_and_distribute.sh
    - must be changed for each study
  - run_convert_and_distribute.sh
    - wrapper or convert & distribute, looks for the subjects that haven't been run & operates on them
  - accession_id_corrections.txt

- **Steps**

  1. Locate your data: As of 2015, your data will be on Blanca.
     1. **ON DREAM**
        1. Data collected at CINC is stored in Dream:

- /dreamio/archive/human/dicom/triotim/twager
  Note: Everyone has *read* access no one has *write* access.
  Confused? Read More about how to log onto Dream
  1. **ON BLANCA**
     1. Data stored in: /work/ics/data/archive/human/dicom/triotim/twager
     2. In terminal type:

- cd /work/ics/data/archive/human/dicom/triotim/twager
- ls

  Now you should see a folder with your project name.
  1. Check here to ensure proper naming and storage of your raw images, but you do not need to pull data from here, the dicomsdicomsdicoms scripts does that for you
  2. **ON COINS**
  3. Alternatively, you can pull your data from COINS (COllaborative Informatics and Neuroimaging Suite), a database Vince & colleagues wrote up and now commercially distribute.
     What does COINS do?
     - Keeps track of recruitment, raw copies
     - live support
     - neuroinformatics database & server
     - can be used to acquire data online (qualtrics)
     - occasion ID

  ----------relevant-bash-tips----------

- pushd / popd
  maintains current location, while going into another new directory temporarily, until you pop out of it
- ---
  missing a mv files line?

1. Organize Your Subjects

   Do you have access to the CANLab Subject Data Base?
   canlab.colorado.edu/db
   If no:
   - sign up at that link
   - remind **Yoni** to confirm your new account

   Why?
   - This contains study questionnaires (however many have switched to qualtrics).
   - This is useful to track subjects
   - Here you can register subjects with basic demographic info & subj id for that person & occasion id for the day they came in (you can get more occasion IDs for every visit)

   go to: **/dreamio3/projects/wagerlab/labdata/current**
   You need to link URSI numbers (from MICIS [Medical Imaging Computer Information System] in the COINS system) with internal numbers. Do this by accessing **accession_id_corrections.txt**
   - type:
     - **more** accession_id_corrections.txt

     where:
     S=subject; number after is subj id #
     OC=occasion number; number after is the occasion # you get from the CANLab database

   Note: This text file is manually updated & called upon by dicomsdicomsdicoms.sh
   You need access to the fMRI data collection log Google Doc for CANLab
   - this log includes URSI number and IDs and basic infos
   - can use for URSI to ID conversion

   ----------relevant-bash-tips----------
   How many subject folders are in your directory? Let bash count:

- ls -l | wc -l

---

needs more explaining here, what are the acronyms?

1. Copy, rename, and move the data from the scanner

   Use this script from **.../labdata/current**
   dicomsdicomsdicoms.sh
   Example Usage:
   ./dicomsdicomsdicoms.dh bmrk5 -u -b bmrk5
   Comment: if you want to create a merged nifti file without copying all dicoms, you can change the dcm2nii command to be "dcm2nii -o (directory you want to output to)"
   This script [created by Luka] pulls data from the archives in Dream, changes the format of the names to conform with CANLab style, and creates the folders it needs to go into. The options used above all you to identify the source & target, and to confirm the nameage (by saying "Y" for yes, interactively).

   Want to know more? Type:

- dicomsdiscomsdicoms.sh --help

   You can also use this script to print out all the dicom files you have for a specific study. You might want to do this so you can see if you need to correct some of the accession numbers.
   Common Errors:
   NOT FOUND : means there isn't an accession number
   edit/watch dicomsdicomsdicoms.sh : emacs
   ----------relevant-bash-tips----------
   Creating a dot profile on Dream or Blanca
   ./ → hidden by standard ls

DATA PREPARATION

- **CONVERT DATA**

Motivation: We analyze Nifti but collect in Dicom format. You must convert appropriately.

- **Scripts & Files You Will Need For This Section**
  - convert_and_distribute.sh
  - disdaq.sh
- **Steps**

1. Convert dicoms to nifti

   Use this script from **.../labdata/current**
   You will have to move the script to that folder. A copy of it is in **CANLabRepos/trunk/Misc/Dream-specific** also it can be found in ruzic/scripts or in other experiment folders
   on blanca, this script is at **/data/projects/wagerlab/git/Repository/CanlabPrivate/Misc/Dream-specific**
   on blanca, this script is at **/data/projects/wagerlab/git/Repository/CanlabPrivate/Misc/Dream-specific**

- convert_and_distribute.sh
  - this is **study specific** and must be modified for your study. To do this, type in terminal:

- emacs convert_and_distribute.sh
  - you should change:
    - TRIM = 0 [number of scans to discard]
      - NOTE: Siemens automatically throws out the ramp up scans so you likely do not have to trim anything (set to 0), but if you are using data collected elsewhere, check with them if this happens. If it doesn't you will likely need to throw out the first 10 seconds of any functional run.
    - FUNC= "NAME"
      - specify some string (it will automatically append a wildcard so it does not need to be too specific) that identifies the name of your functional run folders
    - STRUCT= "NAME"
      - specify some string (also auto appends wildcare) to identify your anatomical folder
  - NOTE: You might need to change the permissions on this bash script. To do so, in terminal type:

- chmod 775 convert_and_distribute.sh

- You can *ONLY* run convert_and_distribute on Blanca. Not engram.
- Before you can run it, you must update your path. You should update this in your bash_profile as well. In terminal type:

- PATH=/data/projects/wagerlab/labdata/current/scripts:${PATH}

- For more information on Blanca configurations, visit this page of the wiki
- Now run it by giving it a wildcard for the name of your data folders:

- ./convert_and_distribute.sh studyname*

- If you have issues where it is just sitting in queue, you should contact rc_help and Zeb.

1. IF YOU ARE RUNNING CONVERT_AND_DISTRIBUTE YOU DO NOT NEED TO DO THIS STEP: Remove the initial scans where the scanner is ramping up

   NOTE: Siemens automatically throws out the ramp up scans so you likely do not have to trim anything (set to 0)
   Use this script from **.../labdata/current**

- Convert_and_distribute automatically calls disdaqs, you wil not need to do this if you use that and/or if you do not need to discard scans.

- disdaq.sh
- qstat -u username

- see what you are running on dream

- dreamon -w
  - meta program written by luka to check all the activity on dream, you need to have the banichlab settings included in your profile

- **CHECK THE SPM DEFAULTS**

Tor has preferences for the SPM defaults which will influence preprocessing & analysis, so update the **spm_defaults.m** file accordingly: **Remember not to edit the file directly -- see the documentation at the top of the file.**

In MATLAB:

- edit spm_defaults

- defaults.stats.maxmem = 2^35; %modified from 2^26
- defaults.mask.thresh = -Inf; %modified from 0.8
- defaults.stats.fmri.hpf = 180; %modified from 128
  - this is task dependent; 180 is common for pain tasks
  - rule of thumb is 4x the length of your task
- defaults.stats.fmri.cvi = 'None'; %modified from 'AR(1)'
  - Tor recommends turning autocorrelation off, because this algorithm pools across the whole brain
  - If you are performing a group analysis, the autocorrelation problem is not as concerning
  - If you are concerned with autocorrelation "pre-whitening" methods are preferred

- FILE STRUCTURE

File structure is hard-coded in much of the CANLab software. Please follow the following directory structure:

more examples:

- /Volumes/engram/labdata/current/[studyname]/Imaging/Analysis
- /Volumes/engram/labdata/current/[studyname]/Imaging/[SUBJ]/

Note: You will likely have to create structural & functional folders in each subject folder (preprocessing scripts will look for them but not create them). You can write a bash script for this (can ask Luka for help), use automator, or do it manually as you collect data.

more information: http://wagerlab.colorado.edu/wiki/doku.php/ku_fmri_course_interactive_session

PREPROCESSING STANDARD SEQUENCES

RUNNING CANLAB PREPROC

Motivation: Prior to analysis, fMRI data undergoes a series of preprocessing steps aimed at identifying and removing artifacts and validating model assumptions (Lindquist, coursera).

Goals:

- to minimize the influence of data acquisition and physiological artifacts

- to check statistical assumptions and transform the data to meet assumptions
- to standardize the locations of brain regions across subjects to achieve validity and sensitivity in group analysis

- **Scripts & Files You Will Need For This Section**

- canlab_preproc_2012_batch.m
- run_canlab_preproc_2012.m
  - wrapper written by Luka that allows you to more flexibly determine the preprocessing streamline (particularly useful for batch mode).
- both will call upon:
  - canlab_preproc_2012 (Note: help contains useful documentation)
  - preproc_part1
  - preproc_part2

- **Steps**

1- Edit run_canlab_preproc_2012 or create your own batch script for preproc. In this script you will need to specify:

- the base directory of your project

- basedir='/Volumes/engram/labdata/current/PROJECTNAME';
- cd(basedir)
  - note: you might need to cd to the Imaging folder (sometimes there are path errors)
  - the TR in seconds

- tr=2;
  - the type of acquisition sequence (crucial for slice timing correction). options for this are built in, and are as follows:

- acq='interleaved_BU' %interleaved, bottom-up (default)
- acq='interleaved_TD' %interleaved, top-down
- acq='ascending' %sequential, bottom to top
- acq='descending' %sequential, top to bottom
- acq='interleaved MT' %interleaved, middle to top
  - a wildcard for subject folder names, commonly denoted as: subjwc

- subjwc=filesnames(fullfile('IE*'));

Put it all together:

- **canlab_preproc_2012**(basedir, runFOLDERname_wildcard, BOLDimageNAME_wildcard, TR, STRUCTURALimageNAME_wildcard, [options])

see

- **help** canlab_preproc_2012

for more optional arguments (mainly skipping certain parts of preproc or skipping the creation of certain outputs).

Here's an example script:

- basedir='/Volumes/engram/labdata/current/PROJECTNAME';
- cd(basedir)
- tr=2;
- acq='interleaved_BU'
- subjwc=filesnames(fullfile('IE*'));
- for s=1:numel(subjwc)
- canlab_preproc_2012_batch(basedir,'r*','*.nii','*mprage*.nii,tr,acq,0,'subjwc',[subjwc{s}'*'],'wh_subjs',1);
- end

Note: There's a bug in the script when it tries to move the implicit mask images. It will give you an error (but not quit) saying it failed to move the file. You can ignore this.

However, we should fix this.

Tip: close all & clear all before and after running this batch operation

2- User Input

This preprocessing script is interactive. You can't set it and forget it. Now we will discuss user input it takes.

- **A) Reorientation**

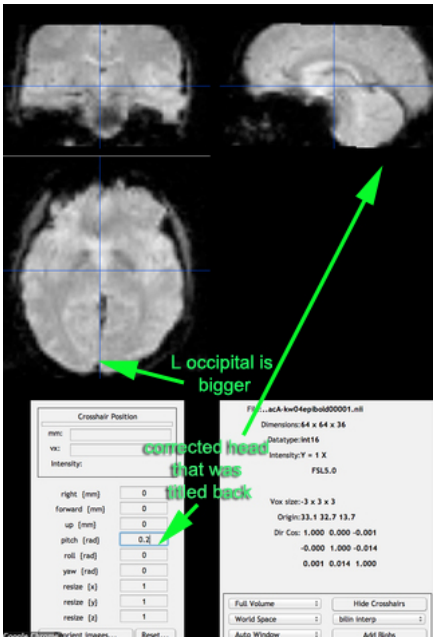
Motivation: Within subjects you want to make sure the data are in the proper alignment. Preproc will give you the option to reorient your images. If the head is tilted, or the brain is being displayed improperly (R/L swap), or something else is funky, now is the time to fix the orientation in the SPM GUI.

Steps:

Manually alter pitch/roll/yaw. May involve some trial and error. Best to try small numbers like: 0.1-0.5 first.

If your images are dark, change the image brightness/contrast. In the SPM GUI during the orientation interactive sessions click

SPM Figure --> colours --> effects --> brighten



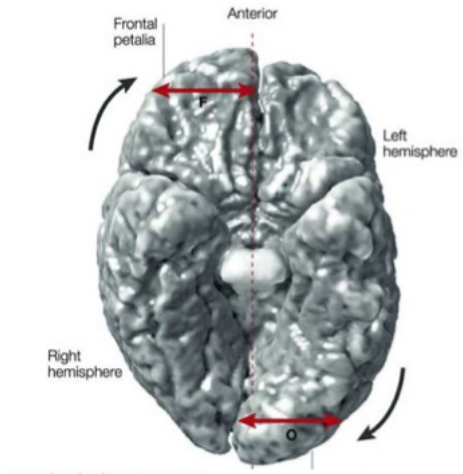
The image above is flipped left/right

SPM uses **neurological convention**: the **Left is on the Left**. Lots of data are collected in radiological convention (the Left is on the Right). SPM isn't going to tell you if it is flipping or not, however the X-coordinate will be negative if SPM thinks it is in radiological and is is therefore flipping it for your viewing convenience. You should mark with vitamin E the Right side of all subjects. If you haven't done this, in order to confirm that what you are analyzing as R side in SPM actually is R side in real life, consider natural asymmetries in the brain: The right hemisphere protrudes anteriorly beyond the left, and the left hemisphere extends posteriorly beyond the right (Toga & Thompson, 2003).

In **FSL** and BrainVoyager, Left is Right because they use radiological convention.

Of course, marking the right side with vitamin E before you scan your subject would be a safer test.

aka the **Left Occipital Lobe** will appear slightly larger than the right.



Mapping brain asymmetry
Arthur W. Toga & Paul M. Thompson
Nature Reviews Neuroscience 4, 37-48 (January 2003)
doi:10.1038/nrn1009

- **B) Setting the Origin**

Motivation: The anterior commissure (AC) is a small, white-matter tract at the end of the fornix. It is the canonical origin of the Tailarach-Tournoux atlas, e.g, the (0, 0, 0) point is the center of the AC.

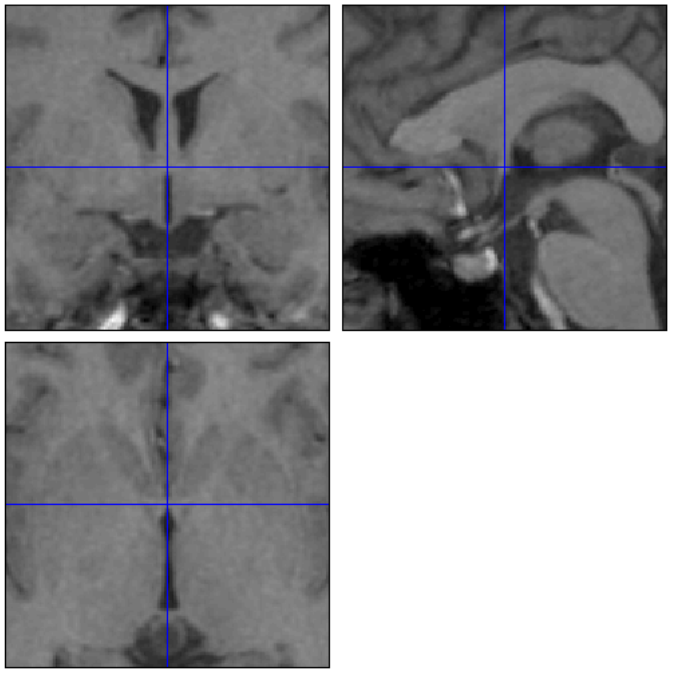
The origin of the MNI average brains is **NOT** at the anterior commissure, but a little more dorsal and posterior. This was done to improve *overall* matching between the MNI average brain and the original Tailarach and Tournoux brain, at the cost of having the origins a little different. Don't be surprised if, after warping, the origin is no longer at the AC.

- **Steps:**

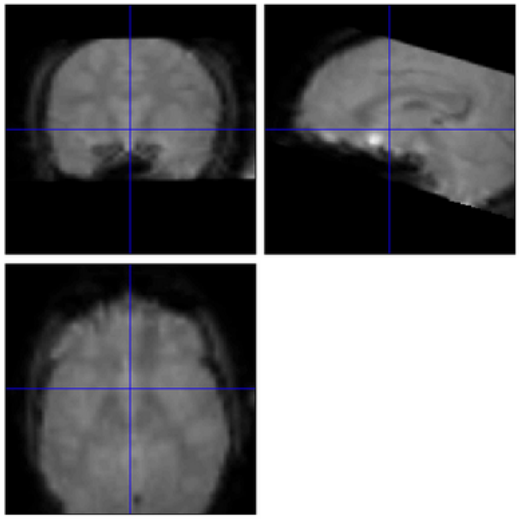
To set the origin, click in the figure until you have the cursor set as so. Then in the MATLAB command window, hit enter. You will have to do this once for the structural and once for functional (independent of the number of runs you have).

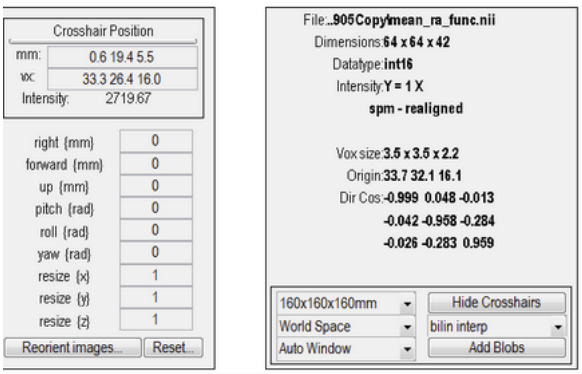
Structural:

The Anterior Commissure:
Structual:



in functional:





Don't stress too hard about setting the origin in those funky looking EPI images. Do your best. The algorithm will take care of the rest (quite robustly, in my opinion).

for more information: http://wagerlab.colorado.edu/wiki/doku.php/help/fmri_help/preprocessing/anterior_commissure

You will then be asked to confirm that the origin is appropriate for all images (so now all of the runs). Visually verify, and click enter in the command window.

- **C) Slice Timing Correction**

Motivation: adjust for variable acquisition time over slices.

This will happen without input from you, and be the most time consuming part. More on this when we consider quality control.

Note: Do not be alarmed when you see mean functional plots widely off the anatomical SPM template during preprocessing **BEFORE** coregistration. The plotting during that time is arbitrary. Do be alarmed if your coregistration plots are wildly off kilter.

- **D) Coregistration**

Motivation: Within subjects you the BOLD to align spatially on the anatomical.

You will be asked to confirm that coregistration looks okay. If something is drastically wrong, it is highly recommended that you deleted all the preproc files and start over with this subject. There are probably issues with original orientation or alignment (possibly corrupted header information).

UNDERSTANDING THE OUTPUT

Motivation: Preproc spits out a lot of images. These are very useful in determined the quality of your data and the quality of the preprocessing procedures. You want to verify things are in order. This section will break down what you need to look for.

