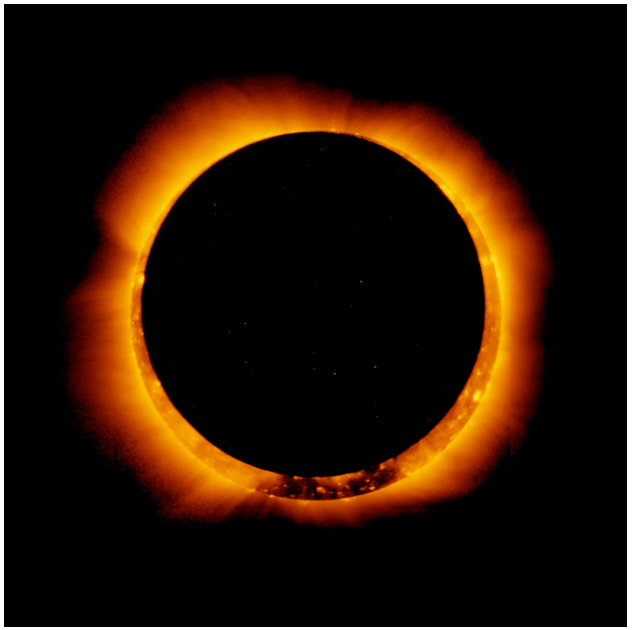
Alignment and Atlases

(volume registration and Talairach transformation)



What does aligning mean and why do we want to do it?

- Alignment means to bring two objects into the same space so that each location within one object corresponds to the same location in the other
- Why?
  - ✧ motion correction across time
  - ✧ align EPI to anatomical data or vice versa – to assign a location with a functional result
  - ✧ compare data from longitudinal studies
  - ✧ compare data from different scanners, sites
  - ✧ compare results with a standard template or atlas for standardized locations and structures

Alignment goals and tools in AFNI

- EPI data across time in a single run or across runs to a base image
  - **3dvolreg** – motion correction (rigid)
- align data to template
  - **3dWarpDrive, @auto_tlrc** – align similar volumes (affine) even across subjects
  - **3dQwarp, auto_warp.py** – align similar volumes nonlinearly to template
- align images across modalities – EPI to anat
  - **3dAllineate** – align different or similar volumes
  - **align_epi_anat.py** – general alignment script to align EPI with anatomical data
- Include motion correction, alignment of EPI to anatomical in fMRI processing pipeline script
  - **afni_proc.py**
- Correct for motion between two volumes by aligning in two dimensions using corresponding slices
  - **@2dwarper.Allin** – non-linear alignment of slices
  - **@2dwarper**, **2dimreg** limit alignment to specific plane

Alignment tools in AFNI (continued)

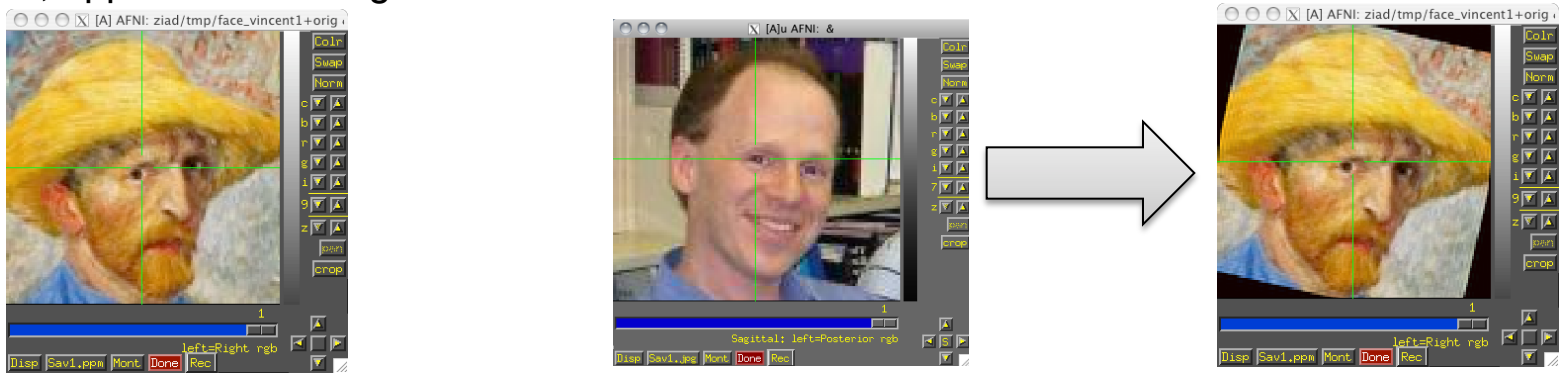
- align partial data to roughly the right part of the brain
  - **Nudge plug-in** - visually align two volumes
- rotate by known amount between volumes
  - **3drotate** – moves (shifts and rotates) volumes
  - **3dWarp** – make oblique, deoblique to match another dataset
- Put centers of data from outside sources in roughly the same space
  - **@Align_Centers, 3dCM** – put centers or centers of mass of dataset in same place
- align specific regions across subjects
  - **3dTagalign, tagset plugin** – place and align volumes using corresponding fiducial marker points
- align one jpeg image to another
  - **imreg** – align two 2D images

Image and Volume Registration with AFNI

- Goal: bring images collected with different methods and at different times into spatial alignment
- Facilitates comparison of data on a voxel-by-voxel basis
  - ✧ Functional time series data will be less contaminated by artifacts due to subject movement
  - ✧ Can compare results across scanning sessions once images are properly registered
  - ✧ Can put volumes in standard space such as the stereotaxic Talairach-Tournoux coordinates
- Most (all?) image registration methods now in use do pair-wise alignment:
  - ✧ Given a base image **J(x)** and target (or source) image **I(x)**, find a geometrical transformation **T[x]** so that **I(T[x])≈J(x)**
  - ✧ **T[x]** will depend on some parameters
    - ➥ Goal is to find the parameters that make the transformed **I** a 'best fit' to **J**
  - ✧ To register an entire time series, each volume $\mathbf{I}_n\mathbf{(x)}$ is aligned to **J(x)** with its own transformation $\mathbf{T}_n\mathbf{[x]}$, for n=0, 1, …
    - ➥ Result is time series $\mathbf{I}_n\mathbf{(T}_n\mathbf{[x])}$ for n=0, 1, …
    - ➥ User must choose base image **J(x)**

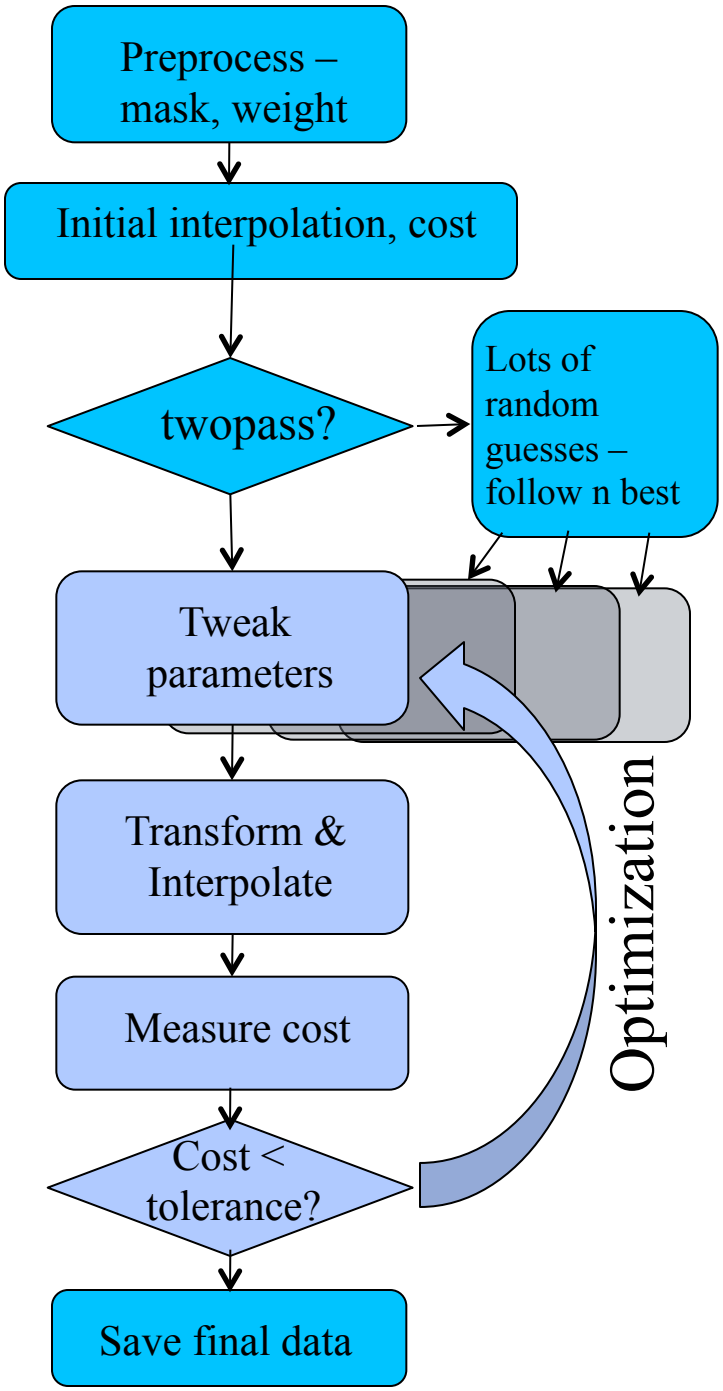
- Most image registration methods make 3 algorithmic choices:
  - ✧ How to measure mismatch **E** (for error) between **I(T[x])** and **J(x)**?
    - ➥ **Or** … How to measure goodness of fit between **I(T[x])** and **J(x)**?
      - ➪ **E**(parameters) ≡ **–Goodness**(parameters)
  - ✧ How to adjust parameters of **T[x]** to minimize **E**?
  - ✧ How to interpolate **I(T[x])** to the **J(x)** grid?
    - ➥ So we can compare voxel intensities directly
- The input volume is transformed by the optimal **T[x]** and a record of the transform is kept in the header of the output.
- Finding the transform to minimize **E** is the bulk of the registration work. Applying the transform is easy and is done on the fly in many cases.
- If data starts off far from each other, may add a coarse pass (twopass) step
  - ✧ guess a lot among all the parameters (rotations, shifts, ...), measure cost
  - ✧ best guesses, tweak the parameters (optimize) and measure again

Now, applications of alignment…

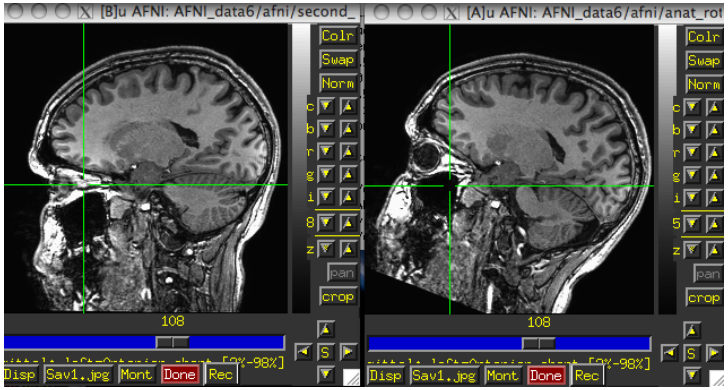


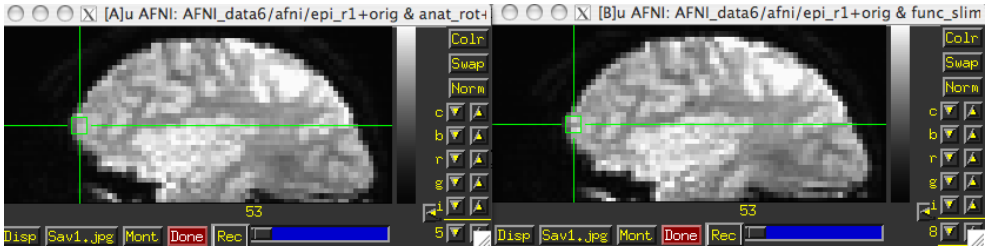
Alignment process - overview

- Preprocess – mask data, weight data
- If far off, take some random guesses (-twopass)
- Optimize parameters on initial or best sets (6,12,39,1000's)
  - Use new parameters to transform input
    - Interpolate onto base data's grid
  - Measure alignment error with cost functional
    - Less than minimum error - finished
    - Better - keep adjusting with same direction
    - Worse – try other parameters
- Create final output by interpolating onto output grid
  - save datasets, transform parameters



Within Modality Registration





- AFNI **3dvolreg** and **3dWarpDrive** programs match images by grayscale (intensity) values
  - **E** = (weighted) sum of squares differences = $\sum_x w(x) \cdot \{I(T[x]) - J(x)\}^2$
    - Only useful for registering 'like images':
      - Good for SPGR↔SPGR, EPI↔EPI, but **not** good for SPGR↔EPI
  - Several interpolation methods :
    - Fourier, linear, cubic, quintic, and heptic polynomials
  - **3dvolreg** is designed to run VERY fast for EPI↔EPI registration with small movements — good for FMRI purposes but restricted to 6-parameter rigid-body transformations.
  - **3dWarpDrive** is slower, but it allows for up to 12 parameters affine transformation. This corrects for scaling and shearing differences in addition to the rigid body transformations.

- AFNI program **3dvolreg** is for aligning 3D volumes by rigid movements
  - **T[x]** has 6 parameters:
    - Shifts along *x*-, *y*-, and *z*-axes; Rotations about *x*-, *y*-, and *z*-axes
  - Generically useful for intra- and inter-session alignment
  - Motions that occur within a single TR (2-3 s) cannot be corrected this way, since method assumes rigid movement of the entire volume
- AFNI program **3dWarpDrive** is for aligning 3D volumes by affine transformations
  - **T[x]** has up to 12 parameters:
    - Same as **3dvolreg** plus 3 Scales and 3 Shears along *x*-, *y*-, and *z*-axes
  - Generically useful for intra- and inter-session alignment
  - Generically useful for intra- and inter-subject alignment
  - AFNI program 2dImReg is for aligning 2D slices
  - **T[x]** has 3 parameters for each slice in volume:
    - Shift along *x*-, *y*-axes; Rotation about *z*-axis
    - No out of slice plane shifts or rotations!
  - Useful for **sagittal** EPI scans where dominant subject movement is 'nodding' motion that may be faster than TR
  - It is possible and sometimes even useful to run **2dImReg** to clean up sagittal nodding motion, followed by **3dvolreg** to deal with out-of-slice motion

- Intra-session registration example:

```
3dvolreg -base 4 -heptic -zpad 4    \
         -prefix fred1_epi_vr       \
         -1Dfile fred1_vr_dfile.1D  \
         fred1_epi+orig
```

Input dataset name

  - **-base 4** ⇒ Selects sub-brick #4 of dataset **fred1_epi+orig** as base image **J(x)**
  - **-heptic** ⇒ Use 7th order polynomial interpolation
  - **-zpad 4** ⇒ Pad each target image, **I(x)**, with layers of zero voxels 4 deep on each face prior to shift/rotation, then strip them off afterwards (before output)
    - Zero padding is particularly desirable for **-Fourier** interpolation
    - also good for large rotations, some data may get 'lost' if no zero padding
  - **-prefix fred1_epi_vr** ⇒ Save output dataset into a new dataset with the given prefix name (e.g., **fred1_epi_vr+orig**)
  - **-1Dfile fred1_vr_dfile.1D** ⇒ Save estimated movement parameters into a 1D (i.e., text) file with the given name
    - Movement parameters can be plotted with the **1dplot** command and used later....

Try this (in AFNI_data6/afni) :

3dvolreg -base 3 -cubic -prefix epi_r1_vrt -1Dfile vr_dfile.1D epi_r1+orig

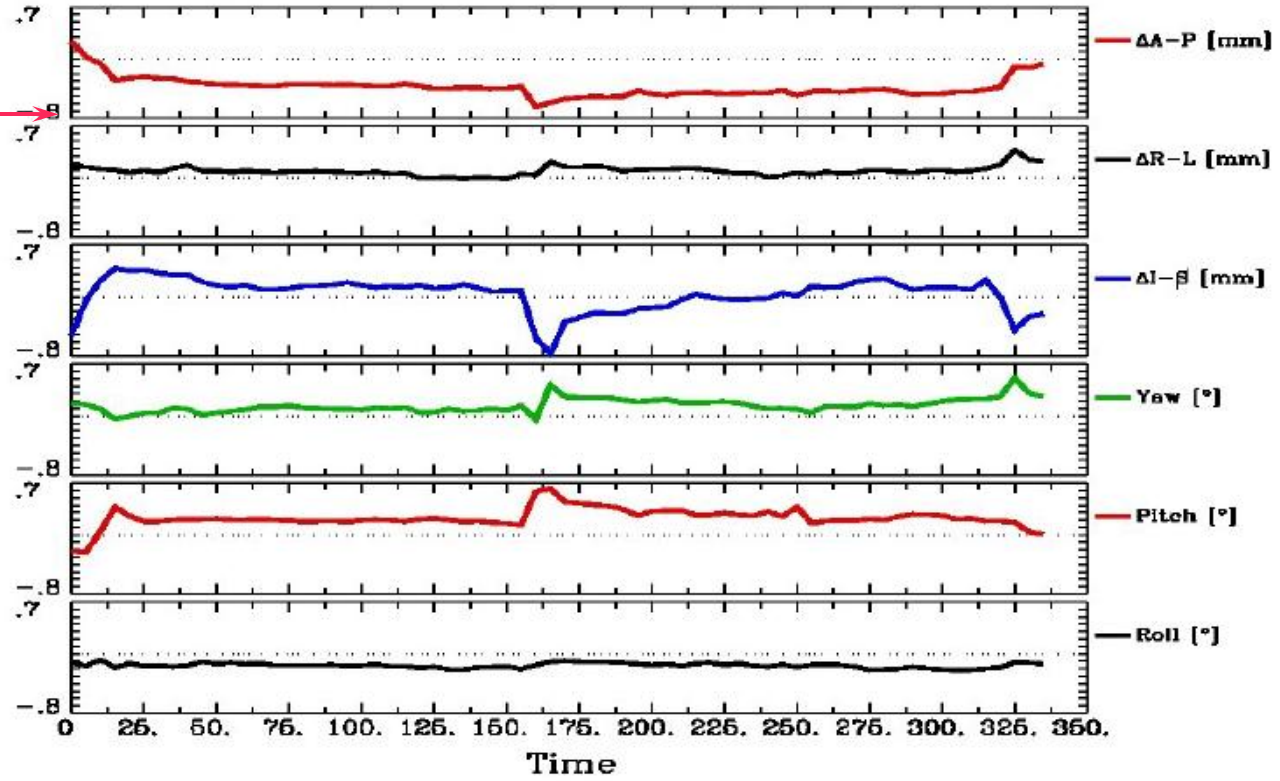
1dplot -volreg vr_dfile.1D &

- Can now register second dataset from same session:

```
3dvolreg -base 'fred1_epi+orig[4]' -heptic -zpad 4      \
         -prefix fred2_epi_vr -1Dfile fred2_vr_dfile.1D \
         fred2_epi+orig
```

  - Note base is from different dataset (**fred1_epi+orig**) than input (**fred2_epi+orig**)
    - Aligning all EPI volumes from session to EPI closest in time to SPGR (if not aligning to anatomical)

```
1dplot -volreg -dx 5 -xlabel Time fred2_vr_dfile.1D
```



About A-P

About R-L
Typical

About I-S

  - Note motion peaks at time ≈ 160s: subject jerked head up at that time

"Fixing Motion"

- Motion occurs over slices and not volumes and moves data off original grid
- "Regressing out" – motion parameters, derivatives, displacement, euclidean norm of derivatives (summary parameters)
- Censoring ("scrubbing", "sweeping under the mat",...)
- Experimental design – kids, monkeys, juice, talking, waving hands wildly....
- Interpreting results – differences in motion between groups or something physiological
- Notice – activation following high contrast borders, "blinds" effects

Try this (in AFNI_data6/afni) :

1d_tool.py –infile vr_dfile.1D -set_nruns 1 -censor_motion 1 ett

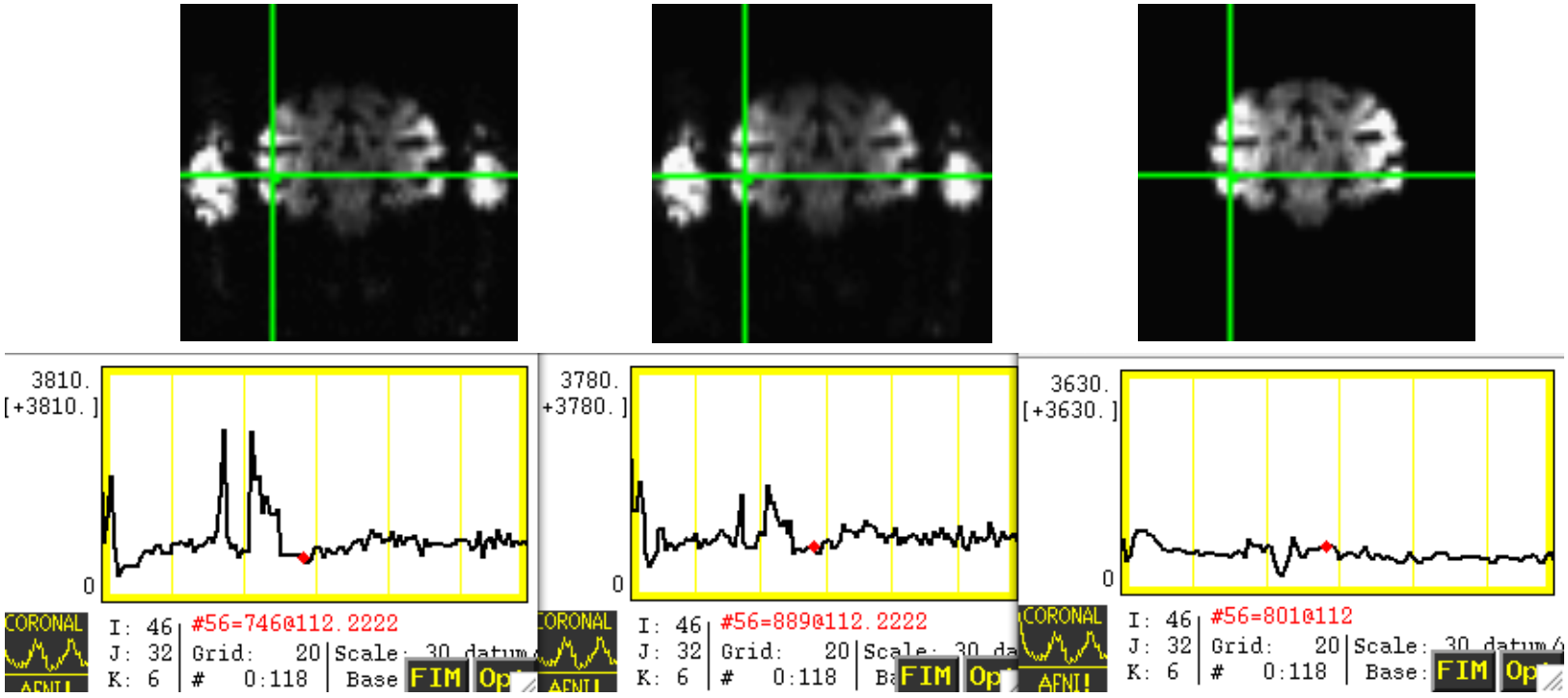
1dplot ett_enorm.1D &

cat ett_CENSORTR.txt

Motion correction – caveats

- Motion is usually not completely correctable, so set motion parameters as regressors of no interest. Interpolation generally blurs data and depends on method and grid/resolution of EPI.
- Check in the AFNI GUI to be sure the data is not bouncing around after correction
- Example – Monkey sips juice at stimulus time, and large jaw muscles move. If the muscles are not masked, then motion correction may track muscles rather than brain.

original | 3dvolreg | automask, 3dvolreg



Cross Modality Registration

- **3dAllineate** can be used to align images from different methods
  - For example, to align EPI data to SPGR / MPRAGE:
    - Run **3dSkullStrip** on the SPGR dataset so that it will be more like the EPI dataset (which will have the skull fat suppressed)
    - Use **3dAllineate** to align the EPI volume(s) to the skull-stripped SPGR volume
    - Program works well if the EPI volume covers most of the brain
  - Allows more general spatial transformations – affine, bilinear, non-linear (polynomial warping)
- **3dAllineate** has several different "cost" functions (**E**) available
  - **leastsq** = Least Squares (**3dvolreg, 3dWarpDrive**)
  - **mutualinfo** = Mutual Information
  - **norm_mutualinfo** = Normalized Mutual Information
  - **hellinger** = Hellinger Metric [the **default** cost function]
  - **corrratio_mul** = Correlation ratio (symmetrized by multiplication)
  - **corratio_add** = Correlation ratio (symmetrized by addition)
  - **corratio_uns** = Correlation ratio (unsymmetric)
  - **lpc** = Local Pearson Correlation (negative)
  - **lpa** = Local Pearson Correlation (absolute value)

align_epi_anat.py

- Goal: Want to align anat and EPI (anat to EPI or EPI to anat or dset1to2 or dset2to1)

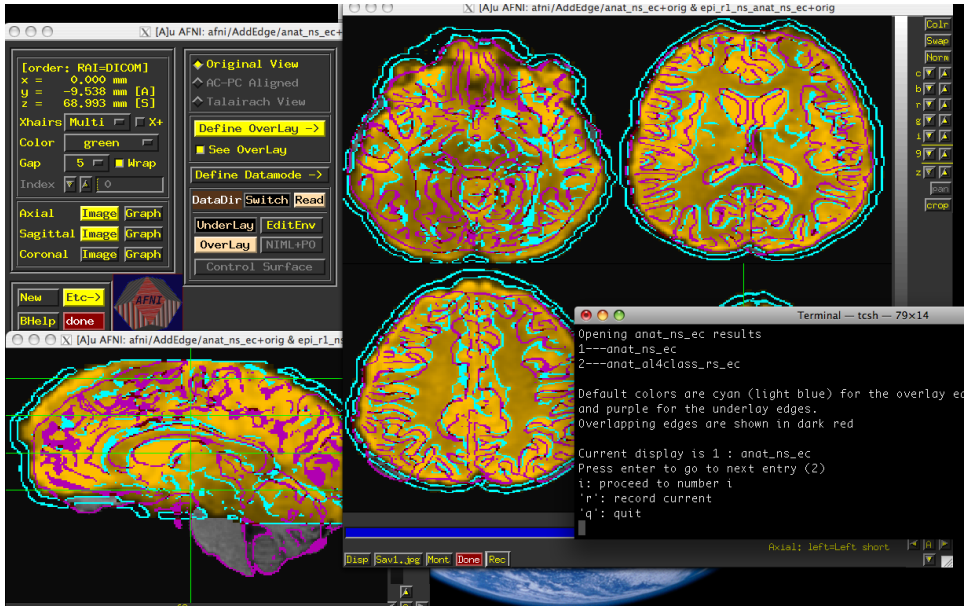
LPC method – Local Pearson Correlation to match dark CSF in anatomical data with bright CSF in EPI data.

- align_epi_anat.py script – preprocessing and calls 3dAllineate for alignment
- @AddEdge – for visualization

- Simple Example:

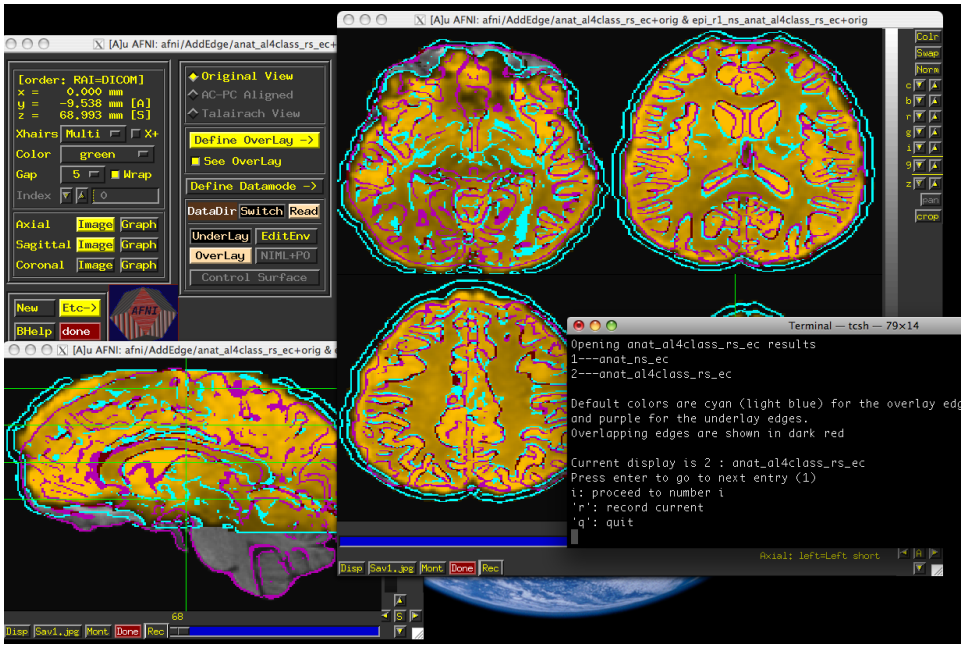
```
align_epi_anat.py -anat anat+orig \
                  -epi epi_r1+orig \
                  -AddEdge -epi_base 0 -suffix _al4class
cd AddEdge
afni -niml -yesplugouts &
@AddEdge
```

Combines deoblique, motion correction, alignment and talairach transformations into a single transformation. Also performs slice timing correction and applies transformations to “child” datasets.



@AddEdge display

Before



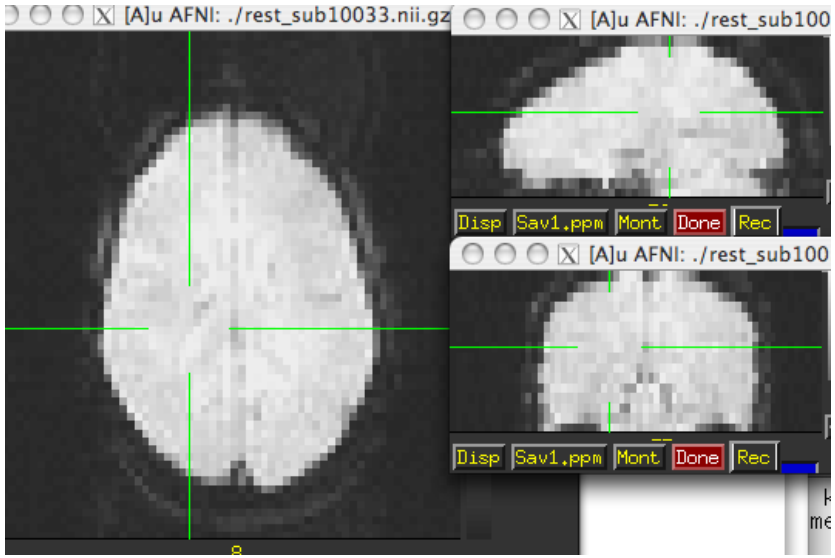
After

Alignment Visualization in AFNI

- Graph and image – travel through time for motion correction or for a thousand datasets in a row.
- Multiple controllers and crosshairs – up to ten datasets at a time, quick and rough.
- Overlay display – opacity control, thresholding. A single pair – good for different or similar datasets.
- Overlay toggle, Underlay toggle – wiggle, good but a little tricky ('o' and 'u' keys in image viewer)
- Sliding Overlay ('4'/'5' keys), Checkerboard Underlay – two similar datasets in underlay but must be virtually identical. Good for comparing two processing methods
- Edge display for underlay – effective pairwise comparison for quick fine structure display and comparison with overlay dataset with opacity. One dataset should have reliable structure and contrast. Now with 'e' toggle.
- @AddEdge – single or dual edges with good contrast for pairwise comparison.

Alignment strategies with align_epi_anat.py

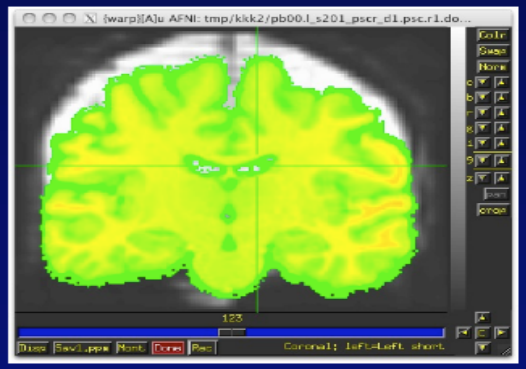
- Defaults work usually (>90% - FCON1000)
- Problems:
  - ✧ Far off start – “-giant_move”, “-big_move”
  - ✧ Poor contrast – “-cost lpa”, “-cost nmi”, “-cost lpc+ZZ”
  - ✧ Poor non-uniformity – “-edge”, “-cost lpa”
  - ✧ stroke/MS lesions, tumors, monkeys, rats, multi-modality (CT/PET/DTI/...), something else? – see us, post message

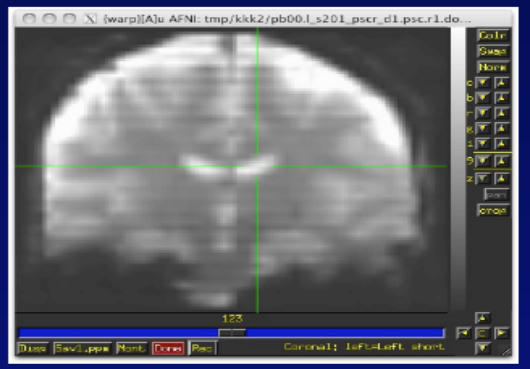


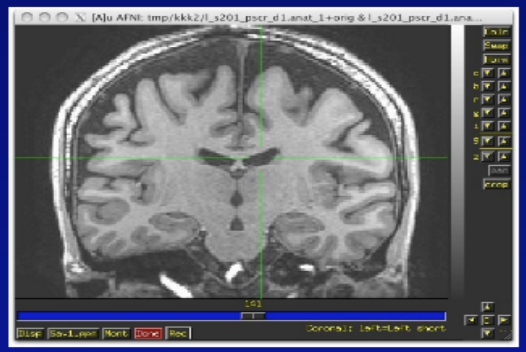
Real and Imaginary Problems

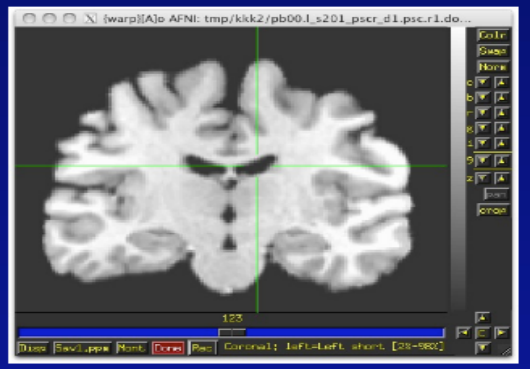
Saved by the contrast

- However, bias may result in erroneous group differences



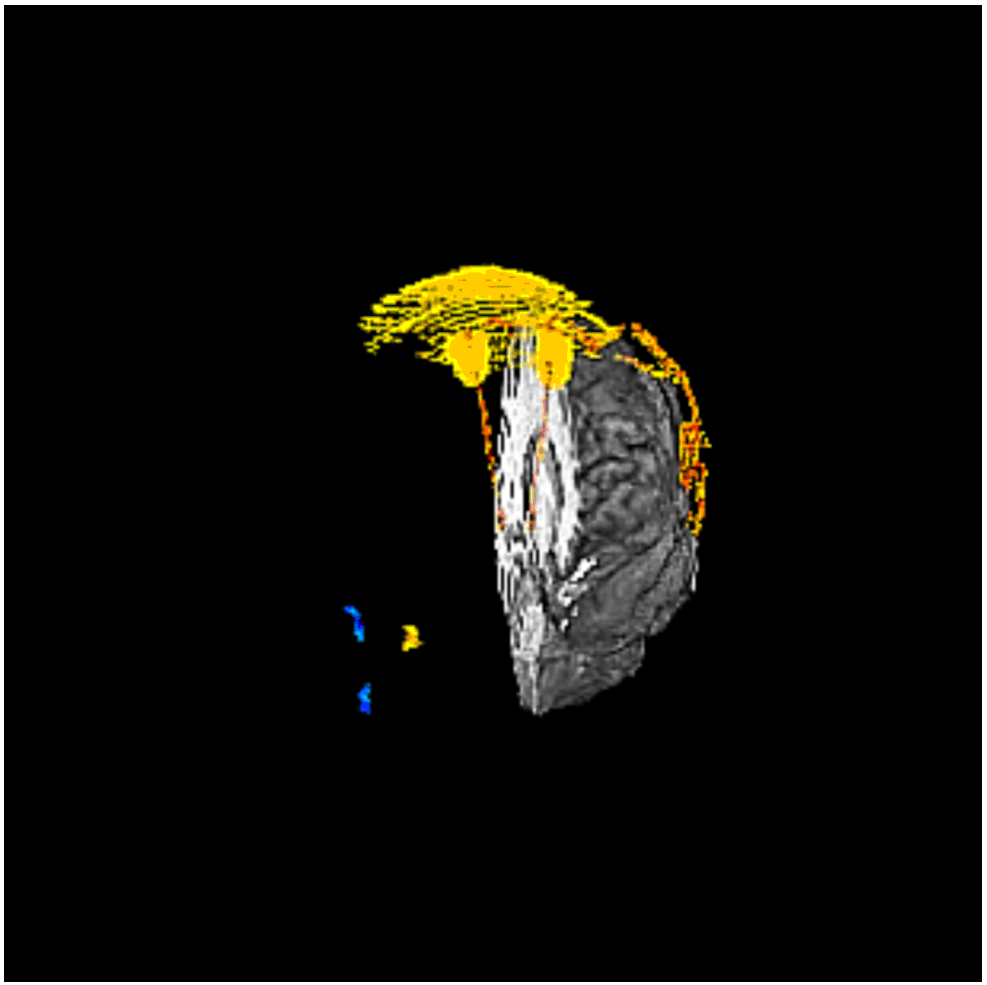




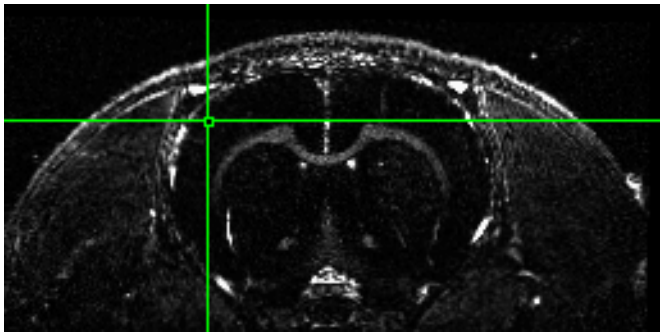


Alignment –special cases

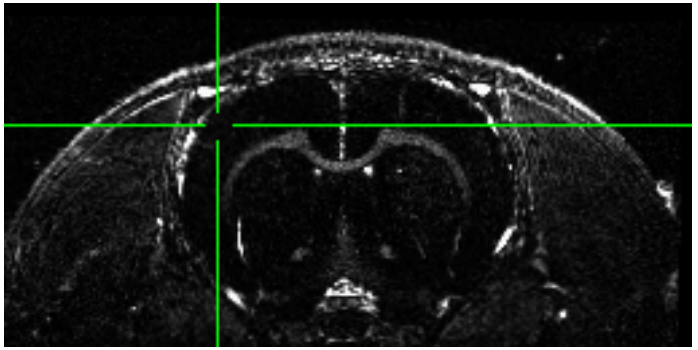
- DBS – align CT with electrodes to pre-surgical MRI



Rat Brains



Alignment of 12 hour Manganese enhanced MRI scan (MEMRI) to start



```
#!/bin/tcsh
# align_times.csh
set basedset = 14_pre+orig
foreach timedset ( 14_*hr+orig.HEAD)
  align_epi_anat.py -prep_off -anat $timedset -epi $basedset \
  -epi_base 0 -anat_has_skull no -epi_strip None -suffix _edge2prep \
  -cost lpa -overwrite -edge -rat_align
end
3dTcat -prefix 14_timealigned_edge 14_pre+orig. 14*edge2prep+orig.HEAD
```

Data from Der-Yow Chen (NINDS)

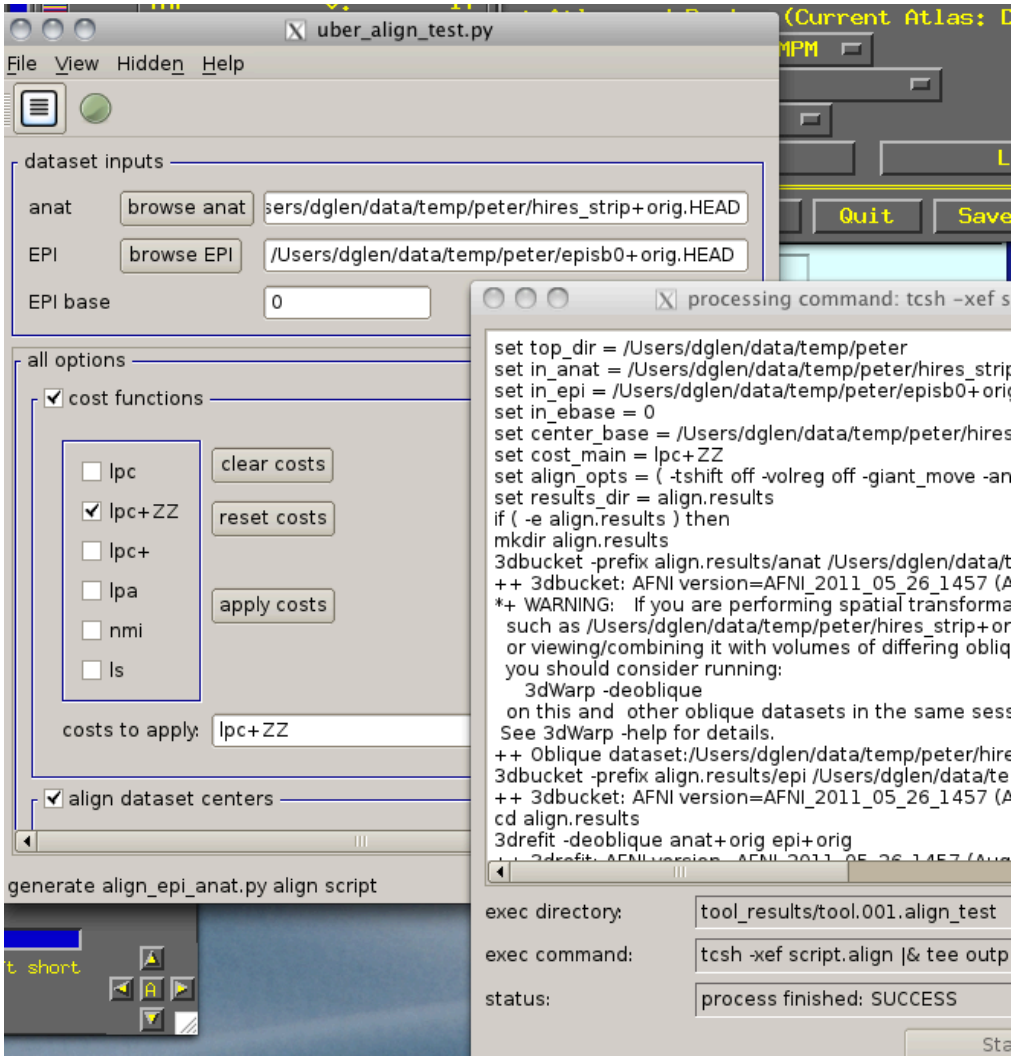
uber_align_test.py

select input data

set options

create script

run script



afni_proc.py – alignment handling

- Single script to do all the processing of a typical fMRI pipeline including motion correction (3dvolreg), alignment (align_epi_anat.py)
- combines transformations when possible
- from example 6 in afni_proc.py's prodigious help:

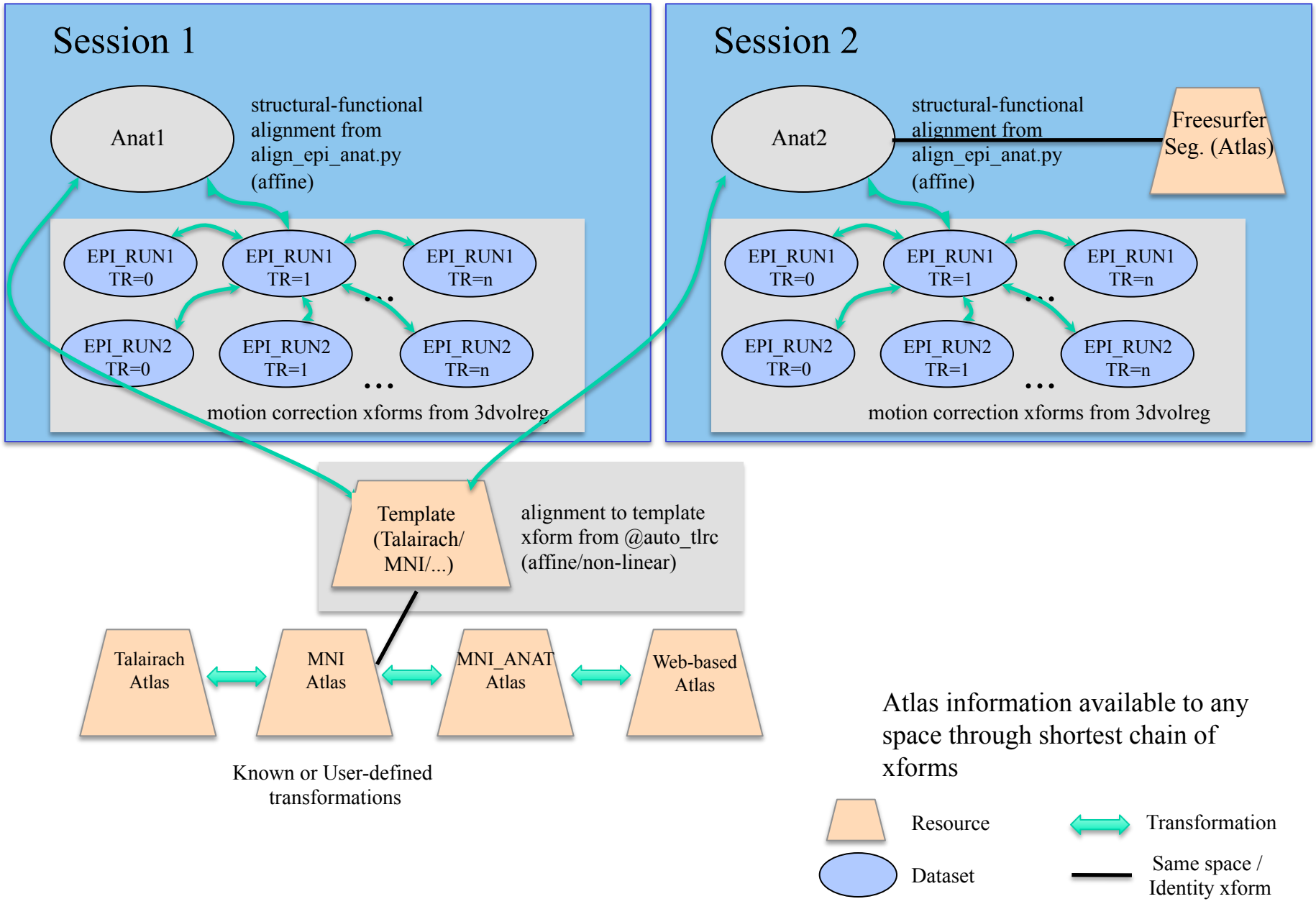
```
afni_proc.py -subj_id sb23.e6.align    \
        -dsets sb23/epi_r??+orig.HEAD  \
        -do_block align tlrc           \
        -copy_anat sb23/sb23_mpra+orig \
        -tcat_remove_first_trs 3       \
        -volreg_align_to last          \
        -volreg_align_e2a              \
        -volreg_tlrc_warp              \
                ...
```

To process in orig space, remove -volreg_tlrc_warp.

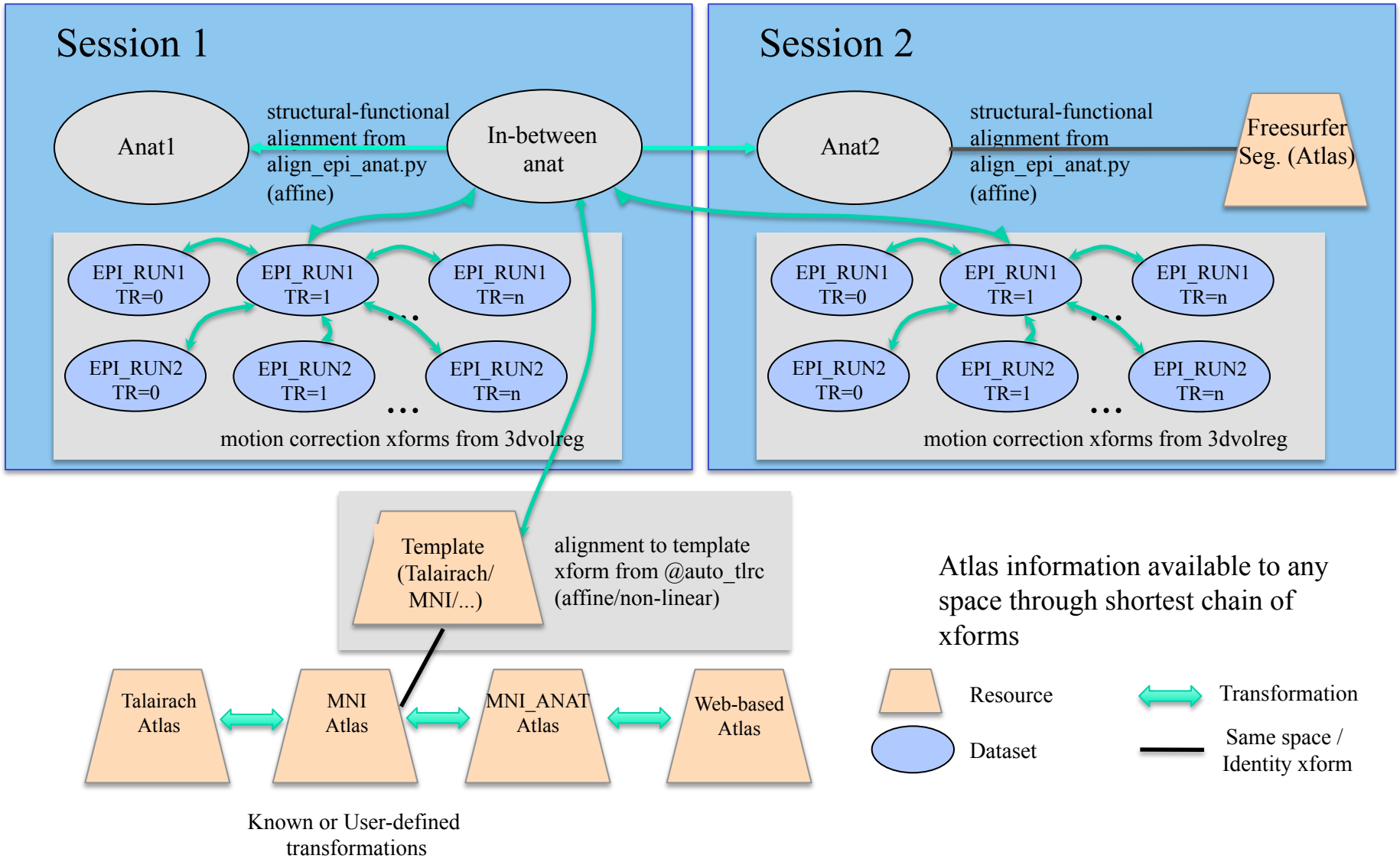
To apply manual tlrc transformation, use -volreg_tlrc_adwarp.

To process as anat aligned to EPI, remove -volreg_align_e2a.

Transformation Chains Example



Alignment across two sessions



http://afni.nimh.nih.gov/sscc/dglen/alignmentacross2sessions

Nonlinear Warping

3dQwarp

- $I_{new}(\mathbf{x}) = I_{old}(\ \mathbf{W}(\mathbf{x})\ )$
  - ✧ where $\mathbf{W}(\mathbf{x})$ depends on a ***lot*** of parameters (1000-50000+)
  - ✧ Method: Incremental transformation with Hermite cubic polynomials over finer and finer 3D patches. Output is both aligned dataset and the warp and the inverse warp deformations in Dx,Dy,Dz
  - ⋆ Better alignment of anatomical volumes to template space
    - ➥ Then carry the functional results to template space for better group analyses***?***
    - ➥ As an aid to brain segmentation and atlas-ing accuracy***?***

★ Pre- and post-surgical alignment, EPI distortion correction

- Related programs and scripts:
  - ✧ ***3dNwarpApply, 3dNwarpCat, 3dNwarpCalc, 3dNwarpAdjust, 3dNwarpFuncs***
  - ✧ ***auto_warp.py, afni_proc.py***

ATLASES

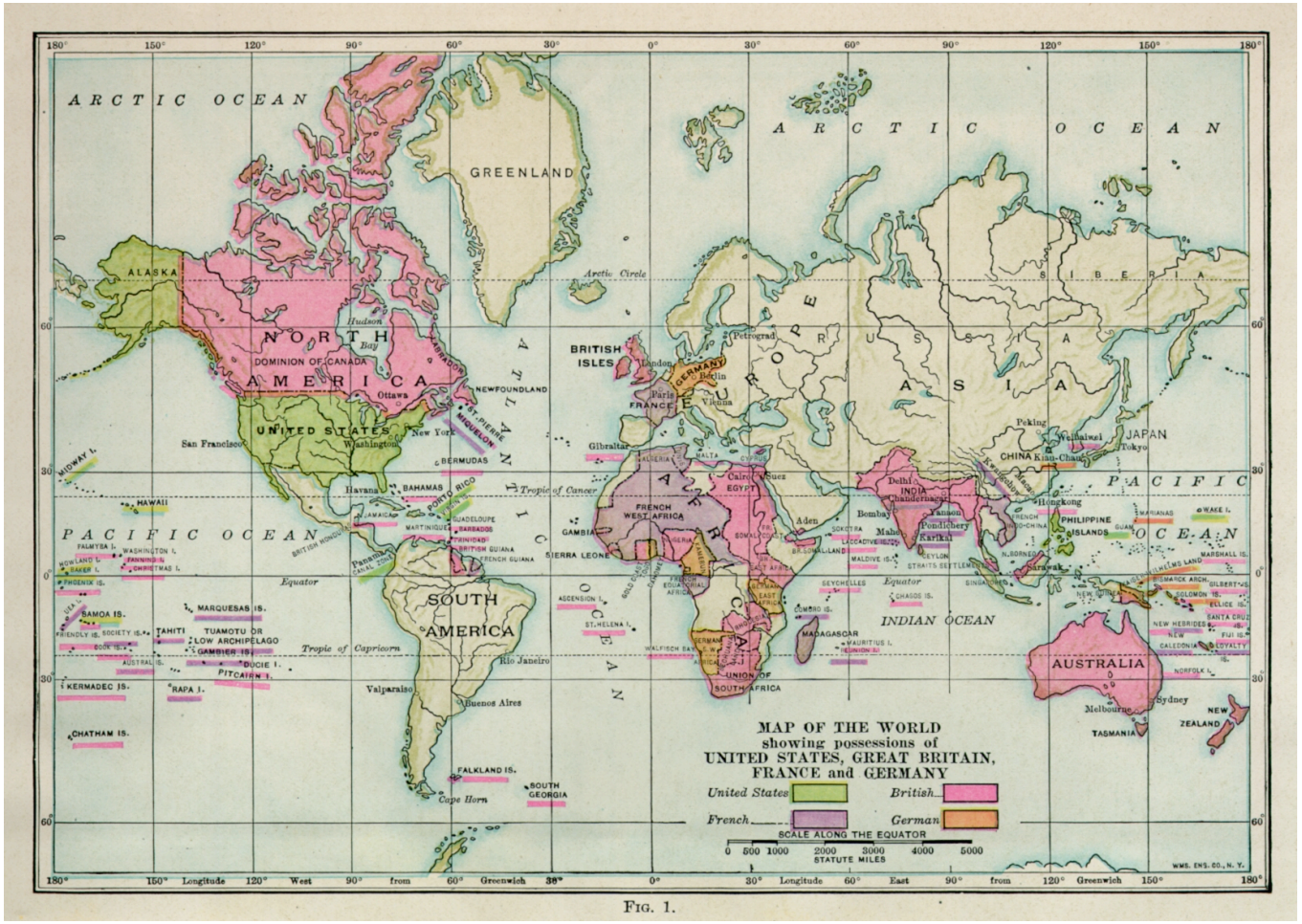
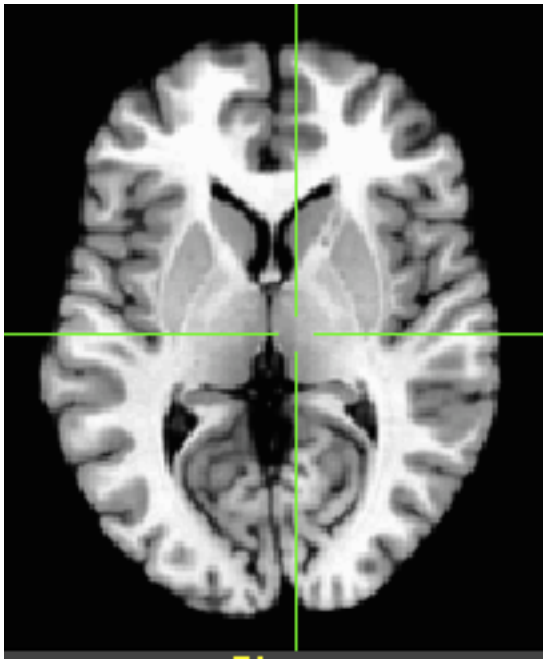


FIG. 1.

ATLAS DEFINITIONS

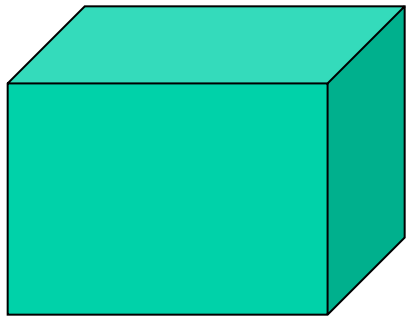
Template - a reference dataset used for matching shapes.
Examples: TT_N27+tlrc, MNI_EPI+tlrc, TT_ICBM452+tlrc.



TT_N27+tlrc

ATLAS DEFINITIONS

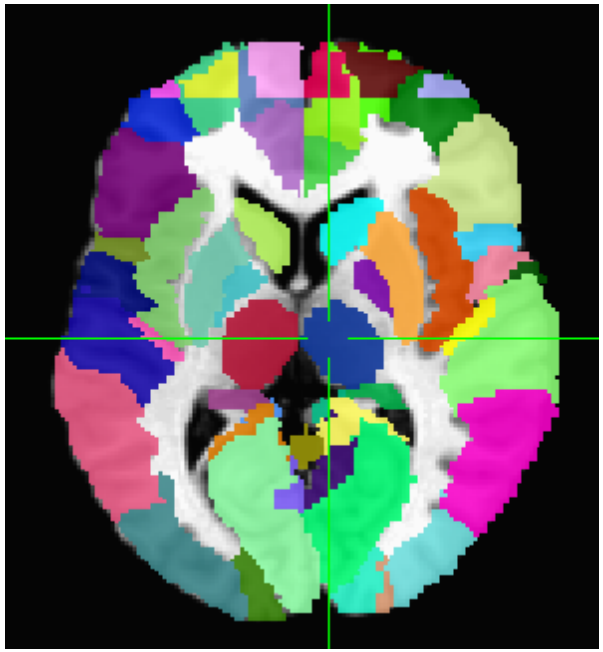
Template Space - x,y,z coordinate system shared by many datasets (the basic shoebox) Examples: TLRC (Talairach-Tourneaux), MNI, MNI_ANAT, ORIG.



ATLAS DEFINITIONS

Atlas - segmentation info.

Examples: TTatlas+tlrc, TT_N27_EZ_ML+tlrc, roidset+orig.



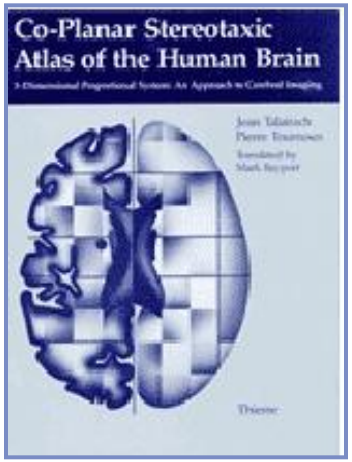
TT_N27_EZ_ML+tlrc

Registration To Standard Spaces

Transforming Datasets to Talairach-Tournoux Coordinates

- The original purpose of AFNI (*circa* 1994 A.D.) was to perform the transformation of datasets to Talairach-Tournoux (stereotaxic) coordinates
- The transformation can be manual, or automatic
- In manual mode, you must mark various anatomical locations, defined in

  Jean Talairach and Pierre Tournoux
  "Co-Planar Stereotaxic Atlas of the Human Brain"
  Thieme Medical Publishers, New York, 1988



  - ✧ Marking is best done on a high-resolution T1-weighted structural MRI volume
- In automatic mode, you need to choose a template to which your data are aligned. Different templates are made available with AFNI's distribution. You can also use your own templates.
- Transformation carries over to all other (follower) datasets in the same directory
  - ✧ This is where the importance of getting the relative spatial placement of datasets done correctly in **`to3d`** really matters
  - ✧ You can then write follower datasets, typically functional or EPI timeseries, to disk in Talairach coordinates
    - ➥ Purpose: voxel-wise comparison with other subjects
    - ➥ May want to blur volumes a little before comparisons, to allow for residual anatomic variability: AFNI programs **`3dmerge`** **or** **`3dBlurToFWHM`**

- Hidden in GUI - right click on “DataDir” or set AFNI_ENABLE_MARKERS to YES in .AFNIRC
- Manual Transformation proceeds in two stages:
  1. Alignment of AC-PC and I-S axes (to **+acpc** coordinates)
  2. Scaling to Talairach-Tournoux Atlas brain size (to **+tlrc** coordinates)

- Stage 1: Alignment to **+acpc** coordinates:
  - ✧ Anterior commissure (AC) and posterior commissure (PC) are aligned to be the y-axis
  - ✧ The longitudinal (inter-hemispheric or mid-sagittal) fissure is aligned to be the yz-plane, thus defining the z-axis
  - ✧ The axis perpendicular to these is the x-axis (right-left)
  - ✧ Five markers that you must place using the **[Define Markers]** control panel:

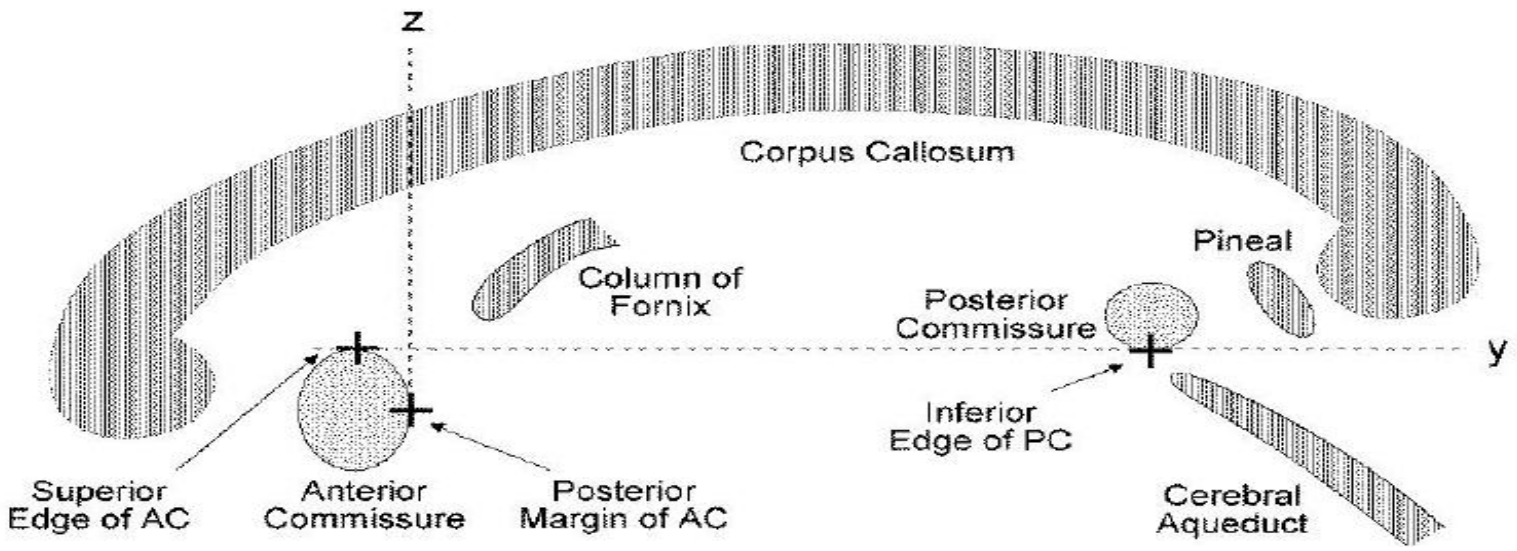
    AC superior edge = top middle of anterior commissure

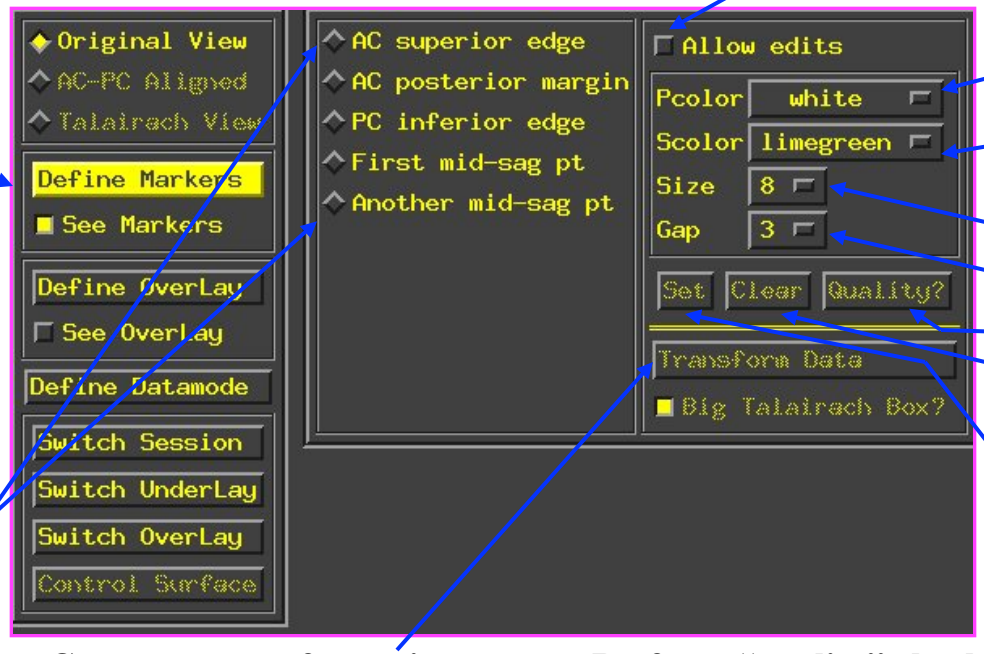
    AC posterior margin = rear middle of anterior commissure

    PC inferior edge = bottom middle of posterior commissure

    First mid-sag point = some point in the mid-sagittal plane

    Another mid-sag point = some other point in the mid-sagittal plane
  - ✧ This procedure tries to follow the Atlas as precisely as possible
    - ➥ Even at the cost of confusion to the user (e.g., you)





Press this IN to create or change markers

Click Define Markers to open the "markers" panel

Color of "primary" (selected) marker

Color of "secondary" (not selected) markers

Size of markers (pixels)

Size of gap in markers

Clear (unset) primary marker

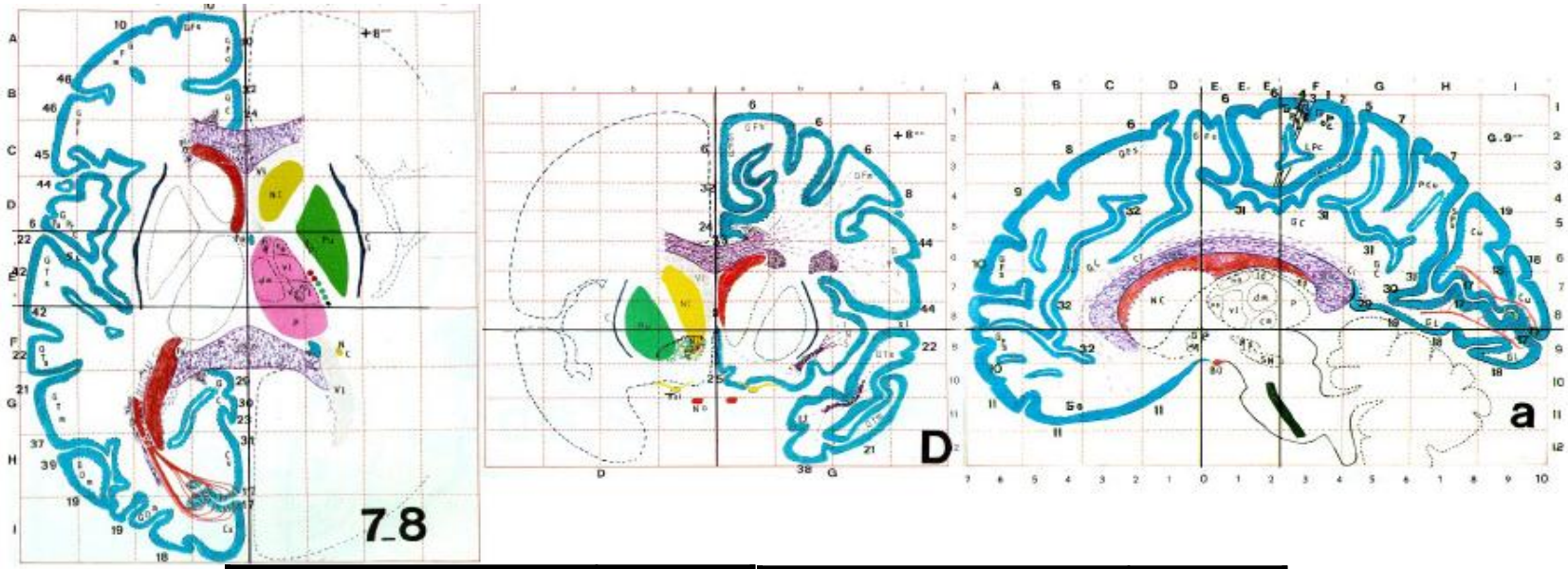
Set primary marker to current focus location

Select which marker you are editing

Carry out transformation to +acpc coordinates

Perform "quality" check on markers (after all 5 are set)

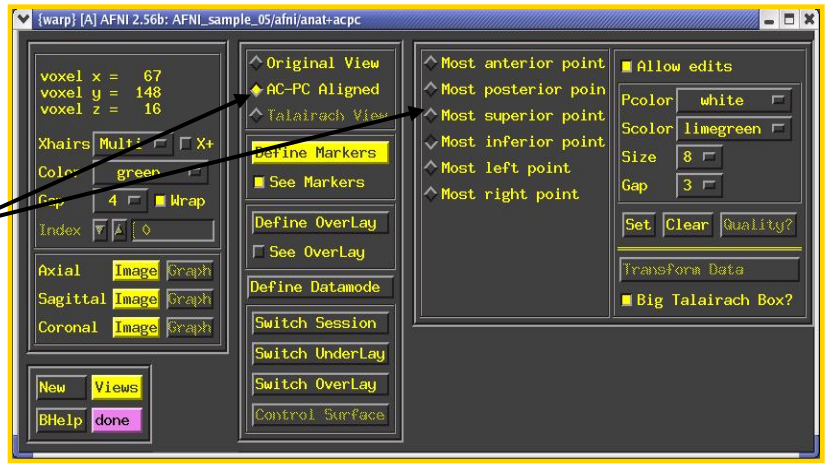
- **Stage 2: Scaling to Talairach-Tournoux (+tlrc) coordinates:**
  - Once the AC-PC landmarks are set and we are in ACPC view, we now stretch/shrink the brain to fit the Talairach-Tournoux Atlas brain size (sample TT Atlas pages shown below, just for fun)



Most anterior to AC	70 mm		
AC to PC	23 mm		
PC to most posterior	79 mm	Length of cerebrum	172mm
Most inferior to AC	42 mm		
AC to most superior	74 mm	Height of cerebrum	116mm
AC to left (or right)	68 mm	Width of cerebrum	136mm

- **Selecting the Talairach-Tournoux markers for the bounding box:**
  - There are 12 sub-regions to be scaled (3 A-P x 2 I-S x 2 L-R)
  - To enable this, the transformed `+acpc` dataset gets its own set of markers
    - Click on the **[AC-PC Aligned]** button to view our volume in ac-pc coordinates
    - Select the **[Define Markers]** control panel
  - A new set of six Talairach markers will appear and the user now sets the bounding box markers (see Appendix C for details):

Talairach markers appear only when the AC-PC view is highlighted



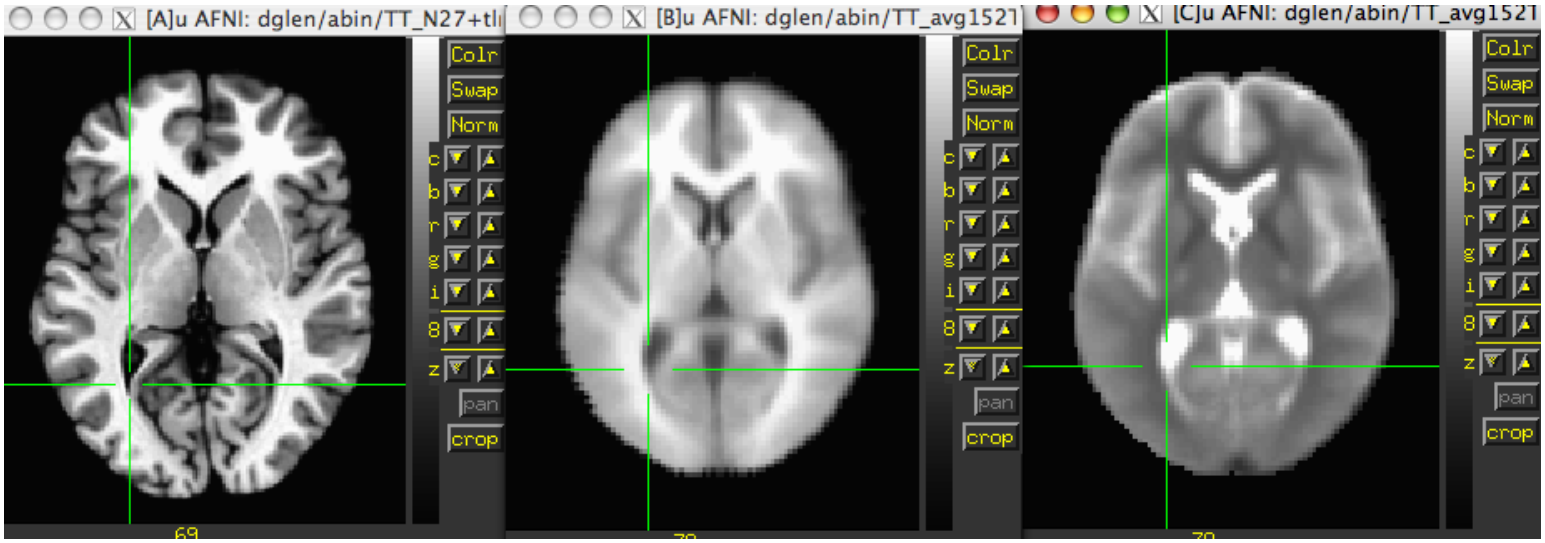
  - Once all the markers are set, and the quality tests passed. Pressing **[Transform Data]** will write new ***header*** containing the Talairach transformations (see Appendix C for details)
    - Recall: With AFNI, spatial transformations are stored in the header of the output

Automatic Talairach Transformation with @auto_tlrc

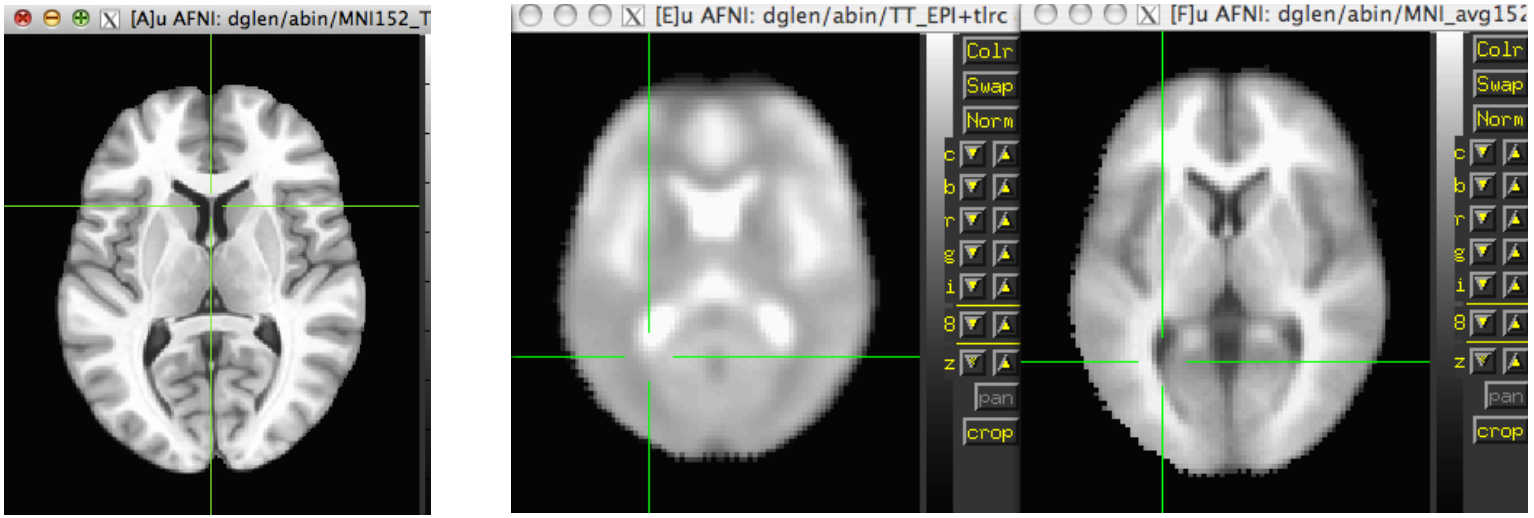
- Is manual selection of AC-PC and Talairach markers bringing you down? You can now perform a TLRC transform ***automatically*** using an AFNI script called **`@auto_tlrc`**.
  - ✧ Differences from Manual Transformation:
    - ➥ Instead of setting ac-pc landmarks and volume boundaries by hand, the anatomical volume is warped (using 12-parameter affine transform) to a template volume in TLRC space.
    - ➥ Anterior Commisure (AC) center no longer at 0,0,0 and size of brain box is that of the template you use.
      - ➪ For various reasons, some good and some bad, templates adopted by the neuroimaging community are not all of the same size. Be mindful when using various atlases or comparing standard-space coordinates.
    - ➥ You, the user, can choose from various templates for reference but be consistent in your group analysis.
    - ➥ Easy, automatic. Just check final results to make sure nothing went seriously awry. AFNI is perfect but your data is not.