- **Scripts & Files You Will Need For This Section**
  - folder: qc_images [found in subject's folder]
    - containing:
      - coregistration.png
  - folder: HTML_output [directory holding preprocessed output of each subject neatly organized into HTML files]

- **SPM terminology**

American	SPM
session	N/A
run	session
volume/image	scan

Letter	Preprocessing Step
a	slice-timing **a**cquisition correction
r	motion correction (**r**ealignment)
w	**w**arping/normalization
s	**s**moothing

this below needs cleaning up

Preprocessing QC checklist

- ☐ **Images**: Confirm that mean imgs from each run look like brains
- ☐ **Dropout**: Look at SNR across the brain. Expect it to be lower in certain areas (i.e. OFC), but know that areas with SNR < about 30 could be quite problematic
- ☐ **Orientation**: Check orientation -- confirm L is L (occipital will be larger on L)
- ☐ **Raw data quality:** Successive differences movie is a way of identifying gradient artifacts in raw data. It is a movie (*.tiff file) produced and saved (though it might not be saved if you skip the movie run in your preproc pipeline)
- ☐ **Head motion**: Check that head motion is minimal (one rule of thumb is < 2mm overall or < 1mm within run)
- ☐ **Coregistration**: For each subj, check structural-functional coreg visually (our preproc code makes you do this, too)
- ☐ **More image registration**: confirm that warped anatomical, warped mean func, and template are all in register (normalization.png?)
- ☐ **Normalization**: scnlab_norm_check to check normalization (takes template (avg152T1.nii), should be copied to a directory where you have permissions, warped T1s for each person, mean functional files, and subject folders as cell array)
- ☐ **Individual subject brain segmentation**: check that each subject's mask looks like expected

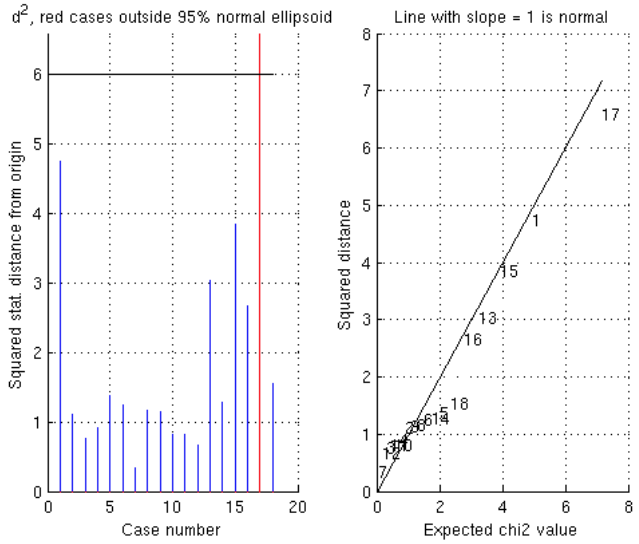
is successive differences being used to construct the nuisance covariates? have to check the code

- anything that is extreme here should be removed before continuing preproc cause come to slice timing correction, a spike will get blurred in time & the motion correction will think it is movement & shift all voxels up
- still a problem if you don't do slice timing corr

Using scnlab_norm check:

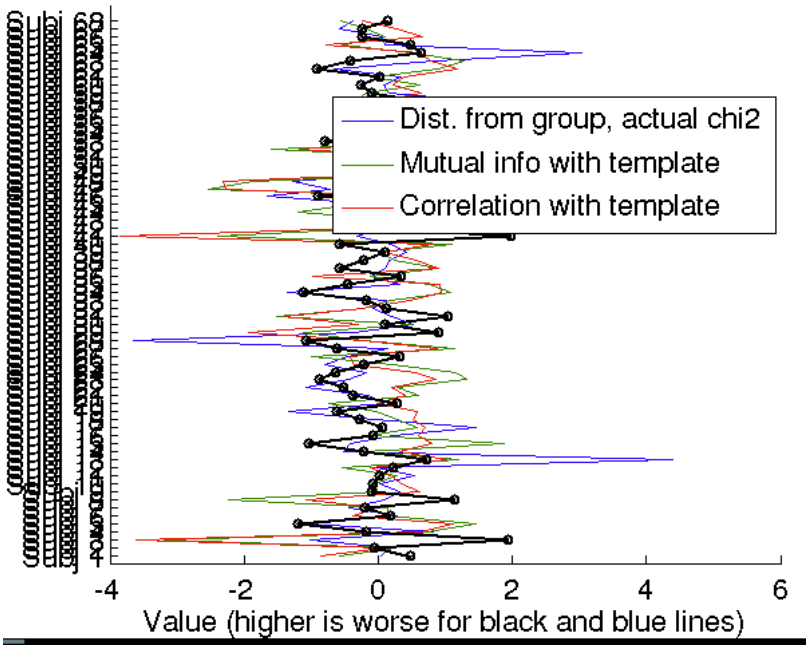
takes template (avg152T1.nii), should be copied to a directory where you have permissions, warped T1s for each person, mean functional files, and subject folders as cell array)

Example plot:



This plot shows the multivariate distance of the subjects (which images? warped normalization or functional files)? you entered in to the function, allows you diagnose outliers.

scnlab_norm_check also produces a plot like this for the structurals and for the functionals:



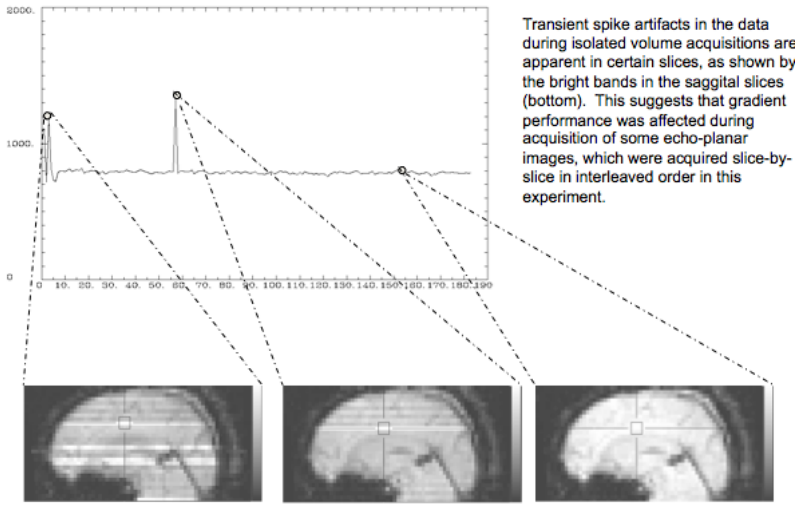
Here, high green and high red is good (i.e., more correlation and more mututal info). High blad is bad (further from group mean). Black is a composite measure, including all three measures of norming (distance, mutual info, corr.), such that higher black indicates worse norming. To the best of Yoni's knowledge, there is no cut-off values for any of these measures; rather, look at the worst subjects' normalization and see how bad it is. Images of the best and worst subjects are saved in the directory in which this function is run.

Visualization & Artifact Removal

Motivation: fMRI data often contain transient spike artifacts and slow drift over time. You should make sure you data are relatively clean. You can use exploratory techniques like principal components analysis (PCA) to look at spike-related artifacts. (Lindquist coursera).

Note: canlab_preproc can produce a movie that displays spikes (example still below), but it is currently disabled due to time & space constraints.

- option to change the threshold for removing crap volumes -- visually determine by watching RMSSD movies and then adjust SD (determine PRIOR to preproc)
- look at ART



Slice Time Correction

Motivation: Each slice is sampled at a slightly different time point. Slice time correction shifts each voxel's time series so that they all appear to have been sampled simultaneously. Want to know more? (Linquist coursera). The movie shows the Root Mean Square Successive Differences (RMSDs) in the data, not the raw data itself.

Options for Correction:

- **Temporal Interpolation**: use information from nearby time points to estimate the amplitude of the MR signal at the onset of the TR; using a linear, spline or sinc function.
- **Phase Shift:** slide the time course by applying a phase shift to the Fourier transform of the time course.
- **Neither**: incorporate the measurement time in your models later.

Note: We do linear temporal interpolation based on a specified slice (and your sequence).

Motion Correction

Motivation: Very small movements during an experiment can be a major source of error if not treated correctly. When analyzing the time series associated with a voxel, we assume that it depicts the same region of the brain at every time point. Head motion could render this assumption as invalid. You can correct this in several ways. We use __.

we use rigid body, correct?

Rigid Body Transformation: involves 6 variable parameters: 3 sets of translations, x, y, & z, and 3 sets of rotations, pitch, roll, & yaw (6 df total).

Alternatives are: translation, rotation, scaling, shearing.

We register to the first volume [in each run? or first volume of first run for all?]

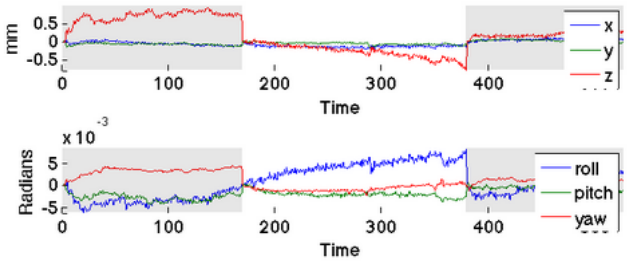
what do we use as the target image (first or mean?)

output example:

```
Building and saving nuisance regressors in PREPROC.Nuisance

ans =

     7
```

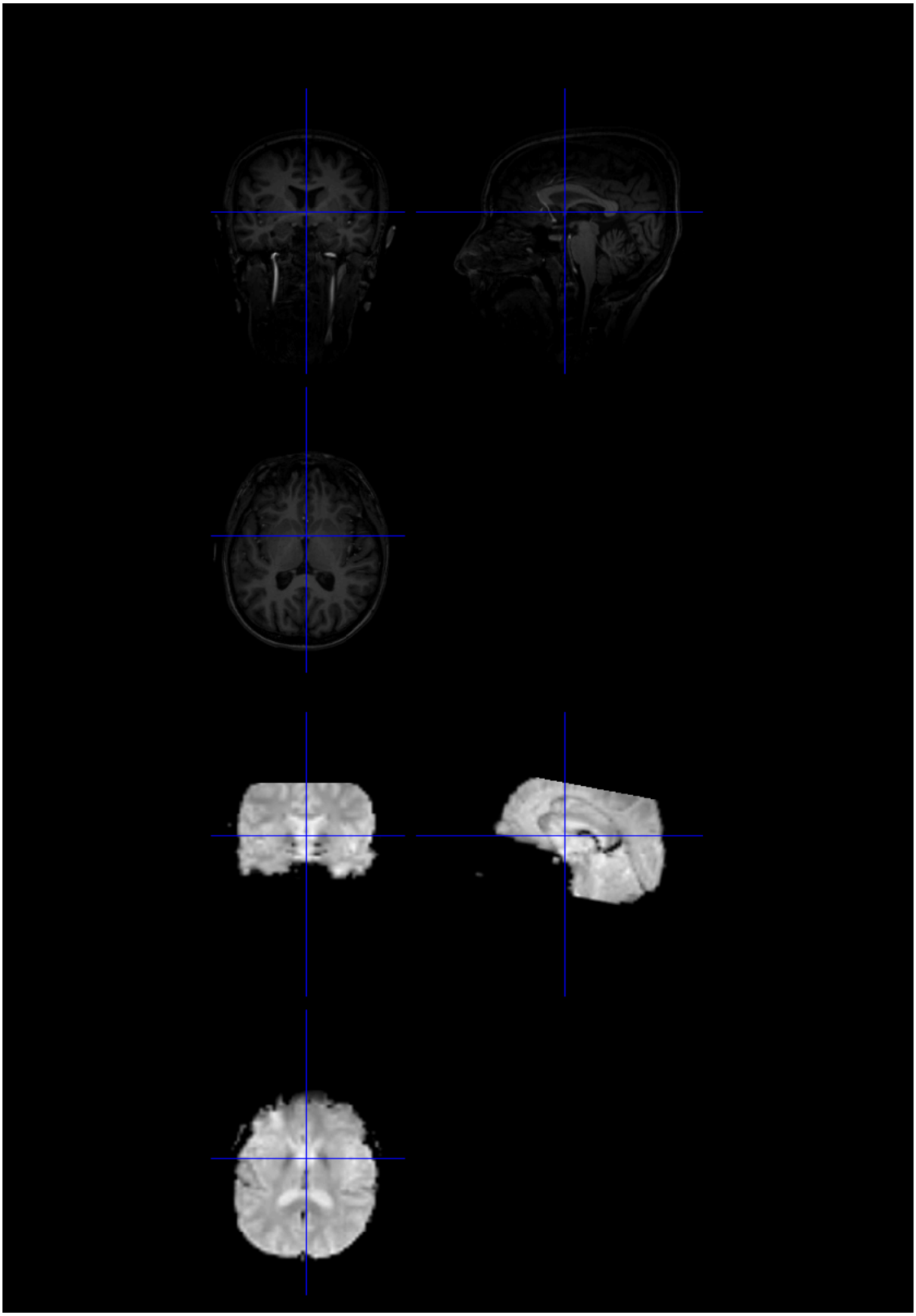


What to Look For: Small deviations from zero with no sharp spikes. Ideally motion parameters should not exceed the range of -2 to +2 mm (but the smaller the range around zero, the better).

Coregistration

Motivations: We want the structural image and fMRI image to be aligned as closely as possible. This allows us to visualize single-subject task activations overload on the individual's anatomical information. This simplifies later transformations of the fMRI images to a standard coordinate system, which will allow us to say where the activations are located.

Example:



what to look for: Try to visually verify (and you can move through the maps in space) that the functional is covering appropriate parts of the anatomical. For example, outer cortex is covered, and functional isn't off the scalp, etc.

put a better image in where anatomical isn't so dark

Normalization/Warping

Motivation: All brains are different. We can stretch, squeeze & warp each brain so that it fits on a standard coordinate system. This facilitates interpretation of results, makes results more generalizable, and allows results to be averaged across subjects. Unfortunately it can hurt spatial resolution and create errors as you warp.

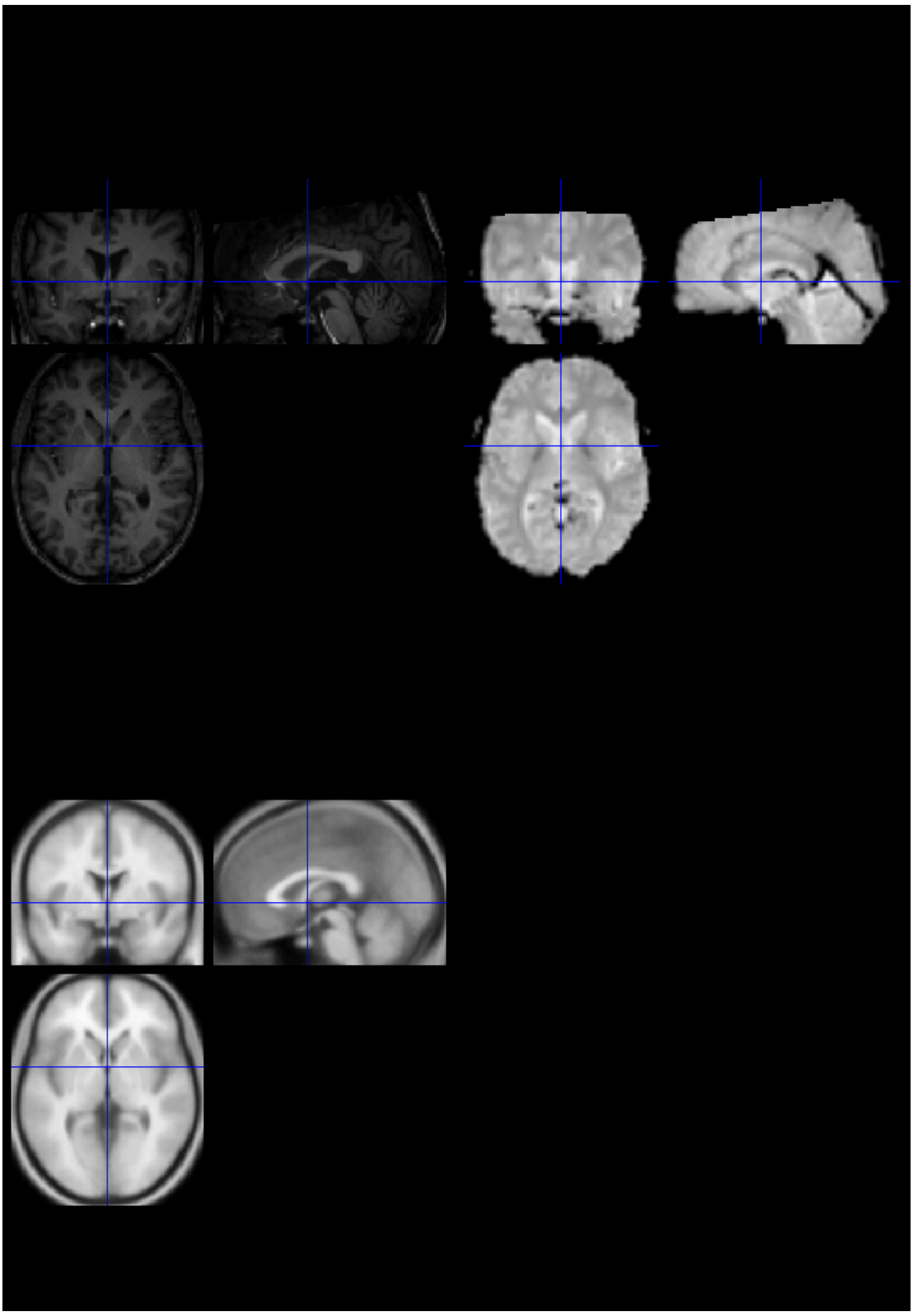
We normalize to the Montreal Neurological Institute (MNI) template, which is a combination of 152 MRI scans on normal, right-handed control subjects.

Options:

- landmarked based: like taliarch
- volume-based: overlap brain volume onto each other in an efficient manner typically using affine transformation or nonlinear transformations
- computational anatomy approached: diffeomorphic transformations
- surface-based methods: work on cortical surfaces

what do we use?

Example:



what to look for:

add what to look fort

Questions about your normalization? You can check it within subjects in your experiment:

scnlab_norm_check(template,wTIs,mean_func_files,subjs)

where:

- template: copy the avg spm template to your directory and put its name here avg152T1
- wTIs is a cell array of paths to structure wmprage.nii files
  - warped anatomical
- mean_func_files char array of paths to functional swr*.nii
  - mean functional
- subjs: array of the subject folder

outputs:

mutual info with template (subj to drop out)

malhalanobis dist : is it representative of your data set (find outliers within your subjects)

shows you the outlier images -- which ones might be the worst/best

in the preproc steps:

warp struct to the functional, warp struct to template, all that warp to all functional

canlab_glm_getinfo('conw') - returns a figure of your design matrix

Spatial Filtering/Smoothing

Motivation: Spatial smoothing can increase signal-to-noise ratio, validate distributional assumptions, and remove artifacts. Helps to overcome limitations in normalization by blurring any residual anatomical differences. It comes at a cost of image resolution.

We smooth at 6mm ??

check this

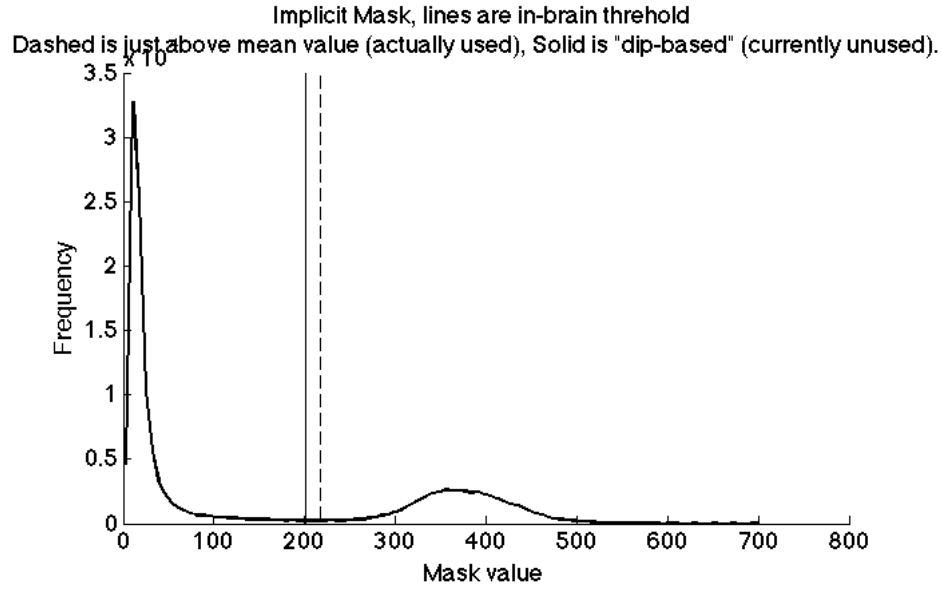
The Gaussian kernel is determined by the full width at half maximum (FWHM): the width of the kernel at 50% of its peak value. Usually measured in terms of mm (typically 4 - 12mm range).