- Templates in **`@auto_tlrc`** that the user can choose from:
  - **`TT_N27+tlrc`**:
    - AKA “Colin brain”. One subject (Colin) scanned 27 times and averaged. (www.loni.ucla.edu, www.bic.mni.mcgill.ca)
    - Has a full set of FreeSurfer (surfer.nmr.mgh.harvard.edu) surface models that can be used in SUMA (l*ink*).
    - Is the template for cytoarchitectonic atlases (www.fz-juelich.de/ime/spm_anatomy_toolbox)
      - For improved alignment with cytoarchitectonic atlases, I recommend using the TT_N27 template because the atlases were created for it. In the future, we might provide atlases registered to other templates.
  - **`TT_icbm452+tlrc`**:
    - International Consortium for Brain Mapping template, average volume of 452 normal brains. (www.loni.ucla.edu, www.bic.mni.mcgill.ca)
  - **`TT_avg152T1+tlrc`**:
    - Montreal Neurological Institute (www.bic.mni.mcgill.ca) template, average volume of 152 normal brains.
  - **`TT_EPI+tlrc:`**
    - EPI template from spm2, masked as TT_avg152T1. TT_avg152 and TT_EPI volumes are based on those in SPM's distribution. (www.fil.ion.ucl.ac.uk/spm/)
  - **`MNI152_T1_2009c+tlrc:`**
    - MNI's 152 nonlinear asymmetric template

Templates included with AFNI



TT_N27 TT_avg152T1 TT_avg152T2



MNI152_T1_2009c TT_EPI MNI_avg152T1

Steps performed by @auto_tlrc

- For warping a volume to a template (Usage mode 1):
  1. Pad the input data set to avoid clipping errors from shifts and rotations
  2. Strip skull (if needed)
  3. Resample to resolution and size of TLRC template
  4. Perform 12-parameter affine registration using **3dWarpDrive**

  Many more steps are performed in actuality, to fix up various pesky little artifacts. Read the script if you are interested.

  - ✧ Typically this steps involves a high-res anatomical to an anatomical template
    - ➥ Example: @auto_tlrc -base TT_N27+tlrc. -input anat+orig. -suffix NONE
  - ✧ One could also warp an EPI volume to an EPI template.
    - ➥ If you are using an EPI time series as input. You must choose one sub-brick to input. The script will make a copy of that sub-brick and will create a warped version of that copy.

Applying a transform to follower datasets

- Say we have a collection of datasets that are in alignment with each other. One of these datasets is aligned to a template and the same transform is now to be applied to the other *follower* datasets
- For Talairach transforms there are a few methods:
  - ✧ Method 1: Manually using the AFNI interface (see Appendix C)
  - ✧ Method 2: With program adwarp

    `adwarp -apar anat+tlrc  -dpar func+orig`

    - ➥ The result will be: **`func+tlrc.HEAD`** and **`func+tlrc.BRIK`**
  - ✧ Method 3: With @auto_tlrc script in mode 2
    - ➥ ONLY when -apar dataset was created by @auto_tlrc
    - ➥ @auto_tlrc -apar SubjectHighRes+tlrc. \
    - ➥ -input Subject_EPI+orig. -dxyz 3
    - ➥ (the output is named Subject_EPI_at+TLRC, by default)
- Why bother saving transformed datasets to disk anyway?
  - ✧ Datasets without `.BRIK` files are of limited use, only for display of slice images

@auto_tlrc Example

- Transforming the high-resolution anatomical:
  - (If you are also trying the manual transform on workshop data, start with a fresh directory with no +tlrc datasets)

```
@auto_tlrc                    \
   -base TT_N27+tlrc          \
   -suffix NONE               \
   -input anat+orig
```

Output:
`anat+tlrc`

- Transforming the function ("follower datasets"), setting the resolution at 2 mm:

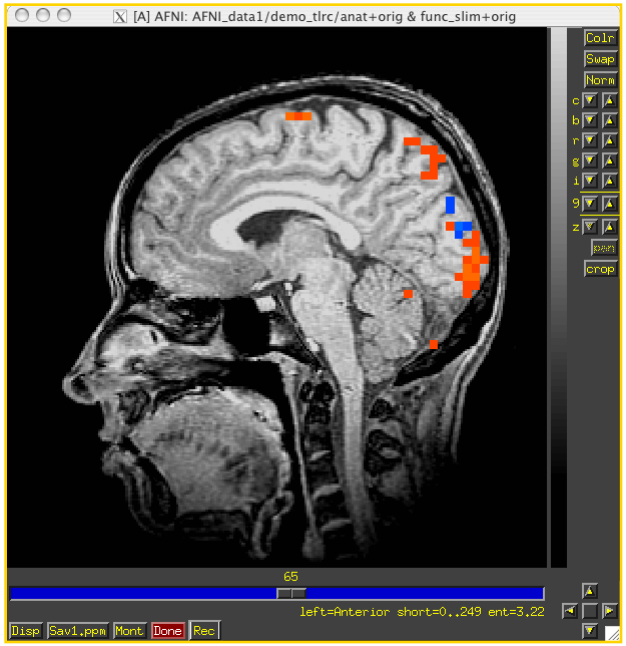
```
@auto_tlrc                    \
   -apar anat+tlrc            \
   -input func_slim+orig      \
   -suffix NONE               \
   -dxyz  2
```

Output:
`func_slim+tlrc`

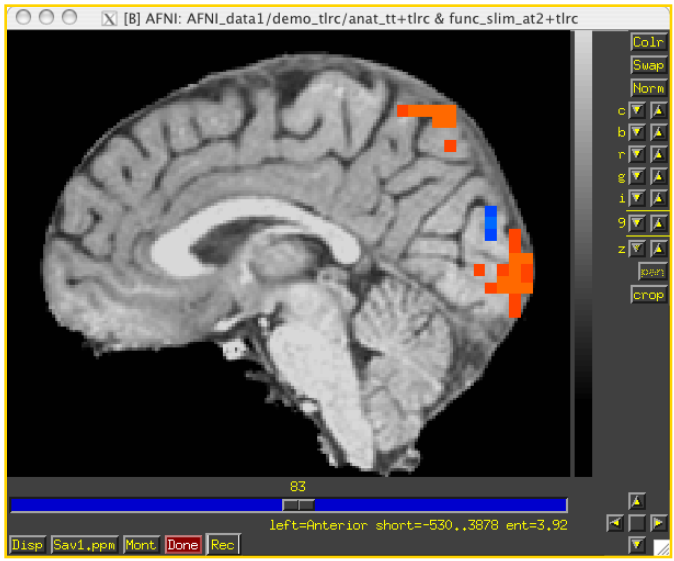
- You could also use the **icbm452** or the mni's **avg152T1** template instead of **N27** or any other template you like (see @auto_tlrc -help for a few good words on templates)

@auto_tlrc Results are Comparable to Manual TLRC:

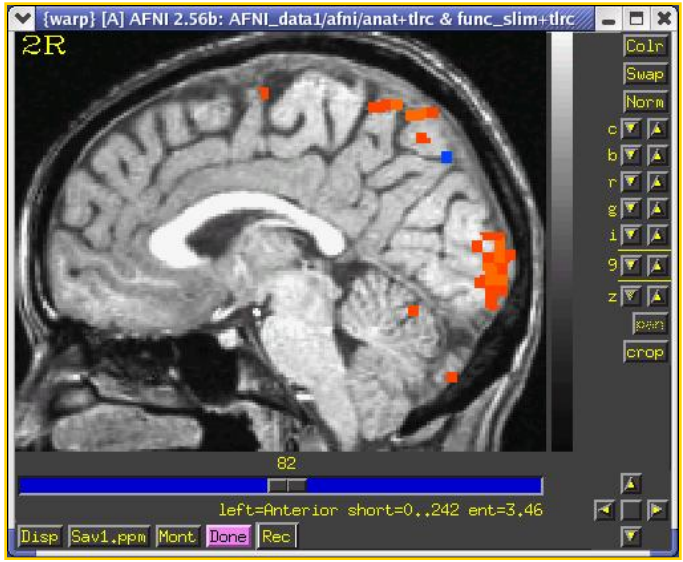
Original



@auto_tlrc



Manual



Standard Spaces

Why use a standard template space?

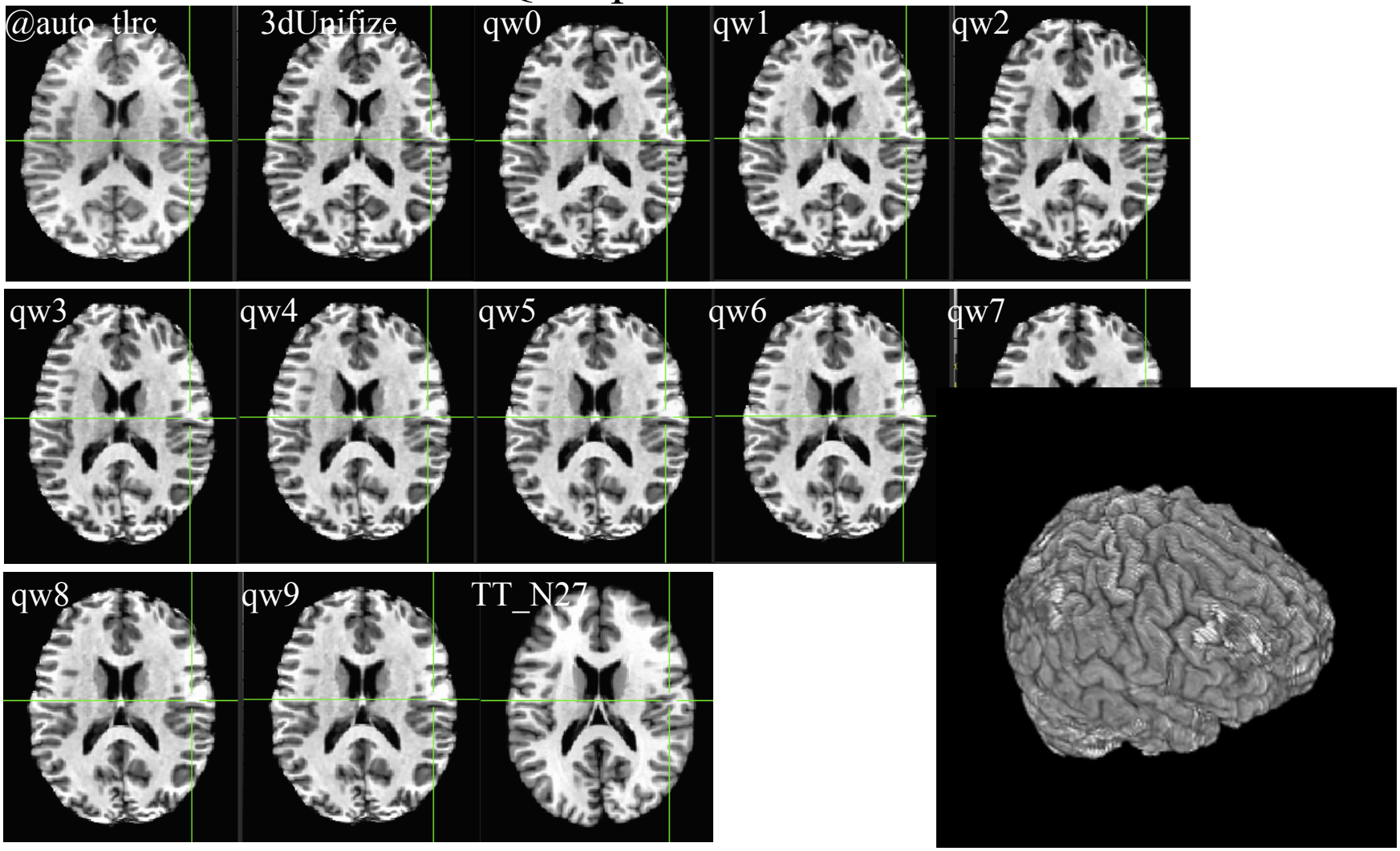
- Compare across subjects and groups easily for every voxel in the brain
- Standardize coordinates with others
- Know where a voxel is automatically from an atlas
- Mostly automated and no specific ROI drawing required

Why not use a standard template space?

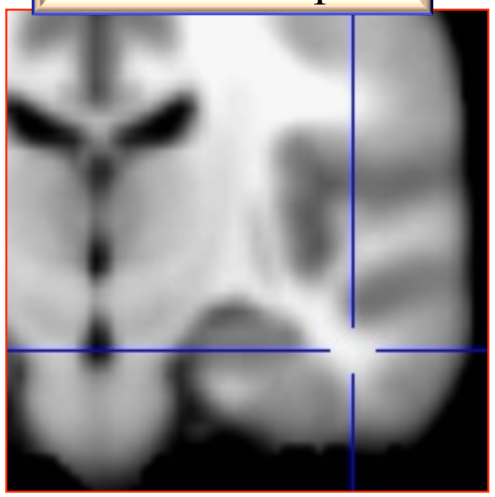
- Inconsistency among subjects
- Inconsistency among groups – elderly versus younger
- Use consistent anatomical ROIs with good anatomical knowledge
- Lower threshold for multiple comparison adjustments

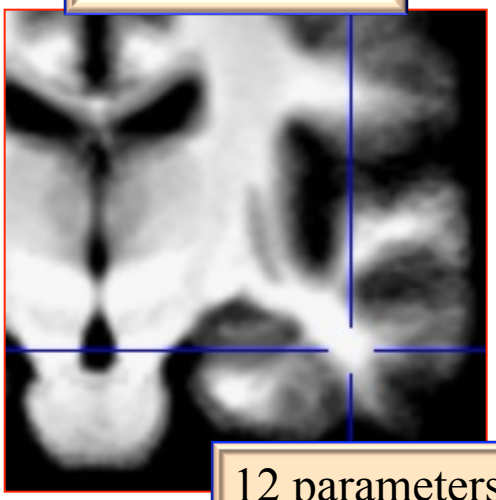
Nonlinear alignment to template

3dQwarp →

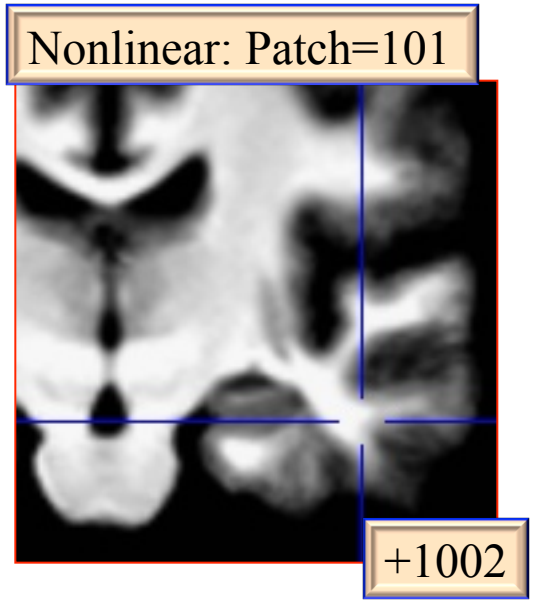


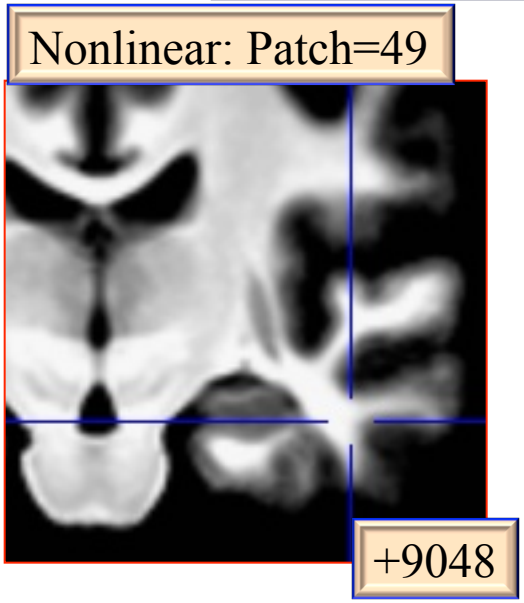
Example: Brain Averages

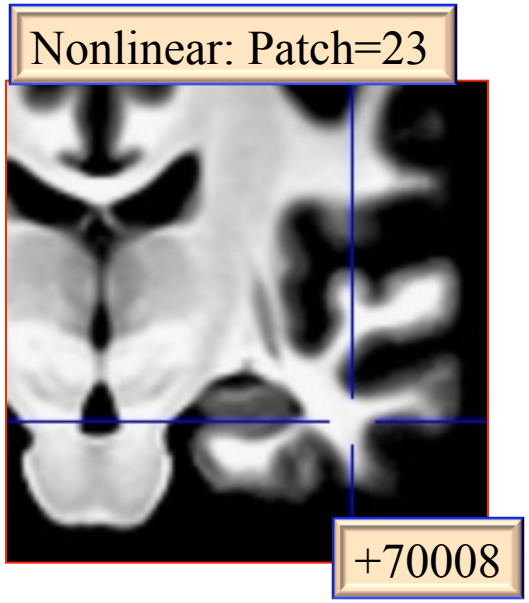




Averages of 21
3 Tesla brain volumes
with varied
levels of refinement in
the nonlinear warping
(smaller patch=more refinement)







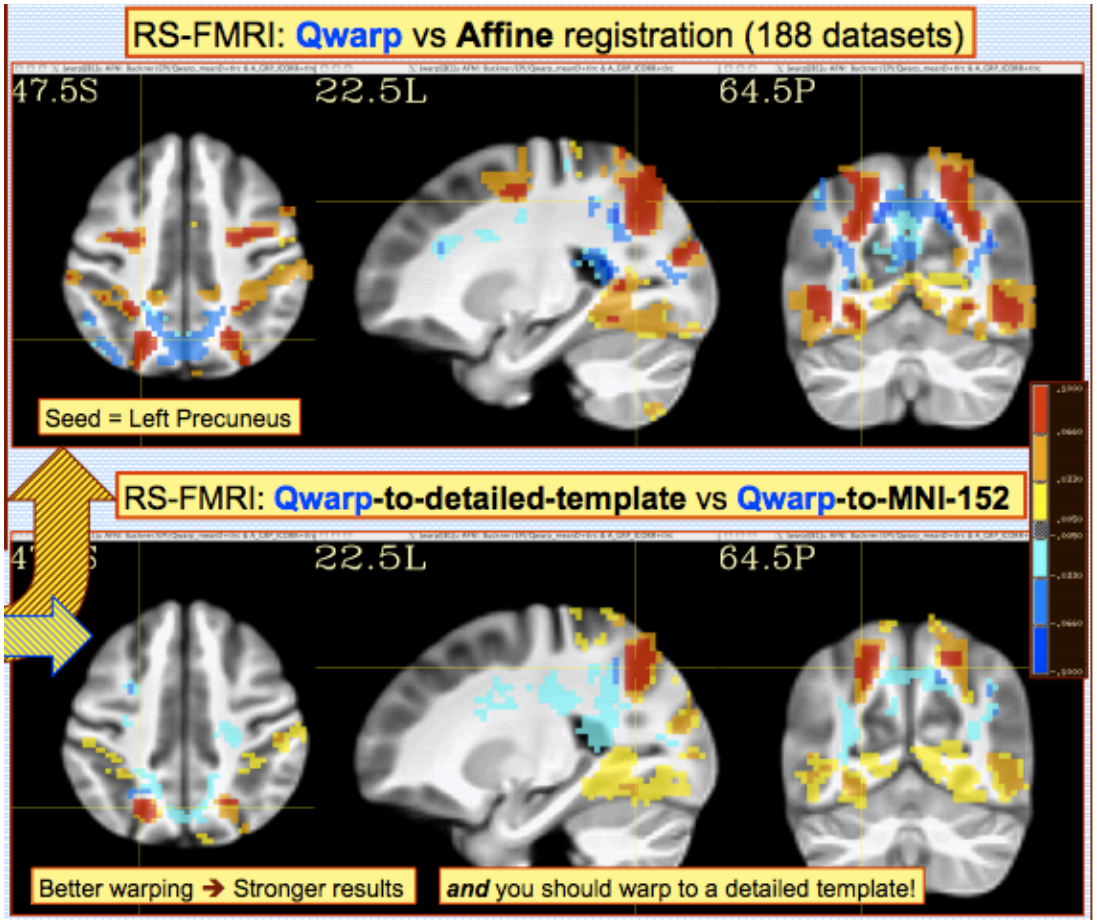
Nonlinear alignment

Advantages:

- Great alignment – better spatial correspondence across data →
- Aligned data matches template

Disadvantages:

- Distortion of individual data
- Aligned data matches template. Choose template carefully
- End limits?
- Skullstripping must be done much more carefully
- Processing time much slower
- Not yet integrated into AFNI GUI for warp-on-demand



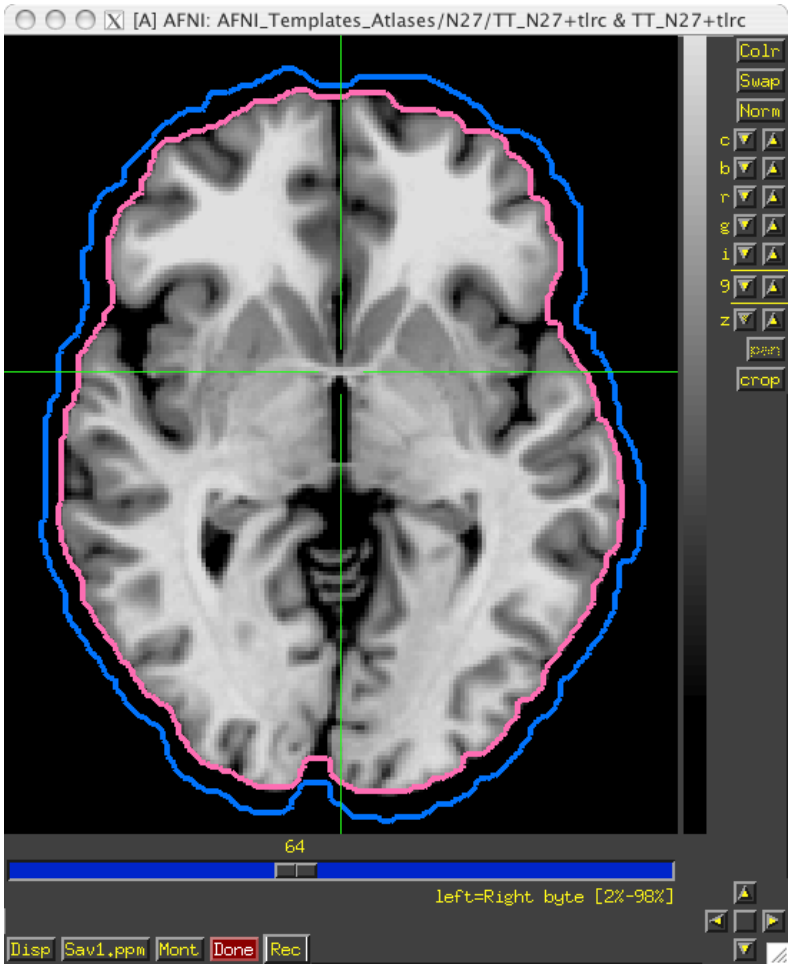
Comparing data

- How can I compare regions/voxels across subjects and groups?
  What works “best”?
  - **@auto_tlrc** – affine registration method to align individual subjects to a template – useful for most applications.
  - **auto_warp.py** – nonlinear alignment to template – maybe better.
  - **manual Talairach** – based on specific markers divides data up based on AC-PC line and brain enclosing boxes. Better for looking at medial structures.
  - **3dTagalign** – place markers on specific corresponding points among datasets and align with affine transformation
  - **ROI creation** – draw ROI’s (Draw Dataset plug-in) for each structure

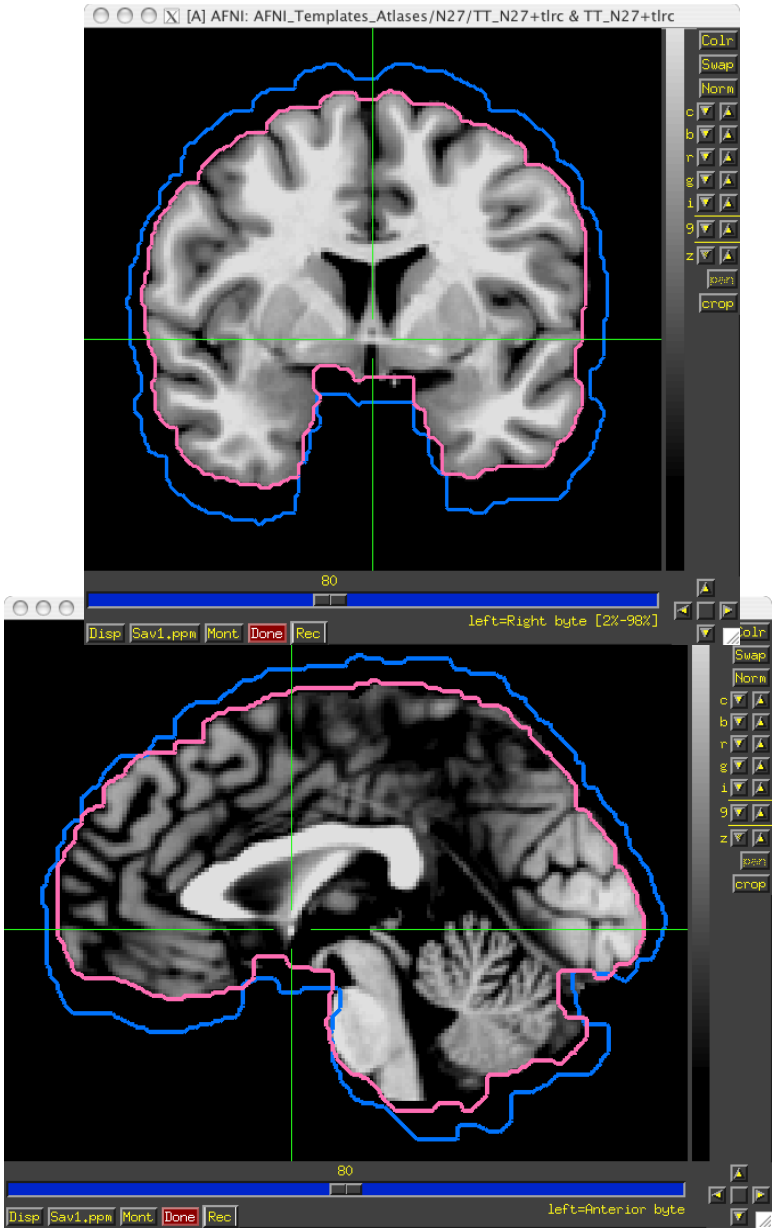
Choosing a template

- Similar to subject group – neonates, pediatric, young adults, elderly, macaque, rabbit...
- Same modality, similar coverage
- Individual or group template
- Make your own template (and maybe an atlas too)
  - Haskins pediatric atlas research
    - affine group averages
    - finding the most "typical" individual in group
    - nonlinear alignment to typical
    - iterative nonlinear alignment
      - @toMNI_Awarp, @toMNI_Qwarpar

Atlas/Template Spaces Differ In Size

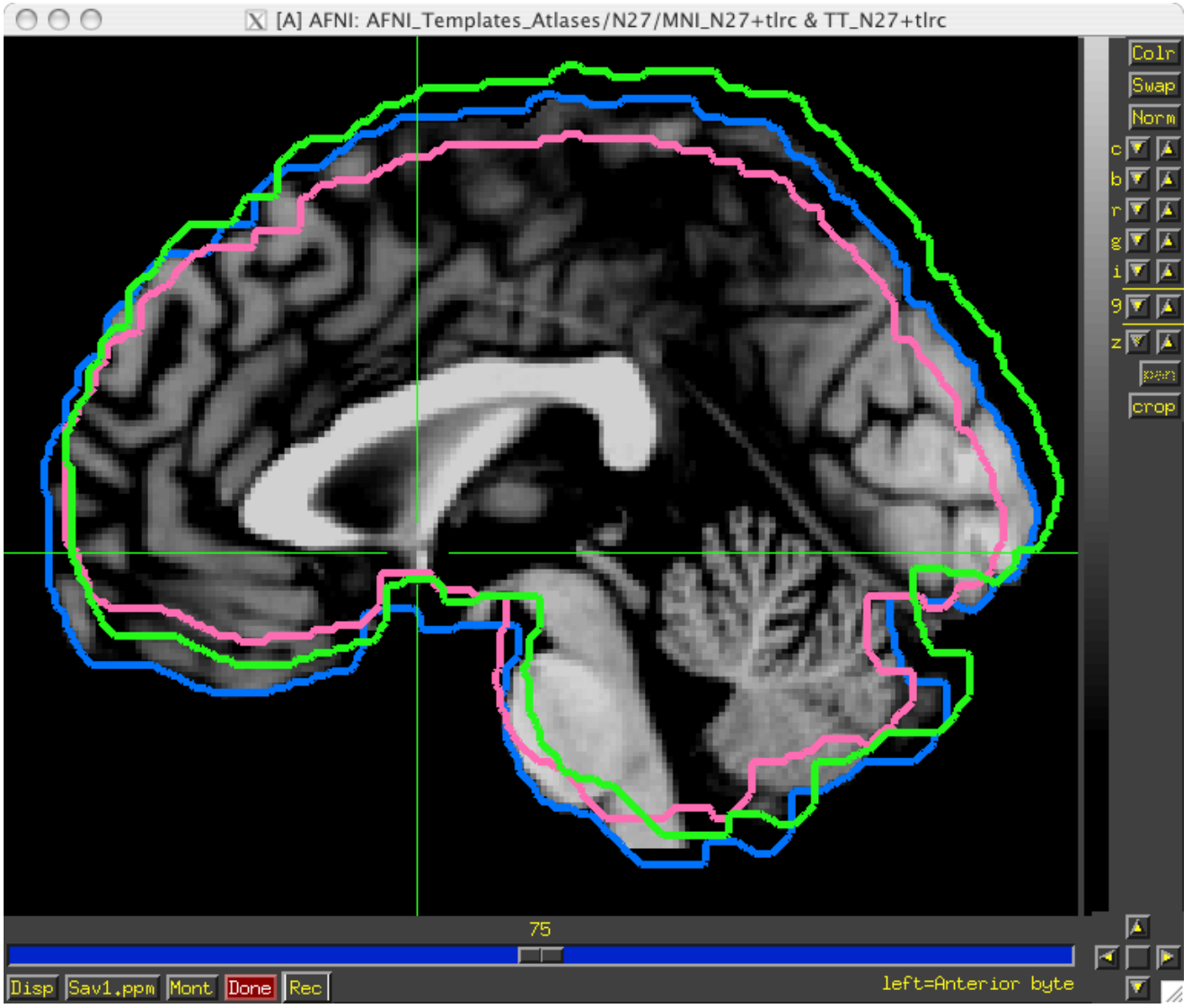


MNI is larger than TLRC space.



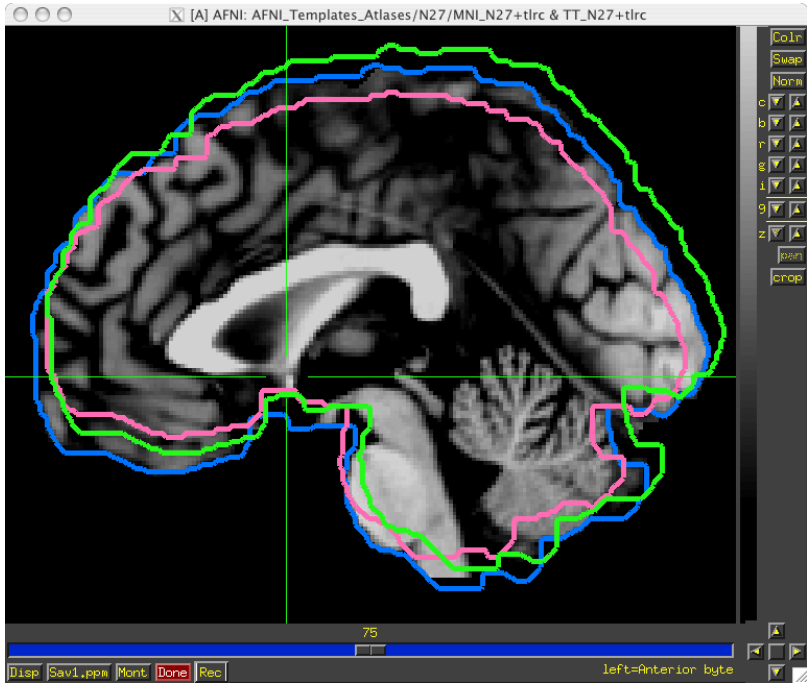
Atlas/Template Spaces Differ In Origin

TLRC
MNI
MNI-Anat.



From Space To Space

TLRC
MNI
MNI-Anat.



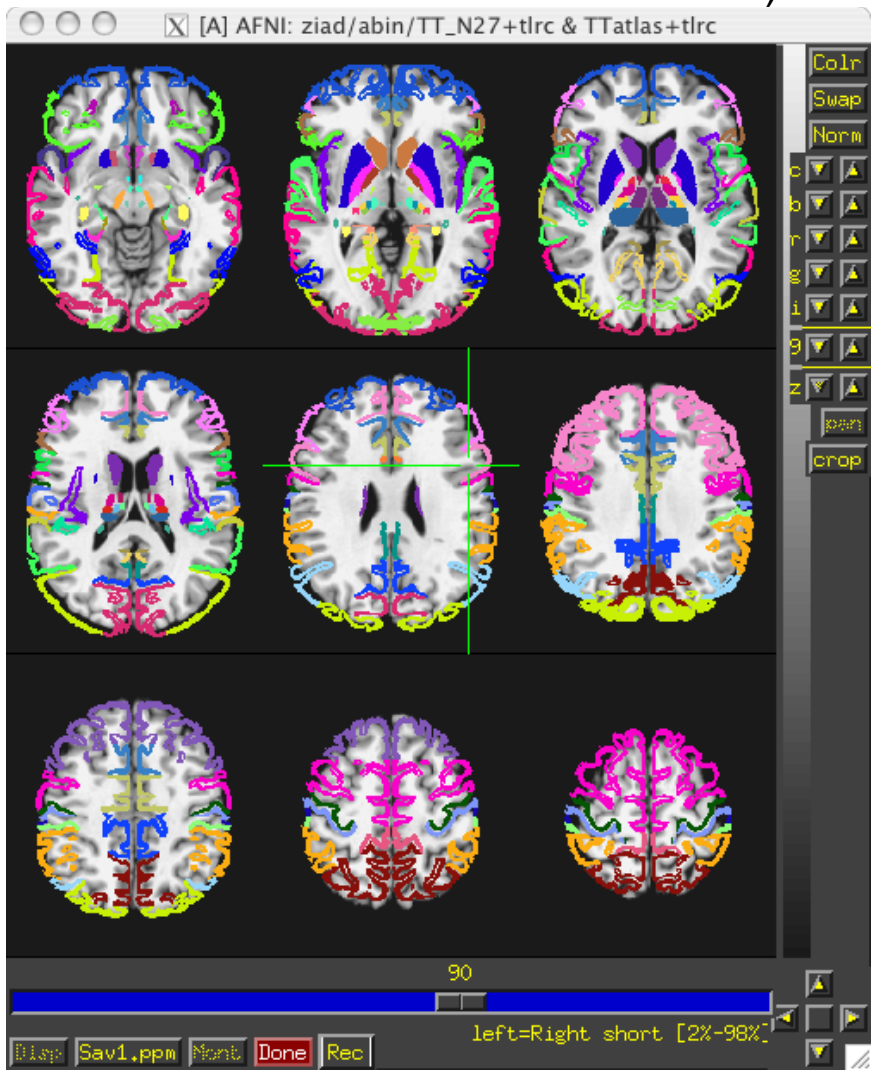
- Going between TLRC and MNI:
  - ✧ Approximate equation
    - ➥ used by **`whereami`** and **`3dWarp`**
  - ✧ Manual TLRC transformation of MNI template to TLRC space
    - ➥ used by **`whereami`** (as precursor to MNI Anat.), based on N27 template
  - ✧ Multiple space coordinates reported in whereami output (AFNI_ATLAS_TEMPLATE_SPACE_LIST)
- Going between MNI and MNI Anatomical (Eickhoff et al. Neuroimage 25, 2005):
  - ✧ MNI + (0, 4, 5) = MNI Anat. (in RAI coordinate system)
- Going between TLRC and MNI Anatomical (as practiced in **`whereami`**):
  - ✧ Go from TLRC (TT_N27) to MNI via manual xform of N27 template
  - ✧ Add (0, 4, 5)

Atlases/Templates Use Different Coord. Systems

- There are 48 manners to specify XYZ coordinates
- Two most common are RAI/DICOM and LPI/SPM
- RAI means
  - X is Right-to-Left (from negative-to-positive)
  - Y is Anterior-to-Posterior (from negative-to-positive)
  - Z is Inferior-to-Superior (from negative-to-positive)
- LPI means
  - X is Left-to-Right (from negative-to-positive)
  - Y is Posterior-to-Inferior (from negative-to-positive)
  - Z is Inferior-to-Superior (from negative-to-positive)
- To go from RAI to LPI just flip the sign of the X and Y coordinates
  - Voxel -12, 24, 16 in RAI is the same as 12, -24, 16 in LPI
  - Voxel above would be in the Right, Posterior, Superior octant of the brain
- AFNI allows for all coordinate systems but default is RAI
  - Can use environment variable AFNI_ORIENT to change the default for AFNI *AND* other programs.
  - See **`whereami -help`** for more details.

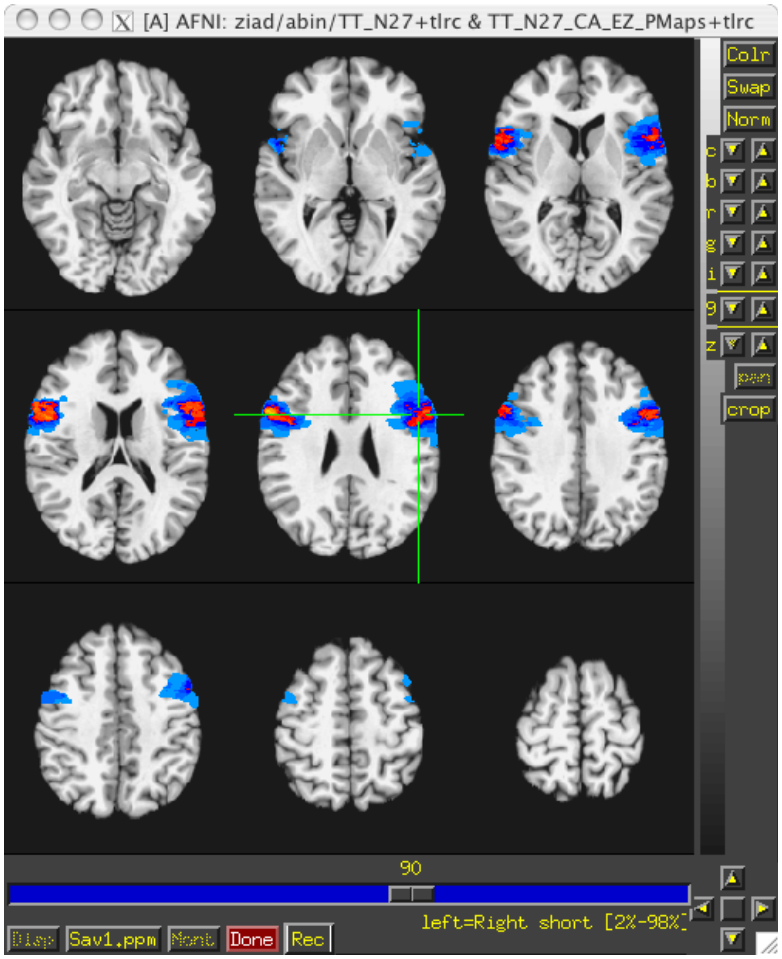
Atlases Distributed With AFNI
TT_Daemon

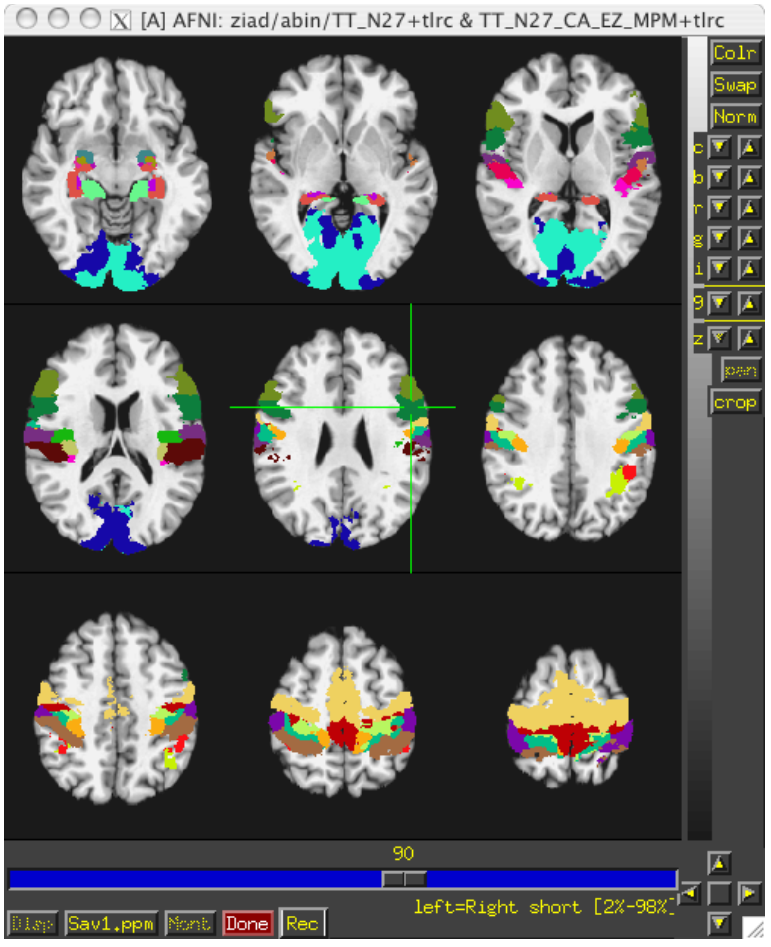
- TT_Daemon : Created by tracing Talairach and Tournoux brain illustrations.
  - ✧ Generously contributed by Jack Lancaster and Peter Fox of RIC UTHSCSA)



Atlases Distributed With AFNI
Anatomy Toolbox: Prob. Maps, Max. Prob. Maps

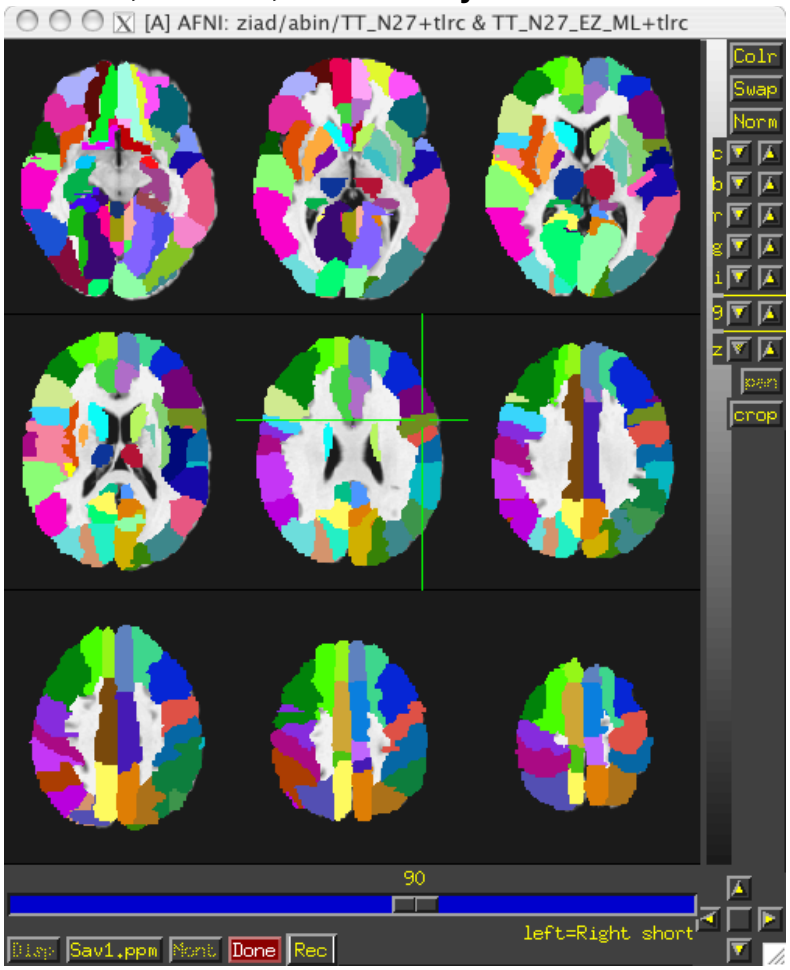
- CA_N27_MPM, CA_N27_ML, CA_N27_PM: Anatomy Toolbox's atlases with some created from cytoarchitectonic studies of 10 human post-mortem brains
  - ✧ Generously contributed by Simon Eickhoff, Katrin Amunts and Karl Zilles of IME, Julich, Germany

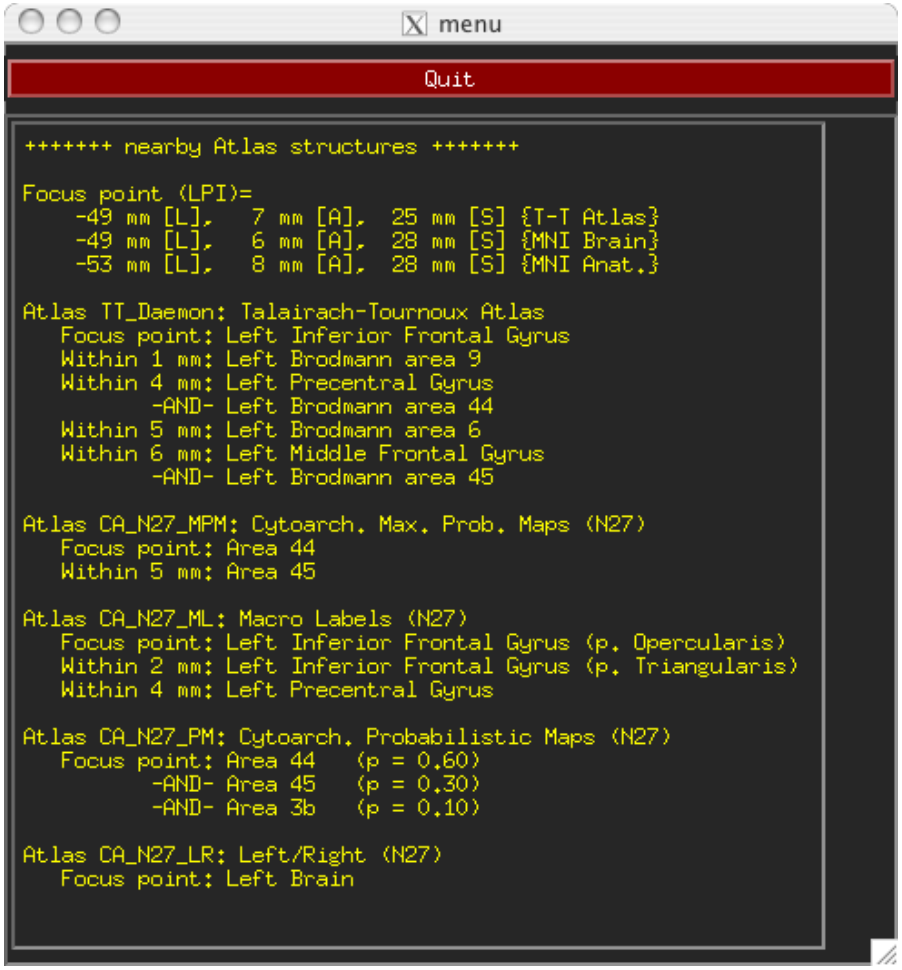




Atlases Distributed With AFNI: Anatomy Toolbox: MacroLabels

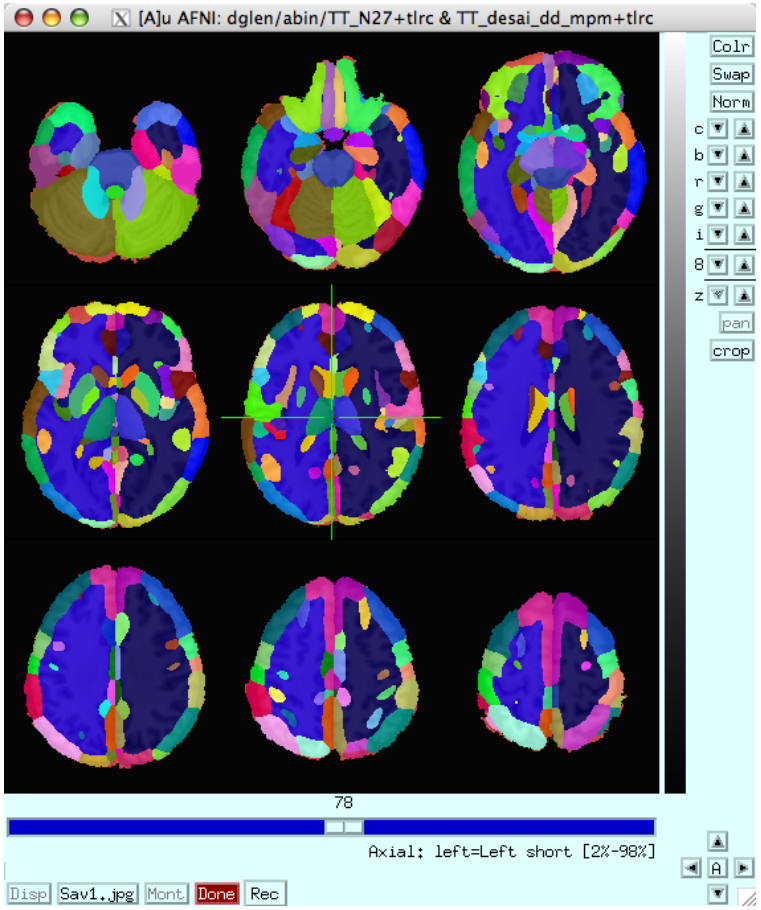
- CA_N27_MPM, CA_N27_ML, CA_N27_PM: Anatomy Toolbox's atlases with some created from cytoarchitectonic studies of 10 human post-mortem brains
  - ✧ Generously contributed by Simon Eickhoff, Katrin Amunts and Karl Zilles of IME, Julich, Germany

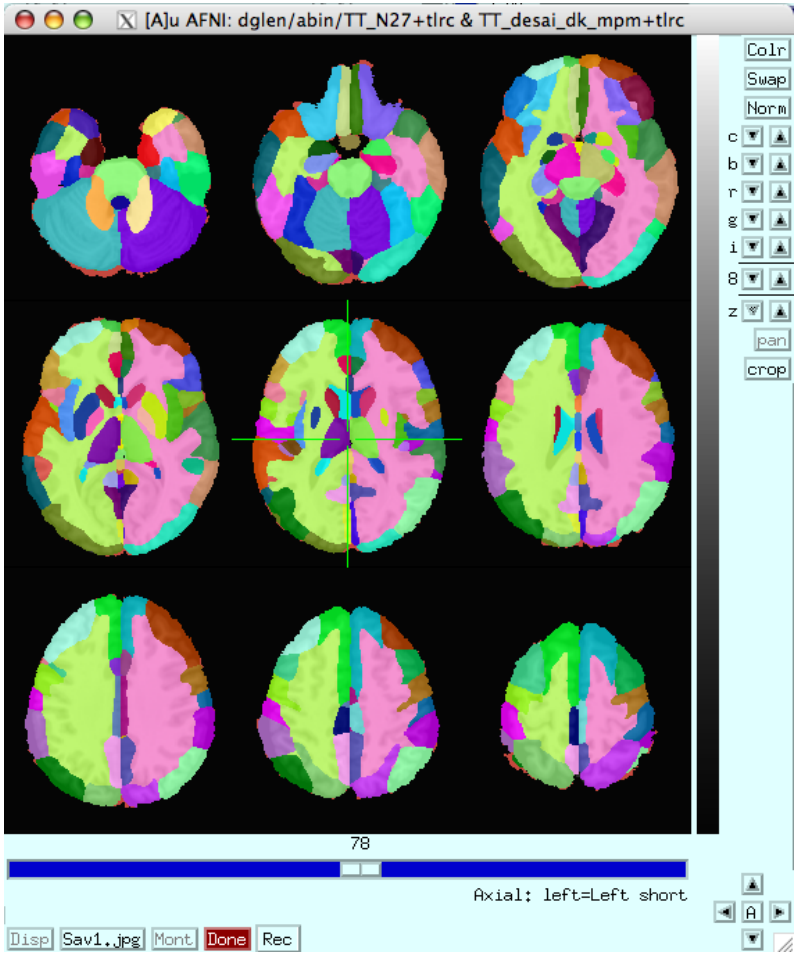




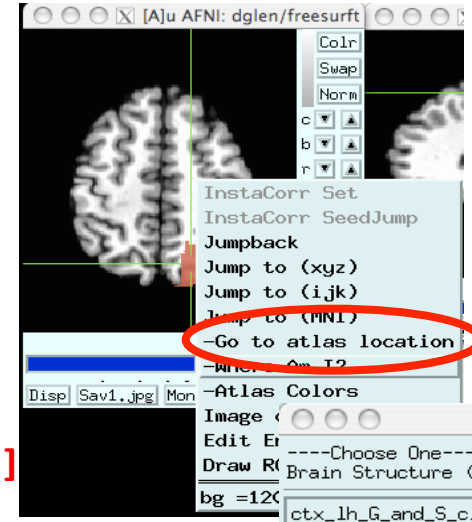
Atlases Distributed With AFNI: Desai PMaps and MPMs

- Atlases generated with typical AFNI pipeline using @auto_tlrc and FreeSurfer segmentation across multiple subjects

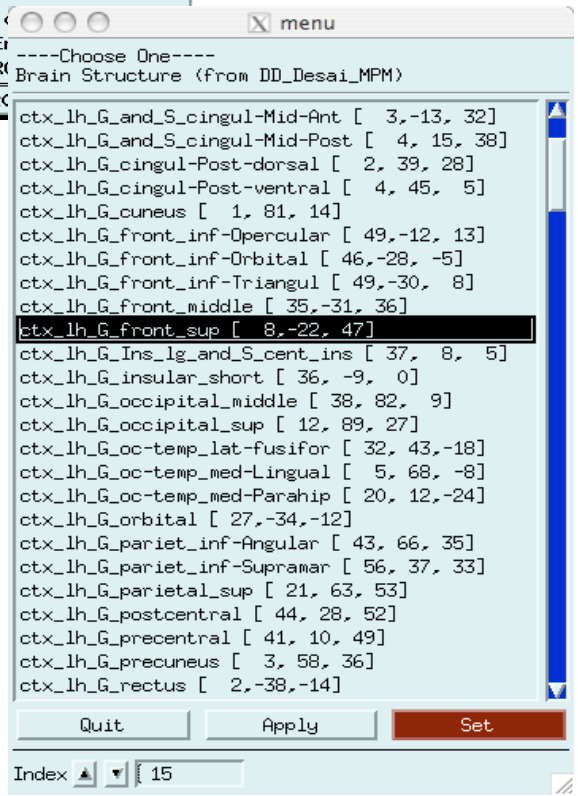




- Some fun and useful things to do with `+tlrc` datasets are on the 2D slice viewer
  Right click to get menu:



✧ [Go to Atlas Location]



Lets you jump to centroid of regions to current default atlas (set by AFNI_ATLAS_COLORS)
Works in `+orig` too

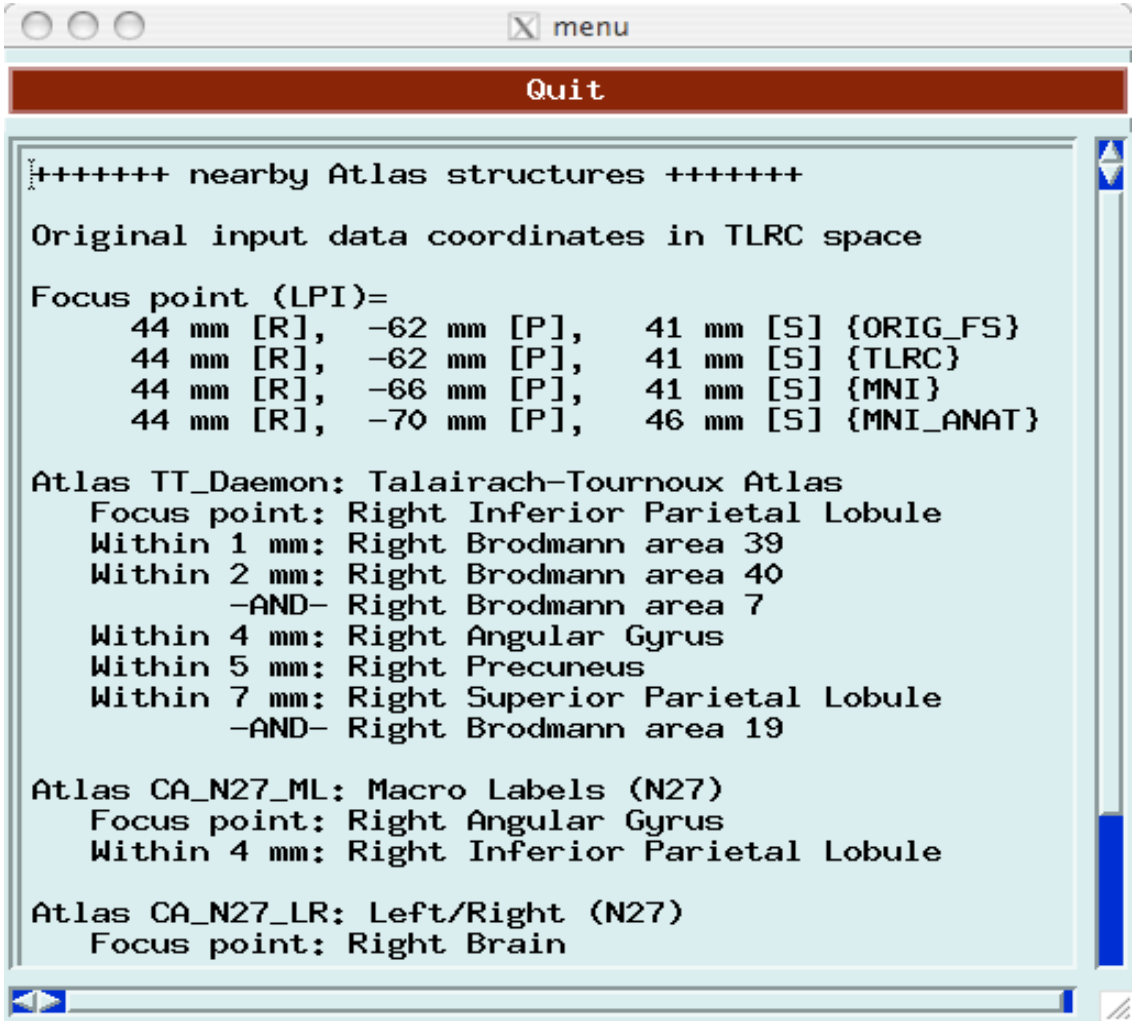
✧ [Where am I?]

Shows you where you are in various atlases and spaces

(works in `+orig` too, if you have a transformed parent)

For atlas installation, and much, much more, see help in command line version:

`whereami -help`



- **whereami** can provide the user with more detailed information regarding the output of **3dclust**
  - For instance, say you want more information regarding the center of mass voxels from each cluster (from the 3dclust output). I.e., where do they fall approximately within the atlases?

```
3dclust -dxyz=1 -1clip 9.5  1 1000 func_FullF+tlrc > clusts.1D
whereami -coord_file clusts.1D'[1,2,3]' -tab | less
```

Center of mass output, columns 1,2,3, from 3dclust output

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

Focus point (LPI)                          Coord.Space
  42 mm [R], -61 mm [P],  -3 mm [I]     {T-T Atlas}
  43 mm [R], -63 mm [P],  -7 mm [I]     {MNI Brain}
  45 mm [R], -69 mm [P],   2 mm [I]     {MNI Anat.}
Atlas           Within  Label                                  Prob.   Code
TT_Daemon       3.0     Right Middle Occipital Gyrus           ---     33
TT_Daemon       3.0     Right Brodmann area 37                 ---     113
TT_Daemon       4.0     Right Inferior Temporal Gyrus          ---     29
TT_Daemon       4.0     Right Middle Temporal Gyrus            ---     35
TT_Daemon       6.0     Right Inferior Occipital Gyrus         ---     28
TT_Daemon       7.0     Right Brodmann area 19                 ---     96
CA_N27_MPM      4.0     hOC5 (V5 / MT+)                        MPM     110
CA_N27_ML       0.0     Right Inferior Temporal Gyrus          ---     90
CA_N27_ML       1.0     Right Middle Temporal Gyrus            ---     86
CA_N27_ML       5.0     Right Inferior Occipital Gyrus         ---     54
CA_N27_ML       6.0     Right Middle Occipital Gyrus           ---     52

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

Shown: Cluster #1's coordinates according to various atlases (TT, MNI, etc), as well as the name of the anatomical structure that is located at or near these coordinates (which may vary by atlas)

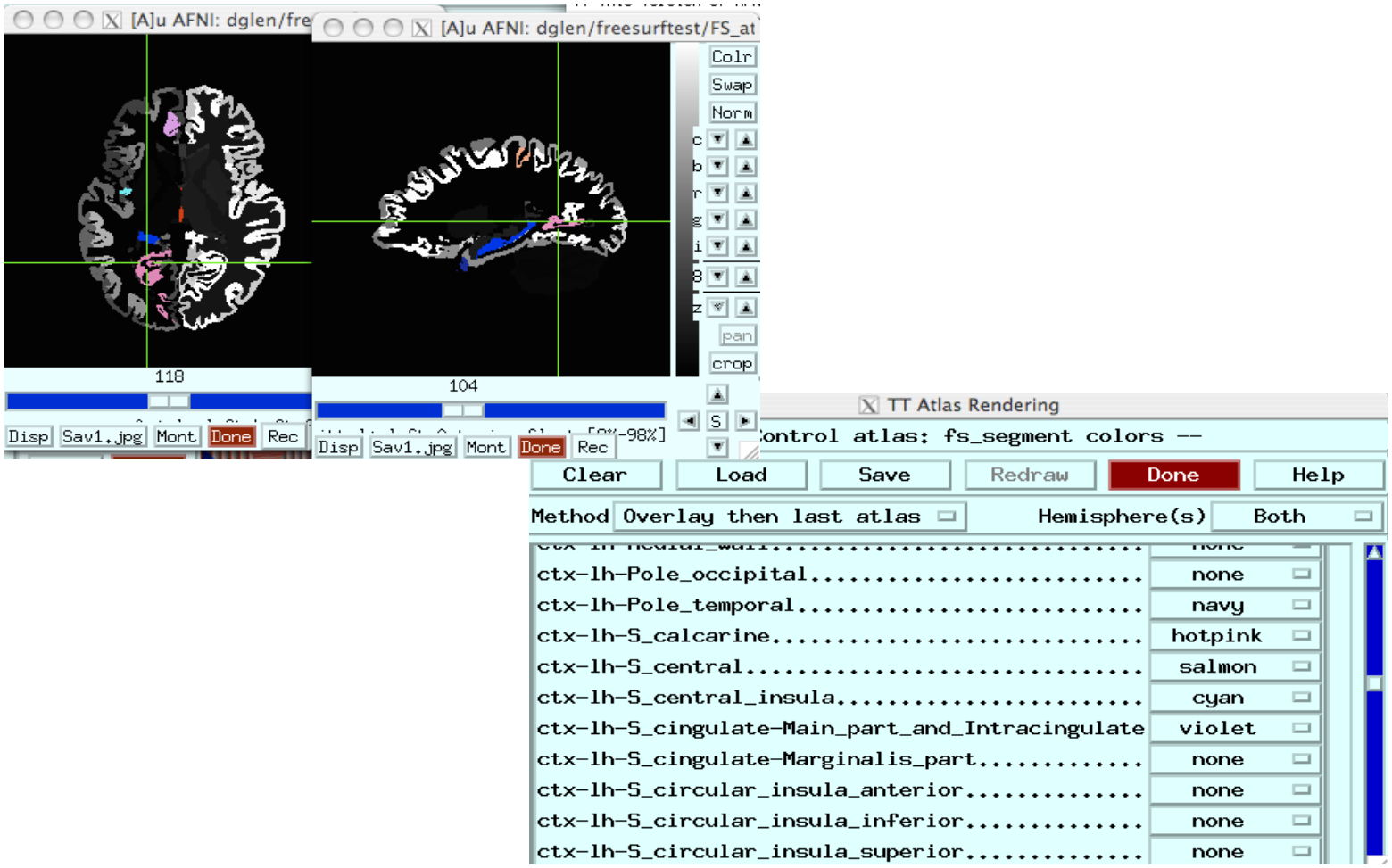
etc…

- **whereami** can report on the overlap of ROIs with atlas-defined regions

whereami -omask anat_roi+tlrc

```
++ Input coordinates orientation set by default rules to RAI
++ Input coordinates space set by default rules to TLRC
++ In ordered mode ...
++ Have 2 unique values of:
   0   1
++ Skipping unique value of 0
++ Processing unique value of 1
++    195 voxels in ROI
++    195 voxels in atlas-resampled mask
Intersection of ROI (valued 1) with atlas TT_Daemon (sb0):
   89.2 % overlap with Middle Occipital Gyrus, code 33
   6.7  % overlap with Middle Temporal Gyrus, code 35
   -----
   95.9 % of cluster accounted for.

Intersection of ROI (valued 1) with atlas TT_Daemon (sb1):
   19.5 % overlap with Brodmann area 37, code 113
   1.5  % overlap with Brodmann area 19, code 96
   -----
   21.0 % of cluster accounted for.

++    195 voxels in atlas-resampled mask
Intersection of ROI (valued 1) with atlas CA_N27_MPM (sb0):
   1.5  % overlap with hOC5 (V5 / MT+), code 110
   -----
   1.5  % of cluster accounted for.

++    195 voxels in atlas-resampled mask
Intersection of ROI (valued 1) with atlas CA_N27_ML (sb0):
   61.0 % overlap with Right Middle Occipital Gyrus, code 52
   20.0 % overlap with Right Middle Temporal Gyrus, code 86
   -----
   81.0 % of cluster accounted for.
```

✧ [Atlas colors]



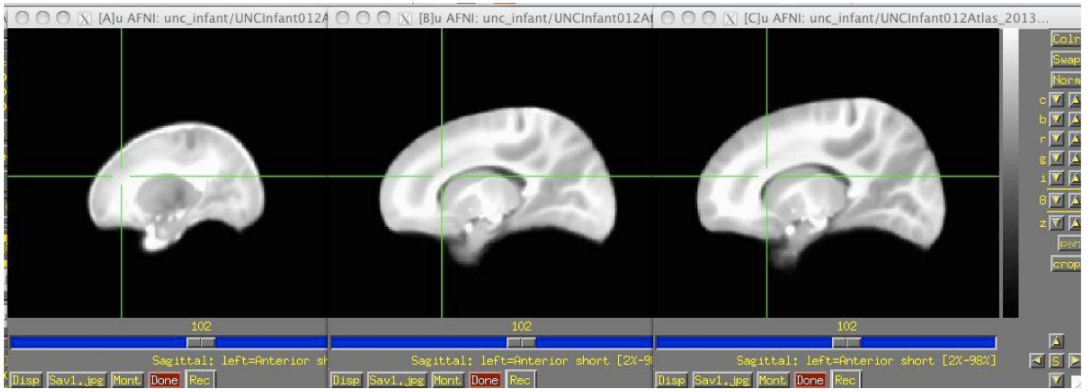
Lets you show atlas regions over your own data (works only in `+tlrc`).

New Atlases and Templates Available!

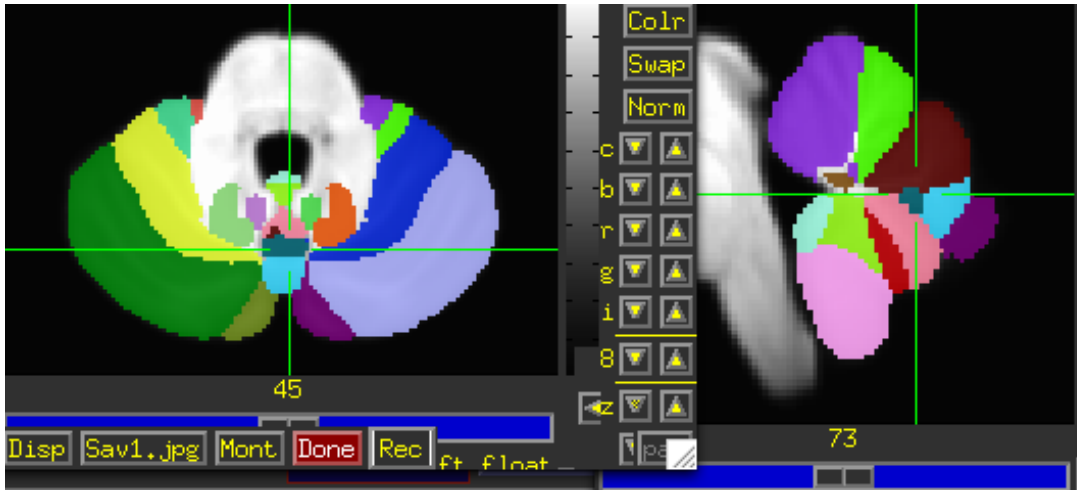
- Infant brain atlas and templates – neonate, 1-year, 2-year. Contributed by Feng Shi, UNC
- Cerebellum atlas and templates – Jorn Diedrichsen,UCL,UK contribution
- Rat brain templates in Paxinos space – Karolinska Institute, Woo Hyun Shim MGH contributions

All of these are user requests or contributions!

What do you need?

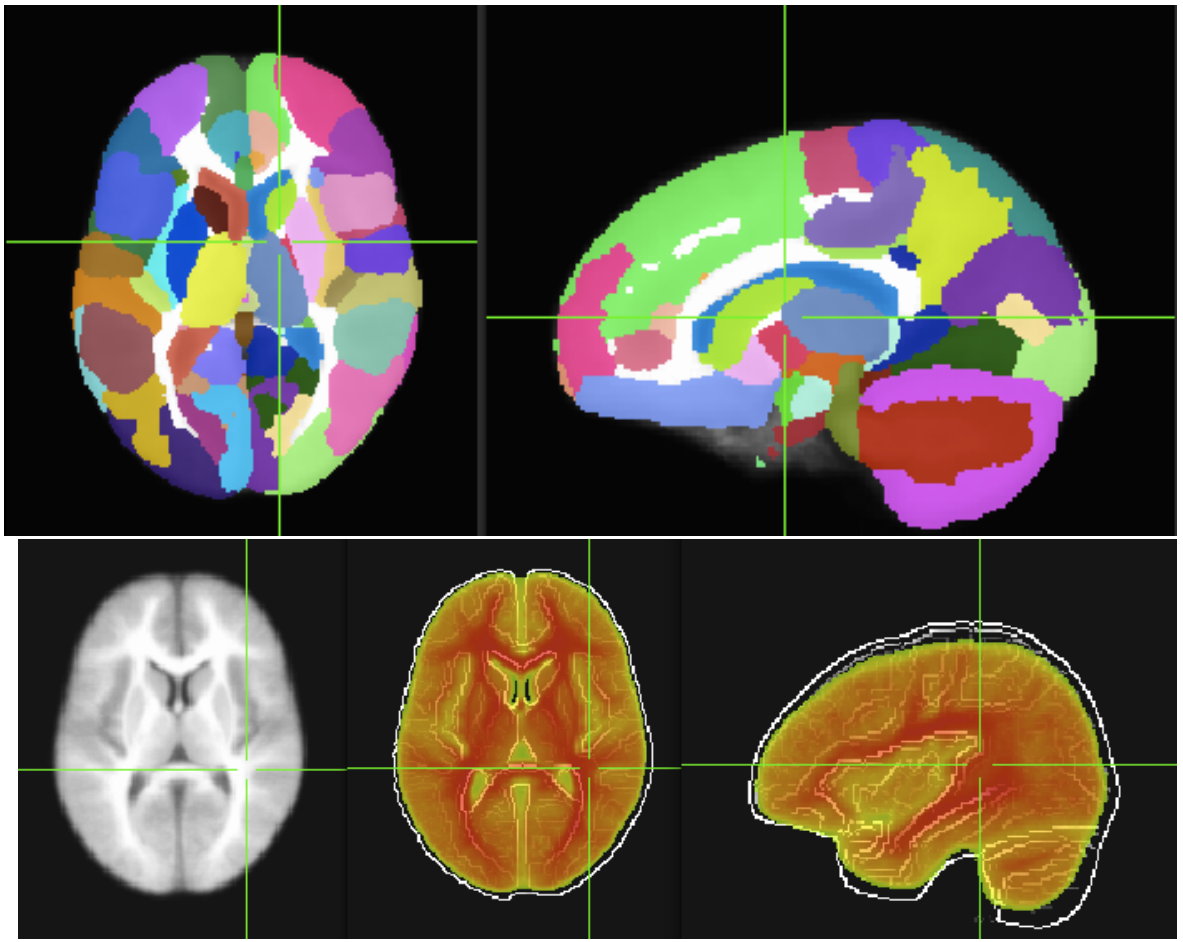


UNC infant templates and atlases – neonate, 1 yr, 2 yr old

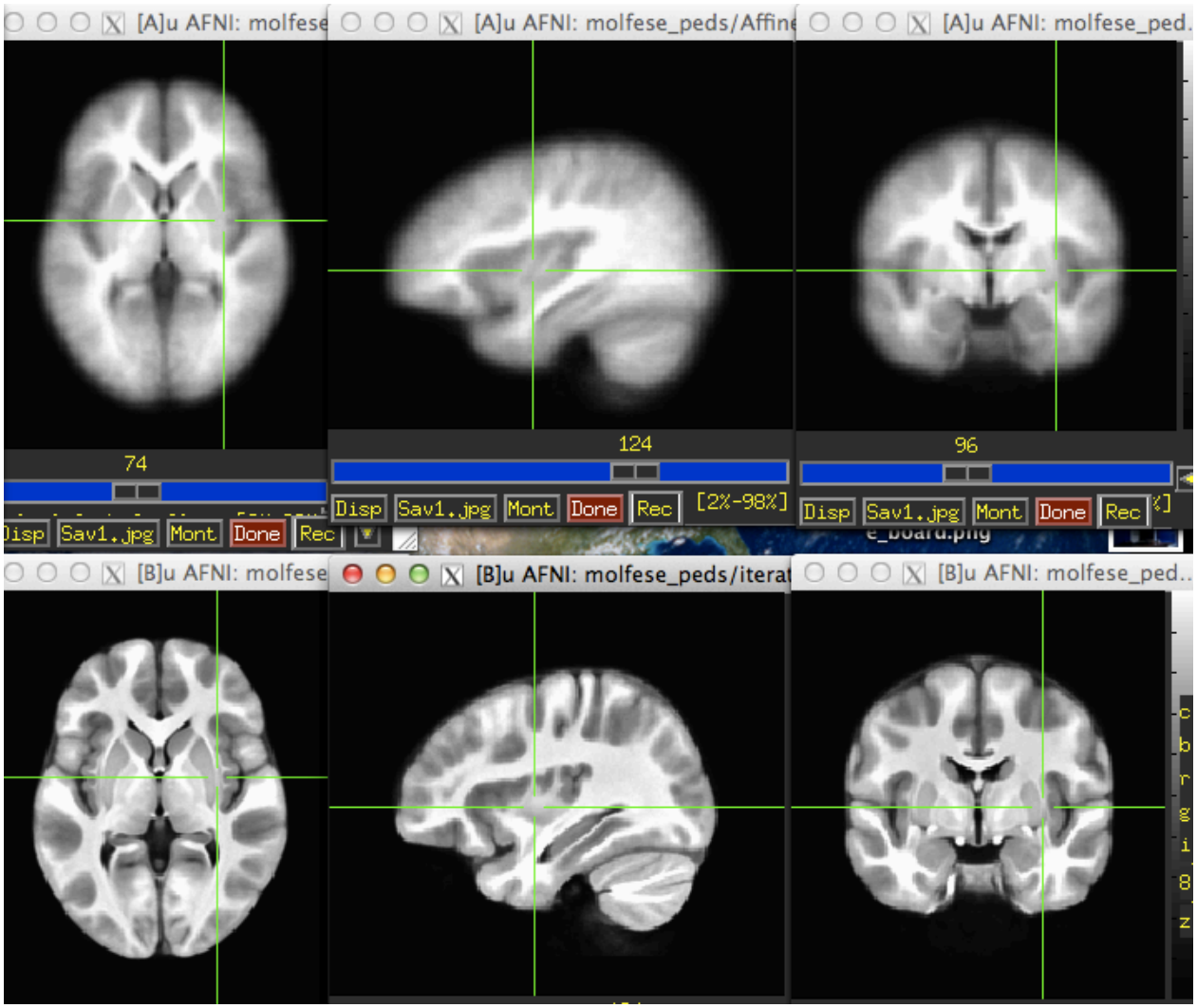


Haskins Pediatric Atlas

- Pediatric brain atlas and templates (7-12 years old) – Peter Molfese, Haskins Laboratories, available very soon
- Manually corrected segmentation from Freesurfer.
- Probabilistic, MPM and template
- ~75 subjects -> 500 (ages 6-13)
- Affine, nonlinear averages, ideal/typical subjects, outliers