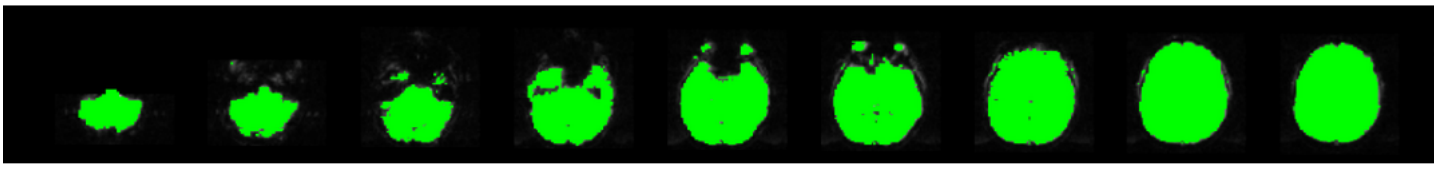
Matched filter theorem: filter that is matched to the signal will give optimum signal to noise.

Implicit Mask

description: a neon green mask superimposed over the subject's EPI BOLD data

example:





By default, SPM masks the images that contribute to an analysis at the Estimation stage. If a voxel is masked out because it fails to exceed an arbitrary analysis threshold (set to a default of 0.8 in SPM5), then its values are replaced with NaNs, and that voxel does not contribute to the final output. Incidentally, this masking contributes to the non-analysis of orbitofrontal and frontopolar regions as a consequence of signal dropout. (source). We removed this threshold completely (see edit spm_defaults.m to Tor's preferences above).

Shows voxels -- one close to zero are outside of brain.

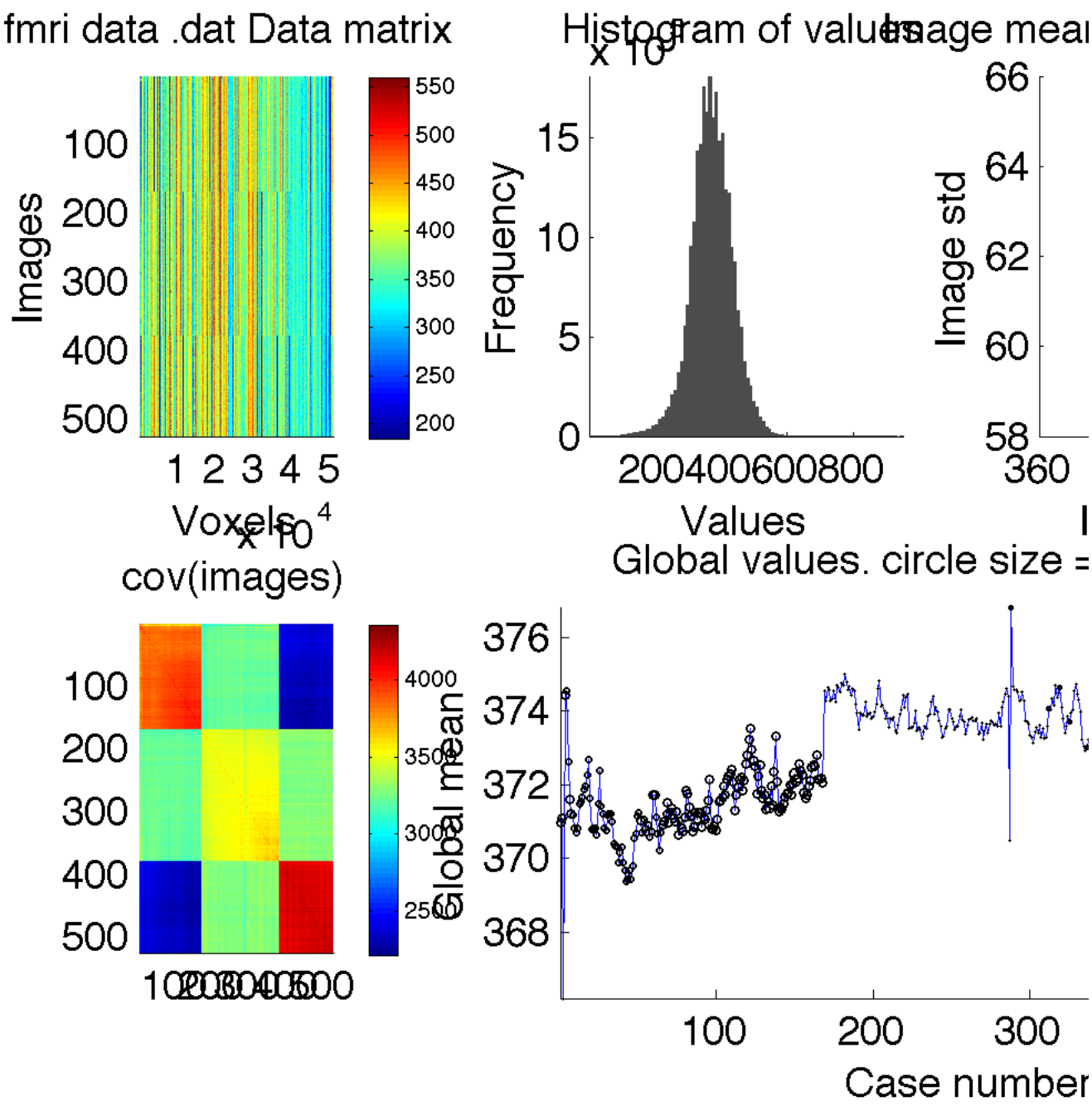
what's a good range?

what to look for: You can scan this for general functional coverage, but you don't have to pay too much attention to this.

fMRI Data Matrix

description:

example:



replace this image with an ideal version

what to look for:

Top L: all the data should be in the same reasonable range, especially if it is in brain data, aka, the full color scale should be used. If there are outliers or very extreme values, it might look all the same color.

Color scale: blue (200) to red (550) that's the range we get signal. You would not want scale to go to 1000.

Top M: look for normal distribution in the many hundreds or thousands, rather than low hundreds (this could depend on how the gain is set during acq). If all the image data are compressed in a very low range, it might be because there are extreme intensities in some parts of the acquired images and this could lead to loss of resolution.

If you're looking at a brain that has not been masked for out of the brain voxels, the distribution will look bimodal. That's ok, but it should be normally distributed for in-brain voxels.

T**op R**: as you turn up the gain the mean should go up & the STD should go up in proportion

>you may want to rescale the image (divide by image mean to reduce noise)

>spm (session by session scaling so that global mean is about 100 per analysis) & fsl do some version of scaling

Bottom L: look for a bright (red) diagonal band. You want strong correlations within a run and negative correlations across runs.

Bottom R: looks like more run to run noise than we want

>the larger the circle the more different the values in that image are across voxels (STD)

>ideally this would be same across runs, you'd want a flat tine

to add to preproc output: how many unique values are there in the functional image in brain? (in implicit mask)

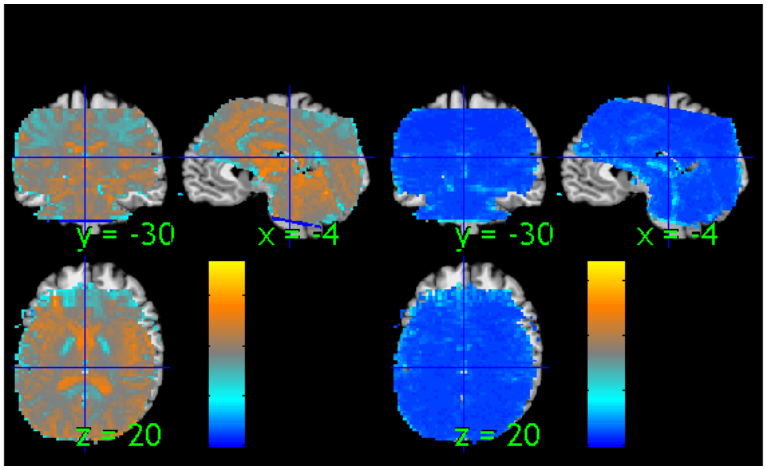
4096 unique values in dicoms

how many discrete levels is it possible to have in this data set? if it is low, this is a problem.

- **this plot?**

description:

example:



what to look for:

functional is on the brain, but does not have to be in register because this is pre-coregistration & pre-normalization

homogeneity across the brain

signal value reasonably high in cortex & regions you really want to analyze

bright line in brainstem --> likely movement

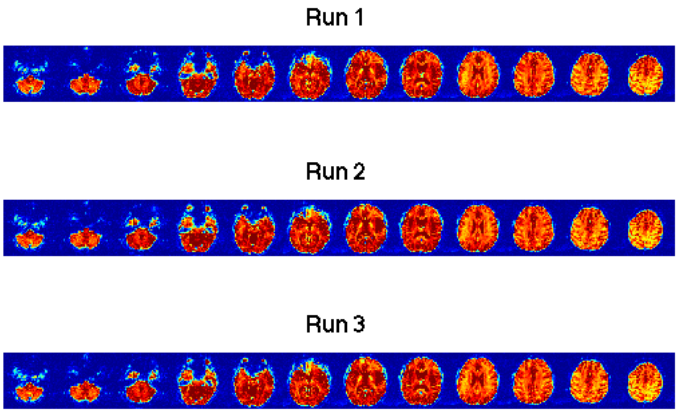
[insert the Outliers based on mahalanobis dist of global vals/satial RSSMD)

SNR maps: high, over 60s (drops to 30s you are losing signal)

- **mean image?**

description:

example:



what to look for: ideally they should look very similar

>scaling changed for each run?

Outlier Detection

description:

example:

what to look for:

drift in bottom slices but not top, we can see the first images of the run are different

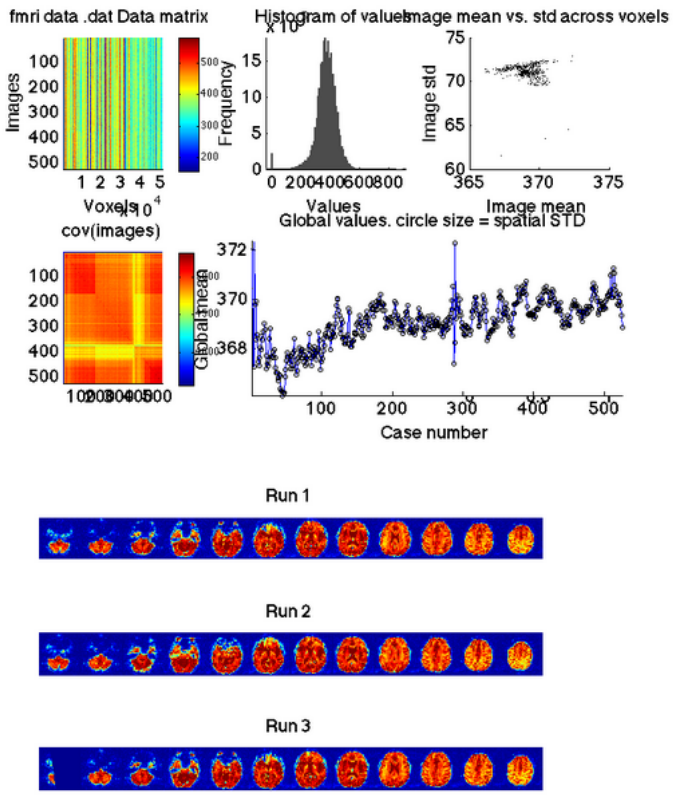
mahalanobis: how different is the slice across all the slices in all runs

>so if image is atypical inmean or STD it will have a high Mahalanobis dist

plots after slice timing & motion correction

description:

example:



what to look for:

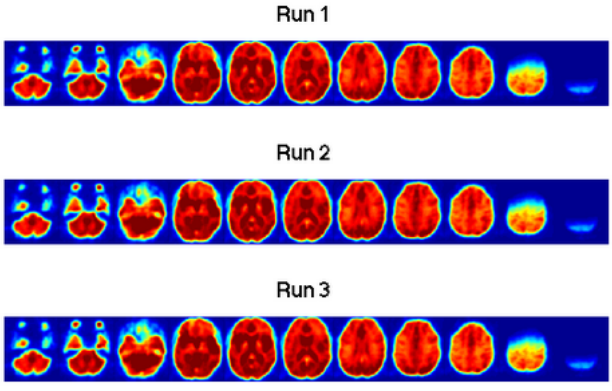
there's a section of data in run 3 that are substantially different from the rest (you can see in STD plot from previous & from the bright yellows in the cov plot here)

-

SNR Visualization

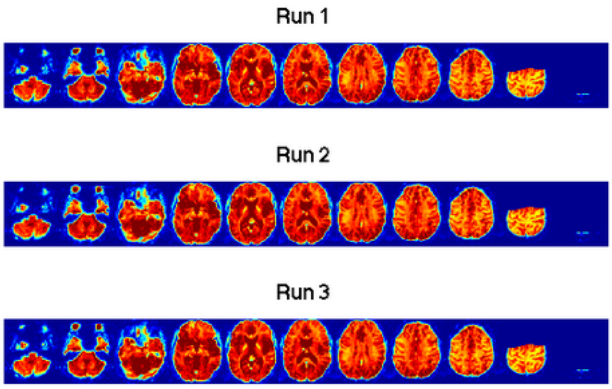
description: Look at the mean signal in your functional images as well as the standard deviation in order to have a sense of how robust your signal is to noise.

STANDARD DEVIATION



What to look for: "Assuming you don't have a major problem in your data, the first thing you'll notice in the std dev images is that brain edges will have higher std dev than central brain regions. This is perfectly normal and is the product of small, typical head movements. (These artifacts are usually well contained with a rigid body realignment in post-processing.) Furthermore, gray matter has higher std dev than white matter, because it's more active metabolically and so its physiologic "noise" level is higher. (Hence the correspondingly lower TSNR for gray matter.) So far so good. But the std dev images will also reveal severe acquisition imperfections to casual inspection, too (e.g. motion-related reconstruction artifacts as can plague GRAPPA)." From PractiCal fmri

MEAN SIGNAL



OBJECT-ORIENTED PREPROCESSING TOOLS

Motivation: x

What is object-oriented programming?

You can think of an object as a single data structure that contains data as well as functions.

Functions of objects are called **methods**.

Type

- methods fmri_data

to see the methods associated with that object

preprocess

rescale

>multiple diff method for session by session scaling, run2run diff get normalized out (so runs are comparable)

>outlier identification

this obv needs more

PREPROCESSING MULTIBAND DATA

32-CHANNEL AUTO-ANALYSIS PIPELINE

Motivation: Multiband images should not be preprocessed in the standard way: we need to correct for distortion, etc. Calhoun's group at MRN developed an auto-analysis pipeline. There is more information about implementation of this pipeline at CINC here.

Contact: Jill Fries, jfries@mrn.org

Note: When you begin a new MB study, email Jill to set up your study in their preprocessing pipeline.

For more information on multiband imaging, visit this in-depth guide.

MRN Preprocessing steps for Multiband fMRI scans:

1) Distortion correction

2) Motion correction

3) Spatial normalization via subject native T1 image

4) Smoothing 6mm

Final Files Key

d - distortion corrected

w - motion corrected/normalized

s - smoothed

File Structure

Your data preprocessed data can be found in:

- /data/auto_analysis/human/twager/icaps_200063/AUTO_ANALYSIS/triotim

In the subject's folder, you will find:

- all raw images converted from dicom to nii
- new folders:
  - analysis
    - fs_5.3
      - auto analysis of your structural data with freesurfer

        - vbm
            - auto analysis of the grey matter volume of your structural data
    - auto_log
        - text files containing log notes on the auto processing pipeline
    - preprocess
        - files which were used and produced to preprocess your functional data (like during dewarping and distortion correction)
- inside your functional runs:
    - [raw_run_name].nii.gz
        - the raw in nifti
    - d[raw_run_name].nii.gz
        - distortion corrected image
    - wd[raw_run_name].nii
        - motion corrected, normalized, and distortion corrected image
    - swd[raw_run_name].nii & swd[raw_run_name].mat
        - smoothed, motion corrected, normalized, and distortion corrected images in two formats: nii and a 4D mat file
    - the following afni files:
        - sanat_al_e2a_only_matt.aff12.1D
            - contains affine transformation that aligns subject EPI to anatomical (T1) image.
        - sepi_al_mat.aff12.1D
            - contains rigid body transformations from aligning EPI volumes to base volume
        - sepi_al_reg_mat.aff12.1D
            - combined output of the above two transformations that is combined with T1 to MNI transformation to take the EPI data to MNI space in a single interpolation step
        - sepi_vr_motion.1D
            - These are the 6 rigid body head motion estimates.
            - These are called lated on by mrn_to_canlab
        - Note: The pipeline does not identify any spikes.
- inside your anatomical scan:
    - awpy
        - preprocessed anatomical files **[add more detail]**

More MRN Code Notes

- **On their scripts**

Currently I have it setup this way in case any additional or individualized attention is needed.

This may change as we improve the scripts.

There is a log directory which has 3 files, one with all standard out from session, one with _TIME.log that shows time for each step, and one to show that the file that was submitted.

- /data/auto_analysis/scripts/triotim/twager/icaps_200063/32ch

- **On fMRI processing**

DICOM TO NII

auto5_convert_20dcm.sh/auto5_convert_t1.sh<http://auto5_convert_20dcm.sh/auto5_convert_t1.sh> - fsl dicom to nifti

T1 CHECK HEADER FOR PARAMS

-auto5_preproc_t1.sh - checks for prescan normalization in header.

PREPROCSS fMRI

- auto6_process_epi.sh
    - run for each task
- Checks for prescan normalization in header
- Checks header for dInPlaneRot and direction and sets flag for AP(APPA) or PA(PAAP) parameters used for distortion correction.
- Continues to auto6_preproc_distcorr.sh for distortion correction using the 2 distortion images and runs fsl topup to correct with appropriate epi_parameters_APPA_MB8.txt file or PAAP file.

AFNI SPATIAL NORMALIZATION TO T1_USING SBREF

-auto1nb_sp_norm_epi.sh

-auto1_normalize_epi.sh

See notes on Motion Correction and Spatial Normalization below

SMOOTHING

-SPM 5 Smooth Guassian smoothing 6mm

On VBM

-SPM Segmentation

-vbm smooth 10mm

-vbm_corr_value.txt - We compute a quick look correlation value, which does spatial correlation between canonical spm grey.nii and individual's unmodulated/smoothed/normalized image. I value is less than .94, we recommend users review it.

SETTINGS PERTAINING TO SEGEMENTATION

---

csprefs.segment.pattern = 't1w_32ch_mpr_08mm_*.nii'

% Grey matter

% Options are as follows:

% [0 0 0] means 'None'

% [0 0 1] means 'Native Space'

% [0 1 0] means 'Unmodulated Normalised'

% [1 0 0] means 'Modulated Normalised'

% [0 1 1] means 'Native + Unmodulated Normalised'

% [1 0 1] means 'Native + Modulated Normalised'

% [1 1 1] means 'Native + Modulated + Unmodulated'

% [1 1 0] means 'Modulated + Unmodulated Normalised'

csprefs.segment.output.GM<http://csprefs.segment.output.GM> = [1, 1, 1];

% White Matter

% Options are the same as grey matter

csprefs.segment.output.WM = [1, 1, 1];

% CSF

% Options are the same as grey matter

csprefs.segment.output.CSF = [1, 1, 1];

% Bias correction

% Options are as follows:

% 1 means 'Save Bias Corrected'

% 0 means 'Don''t Save Corrected'

csprefs.segment.output.biascor = 0;

% Clean up any partitions

% Options are as follows:

% 0 means 'Dont do cleanup'

% 1 means 'Light Clean'

% 2 means 'Thorough Clean'

csprefs.segment.output.cleanup = 0;

PROCESSING FOR DTI

-Bias correction if needed (Custom MATLAB)

-Distortion correction (FSL - TOPUP)

-Motion and Eddy current correction (FSL - DTI)

-Quality Control (Custom MATLAB)

-FA, RD, AD calculation (FSL - dtifit)

-Spatial norrnalization to a MNI template (FSL - FNIRT, TBSS)

On 32-Channel AutoAnalysis

- **Motion Correction and Spatial Normalization**

1)Subject EPI-SBref volume is aligned to subject T1 (affine)

2) All EPI volumes are realigned (affine) to EPI-SBref and estimated motion parameter file (sepi_vr_motion.1D) is written to the epi data directory of that scan. This file has one extra time point (first time point, which is EPI-SBref) .

3)normalize subject T1 to MNI reference T1 (nonlinear transform)

The transforms in 1,2 and 3 are combined together and applied to EPI (3dNWarpApply command) to write a normalized EPI volume in MNI space. No nifti file is written after just motion correction (realignment). This is done to avoid multiple interpolation similar to previous spm5 AA approach.

Setting your data up into the CANLab file structure

note this is still under construction

Use the bash script:

- mrn_to_canlab.sh

run it in the main directory where all of your M* subject folders are

this will move your preprocessed functional images into the proper file structure for analysis with CANLab tools.

Assessing your data quality

The autoanalysis pipeline saves us a lot of headaches! But, we still need to have an intimate look at our data, and make sure each subject's scans are of acceptable quality.

note this is still under construction

Use this matlab script

- mrn_canlab_data_qual_report.m

After your files are in the appropriate structure, run this matlab function with the subject directory, and it will publish html reports on the data quality of each scan using Tor's tools. These reports are akin to the data quality output discussed in the previous section.

This tool *will* assess:

- SNR

Congratulations, now you are ready to run some models.

Things to Know About Your Data And How To Find Them Out

Note: You may have to consult your MR tech about this stuff.

How many slices do you have?

find out:

in terminal:

- fslinfo $functional_data_file_here

  or

- fslval $functional_data_file_here dim3

Also, to check the acquisition of your images, you can use the script check_acquisition_pars_func.sh in labdata/current which outputs the type of acquisition you have.

You want to look at the value of dim3. This value can help you determine your microtime onset.

Hopefully it's odd. Then you should figure out what point in time the spatially middle slice was, and use that for your microtime onset. Example:

5 slices. In order of time, labeled by spatial order: 1 3 5 2 4.

The spatial middle (3) is acquired 2/5ths of the way through the volume acquisition time. Interpolated into 16ths, we get (2/5) * 16 = 6.4 ~ 6. So microtime onset 6.

I think this is right, but I'd first want to look at the SPM slice timing correction function and confirm that reference slice is determined by space and not time. If you have an even number of slices, I'd also have to look up how the reference slice is determined by rounding up or down.

FIRST LEVEL MODELS

MODEL QUALITY CHECKLIST

- ☐ good functions to run
  - ☐ scnlab_spm_design_check
  - ☐ canlab_glm_subject_levels by default runs the above on all your subjects by default
- ☐ do you have all the runs you expect?
- ☐ check colinearity among regressors
  - ☐ getvif (scnlab_spm_design_check does this, and Wani has a script that does it with lower overhead. lets fold those two together)
  - ☐ regressors of interest should not be correlated with regressors of no-interest. OK for regressors of no-interest to be correlated with each other.
- ☐ check mask - have you analyzed the voxels you think you analyzed?
- ☐ look at design matrix -- does it look as expected? run means? etc.
- ☐ check for outlier images: number of spikes, look at successive diffs movie

OVERVIEW

canlab_glm_* are a set of functions to handle the running of glm analyses of fmri data. They use SPM for analysis of individual subjects and Tor Wager's robust regression functions (robfit) for group level analysis.

Please read more about the canlab_glm_* functions before you begin: canlab_glm_README.txt

CONSTRUCTING DESIGN FILES

Motivation: To construct general linear models for single subjects. Proper construction of the design matrix is critical for effective use of the GLM.

- **Scripts & Files You Will Need For This Section**

- canlab_glm_example_DSGN_setup.txt
- canlab_glm_subject_levels.m
  - for info: canlab_glm_subject_levels('README')

- wrapper for the various other canlab_glm family members
- canlab_glm_dsgninfo.txt

- **Steps**

1- Find the template file **canlab_glm_example_DSGN_setup.txt** in the repos, open it and copy its contents into a matlab script which will be unique for your study. You will edit this script to suit your needs in the following steps.

2- Have ready your experiment timing files. You might have timing files unique to each subject (through Eprime logs, etc) or you might have one experiment timing & event structure which was applied to all subjects. Know your event timing. We need to construct a file that contains all the onset times in seconds. Seconds start counting at 0.

> **Note**: Useful tools for converting eprime files to matlab:
> http://www.3tmri.nl/software/88-eprime-to-spm-using-matlab
> This will allow you to export edat files into a csv that can be read with matlab or python.
> The easiest method I've found for reading these csvs so far is using pandas in python, which has a method called read_csv that extracts all the columns in the csv in arrays. Matlab works too, but I've found it easiest to get rid of all strings in the csv before importing it into matlab.

3- Now let's create your **stimulus onset file** in matlab, which will be called by the design file (which is then called when you run the GLM). This is done in SPM style. In this file you need to build and save matlab cells which will specify the:

- **duration** of events in seconds (if this is constant for all trials, you can specify it once at the start of your file)
- **name** of the trials of interest
- **onset** times of each trial of interest in seconds which being counting at 0.
  - Note: Even if you plan to concatenate runs, create separate model files for each functional run (unless you actually have concatenated them before this point, which likely you have not), i.e. If you have 3 runs, you'll have probably three onsets of 0 (instead of counting from 0 once from the first run to the end).

Example Code:

- duration{1}=[4];
- %4sec stimuli presentations constant throughout whole task
- %%
- %ACQ B conditions
- name{1}='acq_cs_min_early';
- onset{1}=[14
- 28
- 84
- 140];
- %save to worksapce as separate matfile
- fout=name{1};
- save(fout,'name','onset','duration');

You will repeat this exact structure for all events you want to model. Do not change anything but the duration, name, and onsets. The contrast conditions will later call these saved files.

Note: This method models events as box functions. This is recommended if stimuli are greater than 3 sec. If stimuli are very short, consider a stick function where duration = 0.

Another Example (from SPM)

- % I   Example SPM-style condition:
- name{1}   = 'heat';
- onset{1}   = [16.97 54.84 98.57 138.25 179.95];
- duration{1} = [10.75 11.25 11.00 11.75 11.50];

for more information see: spm fmri -> help -> spm.stats.fmri_spec -> pages 5-10

b) You should distribute the mat files containing onset times specific to subjects and runs into a modeling folder in the functional run folder it refers to. For example, a file structure like:

> /labdata/current/PROJECT/Imaging/IE202NC/Functional/Preprocessed/r1/spm_modeling

In order to do this, you can make a bash script to distribute your mat files from the MATLAB folder to the files respective subject-run folders. An example script is below. In this script one onset file structure was needed for all subjects in version B run whatever. Subjects in version B are in a list in the text document called in the first line. I ran my onset creation script in matlab, then ran this to create the modeling directory for that subject in its folder and stuck the appropriate mat files into it.

- for i in $(cat ./IESUBJvB.txt);do
- cd /Volumes/engram/labdata/current/Imagination/Imaging/$i*;
- mkdir Functional/Preprocessed/r2EXT/spm_modeling;
- cp ~/Documents/MATLAB/ext*.mat Functional/Preprocessed/r2EXT/spm_modeling;
- echo $i;done

4- Back to your design file. There are many options you will need to select according to your data acquisition. Let's go through them all.

a) Leave yourself a handy descriptive note about this model design:

- DSGN.metadata.notes='This GLM DSGN file is for study X; concatenated runs'

b) Denote an absolute file path to your subject level analyses folder (review CANLab file structure if needed):

- DSGN.modeldir ='/Volumes/engram/labdata/current/STUDYNAME/Imaging/Analyses/first_level/model1';

  Warning: if you do not specify a model directory, things can error as it assumes another directory to be the parent this can result in the permanent deletion of entire subj data folders.

c) Give it a list of all the subjects you are going to run by creating a cell array of subject directories, using absolute paths. See the help file for alternatives.

- %DSGN.subjects=filenames('/Volumes/engram/labdata/current/STUDYNAME/Imaging/SUBJ*','absolute');

d) At this point, if you have more than one BOLD run, you might want to consider if you should concatenate across runs. This means you will model them together, not separately. You will give onset times for the whole experiment in the onset files.

When to concatenate:

- If your runs are related, you most likely should (then you can contrast conditions between runs)
- If you have lots of spikes in your data that get modeled out, leaving you with fewer TRs in individual runs, you might want to concatenate to beef up the data in your model.
- In general, Tor suggests concatenation across runs is more statistically sounds
- **Note**: You should model the mean for each run. Script automatically adds intercept regressors to each but the first run (spm does that one automatically). If you want to add custom intercept regressors, there is an option for that, but that is rare.

When not to concatenate:

- If you had very unrelated runs: a stroop task and maybe a delay discounting task which were interested in different empirical questions, unrelated to each other.
- Any time you'd want separate GLMs for separate runs.

Now you will denote a cell array of strings that specify location of 4D functional data files within subject directories (may include wildcards). If you're concatenating, specify all of the runs you will call. Otherwise, just one.

- DSGN.funcnames{1} = {'Functional/Preprocessed/r1*/swraIE*.nii'};
- DSGN.funcnames{2} = {'Functional/Preprocessed/r2*/swraIE*.nii'};
- DSGN.funcnames{3} = {'Functional/Preprocessed/r4*/swraIE*.nii'};

e) Do you want the design file to error if a functional run is missing from some subject? If so, keep the following value at false. Otherwise, you can alter to true, but you should have a good reason for doing so. Being sloppy here can cost you.

- DSGN.allowmissingfunc = false;

f) Did you decide to concatenate? If so, this section is for you. This will create a cell array of arrays of runs to be concatenated. The numbers correspond to the functional runs identified in step 4-d. You can specify this serially [1:3] to concatenate all three runs together, or you can make things more complex like: {[1 2] [6:10]} which will put together runs 1 & 2, leave separate runs 3, 4, and 5, and combined runs 6 through 10.

Added by this script are:

- intercept regressors for each run (SPM does the first, this is not duplicated)
- linear trend regressors for each run
- regressors are merged in a block diagonal fashion (DSGN.multireg)
  - needs some clarification

- DSGN.concatenation = {[1:3]};
- % l - concatenate conditions (DSGN.conditions)
- % l (NOTE: it is assumed that concatenated runs each have the same conditions in same order)

ask luka about the concateneated condition order comment above

g) Now you need to set parameters specific to your data acquisition. You may need to the "Thing to Know About Your Data" section, if you are unsure how to determine what values you should input here. You should also consult your MR tech if you are confused.

Parameters to be set:

- TR in seconds

- DSGN.tr = 2;

- Value of high pass filter in seconds
  - Note: This was changed to Tor in SPM Defaults from 128 to 180 sec. Use 180 sec.

- DSGN.hpf = 180;

- Microtime resolution
  - This is the number of time bins SPM divides each TR into for calculating regressor values.
  - Value is defaulted to 16. It is recommended that you leave it at the default unless your TR is very long.
  - Commonly, this number corresponds to the number of slices acquired per TR.
  - determining microtime resolution:
    - know your acquisition sequence (interleaved, sequential, etc)
    - which slice is selected as the referent slice (default is the middle)
    - how long your TR is: the longer the TR the more microtime resolution you want
    - If you have 2 second TRs or less, a value of 16 is fine

- DSGN.fmri_t = 16;

- Microtime onset
  - This is the bin number, of however many you've specified in the microtime resolution, from which the regressor value is chosen to be entered into the model. Aka, you are denoting the reference slice. Selecting this slice will depend on how your data were acquired.
  - Example: If you set microtime resolution to 16 and microtime onset to 1, then each TR is divided into 16 time bins, and the value of each regressor in the first bin is entered into the model.
  - Commonly, if slice timing correction is performed during preprocessing, the microtime onset is set to the **middle** slice. This is because:
    - the middle slice (in time) is never more than TR/2 away from any other slice
    - if you set the microtime onset to the first slice, it is nearly twice that TR away from the last slice (source for microtime resolution and onset information & recommendations, in addition to Luka).
  - In the example below, an interleaved sequence was used and slice time correction was performed. 9 was chosen to correspond with slice time correction, and is roughly the middle slice.

- DSGN.fmri_t0 = 9;   %microtime onset

h) If you did not create a separate onset file as discussed in Step 1, you can input your conditions and onsets in the following cell array (one cell per session) of cell arrays (one cell per condition):

- DSGN.conditions

These condition files should be created and saved to each subject in DSGN.modelingfilesdir. All timing is in seconds (not in TRs).

i) Specify where the conditions, parametric modulators, and regressors for each subject should go:

- DSGN.modelingfilesdir = 'spm_modeling';

j) However you created your conditions and onset files, you will call the names of cell arrays containing the duration, onset times, and name information in the following code:

- c=0;
- c=c+1; DSGN.conditions{1}{c} = 'acq_cs_min_early';
- c=c+1; DSGN.conditions{1}{c} = 'acq_cs_min_late';
- c=c+1; DSGN.conditions{1}{c} = 'acq_cs_plus_early';

Note: the use of the value 'c' here is merely to facilitate editing, as it allows you to delete or add conditions without having to change the numbers assigned to the exisiting/remaining conditions.

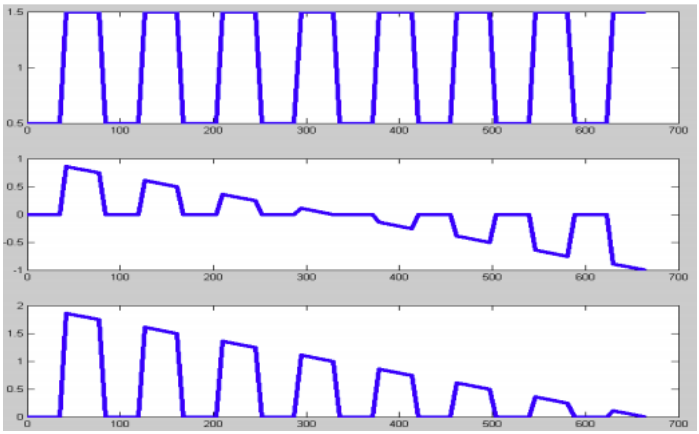
The names called in those cells have to correspond with the names given to the onset files. SPM might alter the names a bit after you run this, and you might get errors. If this happens, add wildcards to the end of the condition names here.

k) You can also specify parametric modulators in the same way.

- DSGN.pmods{1}{c} = {'pain_rating_H'};

PARAMETRIC AND TIME MODULATORS

Motivation: To understand when to use parametric or time modulation



Time / parametric modulations are used to relate the size of a response in a certain

condition throughout the experiment to a third variable. This third variable can be,

e.g., be a linear downward trend (e.g., when responses are expected to habituate -

this is an example of a time modulation), or reaction time measurements (when,

e.g., a shorter reaction time is expected to accompany a larger BOLD response -

this is an example of a parametric modulation). (source)

Parametric modulation allows for trial-specific variation in amplitude. Often a stimulus can be parametrically varied across repetitions, and it is thought that this may be reflected in the strength of neuronal response. For example, reaction times in a Stroop task might quicken as a subject receives the same trial type over and over again. You can include an additional regressor, a parametric modulator, to account for different neuronal responses within trials.

When to use parametric modulators:

- Add parametric modulators (pmods) if you want to add something like pain ratings, single-trial scored skin conductance responses, reaction time, etc
- You can add anything non-continuous & event-related.

When **not** to use parametric modulators:

- If you want to add something continuous like motion, heart rate, a subject's complete SCR file, etc, you can add it as a regressor in your glm (see nuisance regressors). But you can't add this as a pmod.

below needs cleaning up & clarification

do you want to do:
demeaning?
(to manually do this for each pmod, subtract the mean of each pmod from all values for the pmod. Required if you turned off orthgonalization as below)
orthogonalization ?
Another option is to make SPM not orthogonalize. This involves commenting out some lines of code in 2 SPM functions. In spm_fMRI_design.m and in spm_get_ons.m, there are calls to spm_orth. (from http://wagerlab.colorado.edu/wiki/doku.php/help/fmri_help/fmri_statistical_models/parametric_modulation?s[]=parametric&s[]=modulators)

(good idea to turn this off, SPM defaults to stepwise regression where it regresses the first pmod out and then regresses the effect of the other pmods on the residuals)

- **Scripts & Files You Will Need For This Section**
  - inside your design file, DSGN.pmods & DSGN.notimemod
- **Steps**

1. Do you want a time modulator? If you don't expect habituation, and don't want this, set the following option to TRUE.
   - DSGN.notimemod = false;
1. Making Parametric Modulators

```
%  DSGN.pmods  (DEFAULT: no pmods)                                          |
% |    cell array (one cell per session) of cell arrays (one cell per condition)       |
% |      of cell arrays (one cell per modulator) of MAT-file names                     |
% |    each MAT-file contains cell arrays of SPM pmod fields: name, param, poly.  |
% |      example mat contents:                                                         |
% |        name{1}  = 'rating';                                                |
% |        param{1} = [50 13 32 69 54 71 10 6];                                |
% |        poly{1}  = 1;                                                       |
% |    (note: these cell arrays of pmod fields will be converted into arrays of        |
% |          pmod structs for SPM (e.g., name{2} -> pmod(2).name))                     |
% |    location of MAT-files: DSGN.modelingfilesdir (see below)                |
% |    EX: DSGN.pmods{2}{4} = {'temp' 'rating'};                               |
% |      if the 2nd functional file is .../r2/swraPAINTASK.nii, the 4th condition |
% |        will get the pmods defined in:                                      |
% |        .../r2/DSGN.modelingfilesdir/temp.mat                               |
% |        .../r2/DSGN.modelingfilesdir/rating.mat
```

l) Do you want the glm to allow empty conditions? You can change this following option to true if you do. You may want to do this if you want to model errors for some subjects but not all subjects have errors. However, skipped conditions should not be included in contrasts (unless you balance the weights).

- DSGN.allowemptycond = false;

m) Create your nuisance regressors and add them to the model in this section. Here you want to put the name of an SPM style regressor file that can be found in your modeling files directory.

- DSGN.multireg = 'noise_model_1';

NOTE: You must give something to DSGN.multireg or your first level models with break with the current code.

MAKE SOME NOISE (MODELS)

Motivation: In order to better analyze BOLD signal, we should construct models that explain known causes of variance which are not of interest to your hypothesis. Functional MRI data typically exhibit significant autocorrelation caused by physiological noise, low frequency drift, and motion-related 'spin history' artifacts. These known sources of variance can be added to a noise model, which can be added to your first level GLM.

Note: If you have a continuous regressor of interest (not noise) don't put them into this model because it will be a pain to try and run contrasts on it later.

- **Scripts & Files You Will Need For This Section**
  - make_noise_model_1_2012.txt
- **Steps**

1. Find the text file in the repository make_noise_model_1_2012.txt Copy its contents into a matlab script specific for your study. This script will create your noise models (noise_model_1.mat) that will store a SPM-style R matrix containing:
   - spikes from scn_spike_id
     - number: variable per functional run
   - realignment parameters
     - number of parameters: 6
   - squared realignment parameters
     - number of parameters: 6
   - derivatives of realignment parameters
     - number of parameters: 6
   - squared derivatives of realignment parameters
     - number of parameters: 6

   These parameters around found in the Functional/Preprocessed/Nuisance_covariates_R.mat which are output after canlab_preproc_2012. They are in R{1} and R{2} unconcatenated, and redundant spikes are removed.