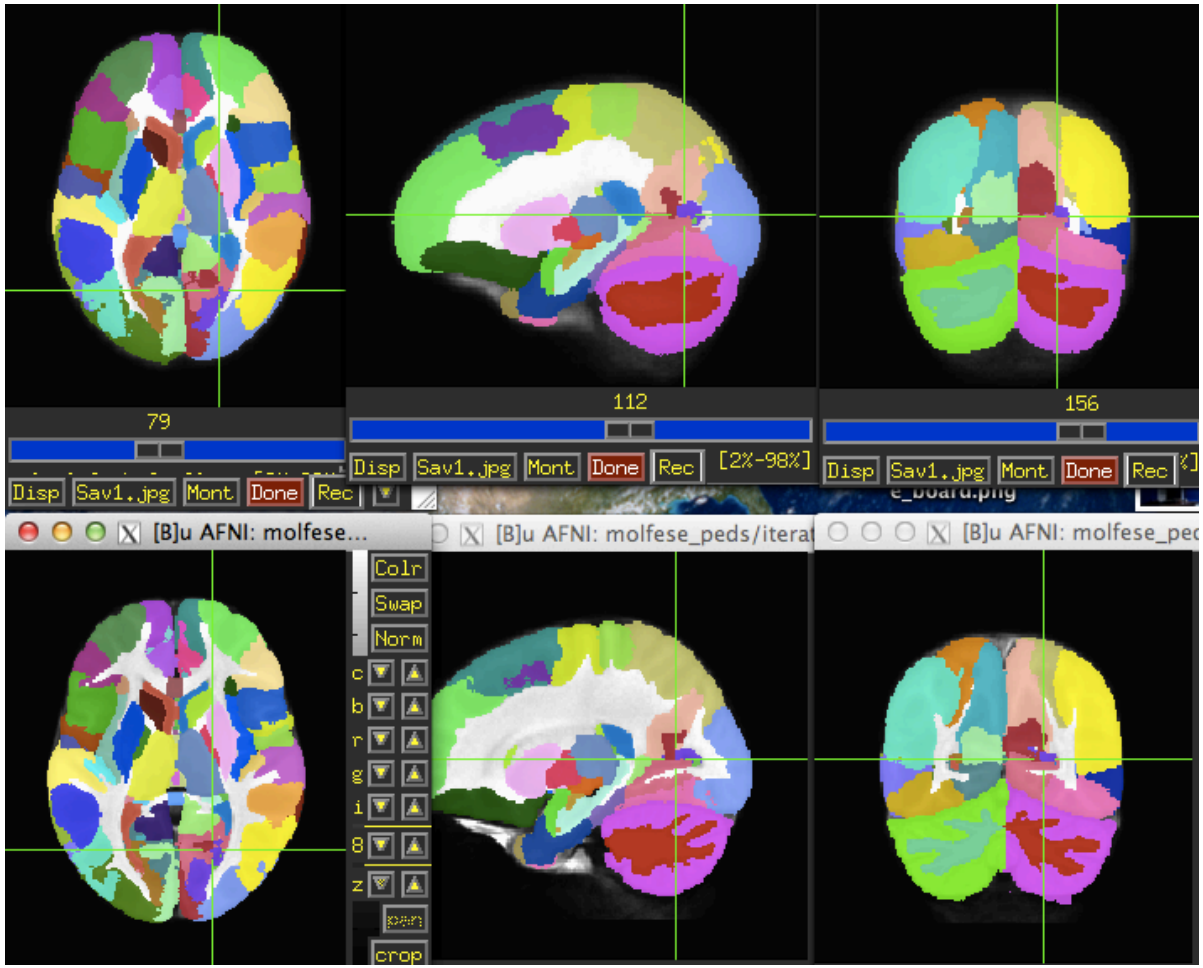
Haskins Pediatric Atlas - templates



Affine Group

Nonlinear Group I – iterative

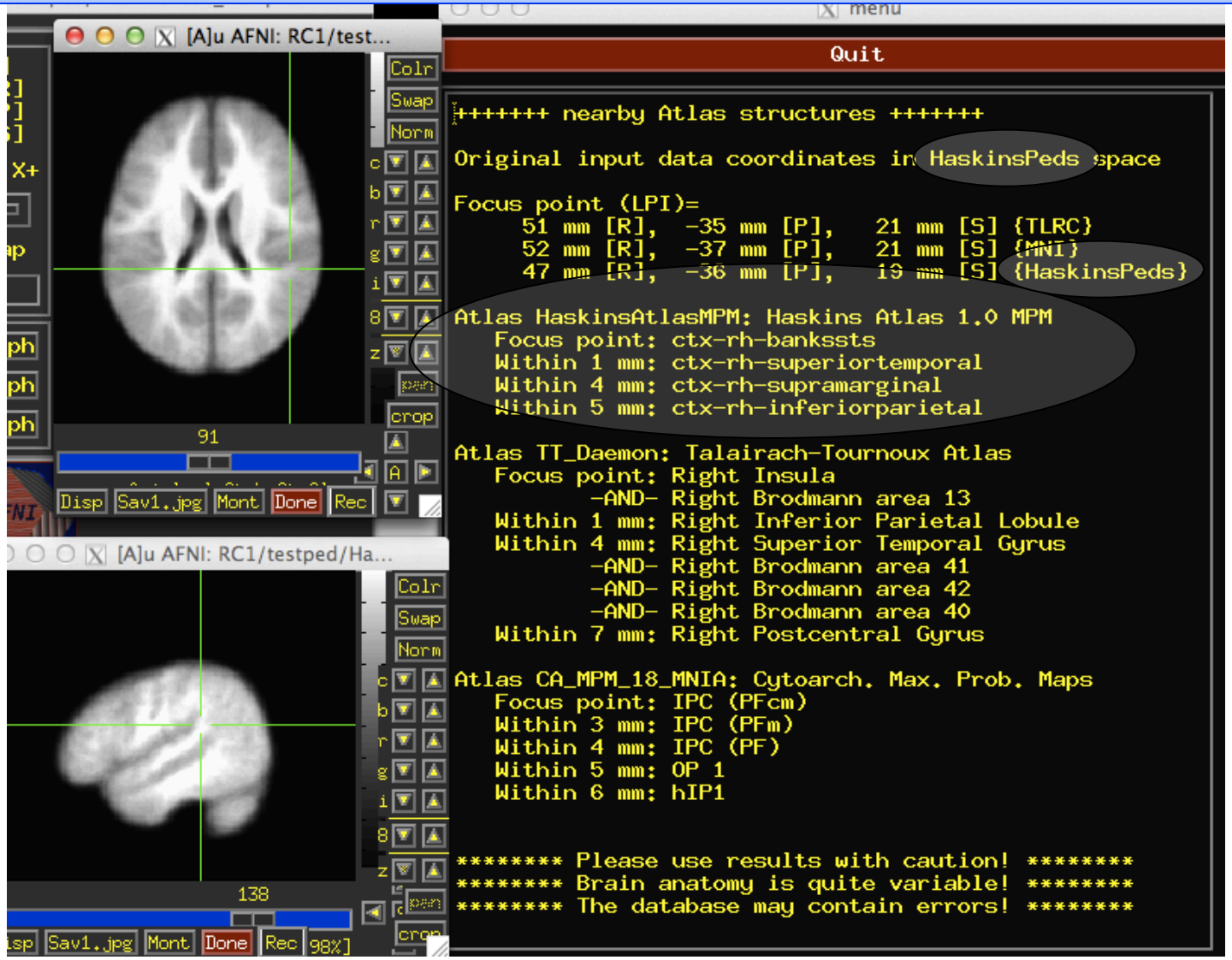
Haskins Pediatric Atlas – MPM atlases



Affine Group

Nonlinear Group I – iterative

Haskins Pediatric Atlas



Make your own atlas!

- New atlases – easy and fun. Make your own!
  - ✧ make available in AFNI GUI and whereami and to other user

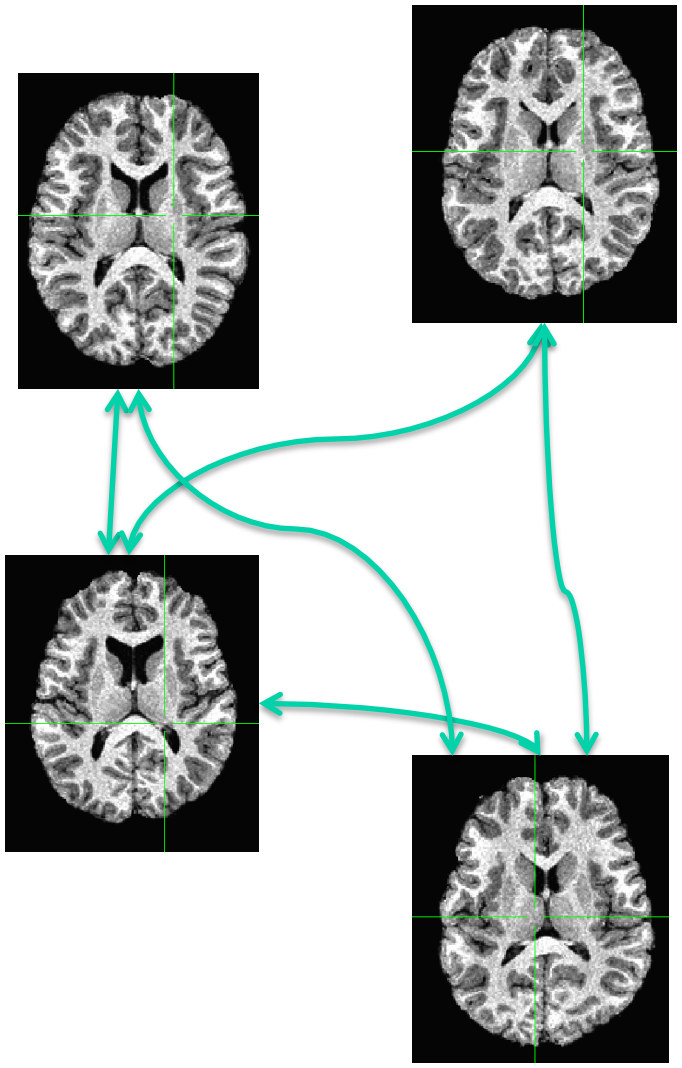
```
@AfniEnv -set AFNI_SUPP_ATLAS_DIR ~/MyCustomAtlases/
```

Then:

```
@Atlasize -space MNI -dset atlas_for_all.nii \
                     -lab_file keys.txt 1 0 -atlas_type G
```

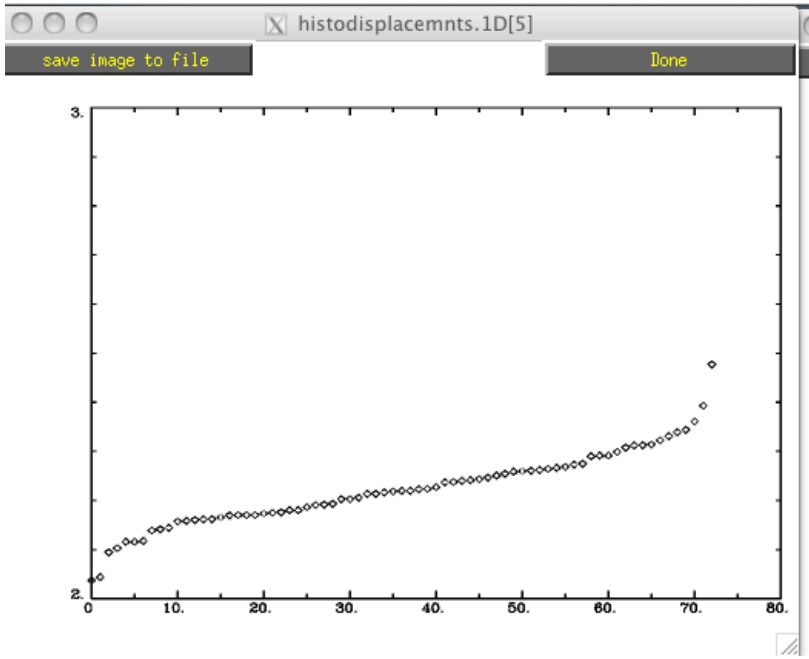
In ~/MyCustomAtlases/ you will now find atlas_for_all.nii along along with a modified CustomAtlases.niml file.

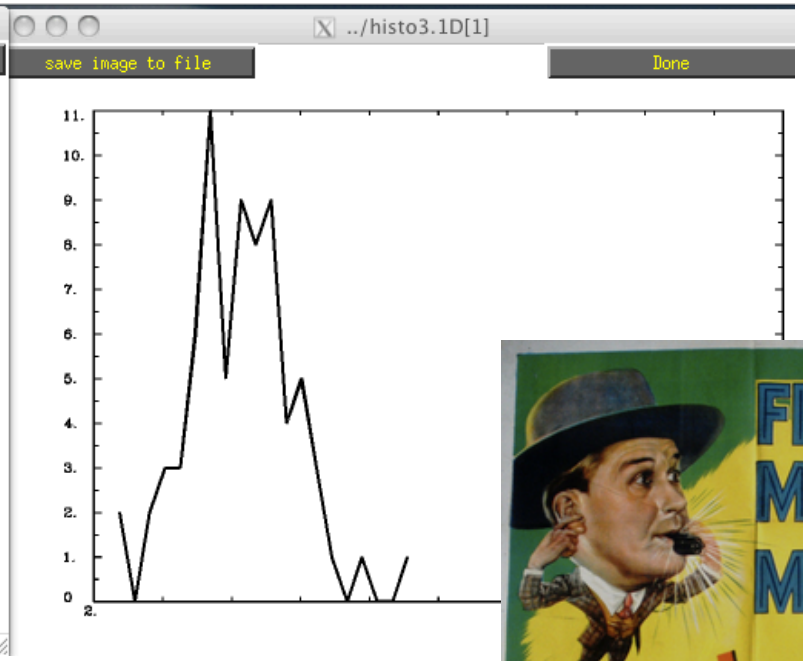
Haskins Pediatric Atlas – Finding the "typical" kid's brain



1. Make roughly same size with affine warp
2. Align every subject to every other
3. Find the mean of the absolute displacement at every voxel in the brain for each subject for warp between that subject and all others
4. Find sum of mean displacement across brain
5. Rank subjects by mean absolute

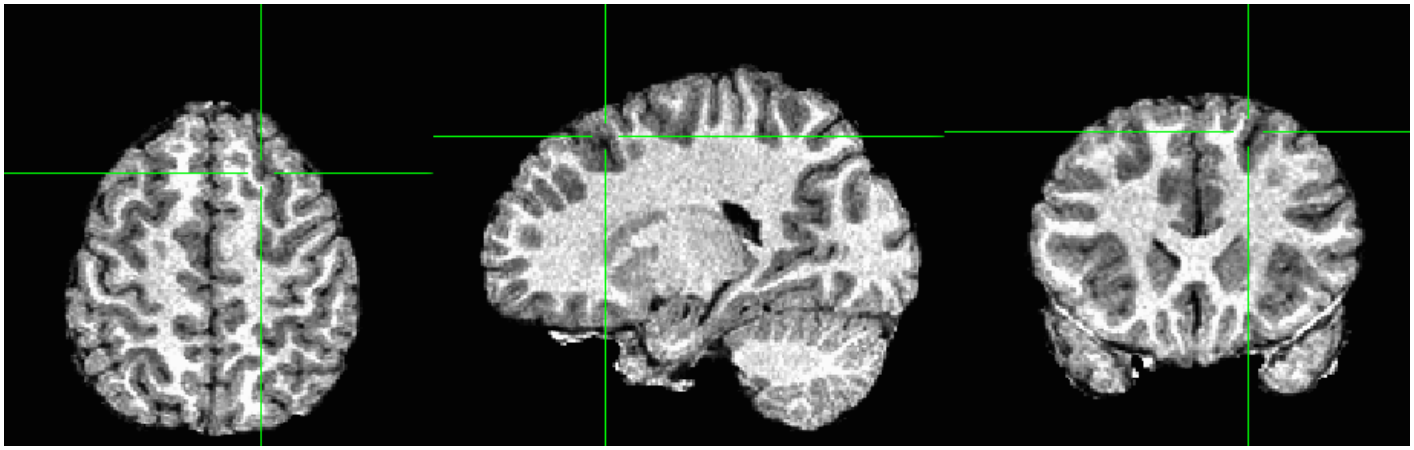
Haskins Pediatric Atlas – Finding the "typical" kid's brain



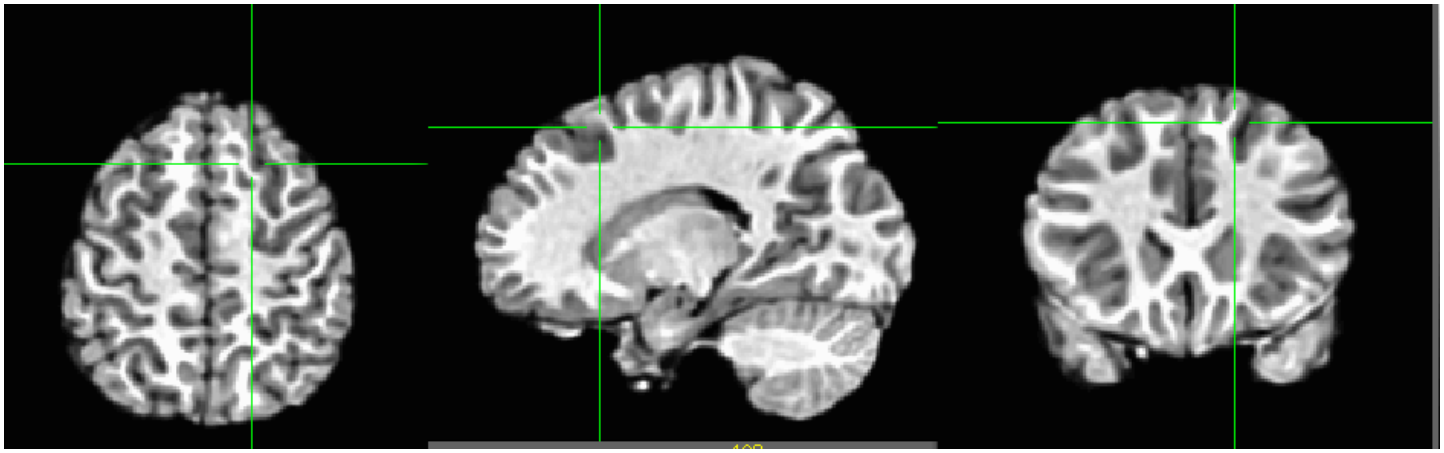


Kochunov (2001), Smith (2006), Fonov (2011)

Haskins Pediatric Atlas – Finding the "typical" kid's brain

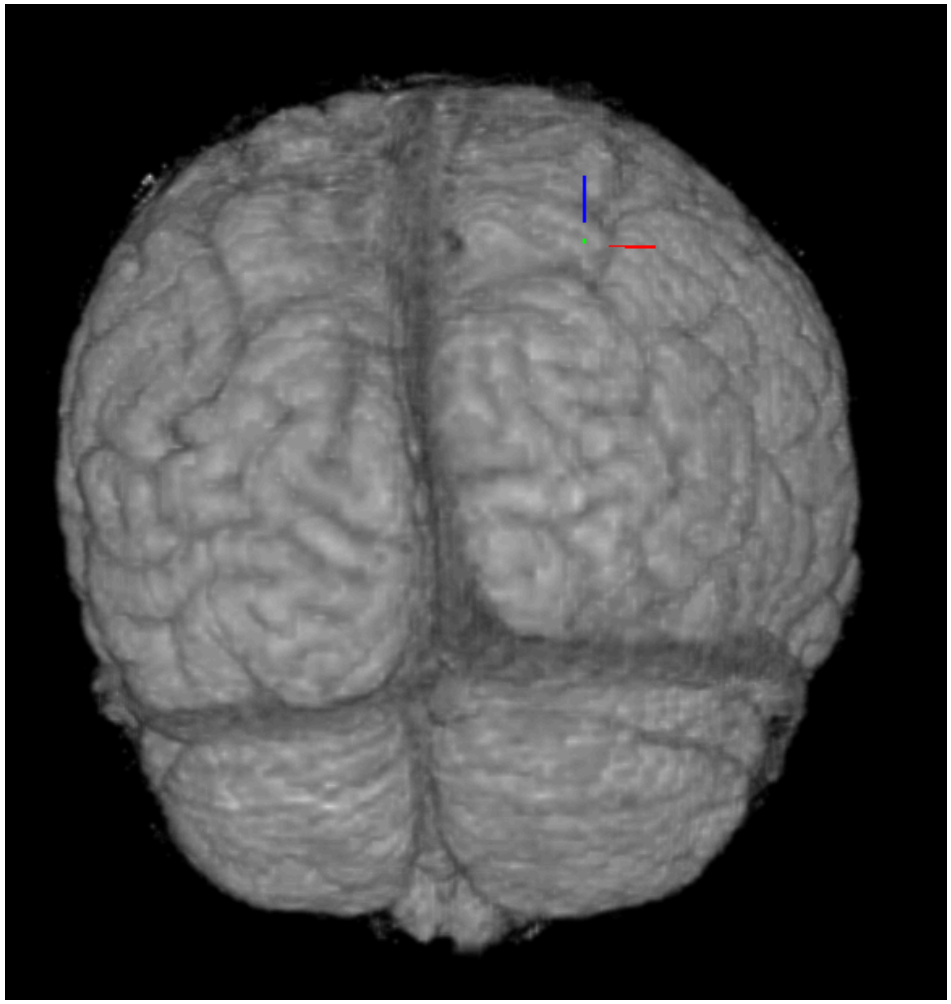


sn_xxx



anisotropically smoothed

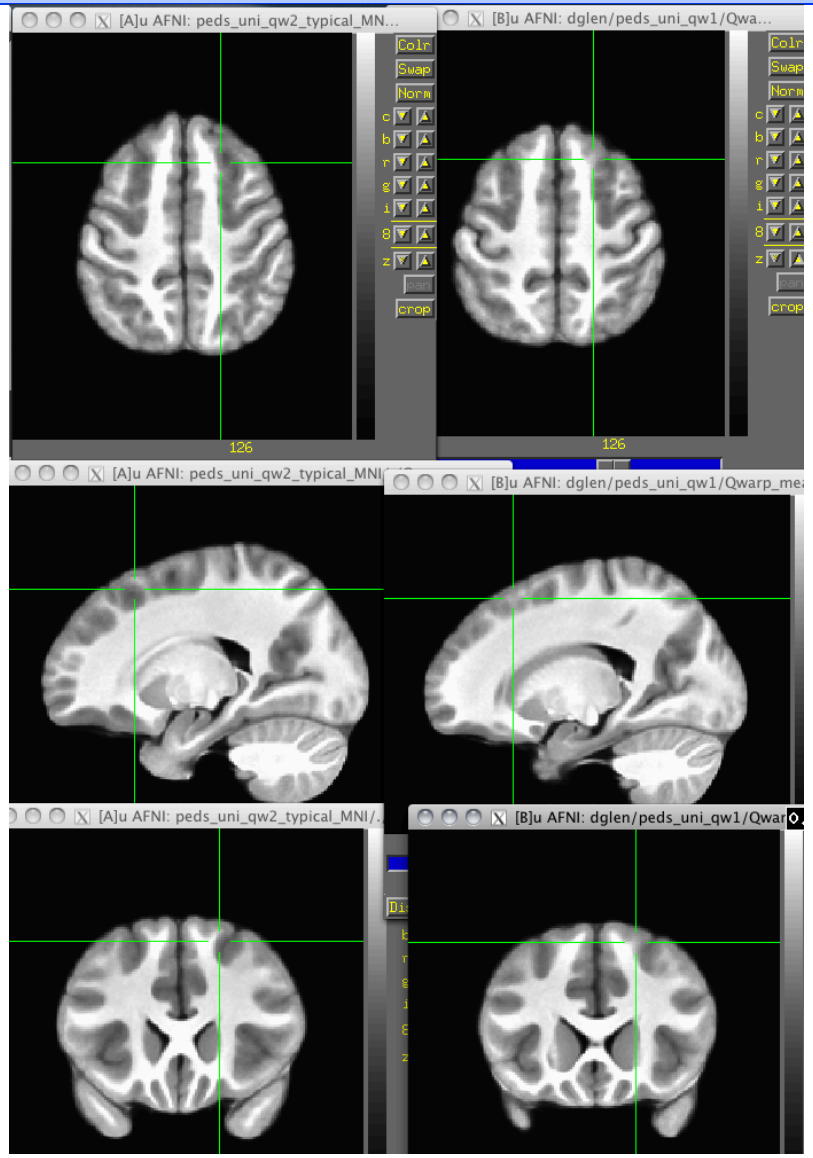
Haskins Pediatric Atlas – Finding the "atypical" kid's brain



Petalia – torque asymmetry

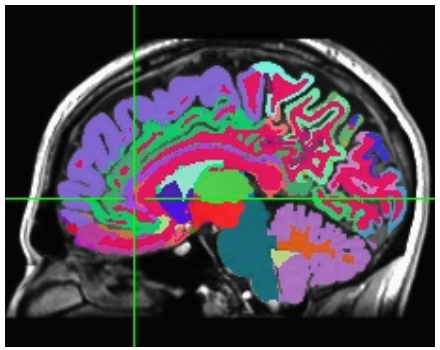
Haskins Pediatric Atlas – Nonlinear iterative group templates

Typical subject starting point

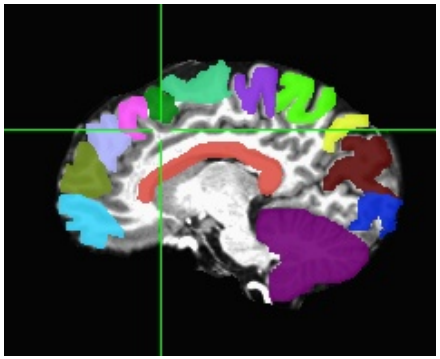


MNI-152 starting point

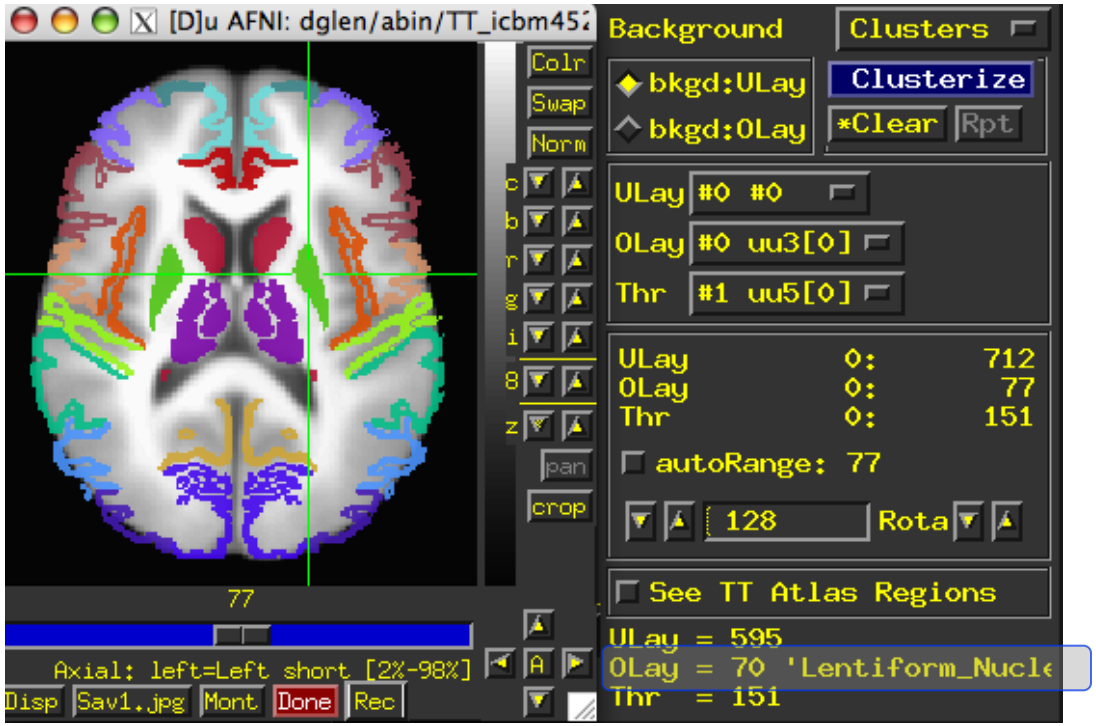
Individual Subjects



FreeSurfer segmentation



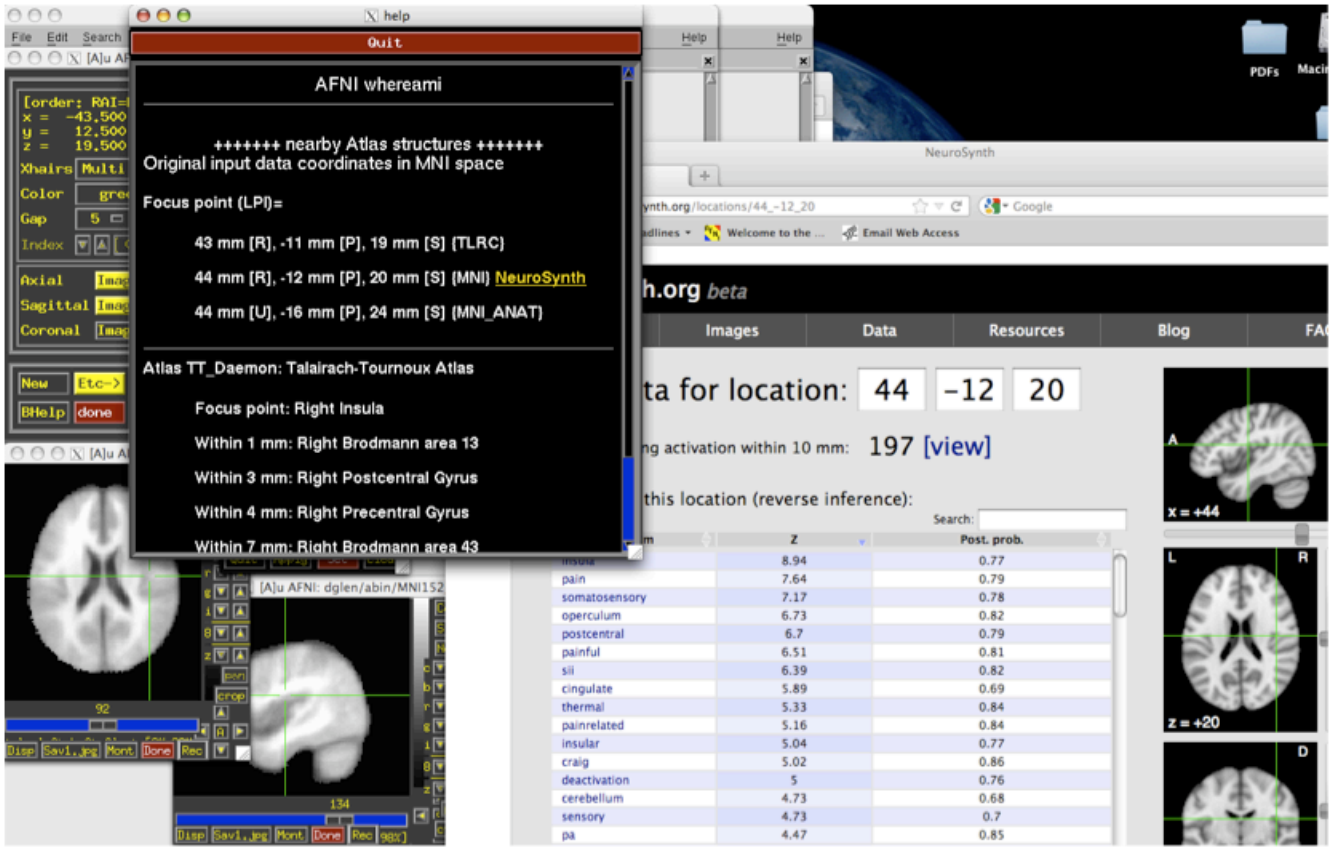
Manual segmentation



Overlay panel shows structure name. Now FreeSurfer segmentation can also be used in whereami

Web-based atlases

NeuroSynth, LinkrBrain, BrainMap.org, Allen Brain, ...



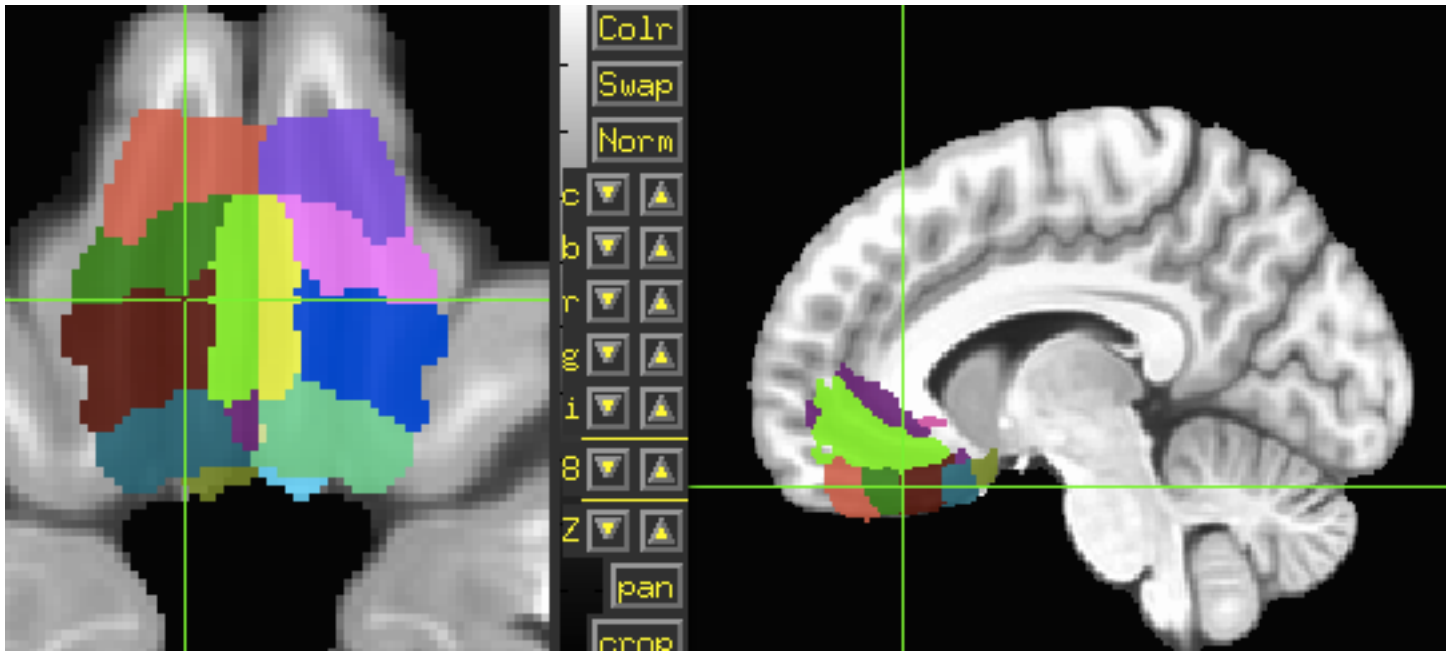
set AFNI_WEBBY_WAMI to YES and AFNI_NEUROSYNTH to YES

In Development

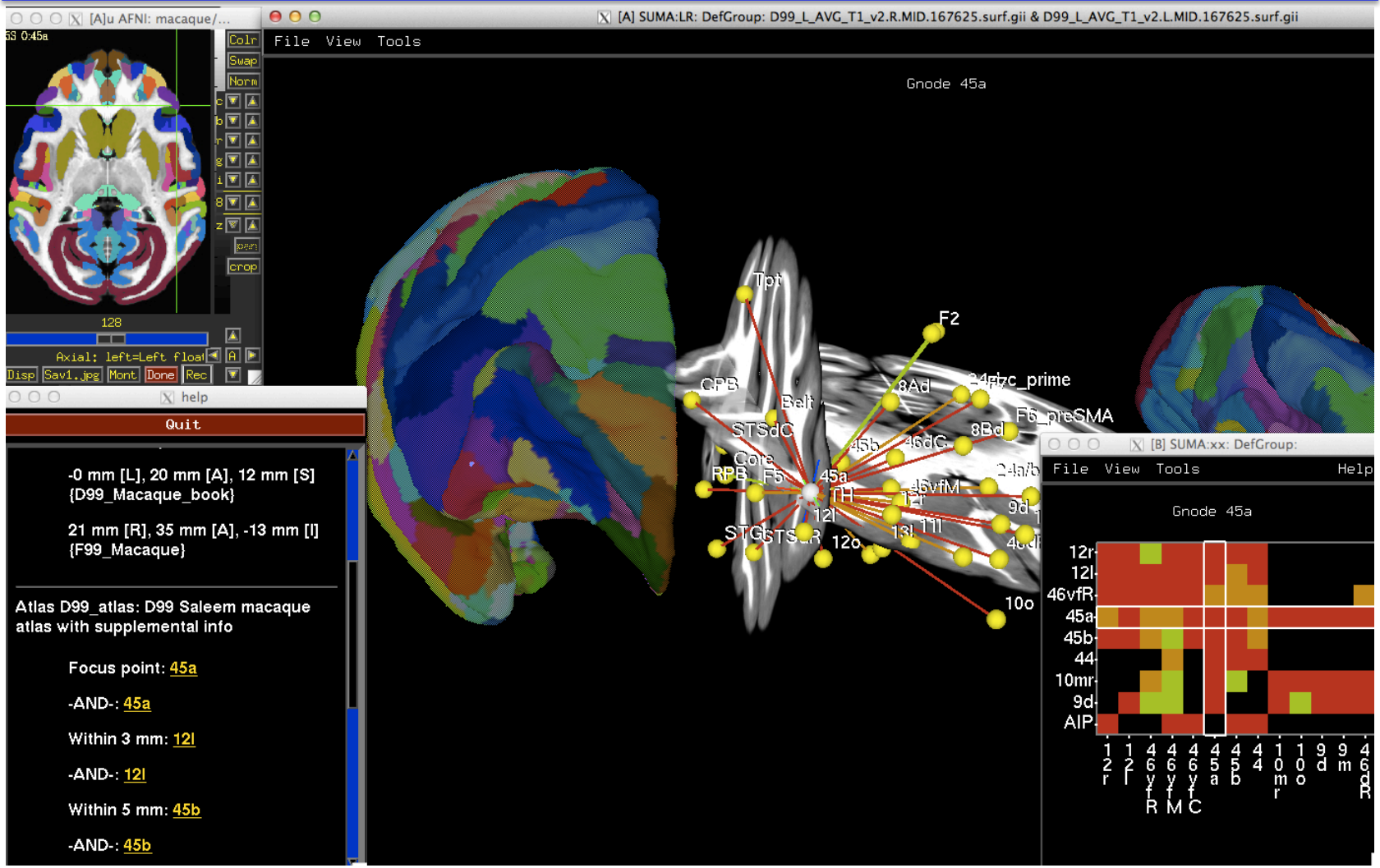
- New templates and template spaces – fully supported in AFNI
  - ✧ macaque
  - ✧ rat, mouse, human
  - ✧ pediatric, …
- On-the-fly transformations through all available template spaces
- Extra information about atlas structures

Upcoming atlases

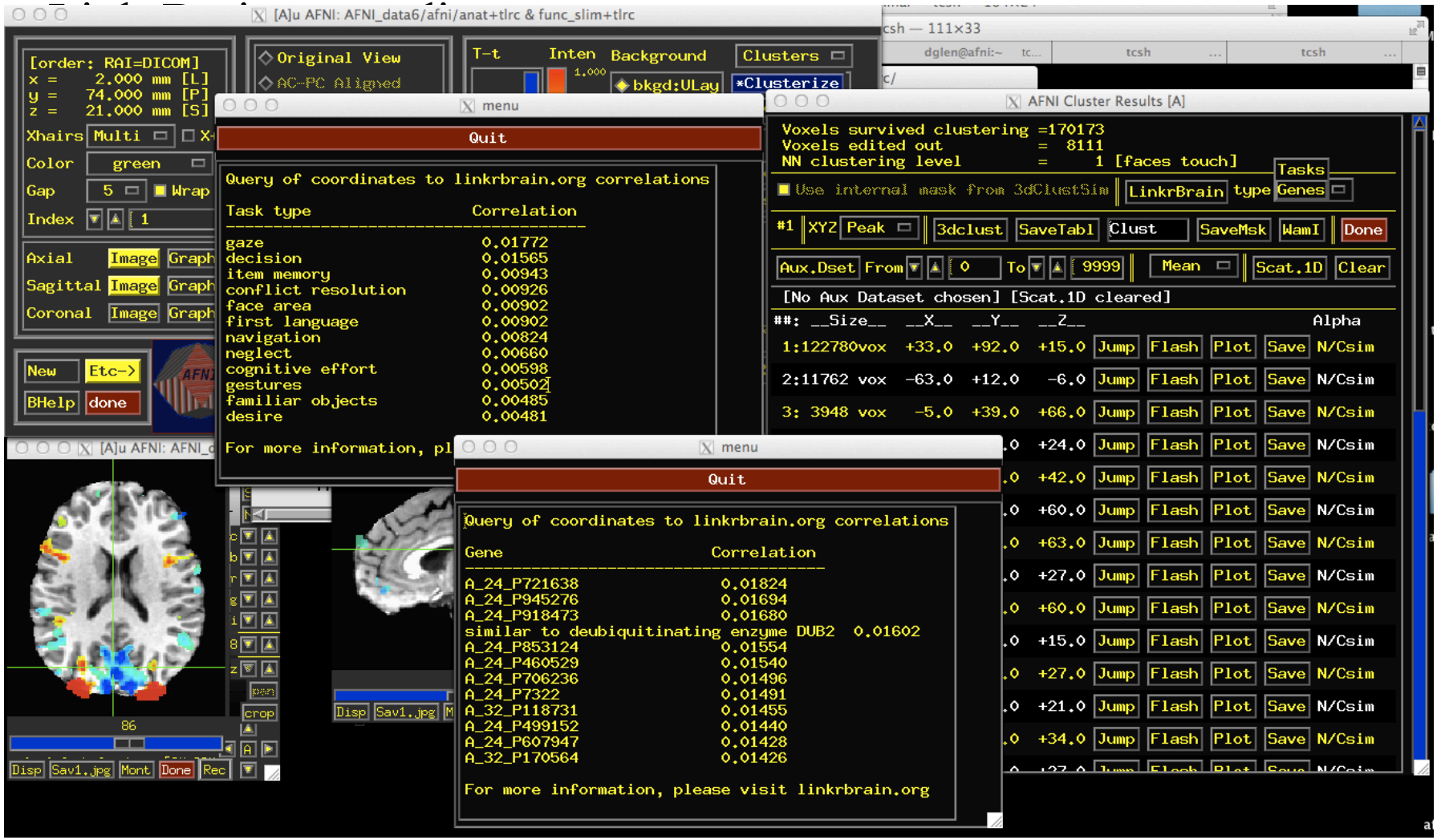
- ventromedial Prefrontral Cortex (vmPFC, Scott Mackey)
- Eickhoff-Zilles 2.0 cytoarchitectonic atlases
- DTI fiber atlas – Susumo Mori