1. Pay attention to the top part where the comments indicate that the following bits of code should be fitted to your study. You will have to maybe change the output directory name (if not 'spm_modeling'), as well as the path to your imaging directory, etc.
2. Run it. If you catch an error like assignment A(I)=B right away, peep around, see if the problem is that maltab is having trouble calling fslval. Make sure you have fsl installed, and on path in your matlab. To fix this, add the full path to fslval.

   /usr/local/fsl/bin/fslval
   Alternatively, find out the path of fslval in terminal by typing:

- type -p fslval

CONVOLUTION OPTIONS

Motivation: Not all parts of the brain have the same HRF. To allow for different types of HRFs we use temporal basis functions. A linear combination of function can be used to account for delays and dispersions in the HRF. There are several options to be considered here.

- **Scripts & Files You Will Need For This Section**
  - inside of the DSGN file, DSGN.convolution
    - this provides more theoretical detail for creating this setting
- Options

There are two available convolution types:

1. Hemodynamic Response Function - 'hrf'
   1. Canonical HRF: the shape is fixed and only the amplitude is allowed to vary. This is not very flexible.
   2. This is the spm default.
   3. Reliable but not accurate.
      - ask luka if this is canonical hrf

1. Finite Impulse Response Function - 'fir'
   1. FIR basis set: contains one free parameter for every timepoint following the stimulation for every cognitive event type. More flexible.
   2. luka: FIR usually isn't usually a starting place, but you'd want windowlength to equal your tr, and order can be between 8 and 20 seconds, I guess. Maybe depends on expectations about the hrf and vifs and such.
   3. Really really accurate, but not reliable. It has low power and you risk overfitting the data.

- **Steps**

1. Inside your design file...

I) Convolution

- % I - DSGN.convolution (DEFAULT: hrf.derivs = [0 0]) I
- % I structure specifying the convolution to use for conditions I
- % I different fields required depending on convolution type I
- % I AVAILABLE TYPES (declared as string DSGN.convolution.type) I
- % I 'hrf' (Hemodynamic Response Function) I
- % I convolution.time (1/0 switches for time derivatives) I
- % I convolution.dispersion (1/0 switches for dispersion derivatives) I
- % I EX: DSGN.convolution.hrf.derivs = [1 0]; % HRF w/ time deriv I
- % I 'fir' (Finite Impulse Response function) I
- % I convolution.windowlength (post-stimulus window length (in seconds)) I
- % I convolution.order (number of basis functions) I
- % I convolution.keepduration (optional flag. DEFAULT: FALSE) I
- % I if FALSE, set all durations for all conditions to 0 I
- % I EX: DSGN.convolution.type = 'fir'; I
- % I DSGN.convolution.windowlength = 46; I
- % I DSGN.convolution.order = 23; I
- % I - DSGN.ar1 (DEFAULT: false) I
- % I if true, use autoregressive AR(1) to model serial correlations I
- % I (see SPM manual) %%tor turns this off in spm_defaults
- % I - DSGN.singletrials I
- % I a cell array (1 cell per session) of cell arrays (1 cell per condition) of I
- % I (corresponding to DSGN.conditions) of true/false values indicating I
- % I whether to convert specified condition to set of single trial conditions I
- % I (e.g., from 1 condition with n onsets to n conditions with 1 onset each) I
- % I - DSGN.singletrialsall (DEFAULT: false) I
- % I set DSGN.singletrials to true for all conditions I
- % I - DSGN.allowmissingcondfiles (DEFAULT: false) I
- % I if true, throw warning instead of error when no file(s) are found I
- % I corresponding to a MAT-file name/wildcard

SPECIFYING CONTRASTS

Motivation: X

- **Scripts & Files You Will Need For This Section**
  - canlab_spm_contrast_job
    - help function useful here
    - do not have to run directly, will be called by canlab_glm_subject_levels later
    - It does 6 things by default:
      1. Construct contrast vectors and names across all sessions and add to matlabbatch SPM job structure
      2. Scale contrast vectors according to the number of regressors/sessions included.
      3. Check the contrast vectors to make sure they sum to zero
      4. Save the matlabbatch job file in the SPM directory
      5. Runs the job, which adds contrasts to the SPM.xCon field and creates con*imgs, t*imgs
      6. Old (pre-existing) contrasts are removed by default!

- onset files with condition names already specified

- **Steps**

1. Decide what contrasts you want to run.
2. Inside the design file, you can specify them in the following fashion:

```
Here is a way to specify an average across one regressor or a set of them
 that share a common name:
 input_contrasts{1} = {  {positive effect name} };

 e.g., if you have two regressors called 'soc_friend_pre' and 'soc_friend_post',
 This contrast will specify the average across both regressors:
 input_contrasts{1} = {  {'soc_friend'} };

 This will do the same thing:
 input_contrasts{1} = {  {'soc_friend_pre' 'soc_friend_post'}   }

 You can use only some strings in the middle of regressor names.
 e.g., input_contrasts{1} = {  {'friend'} };

 This will specify a DIFFERENCE pre - post, because the first cell has one positive condition (pre)
 and the second cell has one negative condition (post):
 input_contrasts{2} = {  {'soc_friend_pre'} {'soc_friend_post'}   };

 ...Since we have entered it as input_contrasts{2}, it will be added as a
 second contrast after the first one.

 The contrast below specifies an interaction, % (Rej - Friend) x (Pre - Post)
 input_contrasts{3} = [{{'soc_rejector_pre' 'soc_friend_post'}} {{'soc_rejector_pre' 'soc_friend_post'}}]

 ...Since we have entered it as input_contrasts{3}, it will be added as a third contrast.

 'Custom' option example:
 This is an example to use your customized contrast (e.g., 1st contrast: soc_friend_pre = -3, soc_friend_post = -1,
 soc_rejector_pre = 1, soc_rejector_post = 3, 2nd contrast: soc_friend_pre = 1, soc_friend_post = 2, soc_rejector_pre =3).

  input_contrasts{1} = { {'soc_friend_pre' 'soc_friend_post' 'soc_rejector_pre' 'soc_friend_post'} };
  input_contrasts{2} = { {'soc_friend_pre' 'soc_friend_post' 'soc_rejector_pre'} };
  canlab_spm_contrast_job(modeldir, input_contrasts, 'custom', {[-3 -1 1 3] [1 2 3]});
```

However, in the design file, you replace input_contrast{} with DSGN.contrasts{c} like so:

- c=0;
- c=c+1; DSGN.contrasts{c} = {{'acq_cs_plus'}}; %CS+ all aq v baseline
- c=c+1; DSGN.contrasts{c} = {{'acq_cs_plus_early'} {'acq_cs_min_early'}}; %early CS+ v CS-
- c=c+1; DSGN.contrasts{c} = {{'acq_cs_plus_late'} {'acq_cs_min_late'}};

3.

scn_design_check

Variance Inflation Factor

Motivation: credit assignment issue (how do you distribute credit for success among the many decision that may have been involved in producing it?)

Run a regression with every regression predicting every other regressor, and if you have a high VIF, then they are highly correlated and you have some issues. For example, motion regressors could be highly correlated to task if you have a task that induces flinches, finger tapping, etc.

In the VIF graphs output after the glm, below 2 is preferable, between 2-5 can be okay, above 5 -- strong issues.

- **RUNNING THE MODEL**
- **Motivation**

  - Time to let your hard work churn out some results! Not done yet, but now you can look at results for each subject.

- **Scripts & Files You Will Need For This Section**

  - canlab_glm_subject_levels
    - Performs lower level GLM analysis with SPM:
      1. specifies model
      2. estimates model
      3. generates contrast images for model
      4. creates directory with named links to spmT and con maps
      5. publishes analyses with scn_spm_design_check

  - your design file

- **Steps**

luka: rules for modeling

- if you're going to contrast against baseline, make sure you model all your task

rules for contrasting

- make sure that relative weighting of conditions is consistent across subjects

rules for concatenating

- only concatenate when you have regressors that will describe data in different runs

UNDERSTANDING THE OUTPUT

- **Motivation**
  - Xxxx
- **Scripts & Files You Will Need For This Section**
  - xxx.xx
- **Steps**

error to discuss:

Saved Variance_Inflation.png in SPM directory

Warning! Different numbers of events of interest in each run.

If this is what you intended, OK, but will use max val and so will include some of no interest

Error using cat

Dimensions of matrices being concatenated are not consistent.

Error in scn_spm_choose_hpfilter (line 56)

X_filtered = filter_X(X_interest, cat(1, SPM.xX.K(:).X0));

Error in scn_spm_design_check (line 143)

scn_spm_choose_hpfilter(spm_results_dir, 'events_only');

Error in canlab_glm_publish_subject_levels (line 23)

scn_spm_design_check(sublevs{i}, 'events_only');

Error in evalmxdom>instrumentAndRun (line 89)

text = evalc(evalstr);

Error in evalmxdom (line 20)

[data,text,laste] = instrumentAndRun(file,cellBoundaries,imageDir,imagePrefix,options);

Error in publish (line 164)

dom = evalmxdom(file,dom,cellBoundaries,prefix,imageDir,outputDir,options);

Error in canlab_glm_publish>spmlower_report (line 134)

fout = publish('canlab_glm_publish_subject_levels.m', p);

Error in canlab_glm_publish (line 63)

spmlower_report(STARTINGDIR,varargin{i});

Error in canlab_glm_subject_levels (line 464)

canlab_glm_publish('s',DSGN.modeldir)

from canlab_glm_subjectlevels_publish.html

or you can run scn_design check solo to get these

how much of your task related variance you are potentionally cutting off with your current filter and what you want to use to get to 5% for your design

NOTE: some bug exists that won't output this if you have an uneven amount of conditions in each run -- something where Rows & Cols are unequal and the code errors.

stick in HP recommendation graph - jess

inset screenshot of VIF output

VIFs

looking at motion regressions

>it is okay if you have high VIFs just in motion regressors, as they might be highly correlated with each other

>You should be concerned if a regressor you want to interpret surpasses [2 5 or 10]

10 hard cut off

2 look into it

>now with luka's script there's a text file output with all the VIF

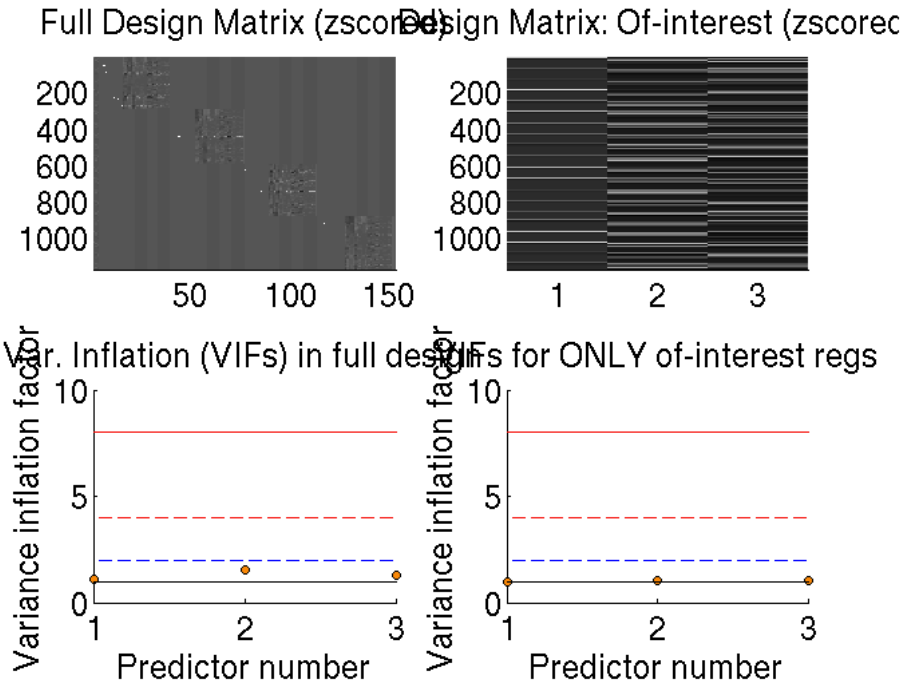
VIF is the correlation with all the other regressors - you won't know which one it is most correlated

You can look at the covar matrix to look pariwise between regressors

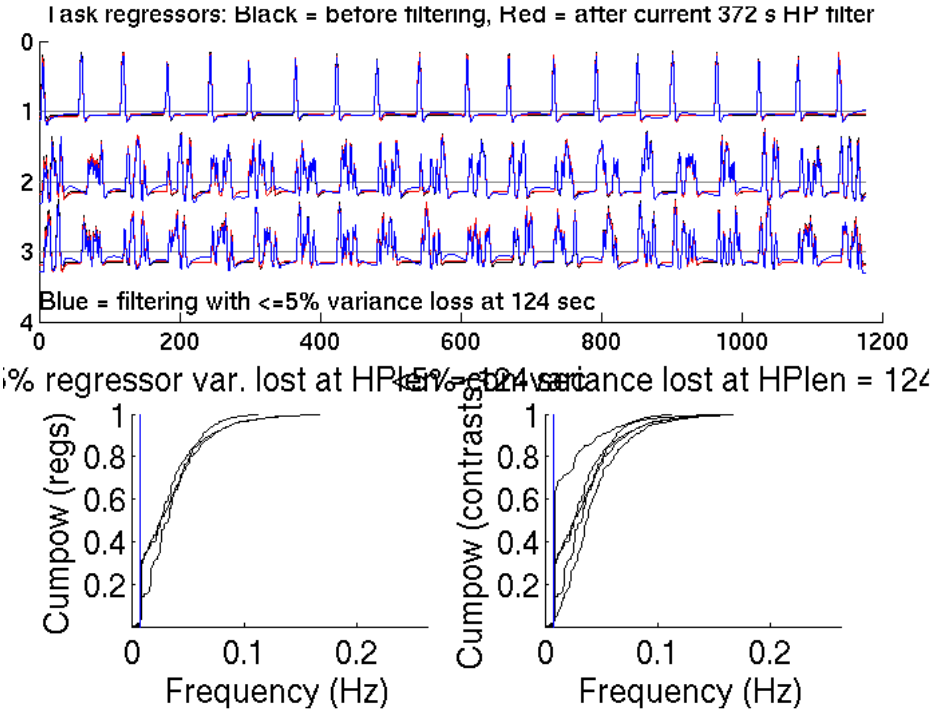
add Wani's VIF code

Example output of canlab_glm_subject_levels:

Prints an html page with the design marix and vifs for each subject, created by running scn_spm_design_check



Plot of high pass filter, which allows you to compare your current hp filter to the recommended hp filter and unfiltered data. Also plots a graph of your power for your regressors and contrasts at different hp filters.



canlab_glm_getinfo

- outputs information about subj levels from a specified group level model,
  - canlab_glm_getinfo('conw'): it will pull out all the contrasts and weights associated with it

SECOND LEVEL MODELS

RUNNING THE GROUP ANALYSIS

- **Motivation**

- you want to represent the group; not the individual

- **Scripts & Files You Will Need For This Section**
  - DSGN file
  - canlab_glm_group_levels.m

- **Steps**

1. If you don't have anything specific you need to specify, you can feed this your design file. Simply:

- canlab_glm_group_levels(DSGN)

you can also be in the model folder and do not need to specify DSGN. Advantage: if you had subjects that had issues in the first level, they will not be included in the second level.

UNDERSTANDING THE OUTPUT

- **Motivation**
  - Xxxx

- **Scripts & Files You Will Need For This Section**
  -

- **Steps**

1.

2nd level:

robust_regression_batch output --

> enter as robust_regression_batch('SETUP.mat')

>shows you the mask

>check into images (do they look okay, ae they all the same orientation)

>shows you the weights -- see what was downweighted

if a subj is totally downweighted you can find outliers easier

add:

>larger, just easier to figure out titels

>add a 'suppress output' of text // 'no verbose'

contribution each subject makes is the weight in the rob regression (weight according to if you think it is an outlier or not) -- talking about first frmi data plots

>what to make of color bars in wright maps -- should be 0 to 1?

rob_results_threshold

canlab_glm_publish add the 't' option to specify threshold

what to look for

>green covers brain [mask is 0 & 1]

bug? there are lighter parts of the overlay

[insert image of mask] -- why does it cover outside of the brain?

>histograms: look for normdistributions that look uniform across subjects (take note of subjects who are off the mean)

>why are some blue?

>in fmri data objects:

for group analysis cov(image) you wat to make sure indepdnence of subjects exists, you want a bright line diagnol, darker blue in corners look at global w across subjs

same plot style for weights

weight of 1 = good things (no outliers)

sanity check robfit (see that global mean case by case graph with the weight one, see that outliers in mean (with high SD) are downweighted appropriately

rec: plot avg weight map -- luka showed that in fslview

to do: create an output table that gives you contiguous clusters of activation (would be smaller/less in there as most are connected)

**weights are different for every contrast

analyses

>to do 2-way ttest in robfit use contrast coded regressor 1 & -1

on thresholding

>how is it done at the group level?

>how do we correct for multiple comparisons?

>does increasing the number of contrasts in your model change power? change the number of tests to correct for?

on dsgn files //

Referencing real files on dream, here are my steps to running a GLM analysis (steps 1 and 2 are one-time-only, steps 3-5 are done on a per-analysis basis):

% 1. Make multiple regressors file (for me this is just noise (spikes + motion))

% (you can make a copy and adapt to your data, assuming canlab_preproc was used... might need adjustment for canlab_preproc_2012)

>> cd /data/projects/wagerlab/labdata/current/ilcp/Imaging

>> make_noise_model_1.m

% 2. Make conditions files (I make everything I can think of and then mix and match when setting up individual analyses)

>> cd /data/projects/wagerlab/labdata/current/ilcp/Imaging/experiment_logs

>> make_timing_files.m

% 3. Set up analysis

>> cd /data/projects/wagerlab/labdata/current/ilcp/Imaging/analyses/modelscripts

>> setup_pr_model2.m

% (this creates pr_model2.mat and makes the structure "DSGN" in the working environment, either of which can be passed to the model-running functions)

>> addpath(genpath('~ruzicl/matlab_codes_in_flux'))

% 4. Run subject levels

>> canlab_glm_subject_levels('pr_model2', 'dream', 'email', 'ruzic@colorado.edu');

% 5. Run group levels

>> canlab_glm_group_levels(DSGN, 'o', '../second_level/pr_model2', 'invp', 'dream', 'email', 'ruzic@colorado.edu');

Consider this a "beta" release, so be wary and check things carefully.

Also, the optional DSGN.contrastnames is still under review with Tor (may not be an option long term, so perhaps best to not use it and get default contrast-naming instead).

preproc//

I think I've got the preproc pipeline streamlined.

dicoms will be in Imaging/<subdir>/raw. Raw and Preprocessed data will be in Imaging/tr{1300,460}/<subdir>/{Preprocessed,Raw} (because preprocessing must be done separately).

/data/projects/wagerlab/labdata/current/dicomsdicomsdicoms.sh

bash script to transfer dicoms (see help, but generally you should be able to specify bmrk5 and use the -u option)

.../bmrk5/Imaging/run_convert_and_distribute.sh

bash script to do dicom -> NIfTI conversion and populate canlab-style directory structure. It will try to run convert_and_distrubute.sh on all subjects (which will bail out on subjects that appear to already have been done given the existence of a directory for them in .../bmrk5/Imaging/tr[0-9]*[0-9]/).

.../bmrk5/Imaging/run_canlab_preproc_2012.m

matlab function that can take any vector of numbers 1, 2, 3 (which stand for part1, interactive, part2, respectively). Has help if you forget. It should figure out what subjects need the specified phase, but could get messed up if there are partial file products from canceled preproc jobs.

Motion

Motivation: Motion can impair the detection of real functional activation. Random motions of the head appear as additive noise in the voxel time-series and decrease the detectability of functional activation. Task-correlated motion can mask real activation in cortical regions and can also appear as false positive activation at the brain edges and high-contrast regions of the brain. (lifted from HBM, Briggs et al, 2009)

robust regression

>mean centering (default -- do it unless contrast coded groups have unequal Ns, then manually do the mean centering -- can ask scott for steps/code here)

Predict toolbox (copied from Psych 7215 hackpad)

What: Using fMRI data to predict pain intensity level (1-4) in new subjects.

First step: Load in data to predict!

dat = fmri_data(imgs); %load all images as an fmri_data object

%Create a vector indicating linear increasing intensity

y = ones(20, 1);

dat.Y = [y; 2*y; 3*y; 4*y];

Use PREDICT method to predict outcome for new subjects.

%Now we have multiple voxels together predicting the outcome

[cverr, stats, optout] = predict(dat, 'algorithm_name', 'cv_pcr', 'nfolds', 5, 'error_type', 'mse');

figure; plot_correlation_samefig(stats.Y, stats.yfit);

xlabel('Actual');

ylabel('Predicted');

HOWEVER, THIS ANALYSIS IS NOT VALID BECAUSE WE DID NOT LEAVE OUT ONE SUBJECT IN CROSS VALIDATION PROCEDURE

Define custom holdout set.

Leave one subject out entirely.

holdout = repmat((1:20)', 4, 1);

[cverr, stats, optout] = predict(dat, 'algorithm_name', 'cv_pcr', 'nfolds', holdout, 'error_type', 'mse');

Cross-validated prediction with algorithm cv_pcr, 20 folds

figure; plot_correlation_samefig(stats.Y, stats.yfit);

Do the same thing with support vector regression:

[cverr, stats, optout] = predict(dat, 'algorithm_name', 'cv_svr', 'nfolds', holdout, 'error_type', 'mse');

stats.pred_outcome_r

figure; plot(stats.Y, stats.yfit, 'ko'); %plot outcome data(Y) and predicted outcome data (yfit)

refline

xlabel('Actual');

ylabel('Predicted');

THRESHOLDING

FIRST PHASE

Motivation: Xxxx

- **Scripts & Files You Will Need For This Section**
  - to change p values, using fsl
  - canlab_glm_maskstats
  - statistic_image.threshold
- **Steps**

HOW TO MAKE A MASK

Motivation: Xxxx

on the wiki

http://wagerlab.colorado.edu/wiki/doku.php/help/fmri_help/fmri_statistical_models/creating_a_results_mask

canlab_glm_publish

- **Scripts & Files You Will Need For This Section**

These are scripts in Luka's home directory on dream. Yoni (joas2631 on dream) also copied them to his home directory on dream
-subcort 9 -thr 50 -o my_l_amy
mkmask
xyz coordiante in MNI space
makes sphere by default
mkclustermask
all of these are shell scripts by luka in the tools banich lab module on dream (and on Blanca too at some point)
fslmask bin
binarize masks
Probabilistic mask: i.e. create a binary mask of all voxels that have a 60%+ chance of being amygadala
open up fslview, open atlas, show layer of interest, then save as image.
fslmaths <mask name> -thr <percent desired> -bin XXX something to binarize it
mkatlasmask will do all this for us if desired, don't need to do it manually if want standard mask
to apply mask
fslmaths <image> -mul <mask> <-- if mask is 0/1, will zero out non-mask voxels

canlab tools has an equivalent:
Canlab tools:
sphere roi tool

- **Steps**

what is a mask? binary masks - 1 in the region 0 is not

pattern masks? like NPS

mask: an image with no intrinsic value but has relative value to other images

canlab thresholding functions

mediation_brain_results

>>jes has ques about fdr

>>***rename this function? not related to mediation

Extent-based correction:

Howto on cluster extent toolbox:

http://wagerlab.colorado.edu/wiki/doku.php/cluster_extent_thresholding

robust_results_threshold: FDR correction

ROI analysis methods:

-Extract mean of the beta images in a ROI in each subject

Marsbar is a useful SPM toolbox that can calculate percent signal change for you and can read contrasts from an existing SPM model:

http://marsbar.sourceforge.net/index.html

object oriented

fmridata is one structure

plot, filter, preproc (on its way)

methods fmri_data

methods statistic_image

methods regions

GRAY-MATTER MASK

the following is by Tor Wager

When correcting for multiple comparisons, it is advisable to use a gray-matter mask that is limited to regions that you would like to test a priori, but also covers all of those areas adequately. A gray-matter mask based on the canonical template can be used, but it is inadequate without manual clean-up to make sure that all relevant areas are included, and other voxels outside brain are not.

There is now a gray matter mask in the repository called:

gray_matter_mask.img

For more info and code used to make it, see here:

http://wagerlab.colorado.edu/wiki/doku.php/help/core/brain_masks

Tor recommends we use this when getting FDR- or FWE-thresholded maps.

Private Masks

masks_private in the repository:

- 5 emotion-category maps from our forthcoming meta-analysis

- placebo effect-predictive maps from wager 2011

- wolfi's 5-basal-ganglia cluster analysis, still in prep

this is a great place for the rejection and PINES maps if they are not there yet -- the plan is to release all of these eventually and/or make them available through in-browser analyses, but not yet.

PREDICTIONS

USING PREDICT

Motivation: Use machine learning techniques, trained on your data, to make predictive maps of certain stimuli, contrasts etc. Use these maps to predict brain activation in other conditions/tasks.

- **Scripts & Files You Will Need For This Section**

  - fmri_data.predict.m [note: don't confused with matlab function predict]
  - spider toolbox

- **Steps**

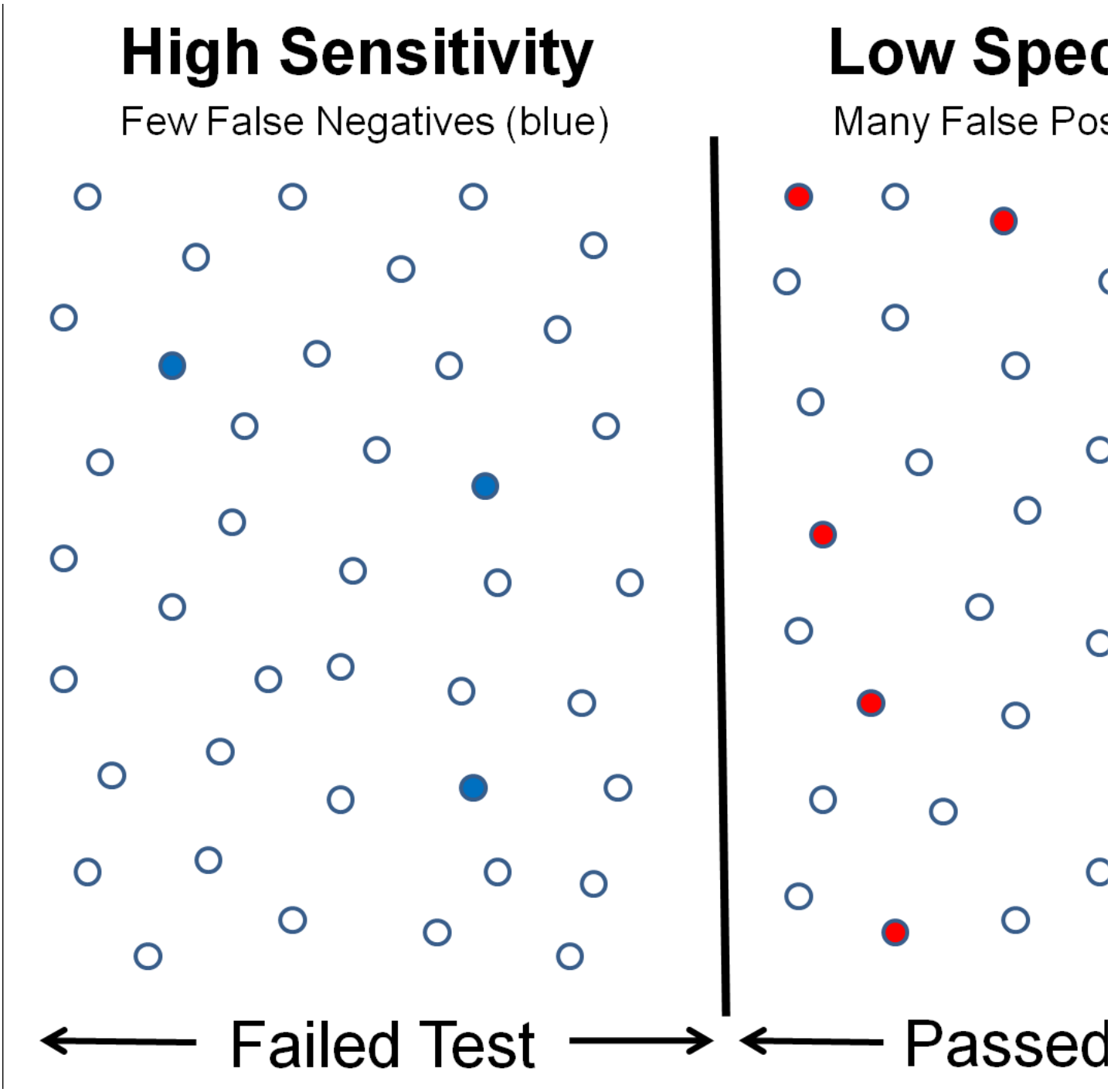
1.

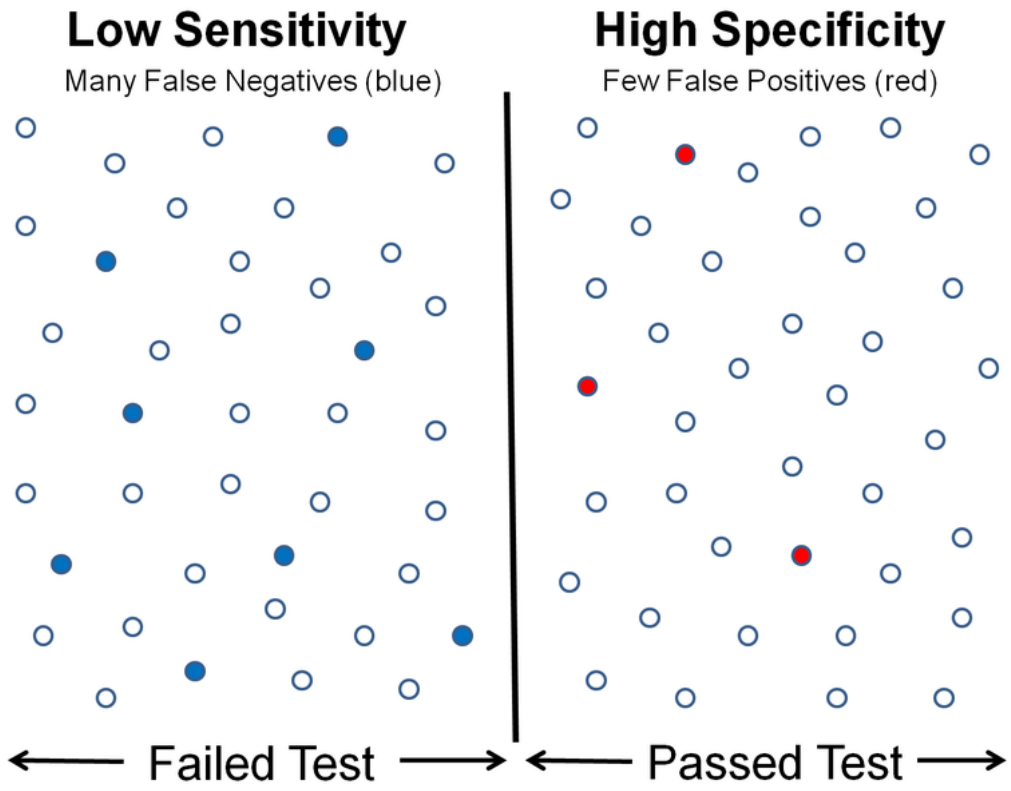
to use the cross-validated weight map you have to go into other_output, don't use weight_obj ... weight_obj will be the same independent of cross val techniques

sensitivity / specificty plot: area under the curve is how good it is

you don't want to see straight diagonals, you want it asymmptotic to Y

stolen from wikipedia:





example code from luke:

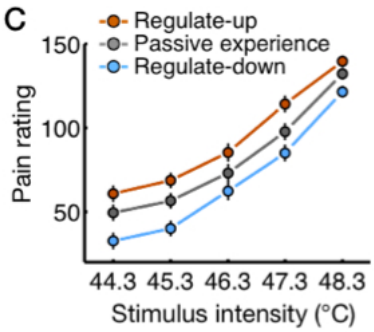
- %Load data
- fPath = '/Users/lukechang/Dropbox/CanlabDataRepository/Data/KrishnanPain/';
-
- high = fmri_data(fullfile(fPath,'Krishnan_High_Pain.nii'));
- low = fmri_data(fullfile(fPath,'Krishnan_Low_Pain.nii'));
-
- highvlow = high - low;
- t = ttest(highvlow,.005,'unc');
- write(t,'thresh','fname','/Users/lukechang/Downloads/test2.nii'); %thresholded
- write(t,'fname','/Users/lukechang/Downloads/test2'); %unthresholded
-
- d = high;
- d.dat = t.dat;
-
- %combine and run svm
- high_low = [high,low];
- high_low.Y = [ones(size(high.dat,2),1);ones(size(low.dat,2),1)*-1];
-
- [cverr, stats, optout] = predict(high_low, 'algorithm_name', 'cv_svm', 'nfolds', 5, 'error_type', 'mcr');
-
-
- %Univariate regression
- r = regress(high_low)
- out = high;
- out.dat = r.b;
- write(out,'fname','/Users/lukechang/Downloads/test2.nii')

DATA VISUALIZATION

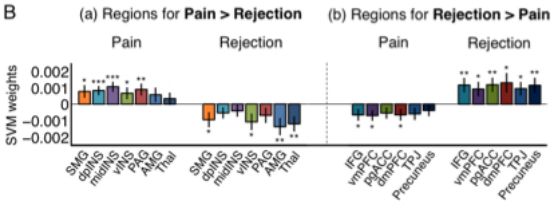
MAKING FIGURES

Motivation: Make better figures for publications

- **Scripts & Files You Will Need For This Section**
  - All resources are in the following dropbox folder
    - https://www.dropbox.com/sh/dvrix5596kzpf2u/AAC9_nLwoz5nw3RSexgA50wMa
  - There are a few functions and example codes that Wani has worked on, including sepplot, bar_wani, roi_contour_map. I've committed these functions to the repository and added them into figure gallery of our wiki (http://wagerlab.colorado.edu/wiki/doku.php/help/core/figure_gallery).
  - scn_export_papersetup (by Tor)
  - sepplot example (fyi, I added legends and axis labels in ppt, not in matlab for convenience).



- bar_wani example



- roi_contour_map example

- Some colormaps that Wani created might be helpful.
  - colormap_blind_wani.mat : this colormap is from http://www.nature.com/nmeth/journal/v8/n6/full/nmeth.1618.html
  - colormap_colorbrewer_wani.mat : http://colorbrewer2.org/
  - colormap_microsoft.mat : this is from microsoft color pallet.

- Tips and Considerations
  - 300 dpi for images (avoid png -- go eps & pdf, which are for creating vectorized images)
  - pagesetup(gcf);
  - saveas(gcf, 'Figure_name.pdf');
    - In powerpoint, copy figures, and right click -- Paste Special... then, paste as pdf.
  - balanced layout

- **Steps**

1.

Resources

- Bang Wong - Nature Methods monthly reviews on how to optimize publishable figures
  - overview figures
  - http://blogs.nature.com/methagora/2013/07/data-visualization-points-of-view.html
- MATLAB tutorials
  - http://blogs.mathworks.com/loren/2007/12/11/making-pretty-graphs/
- MRIcroGL
  - http://www.mccauslandcenter.sc.edu/mricrogl/
  - Note: check that you are overlaying images in the same space; will not get an error if you don't
- Caret
  - http://brainvis.wustl.edu/wiki/index.php/Caret:Download
- Tor Tools (MATLAB)
  - tor_3d.m, canlab_results_fmridisplay.m, addblobs.m (a function of a fmridisplay object)

- %% canlab tools to visualize
- o2 = fmridisplay; % create a fmridisplay object
-
- xyz = [-20 -10 0 10 20]'; % slides for saggital view (x)
- xyz(:, 2:3) = 0;
- o2 = montage(o2,'saggital','wh_slice', xyz, 'onerow'); % create saggital montages
- o2 = montage(o2, 'axial', 'slice_range', [-40 50], 'onerow', 'spacing', 6); % create axis view montages
-
- dat = fmri_data('image_you_want_to_visualize.nii', which('mask_colin27.img')); % read the image you want to visualize
- cl = region(dat);
- o2 = removeblobs(o2); % clean up the activations
-
- o2 = addblobs(o2, cl,'splitcolor', {[0 1 1] [0.05 0.45 1] [1 0.38 0.27] [1 1 0]}); % visualize the activations. You can use different colors.
- o2 = addblobs(o2, cl,'color', [0 0 0], 'outline', 'linewidth', 1.2); % draw outline
- squeeze_axes(o2.montage{1}.axis_handles, 20); % interval between brains

-
- %% canlab_results_fmridisplay
- o2 = canlab_results_fmridisplay('image_you_want_to_visualize.nii'); % one line function

ADDING PHYSIOLOGY

SKIN CONDUCTANCE

Motivation: how to add SCR data as nuisance co-variates or parametric modulators

Using physiological regressors to increase signal to noise ratio

RESPIRATION AND HEART RATE

- **Scripts & Files You Will Need For This Section**

- PHLeM toolbox: https://sites.google.com/site/phlemtoolbox/Home
  - Note: mostly written for users of physio from Siemens, but this page has a link to scripts for using BIOPAC with PHLeM (likely need to be modified- http://psych290z.wikispaces.com/Lab+3+-+FSL), includes sliding window code
  - Can do RETRIOICOR, Variation models (like RVHR), Downsampled model
- FSL toolbox PNM: http://fsl.fmrib.ox.ac.uk/fsl/fslwiki/PNM#PNM:_Physiological_Noise_Modelling
  - Requires column of scanner triggers (how to set these up with BIOPAC??)
    - best to have a trigger sent for each TR
      - In Acq file, Set up acquisition --> External triggering (p 104)
      - you will have to set up a channel that takes those pulses each TR
  - Inputs: Physio in format (one line per sample), fMRI timeseries, sampling rate of physio
  - Outputs: Respiratory, Cardiac EVs in fsl format, Optional: RVT, Heart Rate and CSR regs

Different models of adding physiological nuisance regressors:

RVHR: (more details of Chang et al (09) method in http://www.sciencedirect.com/science/article/pii/S1053811908010355)

RV: respiration volume- calculated based on standard deviation of respiratory waveform in a sliding window of 3 TRs
HR: Average of ECG in sliding window for 3 TRs, divided by 60 to convert to beats per minute

Details of this method (see script make_physio_regressors in PHLeM):

1) compute RV
2) convolve with RRF (respiration response function)

Explained variance beyond RV model

RVT (Birn 06: http://www.sciencedirect.com/science/article/pii/S105381190701083X): respiration volume per unit time (diff between max and min belt positions at peaks of inspiration and expiration, divided by time between peaks)

Model of fMRI signal changes induced by variations in respiration volume (changes in respiration depth and rate), models fMRI changes from breath holding and cued depth and rate changes

Often, MRI signal changes from breathing are bimodal (early decrease followed by strong overshoot (especially if breath hold afterwards)

RETROICOR: Image based retrospective correction (Glover-2000:http://ress.webhost.utexas.edu/pdf/Glover_RetroICorr_MRM2000.pdf)

Determine the cardiac and respiratory phase of each slice in each volume

How to identify cardiac phase (consistent peak in pulse oximeter, R-wave in ECG)

How to identify respiratory phase (need to account for timing, depth of breath, sign change from inspiration to expiration)

These "phases" are then entered in a fourier transformation, and can either be regressed out prior to analysis, or included as nuisance regressors in GLM (likely better, because changes degrees of freedom)

How would this change if multiple slices acquired at same time? Need to get slice timing right also

Disadvantage: does not remove slow changes in rate and depth of breathing

Practical application

What to do with collected data

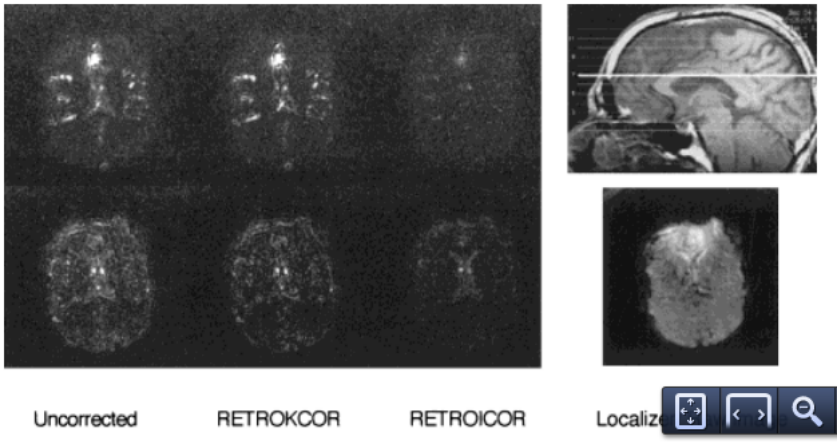
Trim extra time from physio data/add in approximate scanner triggers (wasn't working when collected)?