Saleem macaque atlas – MRI, surfaces, connections, supplemental webpages (in development)



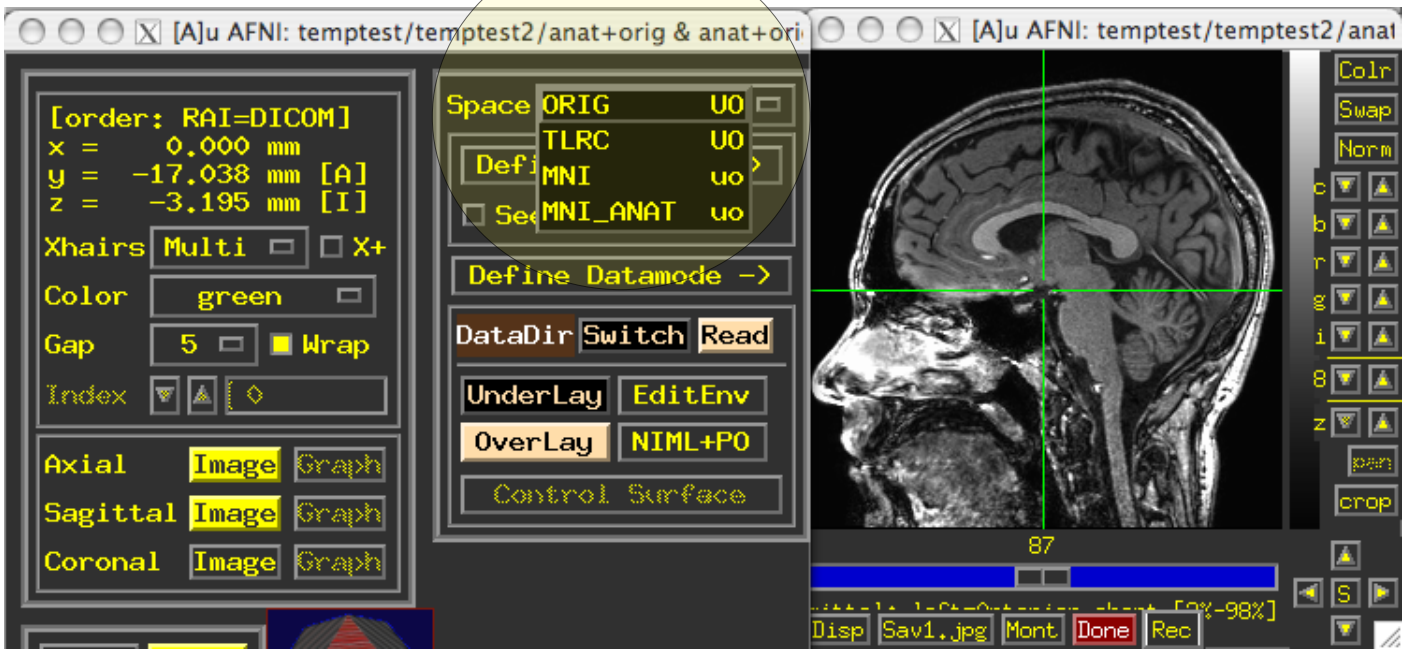
Web-based atlases



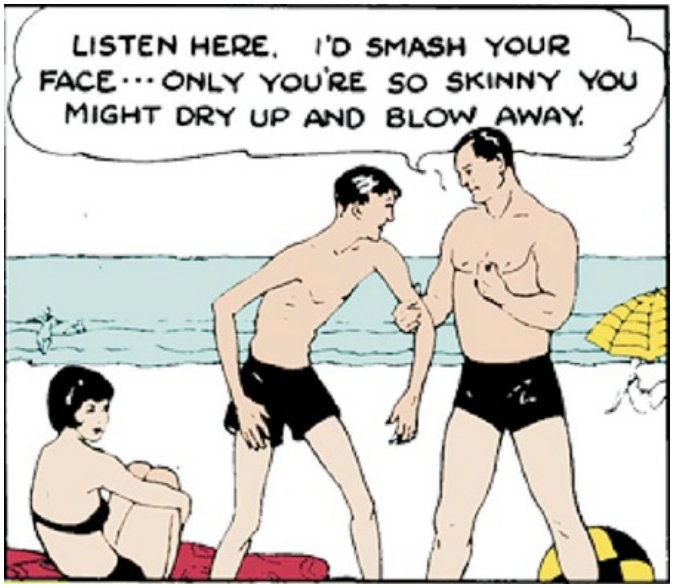
set AFNI_LINKRBRAIN to YES

Atlas Spaces – Warp-on-demand (in development)

Not just your father's Talairach anymore....



Atlas Conclusions and Questions

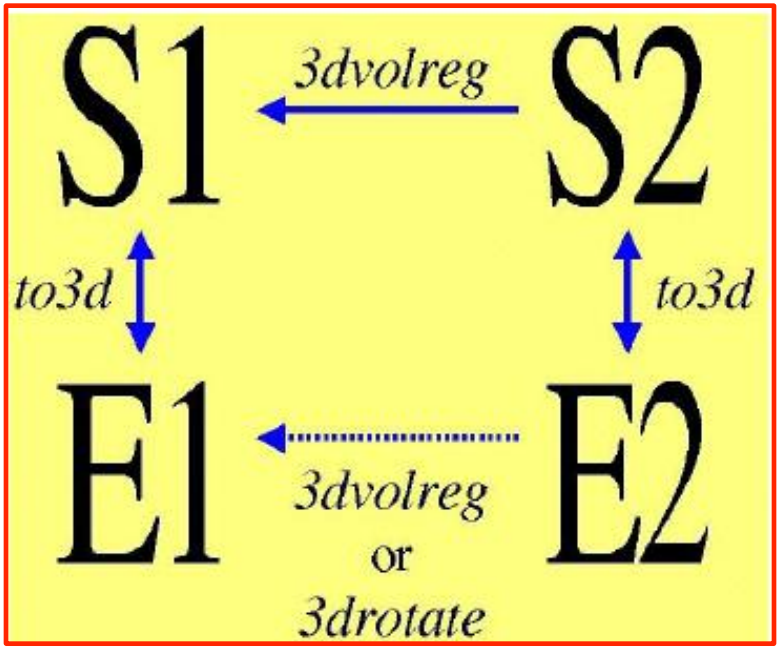


Charles Atlas

Appendix A

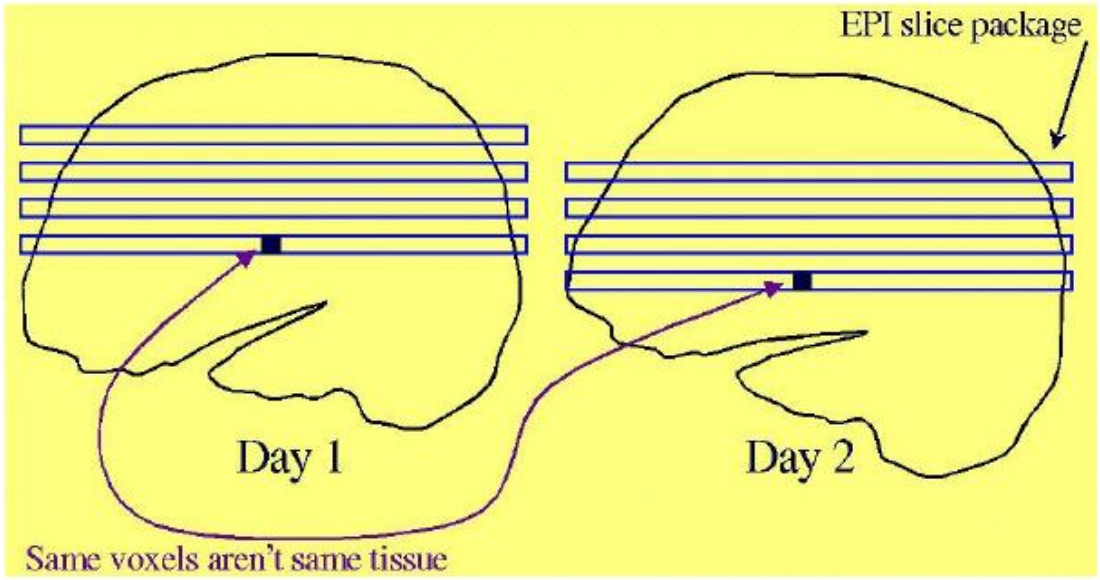
Inter-subject, inter-session registration

- Intra-subject, inter-session registration (for multi-day studies on same subject)
  - ✧ Longitudinal or learning studies; re-use of cortical surface models
  - ✧ Transformation between sessions is calculated by registering high-resolution anatomicals from each session
    - ➥ **`to3d`** defines defines relationship between EPI and SPGR in each session
    - ➥ **`3dvolreg`** computes relationship between sessions
    - ➥ So can transform EPI from session 2 to orientation of session 1

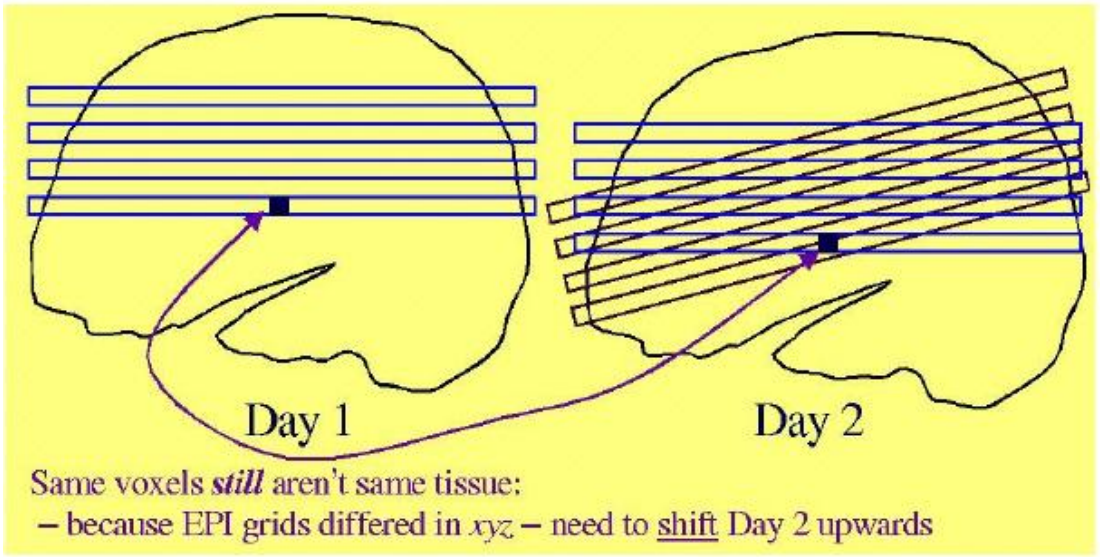


  - ✧ Issues in inter-session registration:
    - ➥ Subject's head will be positioned differently (in orientation and location)
      - ➪ xyz-coordinates and anatomy don't correspond
    - ➥ Anatomical coverage of EPI slices will differ between sessions
    - ➥ Geometrical relation between EPI and SPGR differs between session
    - ➥ Slice thickness may vary between sessions (try not to do this, OK?)

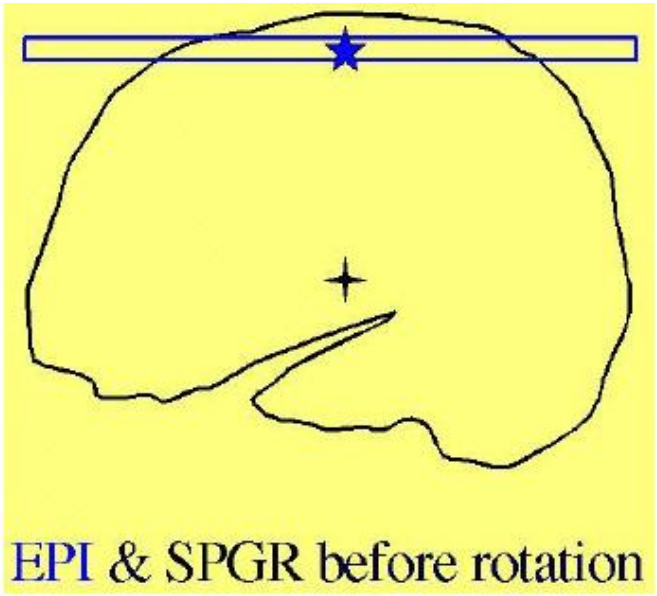
- Anatomical coverage differs



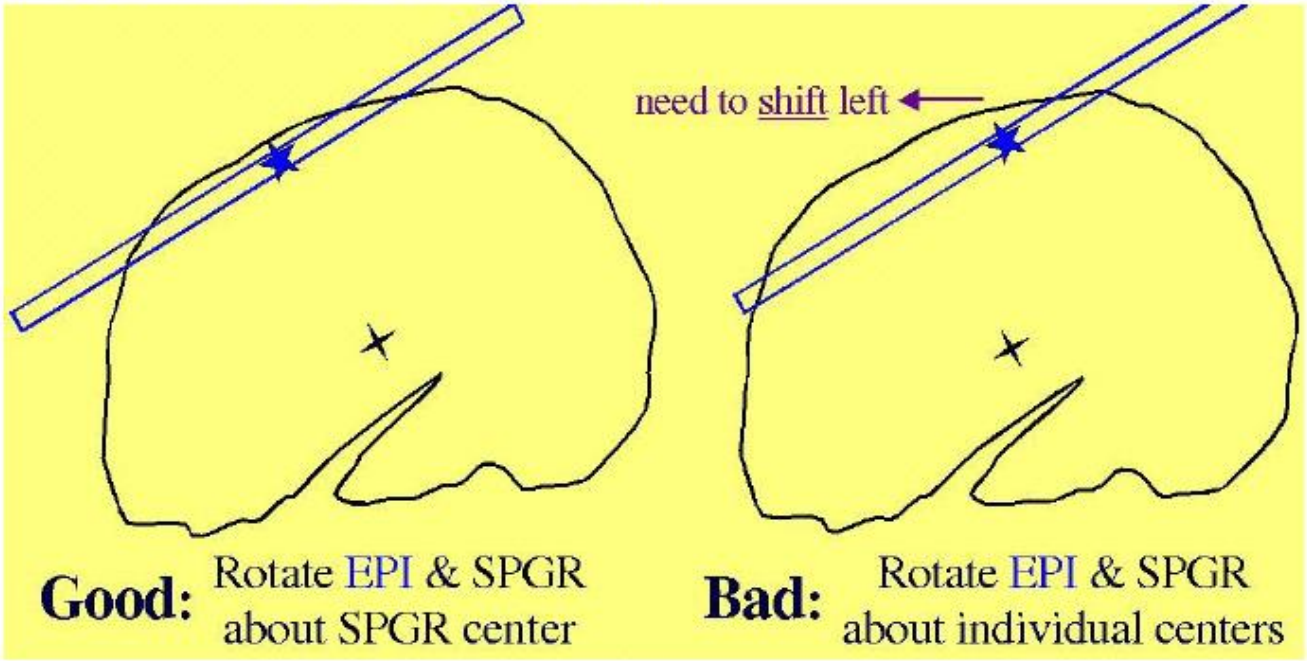
✧ At acquisition:
Day 2 is rotated relative to Day 1



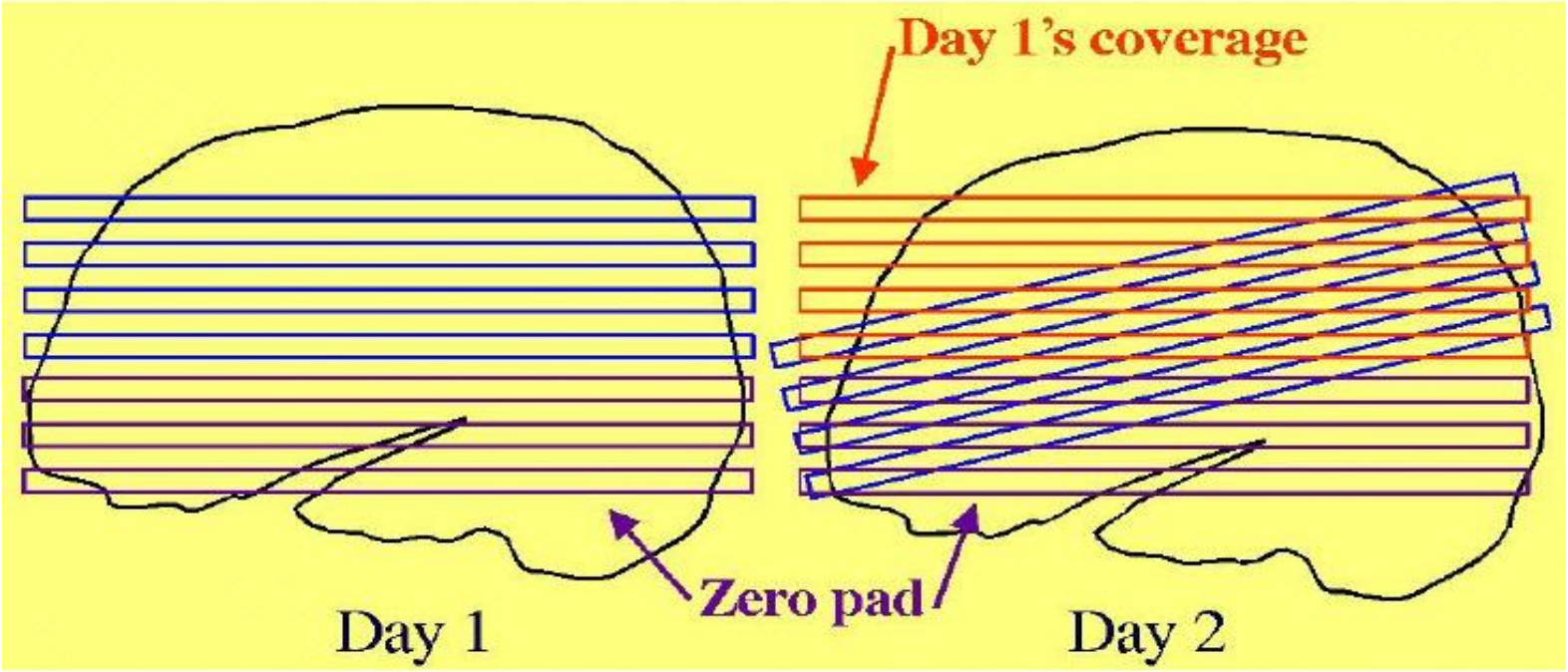
✧ After rotation to same orientation, then clipping to Day 2 xyz-grid



- Another problem: rotation occurs around center of individual datasets



- Solutions to these problems:
  - Add appropriate shift to E2 on top of rotation
    - Allow for xyz shifts between days (E1-E2), and center shifts between EPI and SPGR (E1-S1 and E2-S2)
  - Pad EPI datasets with extra slices of zeros so that aligned datasets can fully contain all data from all sessions
  - Zero padding of a dataset can be done in **`to3d`** (at dataset creation time), or later using **`3dZeropad`**
  - **`3dvolreg`** and **`3drotate`** can zero pad to make the output match a "grid parent" dataset in size and location



- Recipe for intra-subject S2-to-S1 transformation:
  1. Compute S2-to-S1 transformation:

```
3dvolreg -twopass -zpad 4 -base S1+orig \
         -prefix S2reg  S2+orig
```

-twopass allows for larger motions

  - Rotation/shift parameters are saved in **S2reg+orig.HEAD**
  2. If not done before (e.g., in **to3d**), zero pad E1 datasets:

```
3dZeropad -z 4 -prefix E1pad  E1+orig
```

  3. Register E1 datasets within the session:

```
3dvolreg -base 'E1pad+orig[4]' -prefix E1reg \
         E1pad+orig
```

  4. Register E2 datasets within the session, at the same time executing larger rotation/shift to session 1 coordinates that were saved in **S2reg +orig.HEAD**:

```
3dvolreg -base 'E2+orig[4]' \
  -rotparent S2reg+orig      \
  -gridparent E1reg+orig     \
  -prefix E2reg  E2reg+orig
```

These options put the aligned **E2reg** into the same coordinates and grid as **E1reg**

  - **-rotparent** tells where the inter-session transformation comes from
  - **-gridparent** defines the output grid location/size of new dataset
    - Output dataset will be shifted and zero padded as needed to lie on top of **E1reg+orig**

- Recipe above does not address problem of having different slice thickness in datasets of the same type (EPI and/or SPGR) in different sessions
  - Best solution: pay attention when you are scanning, and always use the same slice thickness for the same type of image
  - OK solution: use **3dZregrid** to linearly interpolate datasets to a new slice thickness
- Recipe above does not address issues of slice-dependent time offsets stored in data header from **to3d** (e.g., '**alt+z**')
  - After interpolation to a rotated grid, voxel values can no longer be said to come from a particular time offset, since data from different slices will have been combined
  - Before doing this spatial interpolation, it makes sense to time-shift dataset to a common temporal origin
  - Time shifting can be done with program **3dTshift**
    - Or by using the **-tshift** option in **3dvolreg**, which first does the time shift to a common temporal origin, then does the 3D spatial registration

- Further reading at the AFNI web site
  - File **README.registration** (plain text) has more detailed instructions and explanations about usage of **3dvolreg**
  - File **regnotes.pdf** has some background information on issues and methods used in FMRI registration packages

Appendix B

3dAllineate for the curious

3dAllineate:

More than you want to know

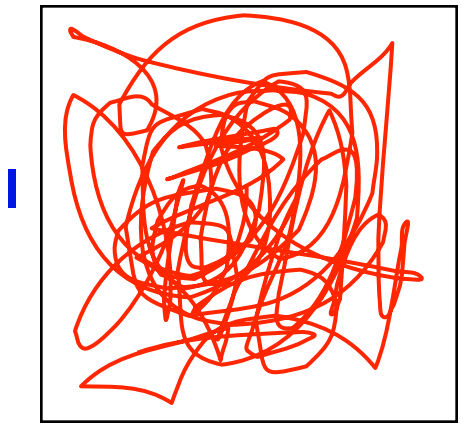
Algorithmic Features

- Uses Powell's NEWUOA software for minimization of general cost function
- Lengthy search for initial transform parameters if two passes of registration are turned on [which is the default]
  - ✧ Random and grid search through hundreds of parameter sets for 15 good (low cost) parameter sets
  - ✧ Optimize a little bit from each 'good' set, using blurred images
    - ➥ Blurring the images means that small details won't prevent a match
  - ✧ Keep best 4 of these parameter sets, and optimize them some more [keeping 4 sets is the default for `-twobest` option]
    - ➥ Amount of blurring is reduced in several stages, followed by re-optimization of the transformation parameter sets on these less blurred images
    - ➥ `-twofirst` does this for first sub-brick, then uses the best parameter sets from the first sub-brick as the starting point for the rest of the sub-bricks [the default]
  - ✧ Use best 1 of these parameter sets as starting point for fine (un-blurred) parameter optimization
    - ➥ The slowest part of the program

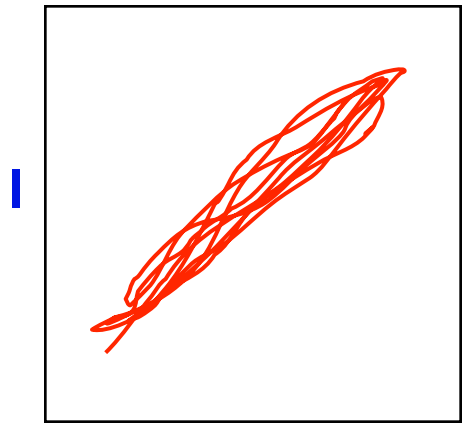
Algorithmic Features

- Goal is to find parameter set **w** such that **E**[**J**(**x**) , **I**(**T**(**x**,**w**))] is small
  - **T**(**x**,**w**) = spatial transformation of **x** given **w**
  - **J**() = base image, **I**() = target image, **E**[] = cost function
- For each **x** in base image space, compute **T**(**x**,**w**) and then interpolate **I**() at those points
  - For speed, program doesn't use all points in **J**(), just a scattered collection of them, selected from an automatically generated mask
    - Mask can be turned off with `-noauto` option
    - At early stages, only a small collection of points [default=23456] is used when computing **E**[]
    - At later stages, more points are used, for higher accuracy
      - Recall that each stage is less blurred than the previous stages
  - Large fraction of CPU time is spent in interpolation of image **I**() over the collection of points used to compute **E**[]

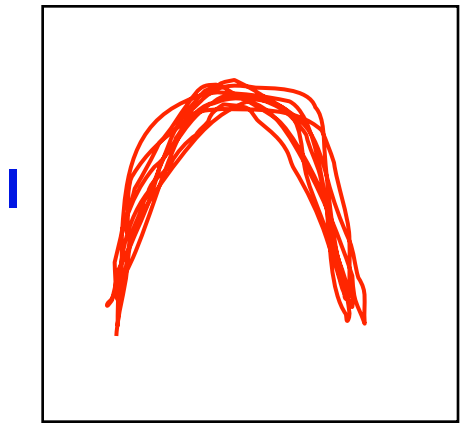
- Histogram cartoons:



• **J** not useful in predicting **I**

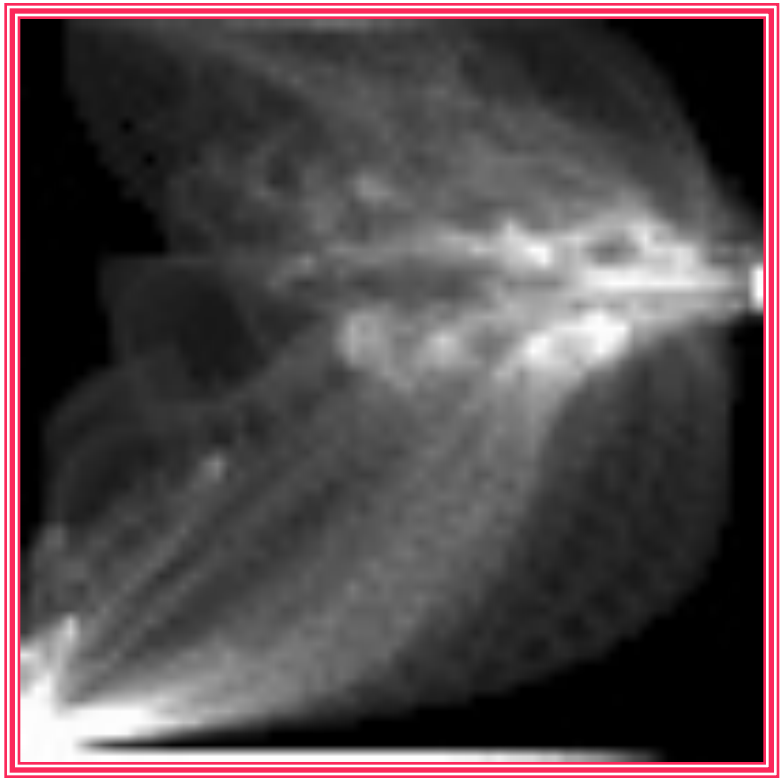


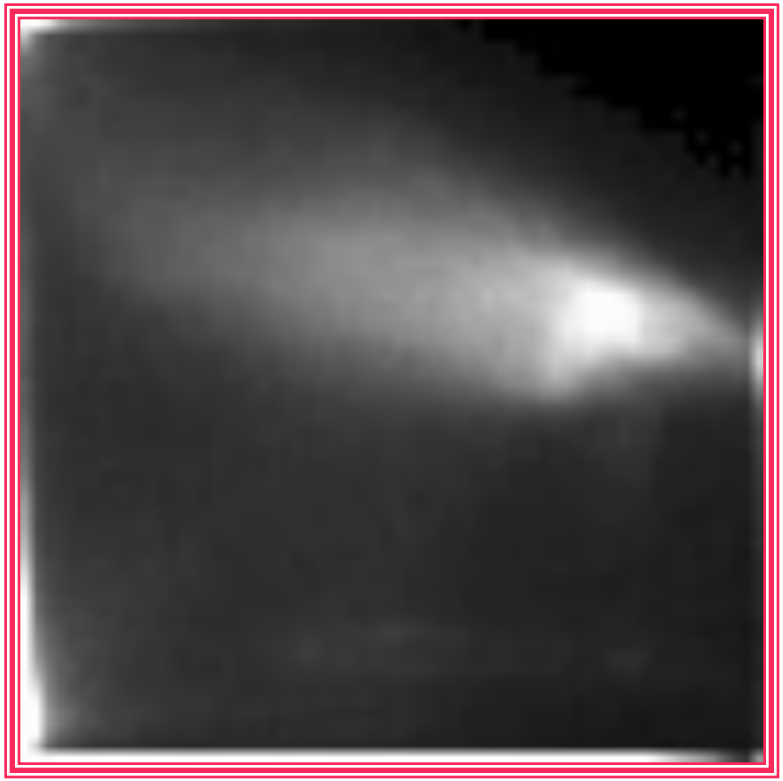
• **I** can be accurately predicted from **J** with a linear formula:
`-leastsq` is OK



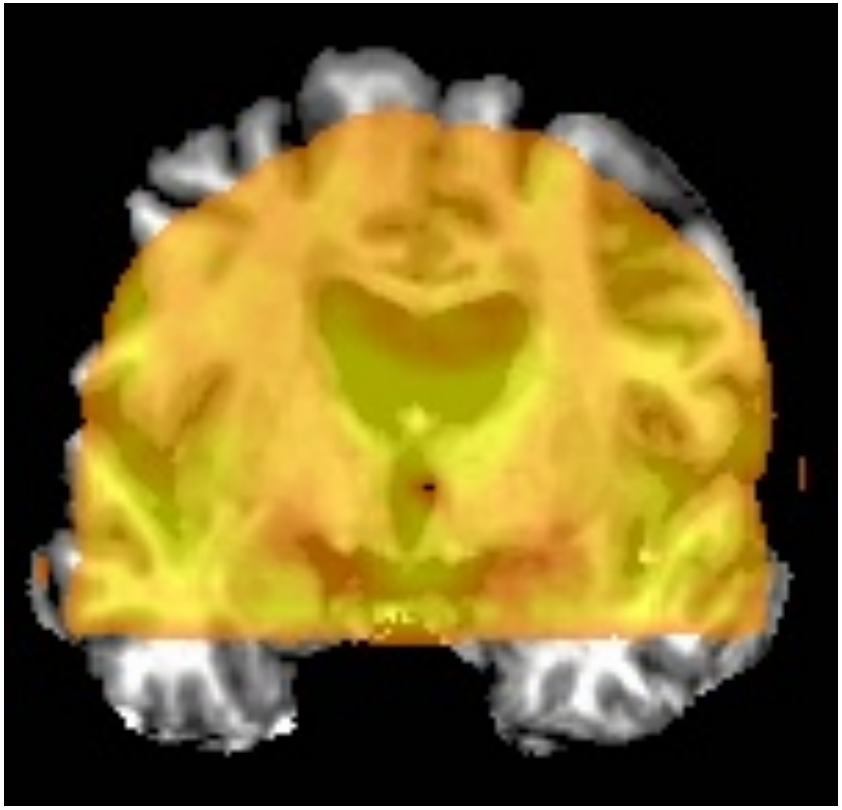
• **I** can be accurately predicted from **J**, but nonlinearly:
`-leastsq` is BAD

- Actual histograms from a registration example
  - ✧ **J(x)** = `3dSkullStrip`-ed MPRAGE **I(x)** = EPI volume

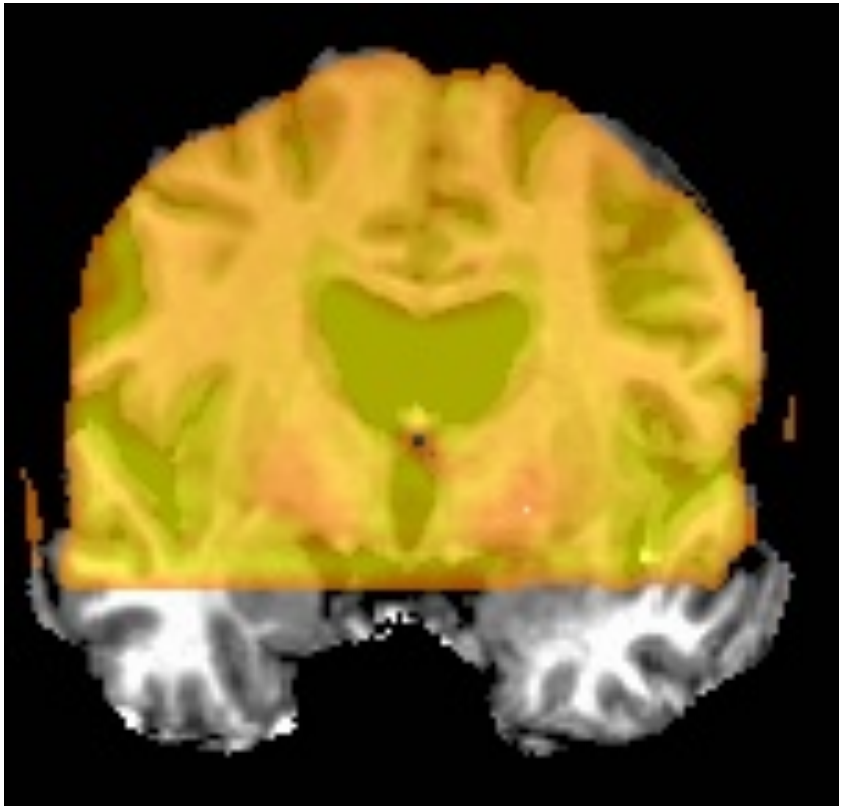




- grayscale underlay = **J(x)** = `3dSkullStrip`-ed MPRAGE
- color overlay = **I(x)** = EPI volume



• Before alignment

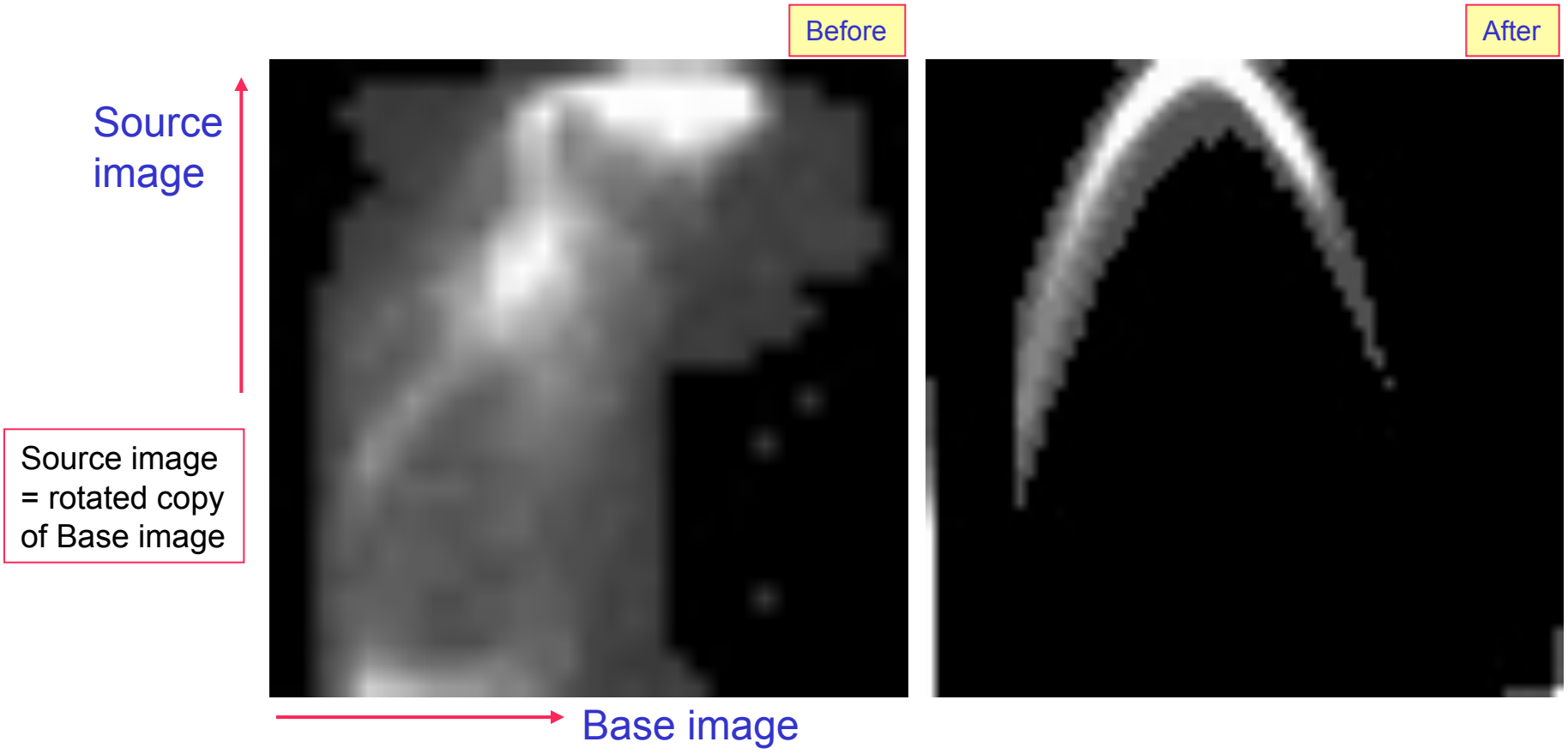


• After alignment
(using **`-mutualinfo`**)

- Other **3dAllineate** capabilities:
  - Save transformation parameters with option **-1Dfile** in one program run
    - Re-use them in a second program run on another input dataset with option **-1Dapply**
  - Interpolation: linear (polynomial order = 1) during alignment
    - To produce output dataset: polynomials of order 1, 3, or 5
- Algorithm details:
  - Initial alignment starting with many sets of transformation parameters, using only a limited number of points from smoothed images
  - The best (smallest **E**) sets of parameters are further refined using more points from the images and less blurring
  - This continues until the final stage, where many points from the images and no blurring is used
- So why not **3dAllineate** all the time?
  - Alignment with cross-modal cost functions do not always converge as well as those based on least squares.
    - See Appendix B for more info.
    - Improvements are still being introduced