For future: How to add scanner triggers as a column in BIOPAC? Better than starting acquisition with first TR trigger? Probably, because many toolboxes require this

How to use other data (breath hold and functional localizer) to test the effect of different physio regressors (potentially before applying to main task)?

Why add physio?

Reduced false positives and extent of connectivity in default network



From Glover (00): example of areas with heart rate (top) and respiratory noise

Comparision of data with physio regressors from Chang 2009:

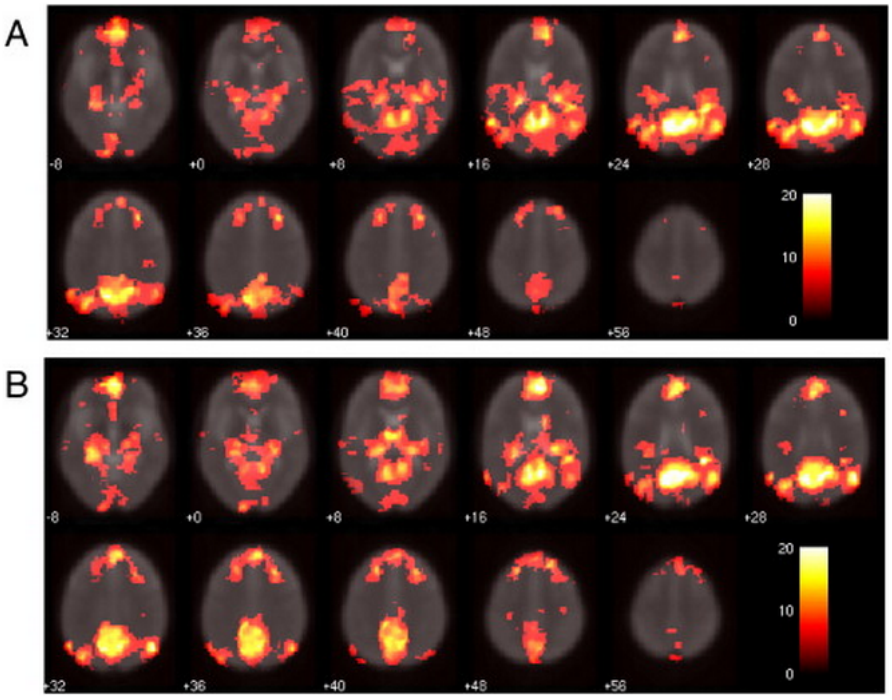


Fig. 5.

Group-level DMN connectivity maps (A) without correction for physiological noise, and (B) with correction using RETROICOR and RVHR. Color bars depict *T*-values.

Other approaches:

Filter out frequency of physiological data: (from Brooks et al: http://journal.frontiersin.org/Journal/10.3389/fnhum.2013.00623/full)

Temporal filtering:

Cardiac cycle: ~1hz, Respiratory: 0.2-0.3hz

Suffers from problems:

Typical TRs have too low a of a sampling rate to distinguish, causing them to alias to different frequencies

less than .4s to avoid aliasing for signals up to 75bpm

Should account for the change in frequency data when doing subsequent analysis

template.

Use ICA to filter out physiological signals? Pro- easier to do Downsides?

ON MULTIBAND

IMAGING PROTOCOL SPECS

- **Tips (from Liz)**

- 2 ways to do distortion correction
  - field map is one (others use this, we don't), newer approach is to apply two different phase encoding directions of distortion correction scans
- Bias correction scans are for hyper intensities with 32 ch, can just use the pre-scan normalize option instead
- Switch from A>P to P>A reduces distortion in frontal cortex
- SBref scans are collected before first real volume, has least distortions and so is ideal for coregistration with oner modalities
- A>P brain is squished in the front, and P>A stretches out the front of the brain (not true in general, it's just because greatest distortion happens in OFC)
- Stretched out brain in regular sequence is easier to correct for (squish the signal back)
- Susceptability gradient across slice causes dropout, within slice causes distortion (distortion is easier to correct for)
- 20-30 deg angle up from AC-PC is better for minimizing dropout
- 29 or 30 ms for TE, same for multiband and non-multiband, higher TE than that = more dropout
- Higher order shims to try to further cancel out dropout in OFC, but will affect other parts of the brain negatively
- Echo spacing must be same in distortion correction scans and main scan

SINGLE TRIAL MODELS

SPECIFYING SINGLE TRIAL MODELS IN YOUR DSGN FILE

Motivation: useful for running multilevel mediation and looking at single trial effects of pain, etc

- **Scripts & Files You Will Need For This Section**
  - DSGN file
  - canlab_glm_subject_levels
- **Steps**

1. Make a single trial onset file. Should contain one condition (though you might have separate runs, so separate onset files for each) with the onsets (in TRs or seconds) for each trial. For example, in the script below I use a log file, per subject, to determine the start time in seconds of the first TR, and then record the start time of each trial relative to that TR in seconds. I created the onset files per subject, per run, like this:

```
exp_dir='/Volumes/engram/labdata/data/IAPS_Marianne/ExperimentLogs';
addpath(genpath(exp_dir));
subs=filenames('/Volumes/engram/labdata/data/IAPS_Marianne/ExperimentLogs/ICAPS*');

for i = 1:length(subs)
    cd(subs{i})
    r1info_file=filenames('IAPSlog_r1*.mat');
    r2info_file=filenames('IAPSlog_r2*.mat');
    load(char(r1info_file));
    name{1} = 'iaps_image_r1';
    duration{1} = [abs(explog(2:end,8))]; % actual durr
    onset{1} = [explog(2:end,6)-explog(2:end,5)]; %subtract time of start tr from each stim
    fout=name{1};
    save(fout,'name','onset','duration');

    clear explog name durr onset fout
    load(char(r2info_file));
    name{1} = 'iaps_image_r2';
    duration{1} = [abs(explog(2:end,8))]; % actual durr
    onset{1} = [explog(2:end,6)-explog(2:end,5)];
    fout=name{1};
    save(fout,'name','onset','duration');

end
```

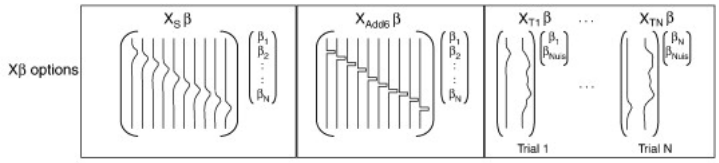
1. Go into your first level model DSGN file and add this after your DSGN.conditions section. This references your conditions. You can indicate (with a 1) if you want that condition to be modeled as single trials or not. Easy!
   - **%% SINGLE TRIAL**
   - **DSGN.singletrials{1} = {1 1 0 0};**
   - **%% this specifies which conditions will be modeled as single trials, in the order that you listed your conditions**
   - %% e,g, in this case the first 2 conditions would be modeled as single trials, but not the third and fourth condition

1. You do not need to specify contrasts in your script (usually not useful on the first level for single trial models)
2. Now you are ready to run your models (see the First Level Models section of this document for more information).

Issues and Concerns

- **Watch Your VIFS**
  - High VIFs (greater than 5 or 2.5, depending on whom you ask ;)) can be harmful to your single trial model results and mediation results
- **Rapid Event Related Designs**

Use what works best for your design:



	Model	Estimation	Regularization parameter	Name
Approaches	$Y=X_s\beta+\varepsilon$	Least Squares	None	LS-A
	$Y=X_s\beta+\varepsilon$	Ridge Regression	λ: Hoerl-Kennard	Ridge (HKB)
	$Y=X_s\beta+\varepsilon$	Ridge Regression	λ: 2 fold CV	Ridge (CV)
	$Y=X_s\beta+\varepsilon$	Partial Least Squares	#components: 2 fold CV	PLS
	$Y=X_s\beta+\varepsilon$	Support Vector Regression	e & C: 2 fold CV	SVR
	$Y=X_{Add6}\beta+\varepsilon$	Least Squares	None	Add6
	$Y=X_{Add4\text{-}6}\beta+\varepsilon$	Least Squares	None	Add4-6
	$Y=X_{T1}\beta+\varepsilon$	Least Squares	None	LS-S

If your ITI is very short, you might have lots of overlap between trials, and high VIFs. One way to get over this would be to use a modeling scheme described by Mumford (2012). Figure is from there.

STANDARDIZING DATA

HOW TO PRODUCE SINGLE TRIAL MODELS FOR CROSS-STUDY COLLABORATIONS

The following instructions are from Tor's email (8/14/14)

What is needed: Share fmri_data objects for each subject with single-trial data and meta-data in a .mat file, in a single folder with your study name.

- This is shared on Dropbox, and copy (manually updated, may not be current) is stored on canlab.colorado.edu

here:

- /Volumes/RAID1/labdata/Wagerlab_Single_Trial_Pain_Datasets
- To share data with the repository, you can put data from individual studies in the Data folder:
- Wagerlab_Single_Trial_Pain_Datasets/Data/studyname
- Let Tor know when you have shared or updated data there.

STEPS

1. fmri_data objects
   - **WHAT**: for each subject, an fMRI data object with single-trial estimates for all pain responses across the standard brain mask
   - **HOW:**
     - locate single-trial model directory
     - identify single-trial regressors for trials of interest (be careful: some regressors may reflect nuisance covariates. There may be different numbers of these for each run)
     - for help: see help and code in
       - scn_spm_design_check.m
       - scn_spm_get_events_of_interest.m
     - create a list of filenames for single-trial beta images (e.g., with filenames.m)
       - dat = fmri_data(filename_list)
       - Note: it's helpful if you call this "dat" for compatibility across studies
     - save file as studyname_subjname.mat, and save filename in meta-data structure (below)

1. Meta-Data Structure
   - **WHAT**: a structure with data for each trial, with temperatures, ratings, index of high VIF trials if available -- see below
   - **What is inside:**

- bmrk_st_comb
  - This is an example structure

- temp: {1x111 cell}
  - temperatures, one cell per subject. in the cell is a column vector for n trials

- ratings: {1x111 cell}
  - ratings, one cell per subject. in the cell is a column vector for n trials

- high_vif_trials_idx: {1x111 cell}
  - indicator for which are high-VIF traials, one cell per subject.

- imgs: {1x61 cell}
  - Image names - I don't really need these, just the fMRI-data objects

- high_vif_descript: 'vif > 2.5'
- subjects: {1x111 cell}
  - Subject names, one cell per subj

- dat_obj: {1x111 cell}
  - Name of data obj filename for each subject
  - **NOTE**: You can also add other fields, and call them whatever you like. It would be very useful to add other data about the trial types for each trial (e.g., regulation/placebo/priming status, depending on your experiment). If you do, please add a _descript field with a descriptive text string describing what the variable is and how it is coded (what the codes mean).

1. Put individual fmri_data object files and meta-data file in folder and share with Tor, either (a) on Dropbox, or (b) in the folder on the canlab server (see above).

- **Checking results and quality control**

If you don't see the results expected, or are otherwise checking the quality of subjects (essential!), then here is a list of "troubleshooting" things to look at.

1. **Do meta-data match up with data file names?**
   - Check subject names in meta-data objects against (a) names of data .mat files, and (b) list of images in those .mat files. Are subject names consistent in all cases?
   - Are trial images ordered sequentially in all .mat files, as expected?
2. **Do data files look reasonable?**
   - Use plot(dat) to plot all single-trial objects for each subject.
   (a) Do the average orthviews maps look like brains?
   (b) Are they in the correct orientation?
   (c) Any dramatic outliers in the data plot (top left)?
   (d) Any dramatic covariance (off-diagonal) in the covariance plot (bottom left)?
   (e) Any dramatic global outliers?
3. **Do meta-data pain ratings look reasonable?**
   (a) Plot for each subject. Are they in the expected range?
   (b) Are there any unusual values (e.g., 0) that could signal invalid or excluded trials, that are not appropriately handled in the analysis?
4. **Examine the design matrix used to estimate single trial responses**
   (a) Plot and save VIFs for each subject in meta-data structure, for single-trial regressors, with and without covariates added
   (b) Are there unusually high VIFs (>2 or so) in the basic design matrix, without covariates? If so, design may be modeled inappropriately
   (c) Are there unusually high VIFs (> 4 or so) after including covariates? If so, task-related head motion or artifacts could be problematic
   * It is a really good idea to save VIFS for all studies in the meta-data object
5. **Consider other misc errors**
   (a) Check the results of standard, non-single trial models. How strong are the NPS responses there? What is the effect size? If results are strong, something may be wrong with the single-trial model or data management.
6. **Consider the possibility of timing errors or mismodeling**
   (a) Are you sure you adjusted for the discarded acquisitions correctly when you specified onset times?
   (b) Extract NPS responses from original preprocessed time series data (e.g., swravols), after removing run intercepts, high-pass filter covariates, and nuisance regressors. Use an FIR model to extract peri-stimulus HRF time courses locked to pain onset
   - Is there a response?
   - Is its timing typical or unusual?
   - Examine cross-correlations. Do some subjects show evidence for a time-shifted response?

- **Consider reasons why your results may not match the NPS or be significant in cross-validated analyses even if the data quality is good:**

- Atypical type of stimulation/modality?
- Small number of trials per subject (small training dataset?)
- Atypical duration?
- Lower intensities than most studies?
- Atypical site of stimulation?
- Different imaging/acquisition sequence?
- Different population?

* Including meta-data about these things in the meta-data structure will be really helpful as we move forward comparing studies

HOW TO BRING YOUR DATA INTO THE CANLAB_DATASET OBJECT FORMAT

The following instructions are from Tor's google doc

What is needed: Your data, your predictors, demographic info on your subjects, and canlab_dataset

- **Canlab_dataset object format**

canlab_dataset is an object specialized for storing information about behavioral and fMRI datasets. Some advantages include the ability to write data to a flat text file with one command. This allows us to store data in a format that we can use in imaging analyses, and also easily store a copy in a format that is readable by anyone in any program. Another important feature is standardized fields for storing information at multiple levels: The experiment level, the subject level, the event or trial level, and a sub-event level (e.g., for physio data).

STEPS

Create an empty one:

- dat = canlab_dataset();
- dat

Now you have a canlab_dataset with properties:

- Description: [1x1 struct]
- Subj_Level: [1x1 struct]
- Event_Level: [1x1 struct]
- Sub_Event_Level: [1x1 struct]
- Continuous: [1x1 struct]
- wh_keep: [1x1 struct]

Methods are operations you can run on the object.

Some are:

write_text

- writes a flat text file (data across all subjects)

concatenate

- returns flat matrix of data across all subjects to the workspace

get_var

- get event-level data from one variable and return in matrix and cell formats

scatterplot

- standard or multi-line scatterplot of relationship between two variables

scattermatrix

- Scatterplot matrix of pairwise event-level variables

All methods for class canlab_dataset:

- add_var
- mediation
- bars
- plot_var
- canlab_dataset
- print_summary
- concatenate
- scattermatrix
- get_var
- scatterplot
- glm
- ttest2
- glm_multilevel
- write_text
- histogram

Adding Experiment Design Info:

There is a matlab script that auto generates these csv files from the EPrime files:

- edat_to_inscanner_behav(edat_in_fname, csv_out_fname, varargin)

There will be a matlab script that reads in these .csv files and creates a struct. The struct has one field per column. Each field contains a cell array, dimensions = 1 x num-rows/trials-in-csv-file.

CANLab Pipeline:

- generate .csvs for in-scanner behav data in aforementioned format. each Functional/Preprocessed/run* folder will have one such file. The script for doing this is complete for EPrime users and can be run in batch mode to bring existing data up to standards.

- use a simple shared script to read those csv files in to a matlab struct

- write custom script for each study to create SPM regressor files (for Luka's GLM scripts) from the matlab structs created in step 2

MEDIATION TOOLBOX

This section on mediation is brought to you by Leonie Koban

OVERVIEW

Motivation: Mediation is a useful tool for inferring causality between changes in brain activity and changes in behavior. It is also allows us to study individual differences. The typical mediation framework looks like this:

Or more generally:

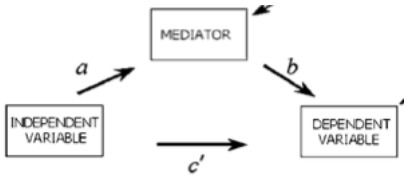


Figure 1
Mediation model.

www.annualreviews.org • Mediation Analysis 595

$$Y = i_1 + cX + e_1,$$
$$Y = i_2 + c'X + bM + e_2,$$
$$M = i_3 + aX + e_3,$$

X should be a manipulated variable. It is important to note that this analysis cannot prove causality. Causality is a logical, theoretical, & experimental issue. For more theoretical information click here.

- **Scripts & Files You Will Need For This Section**
  - mediation.m
    - use this with single dim dta (behav or avg ROI resp)
  - mediation_brain.m
    - use this for voxel wise data
  - mediation_brain_multilevel.m
    - every cell is one subject, and each have M (betas), X (e.g. -1/1 trial type), & Y (e.g. pain rating)
  - mediation_brain_results.m
- **Alternate Resources**
  - http://wagerlab.colorado.edu/wiki/doku.php/help/mediation/m3_mediation_fmri_toolbox
  - Kenny, David A., Josephine D. Korchmaros, and Niall Bolger. "Lower level mediation in multilevel models." *Psychological methods* 8.2 (2003): 115.
  - MacKinnon, David P., Amanda J. Fairchild, and Matthew S. Fritz. "Mediation analysis." *Annual review of psychology* 58 (2007): 593.

SINGLE LEVEL MEDIATION

Motivation: Example Question: Do individual differences in treatment expectations mediate placebo effects? Employ this analysis when you have one measure per individual for X, M, & Y (or additional mediators and covariates).

- **Steps**

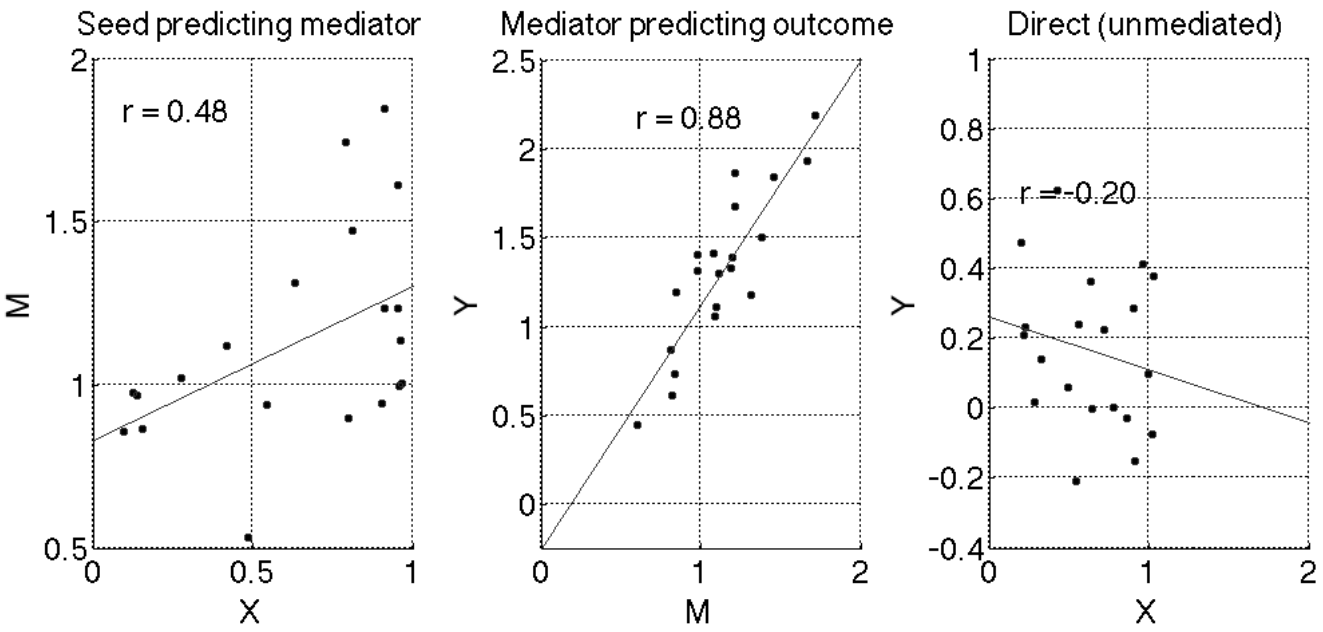
1. define your X, Y, & M and load them into matlab as row vectors
2. optional: install geom2d toolbox for plotting the path diagram. If not, you'll get this message:

- Warning: To create a mediation path diagram, you must have theexternal geom2d toolbox on your path. I can't find it, so thee path diagram will be skipped.

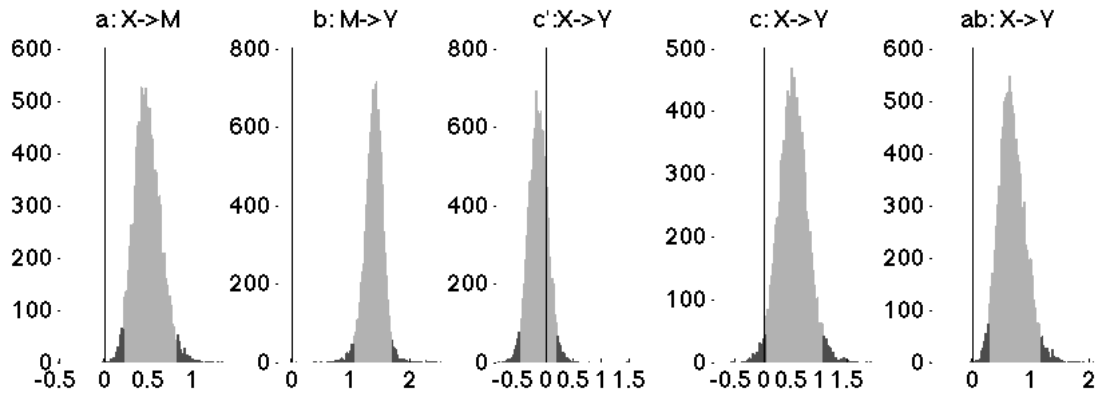
1. Note on Bootstrapping: not necessary at the first level. It is more computationally demanding without the payoff.

- **Example (by Tor):**
  - X=rand(20,1);M=X+rand(20,1);Y=M+rand(20,1);
  - [paths, stats] = mediation(X, Y, M, *'plots'*, *'verbose'*, *'boot'*, *'bootsamples'*, 10000)

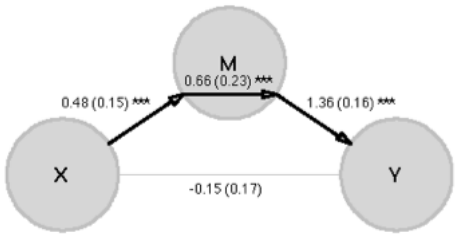
Output:



histograms



optional path diagram



MULTI LEVEL MEDIATION

Motivation: you have repeated measures nested within subjects

Advantages:

- handles uneven group sizes, missing data

Example questions:

Do trial-by-trial expectation mediate social influences on SCR?

- **Steps**

1. setup up your mediation by preparing cell arrays for your variables. You need one cell per subj (1xN), each cell contains a vector with X & Y predictors (X) must be mean centered

output

>look at the covariance between path A & B

Note on Bootstrapping: We bootstrap because the analysis is over conservative.

- We use 10 thousand samples because we are concerned with differences in the tails.
- We sample from the whole row vector: sepcific subjects' X, Y, & M vectors stay together, and therefore we preserve their covariance.
- With 1000 samples, our minimum possible p value is 0.001.

MULTI LEVEL BRAIN MEDIATION

Motivation: You have repeated measures and brain data... same thing as multilvel, but for each voxel in the brain

Steps

1. setup up your mediation by preparing cell arrays for your X, Y, & M variables. You need one cell per subj (1xN)

you need:

>cell arrays (1xN) for X,Y & M

M typically sinlge trial beta est

Here is an example script by Leonie Koban

- % by LK July, 2014, based on DPSP scripts
- % prepare data to see what mediates social influences on pain
-
- clear all
- close all
-
- addpath(genpath('/dreamio/home/maje3857/src/spm8'))
- addpath(genpath('/dreamio3/wagerlab/Repository/'))
- % addpath(genpath('/dreamio/home/leko6010/LeonieScripts'))
-
-
- %% SOCIAL PAIN
-
- behdir = ('/dreamio3/wagerlab/labdata/current/scebl1/Imaging/analyses/behavior/');
- basedir = ('/dreamio3/wagerlab/labdata/current/scebl1/Imaging/analyses/mediation/SimHC_WBgenpain_painrate/');
- st1dir =('/dreamio3/wagerlab/labdata/current/scebl1/Imaging/analyses/first_level/model1_gen_ST/');
- % addpath(maskdir)
-
- subjects = filenames([st1dir, 'SCEBL*']);
-
- cd (behdir)
- load SCEBLmri_data_gen24.mat;
-
- for sub = 1:numel(subjects)
-
- cd (st1dir)
- cd (subjects{sub})
-
- allbetas = filenames('beta_*.img', 'absolute');

```
    load SPM.mat
    p_ind = (regexp(SPM.xX.name,'AllPain'));
    for ii = 1:numel(p_ind)
        if ~isempty(p_ind{ii})
            dummy(ii) = (p_ind{ii});
        end
    end

    painbetas = allbetas(find(dummy == 7));
    % take only those that are actually pain
    SETUP.data.M{1, sub} = char(painbetas);

    clear painbetas allbetas dummy p_ind
end

SETUP.data.X = SCEBLmri_gendata.sim_hc;
SETUP.data.Y = SCEBLmri_gendata.pain;

%% SETUP of mediation

mask = which('brainmask.nii');

cd (basedir)
save medvars SETUP mask

mediation_brain_multilevel(SETUP.data.X, SETUP.data.Y, SETUP.data.M, struct('mask', mask), 'boot', 'nopreproc', 'bootsamples', 1000);
```

Bootstrapping

- Tor recommends 10,000 samples for final analysis on complete data set

>ask tor about the rationale behind bootstrapping here: what's being bootstrapped & why is that the recommended sample
>how does bootstrapping effect thresholding of results?

Make it faster:

>parallel hack by wani: do it in slices, send the slices to diff jobs, and recombine

>less bootstrap samples

>mask out the white matter

conjunction flag (-conj)

>adds another panel to the output with the conjunction results

**dist m sub --> job takes over 24 hr so have to make sure dream/blanca doesn't close the node on you

**find blanca equivalent

looking at the results

- mediation_brain_results
- Display results and set up interactive viewing plots:
  - mediation_results_interactive_view_init

options to explore:

- 'names'
  - naming clusters (can select vmpfc cluster & youll see the name in the output table)
  - name clusters one-by-one before returning cluster/table output
- 'overlay'
  - brain to be overlaid on? default colin?
- threshold
  - set the p value
  - more
  - fdr correct the maps:
    - Get
      - warning: unknown input string option fdrthresh for the following code:
        >mediation_brain_results('all','thresh',[Inf
        >this particular combo seemed to work for me to get fdr thresholded images: [clpos, clneg, clpos_data, clneg_data] = mediation_brain_results('xmy', 'thresh', [Inf], 'size', [5], 'fdrthresh', .05, 'prune', 'conj','overlay',overlay,'mask',brain_mask,'tables');
    - mediation_brain_corrected_threshold
- size
  - extent threshold
  - is this cluster size?
- 'table'
  - get a table print out of clusters
- 'mask'
  - add a mask

3 PATH MEDIATION

by Wani

PRINCIPAL COMPONENTS ANALYSIS (PCA)

UNDERSTANDING IT

PCA is a method of dimensionality reduction which will help us find patterns in our data. PCA finds 'the angle that spreads out the points the most (captures the most variance possible). PCA attempts to preserve linear structure in the data.

This breakdown is helpful.

Motivation:

- **Scripts & Files You Will Need For This Section**
  - xxx.xx
- **Steps**

setup_fastRPCA

from tor's email

- x = dat.dat;
- % can decompose x or x', but faster when rows >> cols
- % 3 solvers (5 variants)
- % constrained
- % lagrangian: two versions: sum of L, S, or max L,S. two params: lambda L,S
- % SPGL1: fixed epsilon (SSE bound), get rid of one parameter
-
- % Pick SSE cutoff for epsilon
- % can choose based on how much residual variance unexplained we want
- S = svd(x, 'econ');
-
- figure; plot(cumsum(S.^2))
- plot(cumsum(S.^2) ./ sum(S.^2))
- cutoff = 40;
- eps = sqrt(sum(S(cutoff:end).^2))
- % choose lower eps in practice
-
- %%
-
- % opts.tol : tolerance for convergence...higher is faster
- % higher eps : faster...
-
- opts = [];
- opts.max = true; % max or sum method. max method preferred in general because more stable solution...for algorithmic reasons...
- A_cell = []; % this is for partial observations (missing data) or partial weights for noisy data/outliers
- lambda_S = 1;   % constraint parameter: larger, more sparsity, smaller, less
- % 0 to Inf... 0 should be no penalty (don't do it)
-
- [L,S,errHist,tau] = solver_RPCA_SPGL1(x, lambda_S, eps, A_cell, opts);
- % try different lamda_S values.
- % once you find good tau value, use it to initialize
- % sparsity in output -> if 0%, then S is zero, all sparse, reduce lambda_S...
- % could use AIC/BIC to optimize sparseness approximately...
-
- % L is 'denoised' data matrix (run SVD to get distributed components)
- % S is sparse data matrix of excluded from L
-
-
- % complexity/slowness: m^2 * v

SVN: COMMITTING CHANGES TO THE REPOSITORY

We are no longer maintaining the SVN repository, please use the CanlabCore repository on Github https://github.com/canlab/CanlabCore

INSTALLING THE REPOSITORY FOR THE FIRST TIME

Motivation: You want the codes!

- **Steps**

1. open terminal and make sure you have svn installed

- svn co svn+ssh://svnclient@canlab.colorado.edu/Volumes/RAID1/resources/svn/Repository NameOfLocalRepositoryFolderToCreate

You might need to enter the common password several times

UPDATING YOUR LOCAL COPY

Motivation: You want to update your local copy of the repository

- **Steps**

1. open terminal
2. cd to your repos folder:

- cd ~/CANLabRepos

1. update

- svn update

1. the password is the lab password

LOOKING AT THE CHANGES YOU'VE MADE

Motivation: You edited your local scripts, maybe you want to share these edits, maybe you don't want to yet. This lets you see what you've done.

- **Steps**

1. open terminal
2. cd to your repos folder:

- cd ~/CANLabRepos

1. check what you've changed... this will print out all the files you've edited to

- svn status

1. You don't remember what you changed in one of the files listed? Let svn tell you, using 'diff' then the full path to the file

- svn diff trunk/SCN_Core_Support/Visualization_functions/barplot_columns3.m

1. You want to go back to the way things were? Use 'revert' and give the full path to the file

- svn revert trunk/SCN_Core_Support/Visualization_functions/barplot_columns3.m

1. You think your edits are golden and should be shared with everyone? Use 'commit' and give a little message about what you have changed. Important: make sure you check your status. You will commit everything you have changed, so revert things you do not want to commit.

- svn commit -m "Added info on single trial models to the README -marianne"

3D Printing Brains

Reconstruct the cortical surface of your brain

Motivation: You want the shape of the outside printed, not the skull, not the insides.

Note: These instructions, though adapted for our lab environment, were really put together by this guy.

- **Scripts & Files You Will Need For This Section**
  - freesurfer: recon-all
- **Steps**

1. Log onto blanca. You can just ssh.
2. Add the freesurfer module. You can add this to your bash_profile if you want to use it always and forever in all terminals.

- $ module add freesurfer/5.3.0

1. Rsync your raw t1 weighted structural images into this folder. Make a new directory with your name or your subject's name.

- /work/ics/data/projects/wagerlab/labdata/projects/3DPrint/[NAME]

1. In terminal, set up the reconstruction of your brain's cortical surface! This might take a while...

- recon-all -i ./02+t1mprage/methisisme+02+t1mprage+00001.dcm -s MR101 -sd ./
- -all
  - where:
    - **-i** takes the first image of the structural .dcm
    - **-s** is some made up subj ID
    - **-sd** is the folder you want the reconstructed files to go in
    - **-all** just comes at the end, I'm not sure why. Someone google why.
  - for more information: http://ftp.nmr.mgh.harvard.edu/fswiki/recon-all

Note: If your data went through the MRN autopreprocessing pipeline then this has already been done for you and you can find these files in your subject's data folder in:

- analysis/fs_5.3_*_mprage/M*/surf/

Convert the reconstructed image into .stl

Motivation: You want it in a file format 3D printer's understand.

- **Scripts & Files You Will Need For This Section**
  - freesurfer: mris_convert
- **Steps**

1. cd to the directory where your reconstructed images are

- cd /work/ics/data/projects/wagerlab/labdata/projects/3DPrint/[NAME]/surf

1. Now call this command for each hemisphere:

- mris_convert lh.pial lh.pial.stl

- mris_convert rh.pial rh.pial.stl

1. Now you want to rsync this onto claustrum or calor or your own machine, anywhere where you can download Meshlab and use it easily.

- rsync -auvR . [your_username]@10.224.20.7:/Volumes/engram/labdata/data/

Downsample your .stl file in Meshlab and Slice it

Motivation: In order to help prevent your printer from crashing, reduce the size of the image to have less than 20,000 faces.

Maybe also need to: You then use a slicing program to cut the model into thin horizontal segments. Slic3r and Cura are open source free options, but use whatever your printer suggests. The slicing program generates the tool path based on the characteristics of the model and your extrusion settings. This program will generate the G-Code that drives the printer. You then pass that file to a program that controls and operates the printer, we recommend Printrun/Pronterface.

Additionally, you can use an option in Slic3r called "Brim" to extrude a thin, one layer surface to help the edges of the print adhere to the print bed.

- **Scripts & Files You Will Need For This Section**
  - Meshlab
- **Steps**

1. Open meshlab and load your stl file
    1. Open --> Import Mesh --> select for R & L hemisphere stl files and import
    2. If it asks about merges vertices say yes
2. Filters --> Remeshing, Simplification, and Reconstruction --> Quadric Edge Collapse Decimation
    1. Target Number of Faces: 19999 (or smaller)
    2. hit Apply, close window
3. Save new file:
    1. Export Mesh As... save

Slice Your Brain, Make Printer Ready

Motivation: When you 3D print something, you want to make sure it has a stable structure (like won't topple over mid print) and that internally it has a stable structure to keep it from caving in). Programs do this for you. You also need to translate the files into G-Code which is what printers read.

Overview: Use a slicing program to cut the model into thin horizontal segments. A popular open source software is Slic3r. The slicing program generates the tool path based on the characteristics of the model and your extrusion settings. This program will generate the G-Code that drives the printer. You then pass that file to a program that controls and operates the printer, like Printrun/Pronterface. Lulzbot comes with its own program called Cura. Makerbot also has its own.

- **Scripts & Files You Will Need For This Section**
  - Slic3r
  - Printrun/Pronterface
- **Steps**

1. Install or update Slic3r, in terminal:

- $ git clone git://github.com/alexrj/Slic3r
- $ cd Slic3r
- $ sudo perl Build.PL
- $ sudo perl Build.PL --gui

Or just install the dmg here: http://dl.slic3r.org/mac/

1. Open Slic3r, drag the .stl file onto it.

Print Your Brain!

Motivation: The moment you've been waiting for.

- **Scripts & Files You Will Need For This Section**
  - A 3D Printer (one currently by Marianne's desk until July, but not ours so please don't use without asking!) Alternatively, Solid State Depot has printers, you can come by on Tuesdays (open nights). Also, ATLAS has printers.
- **Steps**

1. Launch Cura.
2. Load your model into lulzbot (button on upper left of Cura)
3. You can rescale
4. select material you want to print with
5. hit control
6. set head to 210 degrees (for ABS, PLA, etc) research temp for filaments
7. How to change filament
8. Hit print
9.

----------SCRATCH PAD----------

TOPIC

FIRST PHASE

Motivation:

- **Scripts & Files You Will Need For This Section**
  - xxx.xx
- **Steps**

Object oriented tools:

dat = fmri_data(path)

ttest_img = ttest(dat,.005,'unc'); %this will create a statistic_image

to write out the thresholded image:

dat.fullpath = 'writepath.nii'

write(dat)

%to write thresholded (statistic) image

write(dat,'thresh','fname',path);

%to view

orthviews(dat)

%to replace data with a vector of another image's data

dat2 = fmri_data('new_image.nii')

dat.dat = dat2.dat;

dat.Y %vector of values to predict: length of number of images you have in fmri_data

for svm: -1,1 for member of class or not

for regression: predicted value

for logistic regression: 1, 0

%univariate regression:

r = regress(dat) (should output statistic image class)

r.b %beta value for the regression

%no autocorrelation accounted for (shouldn't use for first-level analysis yet)

r.r %residuals

%also possible to residualize w/ preprocess

PCR: principle components regression

options for regression: lasso, ridge (good for collinearity)

cross-validation: does k folds by default (randomly picks them), to do custom holdout set (inputs vector of subjects) %someone add details about this.

Y= classes

Yfit =

cverr= cross validater error (1-cverr) = accuracy

(if applied to new data, how accurate (not exactly accurate), doesn't correct for overfitting

stats.cvpartition

.teIndx: test and training indicex

.other_output (weights for every test)