Cost Functions

- Except for least squares (actually, `ls` minimizes **E**=1.0−Pearson correlation coefficient), all cost functions are computed from 2D joint histogram of **J**(**x**) and **I**(**T**(**x**,**w**))
  - ✧ Start and final histograms can be saved using hidden option `-savehist`



Histogram Based Cost Functions

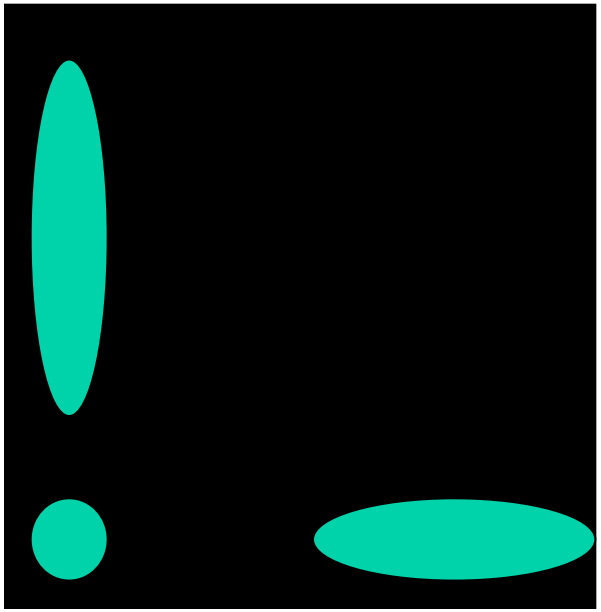
- Goal is to make 2D histogram become ‘simple’ in some sense, as a measurement of ‘predictability’ between **J**(**x**) and **I**(**T**(**x**,**w**))
- Entropy H() of a histogram (finite number of bins):
  - $\{p_i\}$ = probabilities of index i occuring
  - $H(\{p_i\}) = -\Sigma_i\ p_i \log_2(p_i) > 0$
  - $H(\{p_i\})$ = Number of bits needed to encode a single value randomly drawn from the probabilities $\{p_i\}$
  - Smaller entropy H means the values are ‘simpler’ to encode
    - Largest H is for uniform histogram (all p_i equal)

Mutual Information

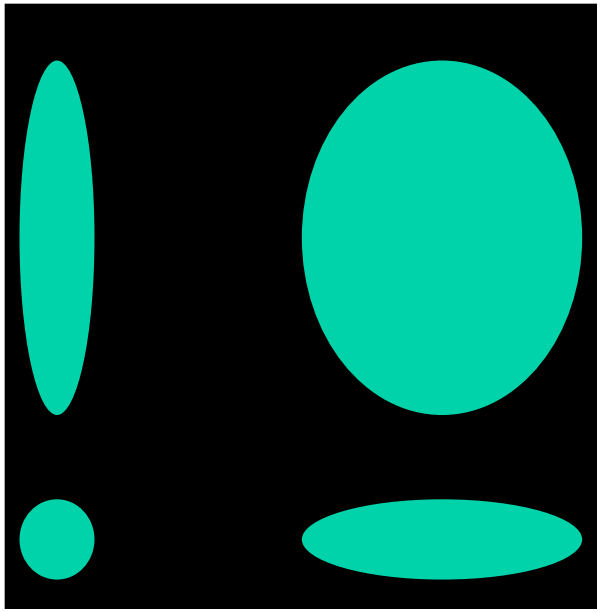
- Entropy of 2D histogram
  - $H(\{r_{ij}\}) = -\Sigma_{ij}\, r_{ij} \log_2(r_{ij})$
  - Number of bits needed to encode value pairs (i,j)
- **M**utual **I**nformation between two distributions
  - Marginal (1D) histograms $\{p_i\}$ and $\{q_j\}$
  - $MI = H(\{p_i\}) + H(\{q_j\}) - H(\{r_{ij}\})$
  - Number of bits required to encode 2 values separately minus number of bits required to encode them together (as a pair)
  - If 2D histogram is independent ($r_{ij} = p_i \cdot q_j$) then MI = 0 = no gain from joint encoding
- `3dAllineate` minimizes **E**[**J**,**I**] = –MI(J,**I**) with `-cost mi`

Normalized MI

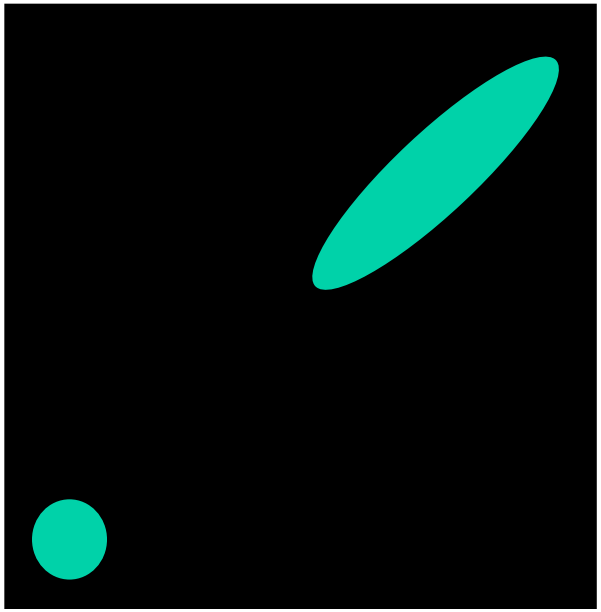
- NMI = $H(\{r_{ij}\}) \;/\; [\, H(\{p_i\}) + H(\{q_j\}) \,]$
  - ✧ Ratio of number of bits to encode value pair divided by number of bits to encode two values separately
  - ✧ Minimize NMI with `-cost nmi`
- Some say NMI is more robust for registration than MI, since MI can be large when there is no overlap between the two images



NO overlap



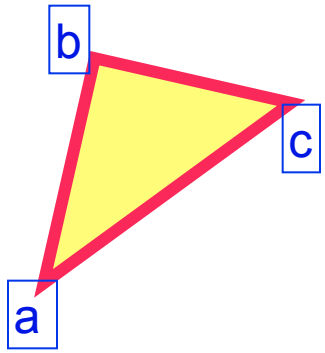
BAD overlap



100% overlap

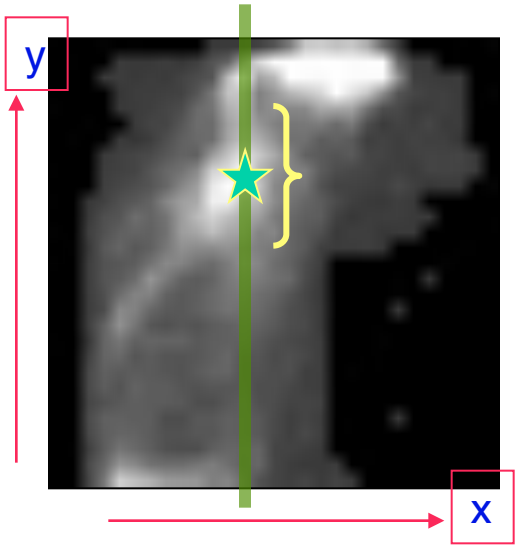
Hellinger Metric

- MI can be thought of as measuring a 'distance' between two 2D histograms: the joint distribution $\{r_{ij}\}$ and the product distribution $\{p_i \cdot q_j\}$
  - MI is not a 'true' distance: it doesn't satisfy triangle inequality d(a,b)+d(b,c) > d(a,c)
- Hellinger metric is a true distance in distribution "space":
  - $HM = \sum_{ij} [\sqrt{r_{ij}} - \sqrt{(p_i \cdot q_j)}]^2$
  - **3dAllineate** minimizes –HM with **-cost hel**
  - This is the default cost function



Correlation Ratio

- Given 2 (non-independent) random variables x and y
  - ✧ Exp[ylx] is the expected value (mean) of y for a fixed value of x
    - ➥ Exp[alb] ≡ Average value of 'a', given value of 'b'
  - ✧ Var(ylx) is the variance of y when x is fixed = amount of uncertainty about value of y when we know x
    - ➥ v(x) ≡ Var(ylx) is a function of x only

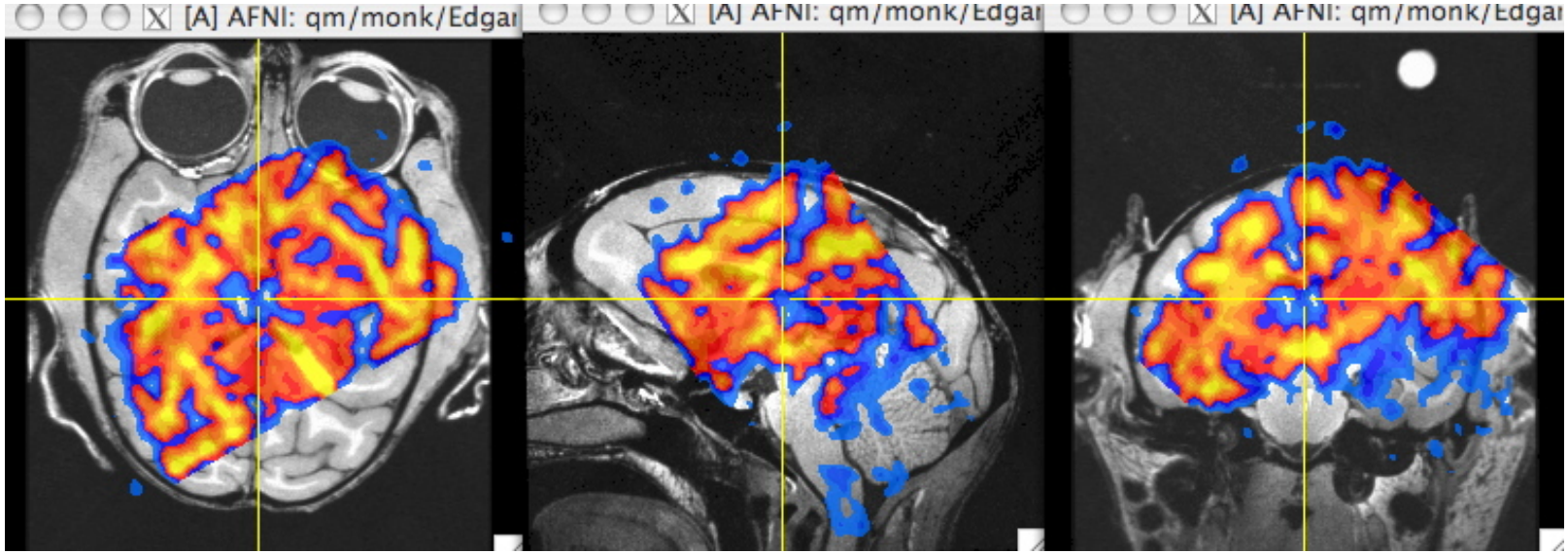


- CR(x,y) ≡ 1 – Exp[v(x)] ⁄ Var(y)
  - Relative reduction in uncertainty about value of y when x is known; large CR means Exp[ylx] is a good prediction of the value of y given the value of x
    - Does ***not*** say that Exp[xly] is a good prediction of the x given y
  - CR(x,y) is a generalization of the Pearson correlation coefficient, which assumes that Exp[ylx] = $\alpha \cdot x + \beta$

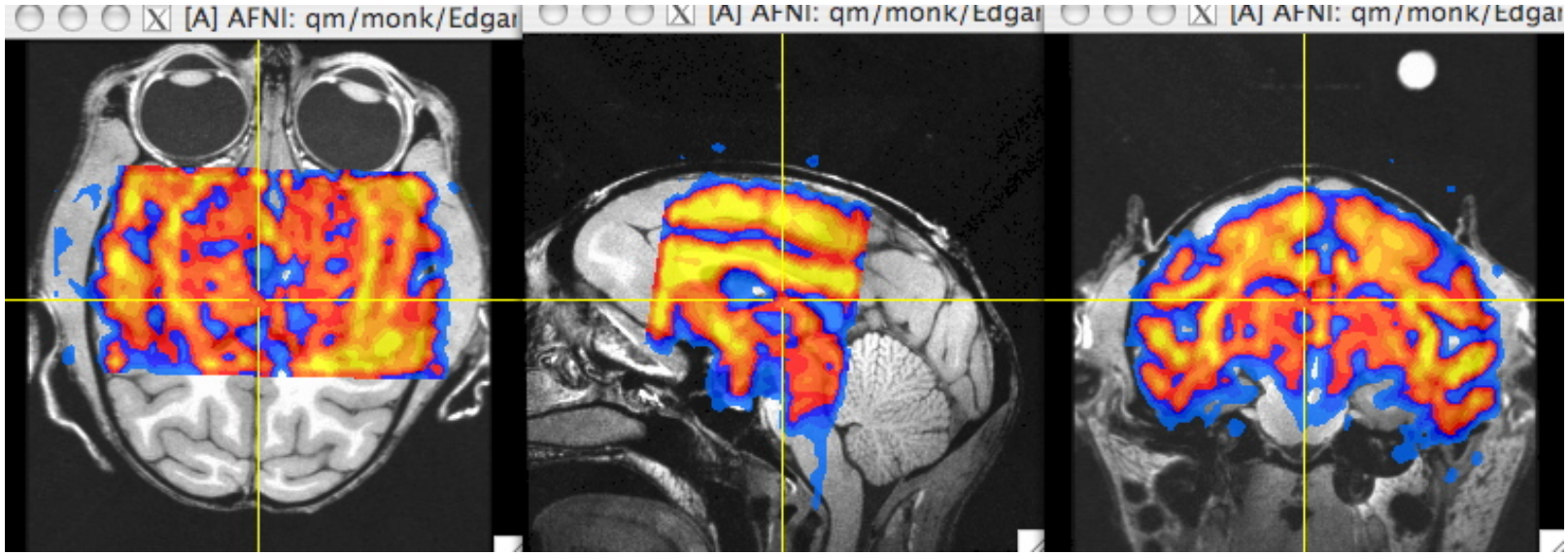
3dAllineate's Symmetrical CR

- First attempt to use CR in 3dAllineate didn't give good results
- Note asymmetry: CR(x,y) ≠ CR(y,x)
- 3dAllineate now offers two different symmetric CR cost functions:
  - ✧ Compute both unsymmetric CR(x,y) and CR(y,x), then combine by Multiplying or Adding:
  - ✧ $CRm(x,y) = 1 - [\ Exp(v(x)) \cdot Exp(v(y))\ ] / [\ Var(y) \cdot Var(x)\ ]$
    $= CR(x,y) + CR(y,x) - CR(x,y) \cdot CR(y,x)$
  - ✧ $CRa(x,y) = 1 - {}^1/_2\,[\ Exp(v(x)) / Var(y)\ ] - {}^1/_2\,[Exp(v(y)) / Var(x)\ ]$
    $= [\ CR(x,y) + CR(y,x)\ ] / 2$
  - ✧ These work better than CR(J,I) in my test problems
- If Exp[y|x] can be used to predict y and/or Exp[x|y] can be used to predict x, then crM(x,y) will be large (close to 1)
- 3dAllineate minimizes 1−CRm(J,I) with option -cost crM
- 3dAllineate minimizes 1−CRa(J,I) with option -cost crA
- 3dAllineate minimizes 1−CR(J,I) with option -cost crU

Test: Monkey EPI - Anat

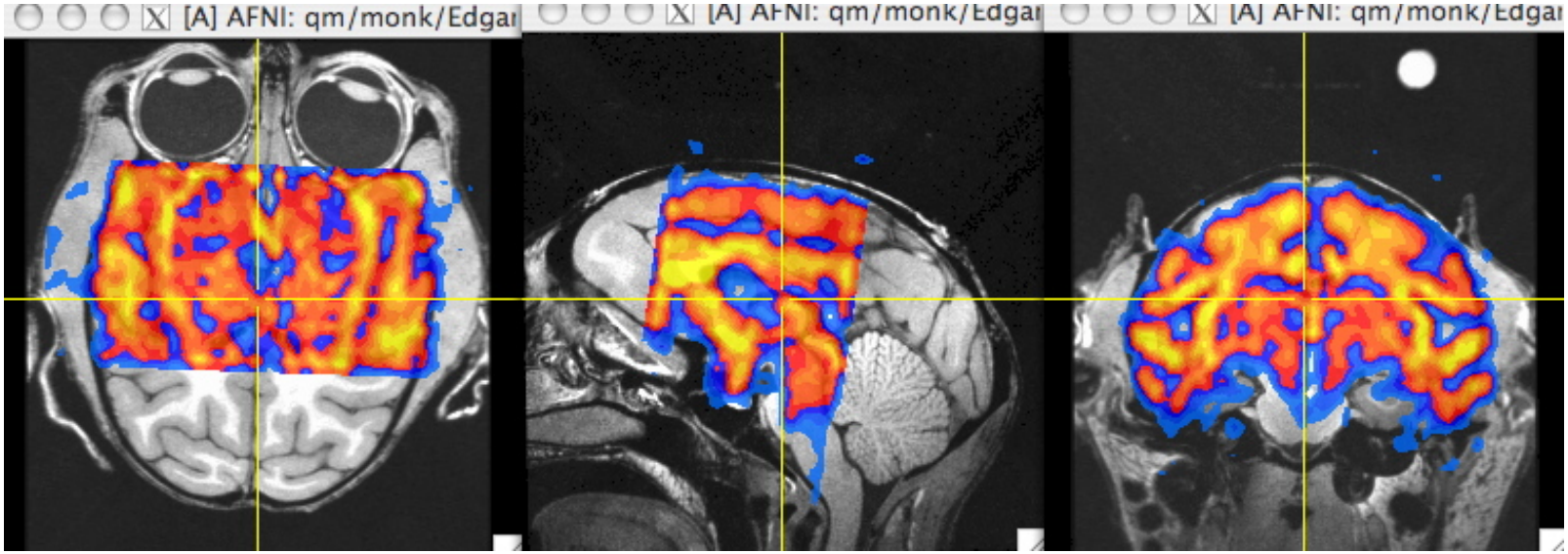


6 DOF
CRm

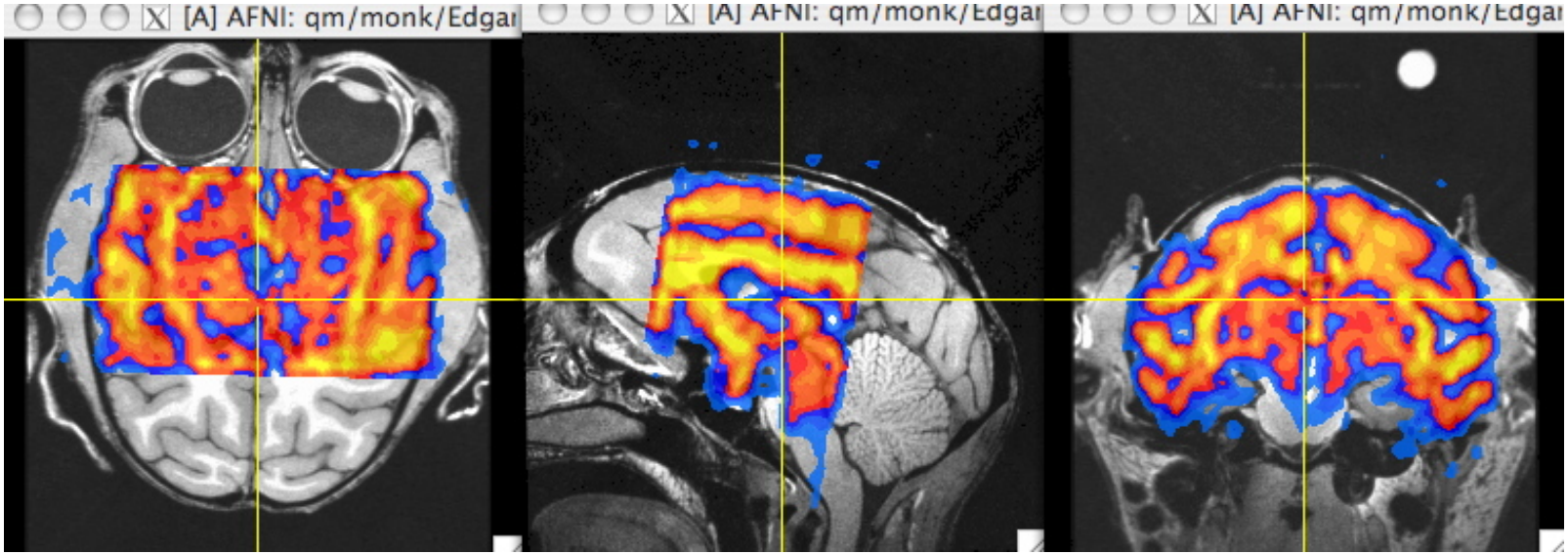


6 DOF
NMI

Test: Monkey EPI - Anat

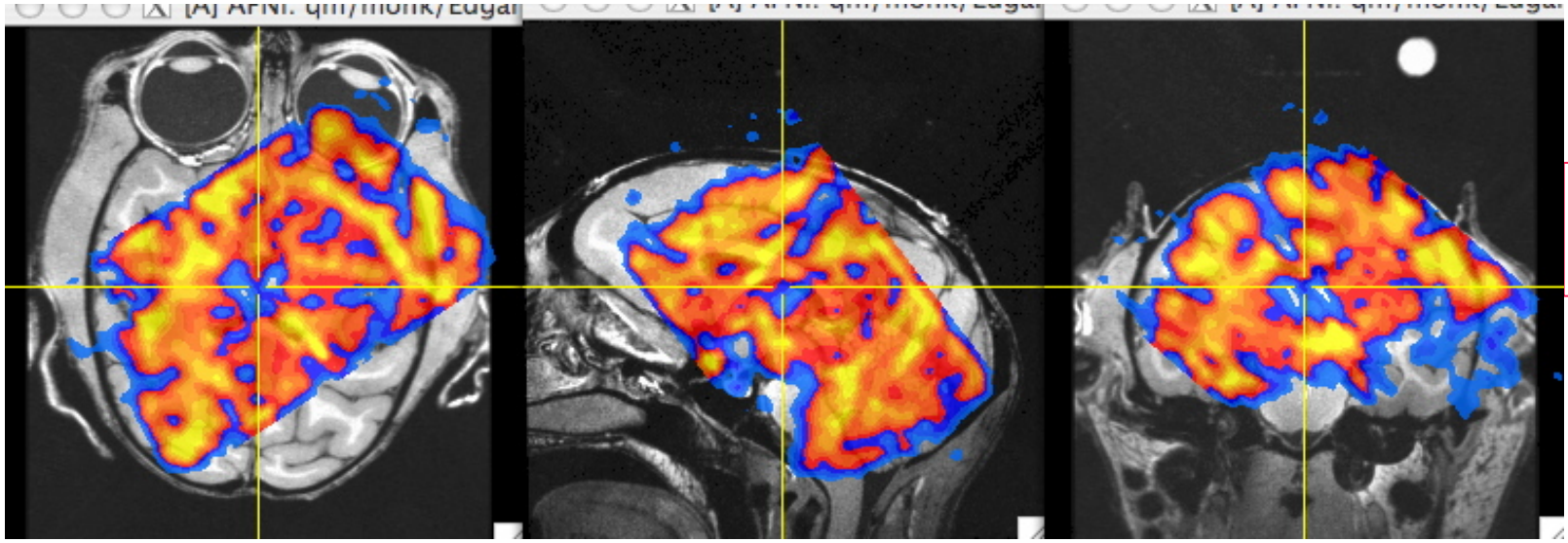


6 DOF
HEL

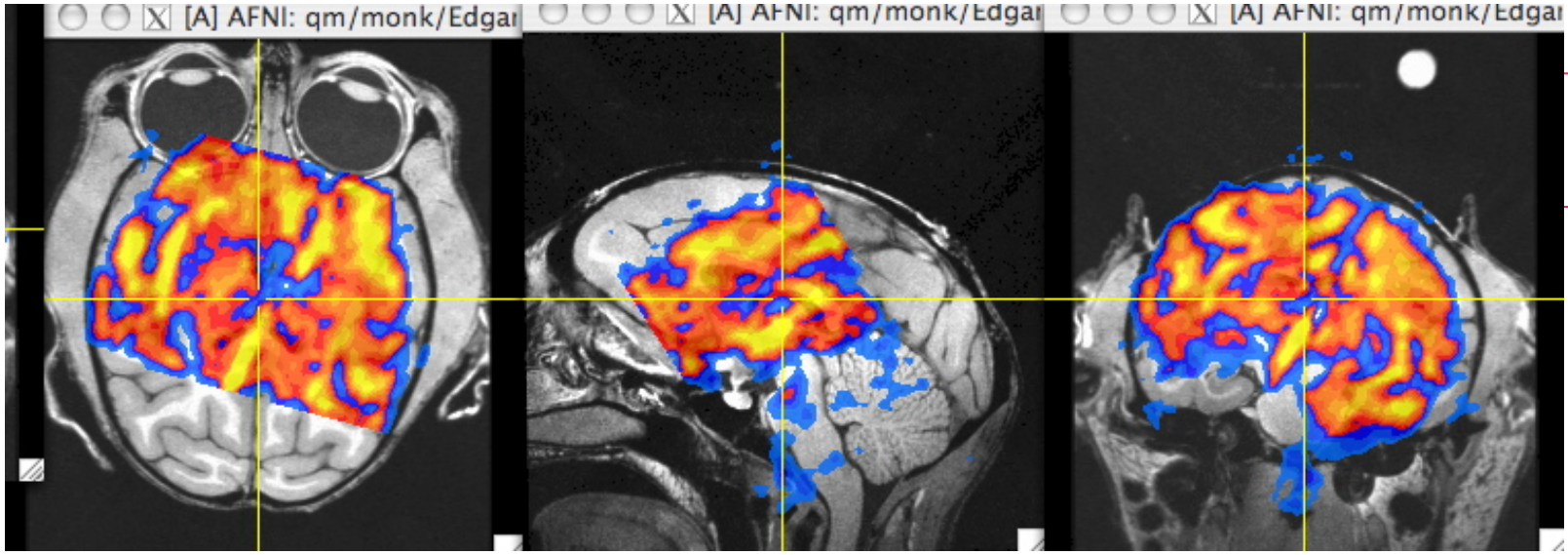


6 DOF
MI

Test: Monkey EPI - Anat

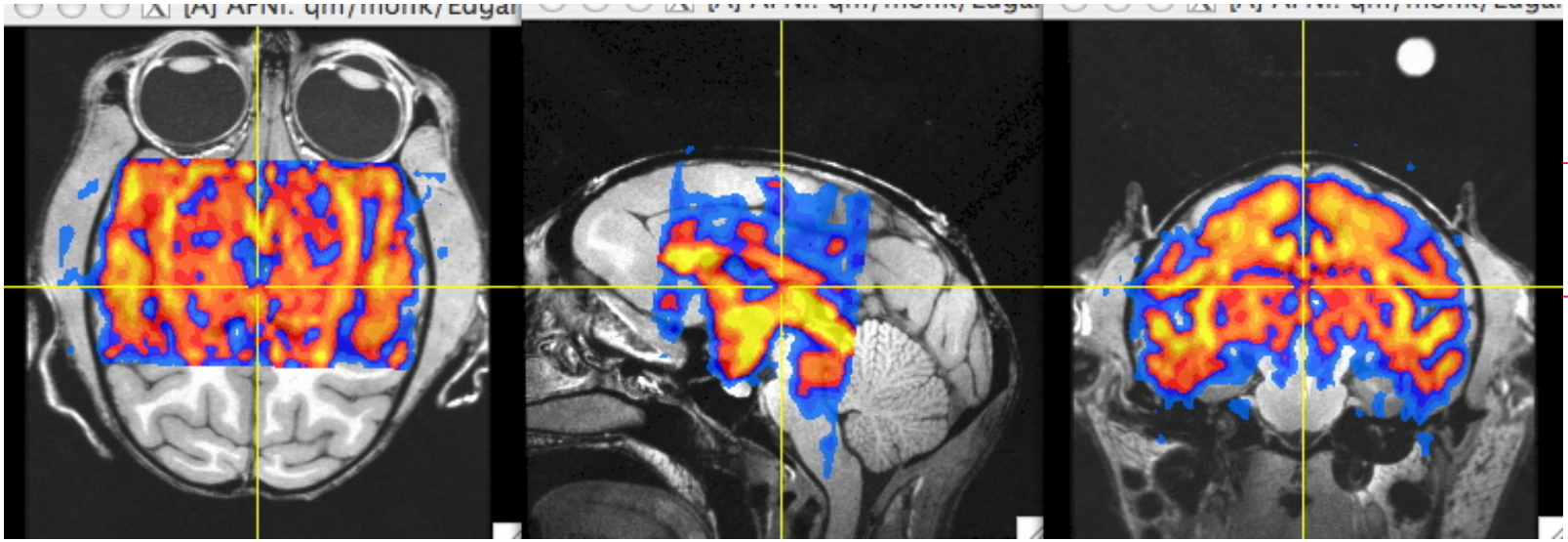


11 DOF
CRm

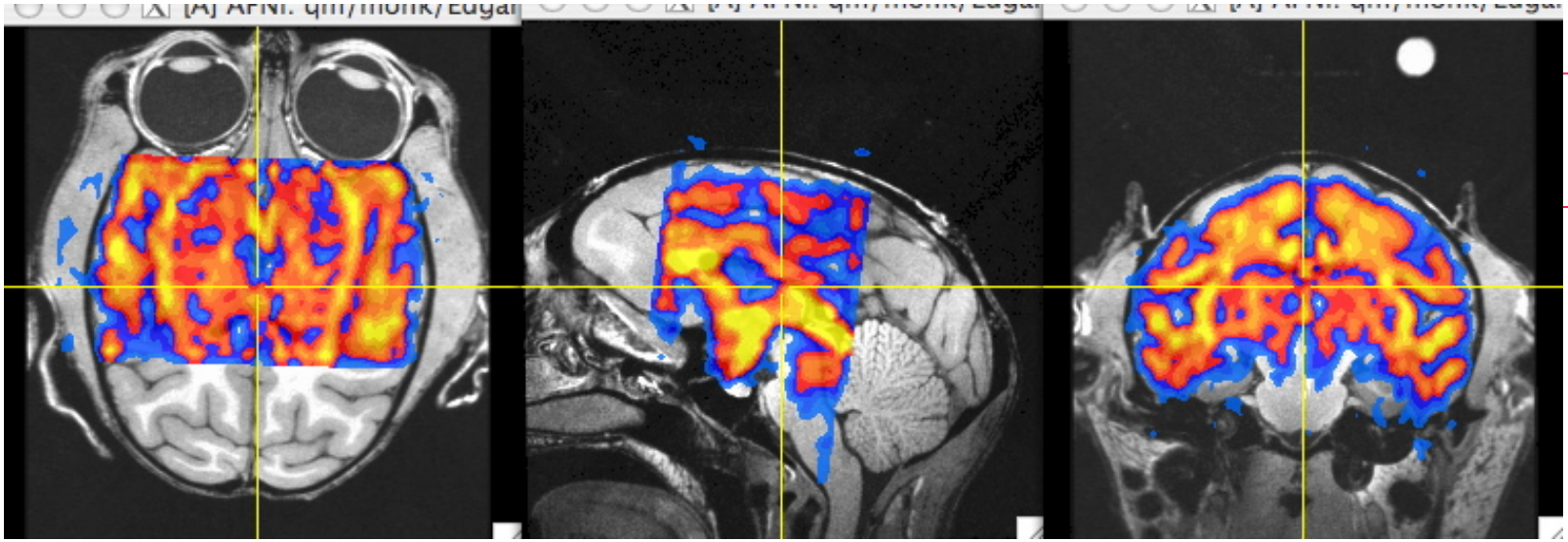


11 DOF
NMI

Test: Monkey EPI - Anat



11 DOF HEL

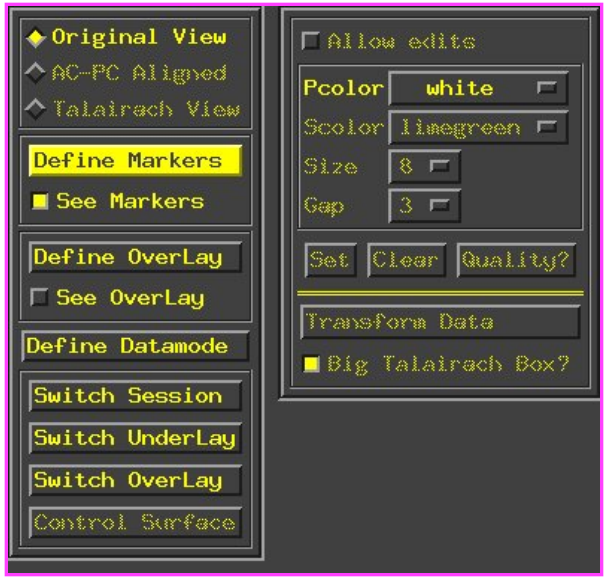


11 DOF MI

Appendix C

Talairach Transform from the days of yore

- **Listen up folks, IMPORTANT NOTE:**
  - Have you ever opened up the **[Define Markers]** panel, only to find the AC-PC markers ***missing*** , like this:



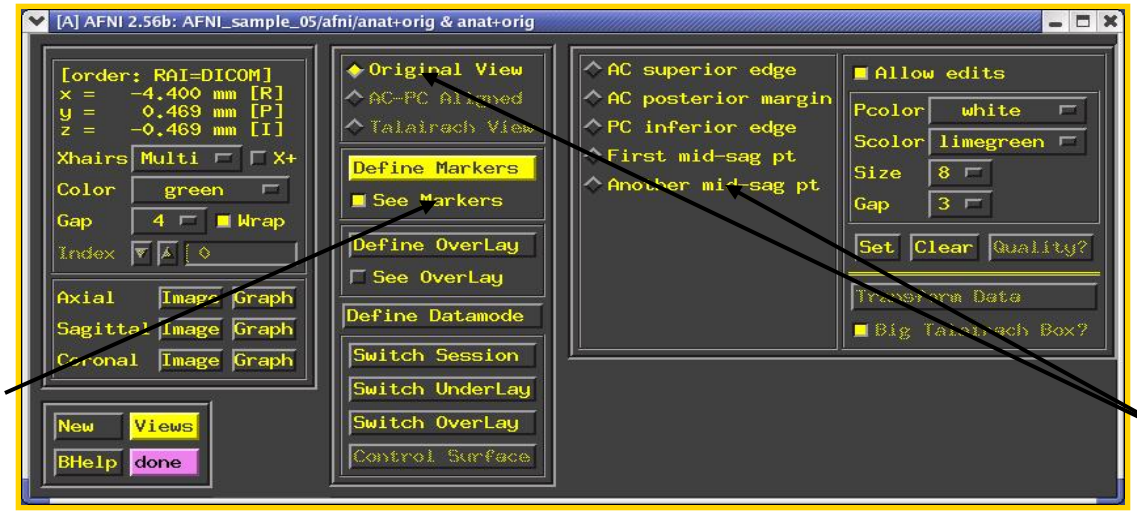
Gasp! Where did they go?

  - There are a few reasons why this happens, but usually it's because you've made a copy of a dataset, and the AC-PC marker tags weren't created in the copy, resulting in the above dilemma.
    - In other cases, this occurs when **afni** is launched without any datasets in the directory from which it was launched (oopsy, your mistake).
  - If you do indeed have an AFNI dataset in your directory, but the markers are missing and you want them back, run **3drefit** with the **-markers** options to create an empty set of AC-PC markers. Problem solved!

```
3drefit -markers <name of dataset>
```

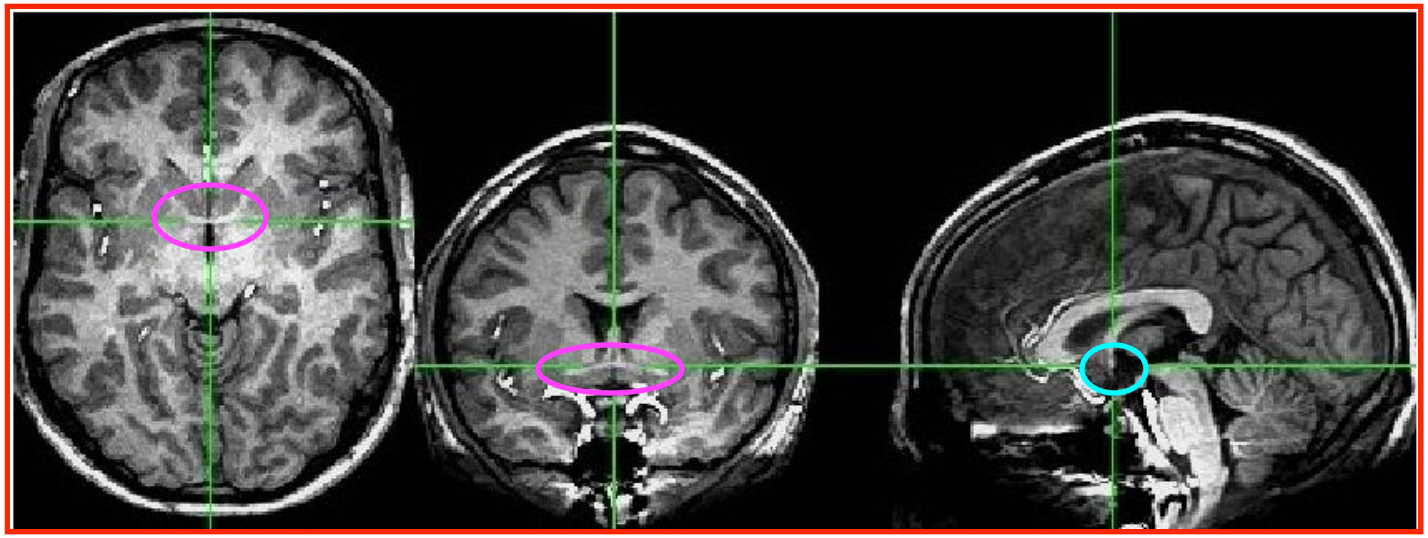
- **Class Example - Selecting the ac-pc markers:**
  - **cd AFNI_data1/demo_tlrc** ⇒ Descend into the `demo_tlrc/` subdirectory
  - **afni &** ⇒ This command launches the AFNI program
    - The "**&**" keeps the UNIX shell available in the background, so we can continue typing in commands as needed, even if AFNI is running in the foreground
  - Select dataset **anat+orig** from the **[Switch Underlay]** control panel

Press IN to view markers on brain volume



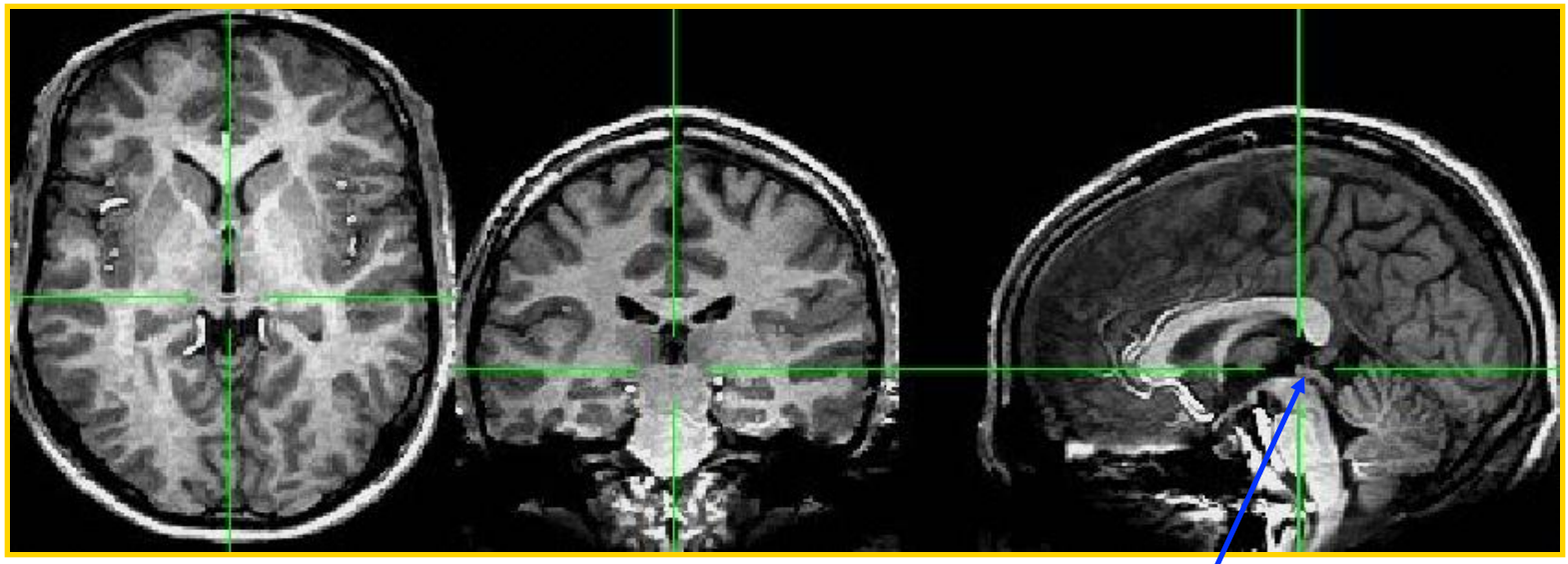
The AC-PC markers appear only when the orig view is highlighted

  - Select the **[Define Markers]** control panel to view the 5 markers for ac-pc alignment
  - Click the **[See Markers]** button to view the markers on the brain volume as you select them
  - Click the **[Allow edits]** button in the ac-pc GUI to begin marker selection



- First goal is to mark top middle and rear middle of **AC**
  - **Sagittal**: look for AC at bottom level of corpus callosum, below fornix
  - **Coronal**: look for “mustache”; **Axial**: look for inter-hemispheric connection
  - Get AC centered at focus of crosshairs (in Axial and Coronal)
  - Move superior until AC disappears in Axial view; then inferior 1 pixel
  - Press IN **[AC superior edge]** marker toggle, then **[Set]**
  - Move focus back to middle of AC
  - Move posterior until AC disappears in Coronal view; then anterior 1 pixel
  - Press IN **[AC posterior margin]**, then **[Set]**

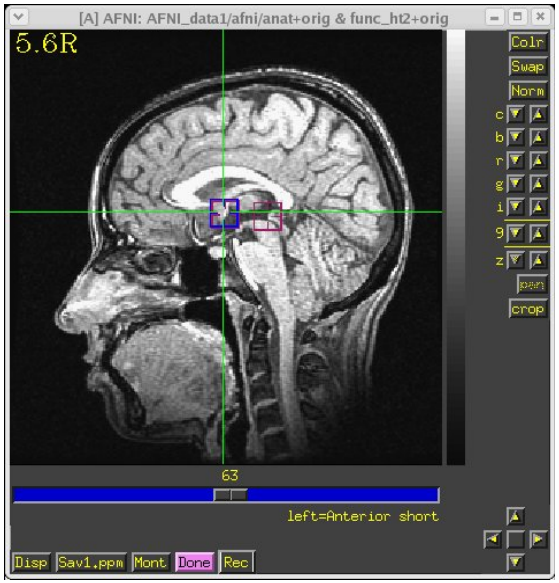
- Second goal is to mark inferior edge of **PC**
  - This is harder, since PC doesn't show up well at 1 mm resolution
  - Fortunately, PC is always at the top of the cerebral aqueduct, which does show up well (at least, if CSF is properly suppressed by the MRI pulse sequence)



  - Therefore, if you can't see the PC, find mid-sagittal location just at top of cerebral aqueduct and mark it as **[PC inferior edge]**
- Third goal is to mark **two inter-hemispheric points** (above corpus callosum)
  - The two points must be at least 2 cm apart
  - The two planes AC-PC-#1 and AC-PC-#2 must be no more than **2º** apart

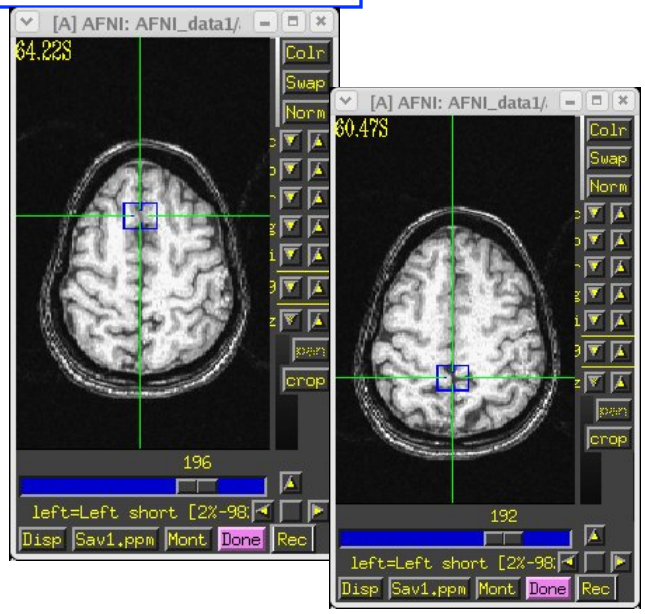
- AC-PC Markers Cheat Sheet
  - ✧ The AC-PC markers may take some time for the novice to master, so in the interest of time, we provide you with a little guide or “cheat sheet” to help you place markers on this example volume:

	i	j	k	to:
AC Superior Edge:	126	107	63	
AC Posterior Margin: 127	108	63		
PC Inferior Edge:	152	109	63	
1st Mid-Sagittal Point:	110	59	60	
2nd Mid-Sagittal Point:	172	63	60	

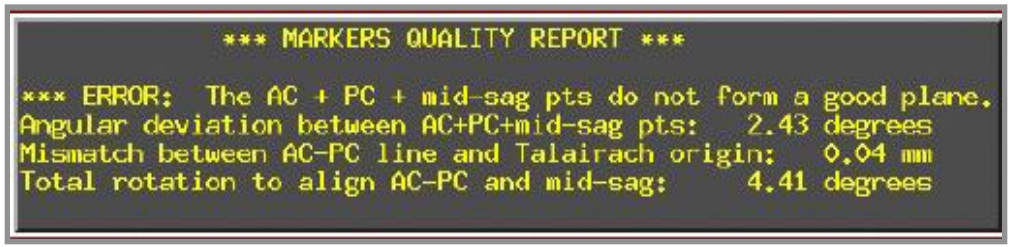


AC-PC markers

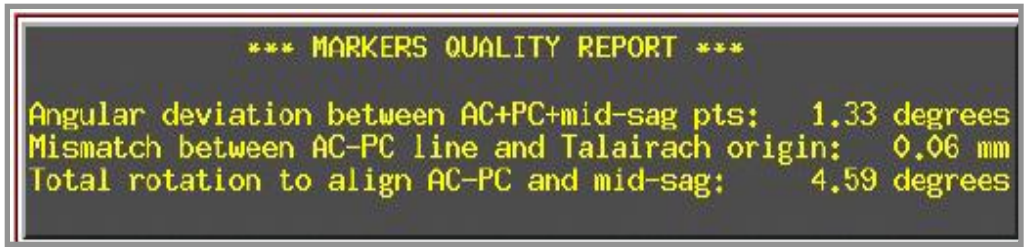
mid-sagittal markers



- Once all 5 markers have been set, the **[Quality?]** Button is ready
  - You can't **[Transform Data]** until **[Quality?]** Check is passed
  - In this case, quality check makes sure two planes from AC-PC line to mid-sagittal points are within 2°
    - Sample below shows a 2.43° deviation between planes ⇒ ERROR message indicates we must move one of the points a little



    - Sample below shows a deviation between planes at less than 2°. Quality check is passed



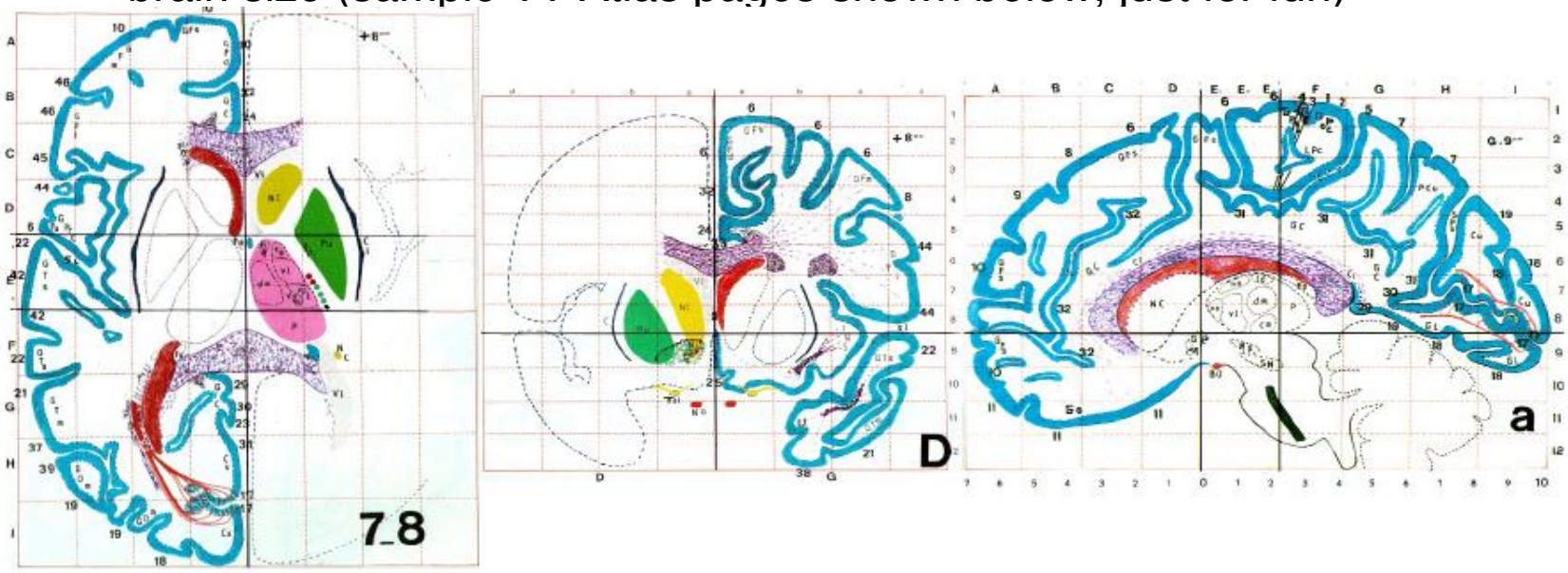
      - We can now save the marker locations into the dataset header

- Notes on positioning AC/PC markers:
  - The structures dimensions are on the order of typical high-res images. Do not fret about a matter such as:
    - Q: Do I put the Sup. AC marker on the top voxel where I see still the the structure or on the one above it?
      - A: Either option is OK, just be consistent. The same goes for setting the bounding box around the brain discussed ahead. Remember, intra-subject anatomical variability is more than the 1 or 2 mm you are concerned about.
  - Typically, all three markers fall in the same mid-saggital plane
- Why, oh why, two mid-saggital points?
  - **[Quality?]** Contrary to our desires, no two hemispheres in their natural setting can be perfectly separated by a mid-saggital plane. When you select a mid-saggital point, you are defining a plane (with AC/PC points) that forms an acceptable separation between left and right sides of the brain.
  - To get a better approximation of the mid-saggital plane, AFNI insists on another mid-saggital point and uses the average of the two planes. It also insists that these two planes are not off from one another by more than 2^o
- I am Quality! How do I escape the tyranny of the **[Quality?]** check?
  - If you know what you're doing and want to elide the tests:
    - Set AFNI_MARKERS_NOQUAL environment variable to YES
    - This is a times needed when you are applying the transform to brains of children or monkeys which differ markedly in size from mature human brains.

- When **[Transform Data]** is available, pressing it will close the **[Define Markers]** panel, write marker locations into the dataset header, and create the `+acpc` datasets that follow from this one
  - The **[AC-PC Aligned]** coordinate system is now enabled in the main AFNI controller window
  - In the future, you could re-edit the markers, if desired, then re-transform the dataset (but you wouldn't make a mistake, would you?)
  - If you don't want to save edited markers to the dataset header, you must quit AFNI without pressing **[Transform Data]** or **[Define Markers]**

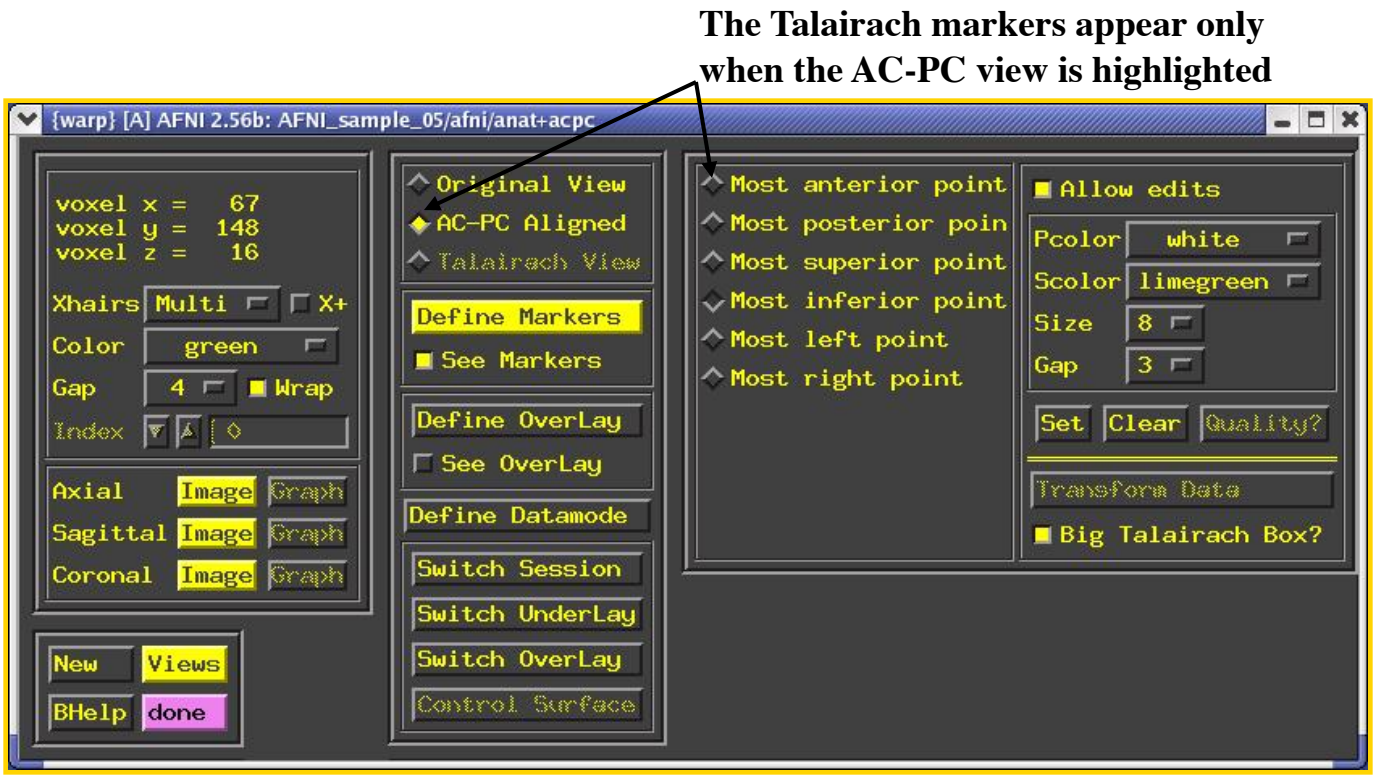
- **`ls`** ⇒ The newly created ac-pc dataset, **`anat+acpc.HEAD`**, is located in our `demo_tlrc/` directory
- At this point, only the header file exists, which can be viewed when selecting the **[AC-PC Aligned]** button
  - more on how to create the accompanying **`.BRIK`** file later…

- **Scaling to Talairach-Tournoux (+tlrc) coordinates:**
  - We now stretch/shrink the brain to fit the Talairach-Tournoux Atlas brain size (sample TT Atlas pages shown below, just for fun)



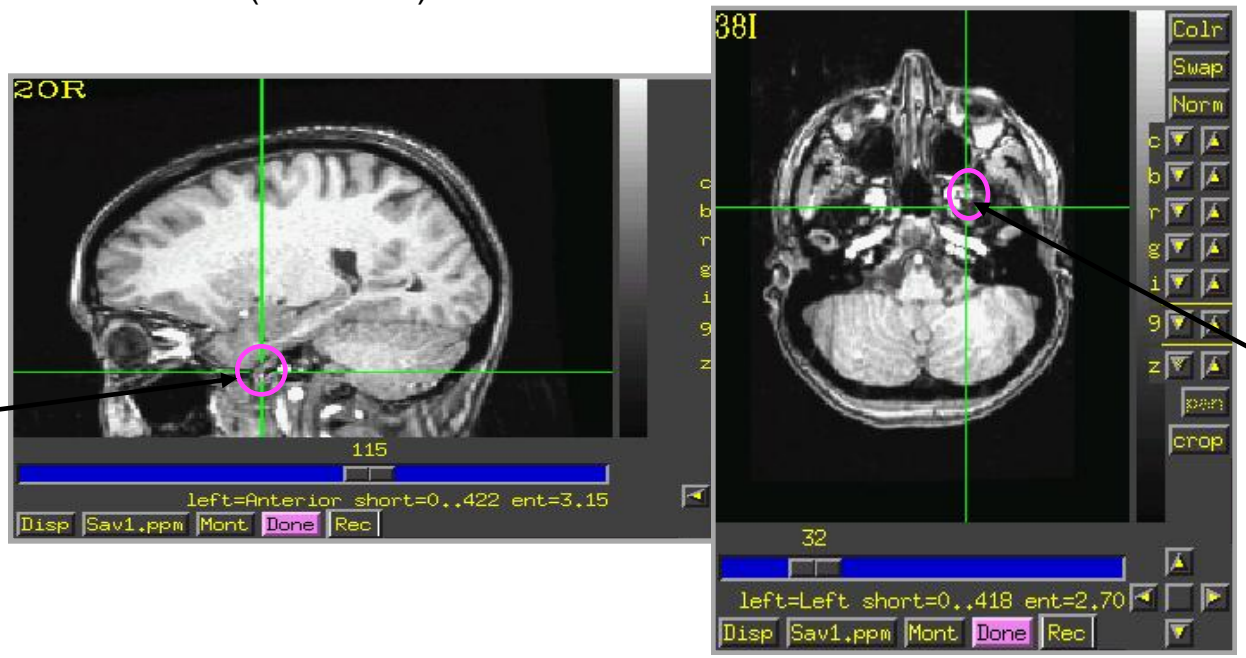
Most anterior to AC	70 mm		
AC to PC	23 mm		
PC to most posterior	79 mm	Length of cerebrum	172 mm
Most inferior to AC	42 mm		
AC to most superior	74 mm	Height of cerebrum	116 mm
AC to left (or right)	68 mm	Width of cerebrum	136 mm

- **Class example - Selecting the Talairach-Tournoux markers:**
  - ✧ There are 12 sub-regions to be scaled (3 A-P x 2 I-S x 2 L-R)
  - ✧ To enable this, the transformed +acpc dataset gets its own set of markers
    - ➥ Click on the [AC-PC Aligned] button to view our volume in ac-pc coordinates
    - ➥ Select the [Define Markers] control panel
  - ✧ A new set of six Talairach markers will appear:



- Using the same methods as before (i.e., select marker toggle, move focus there, **[Set]**), you must mark these extreme points of the cerebrum
  - Using 2 or 3 image windows at a time is useful
  - Hardest marker to select is **[Most inferior point]** in the temporal lobe, since it is near other (non-brain) tissue:

Sagittal view: most inferior point



Axial view: most inferior point

  - Once all 6 are set, press **[Quality?]** to see if the distances are reasonable
    - Leave **[Big Talairach Box?]** Pressed **IN**
    - Is a legacy from earliest (1994-6) days of AFNI, when 3D box size of `+tlrc` datasets was 10 mm smaller in `I`-direction than the current default

- Once the quality check is passed, click on **[Transform Data]** to save the `+tlrc` header
- **`ls`** ⇒ The newly created `+tlrc` dataset, **`anat+tlrc.HEAD`**, is located in our `demo_tlrc/` directory
  - At this point, the following anatomical datasets should be found in our `demo_tlrc/` directory:

    `anat+orig.HEAD` **`anat+orig.BRIK`**

    `anat+acpc.HEAD`

    `anat+tlrc.HEAD`

  - In addition, the following functional dataset (which I -- the instructor -- created earlier) should be stored in the `demo_tlrc/` directory:

    `func_slim+orig.HEAD` **`func_slim+orig.BRIK`**

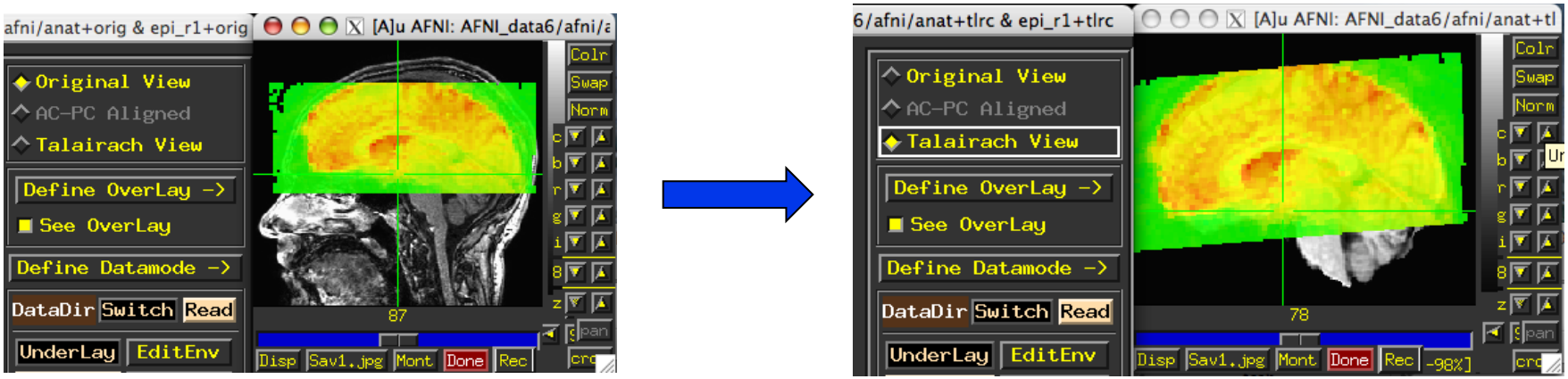
    - Note that this functional dataset is in the `+orig` format (not `+acpc` or `+tlrc`)

- **Automatic creation of "follower datasets":**
  - ✧ After the anatomical `+orig` dataset in a directory is resampled to `+acpc` and `+tlrc` coordinates, all the other datasets in that directory will *automatically* get transformed datasets as well
    - ➥ These datasets are created automatically inside the interactive AFNI program, and are not written (saved) to disk (i.e., only header info exists at this point)
    - ➥ How followers are created (arrows show geometrical relationships):

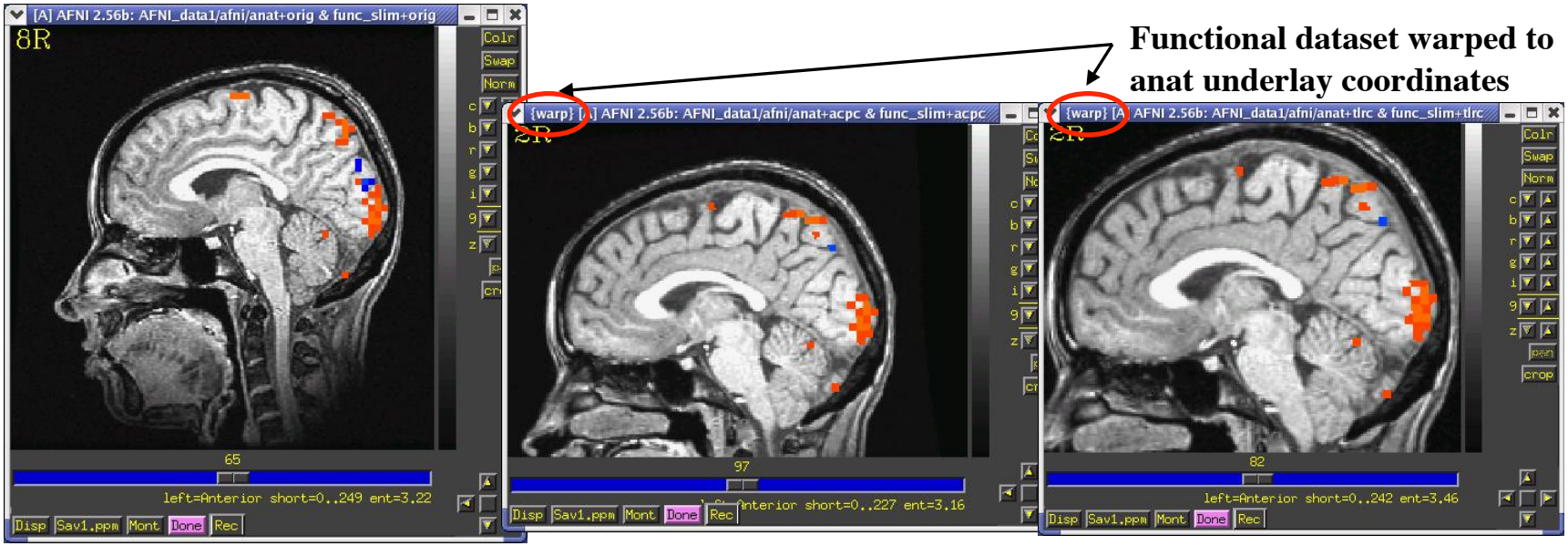
anat+orig → anat+acpc → anat+tlrc

↑ ↓ ↓

func+orig func+acpc func+tlrc

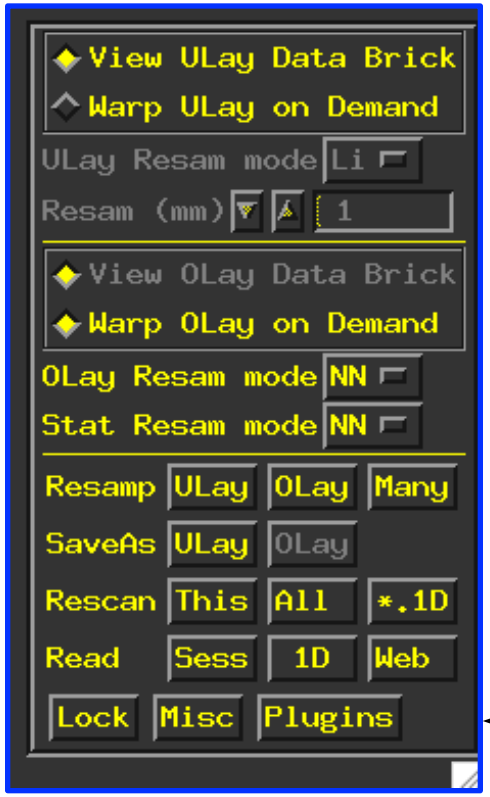


- How does AFNI actually create these follower datsets?
  - After **[Transform Data]** creates **anat+acpc**, other datasets in the same directory are scanned
    - AFNI defines the geometrical transformation ("warp") from **func_slim+orig** using the **to3d**-defined relationship between **func_slim+orig** and **anat+orig**, AND the markers-defined relationship between **anat+orig** and **anat+acpc**
      - A similar process applies for warping **func_slim+tlrc**
    - These warped functional datasets can be viewed in the AFNI interface:



- Next time you run AFNI, the followers will automatically be created internally again when the program starts

- "Warp on demand" viewing of datasets:
  - AFNI doesn't actually resample all follower datasets to a grid in the re-aligned and re-stretched coordinates
    - This could take quite a long time if there are a lot of big 3D+time datasets
  - Instead, the dataset slices are transformed (or warped) from `+orig` to `+acpc` or `+tlrc` for viewing as needed (on demand)
  - This can be controlled from the **[Define Datamode]** control panel:



If possible, lets you view slices direct from dataset .BRIK

If possible, transforms slices from 'parent' directory

Interpolation mode used when transforming datasets

Grid spacing to interpolate with

Similar for functional datasets

Write transformed datasets to disk

Re-read: datasets from current session, all session, or 1D files

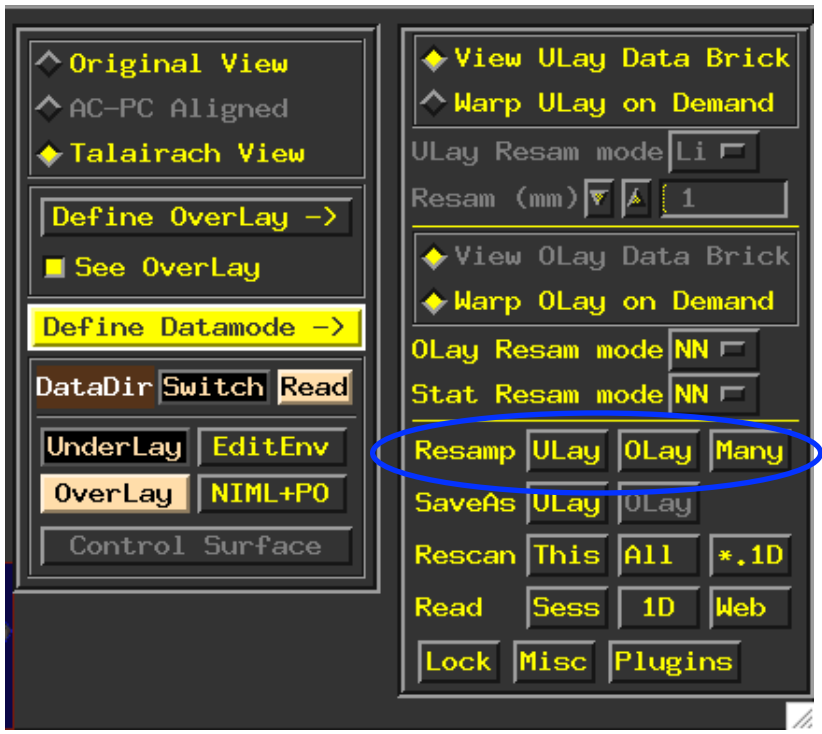
Read new: session directory, 1D file, dataset from Web address

Menus that had to go somewhere

AFNI titlebar shows warp on demand: `{warp}[A]AFNI2.56b:AFNI_sample_05/afni/anat+tlrc`

Creating follower data

- **Writing "follower datasets" to disk:**
  - Recall that when we created **anat+acpc** and **anat+tlrc** datasets by pressing **[Transform Data]**, only **.HEAD** files were written to disk for them
  - In addition, our follower datasets **func_slim+acpc** and **func_slim+tlrc** are *not* stored in our `demo_tlrc/` directory. Currently, they can only be viewed in the AFNI graphical interface
  - Questions to ask:
    1. How do we write our anat **.BRIK** files to disk?
    2. How do we write our warped follower datasets to disk?
  - To write a dataset to disk (whether it be an anat `.BRIK` file or a follower dataset), use one of the **[Define Datamode]** ⇒ **Resamp** buttons:



ULay writes current underlay dataset to disk

OLay writes current overlay dataset to disk

Many writes multiple datasets in a directory to disk

- **Class example - Writing anat (Underlay) datasets to disk:**
  - ✧ You can use **[Define Datamode]** ⇒ **Write** ⇒ **[ULay]** to write the current anatomical dataset `.BRIK` out at the current grid spacing (cubical voxels), using the current anatomical interpolation mode
  - ✧ After that, **[View ULay Data Brick]** will become available
    - ➥ **ls** ⇒ to view newly created `.BRIK` files in the **demo_tlrc/** directory:

      anat+acpc.HEAD anat+acpc.BRIK

      anat+tlrc.HEAD anat+tlrc.BRIK

- **Class example - Writing func (Overlay) datasets to disk:**
  - ✧ You can use **[Define Datamode]** ⇒ **Write** ⇒ **[OLay]** to write the current functional dataset `.HEAD` and `BRIK` files into our `demo_tlrc/` directory
  - ✧ After that, **[View OLay Data Brick]** will become available
    - ➥ **ls** ⇒ to view newly resampled func files in our `demo_tlrc/` directory:

      func_slim+acpc.HEAD func_slim+acpc.BRIK

      func_slim+tlrc.HEAD func_slim+tlrc.BRIK

- Command line program **adwarp** can also be used to write out .BRIK files for transformed datasets:

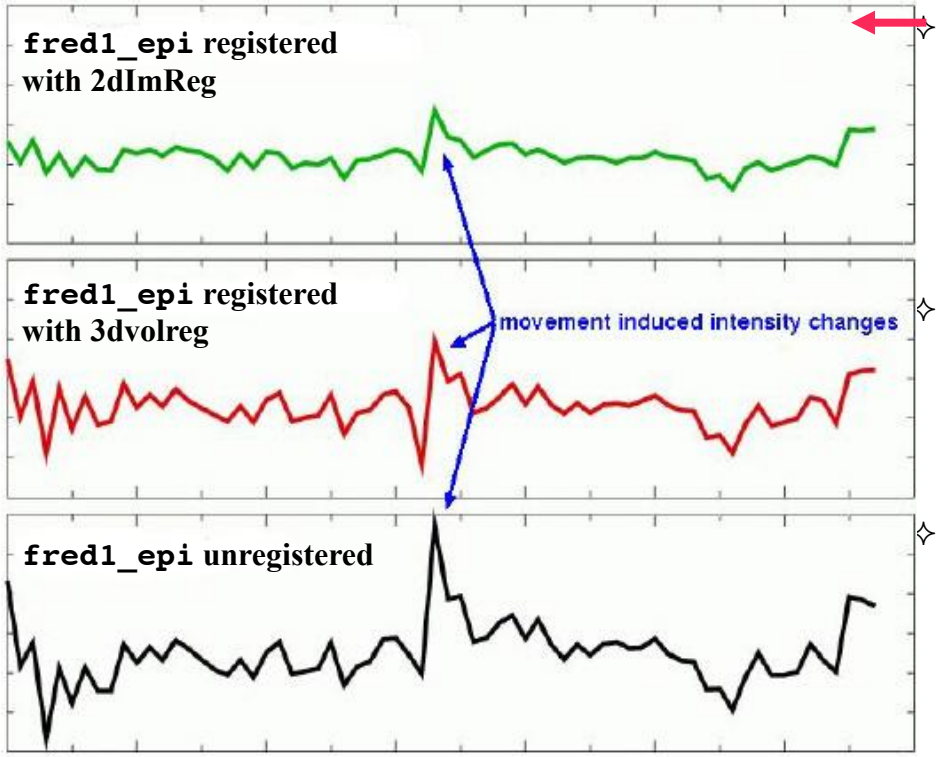
```
adwarp -apar anat+tlrc  -dpar func+orig
```

  - The result will be: **func+tlrc.HEAD** and **func+tlrc.BRIK**

- Why bother saving transformed datasets to disk anyway?
  - Datasets without .BRIK files are of limited use:
    - You can't display 2D slice images from such a dataset
    - You can't use such datasets to graph time series, do volume rendering, compute statistics, run any command line analysis program, run any plugin…
      - If you plan on doing any of the above to a dataset, it's best to have both a .HEAD and .BRIK files for that dataset

- Examination of time series **fred2_epi+orig** and **fred2_epi_vr_+orig** shows that head movement up and down happened within about 1 TR interval
  - Assumption of rigid motion of 3D volumes is not good for this case
  - Can do 2D slice-wise registration with command

```
2dImReg -input fred2_epi+orig      \
 -basefile fred1_epi+orig          \
 -base 4 -prefix fred2_epi_2Dreg
```



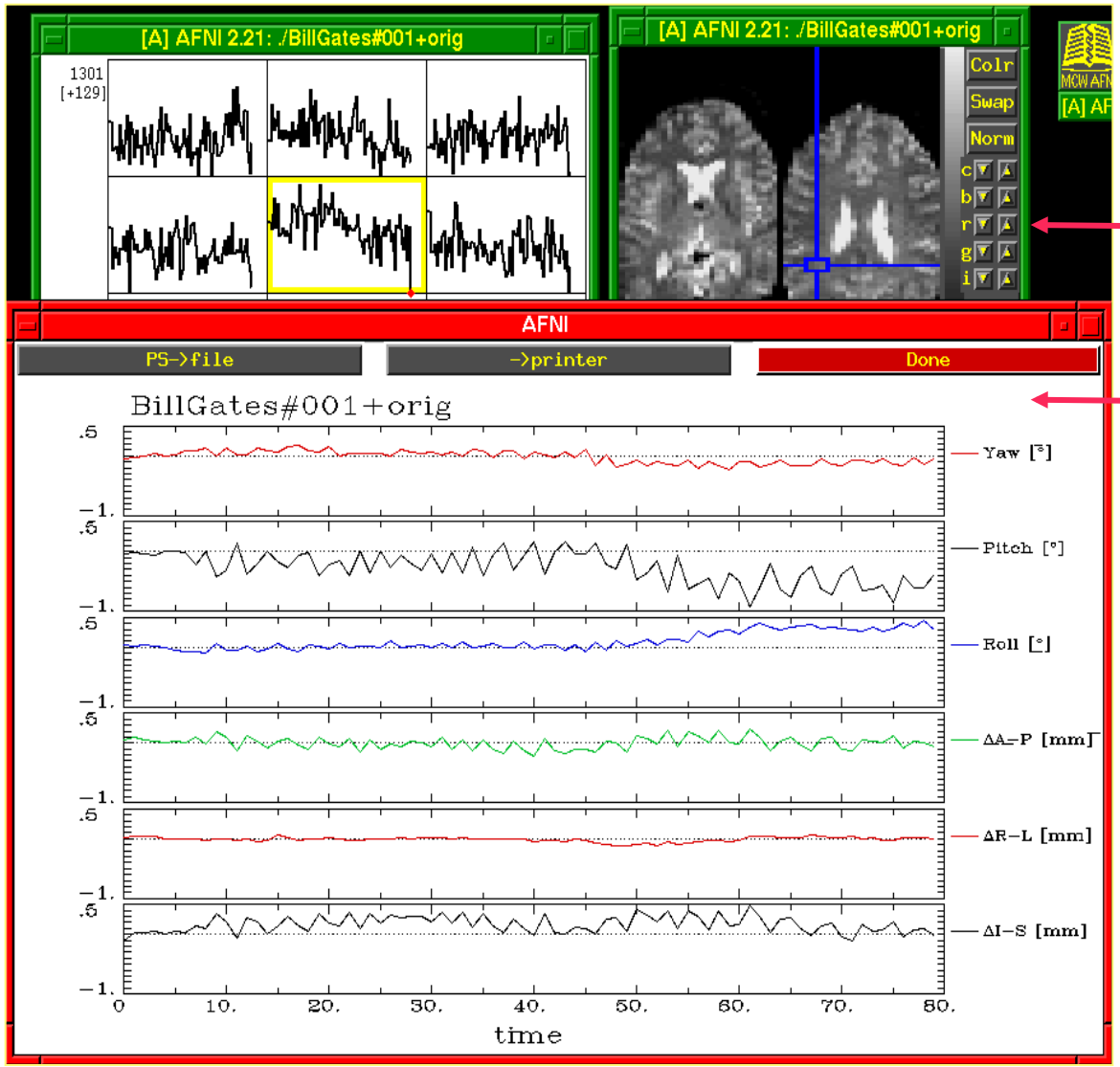
- Graphs of a single voxel time series near the edge of the brain:
  - Top = slice-wise alignment
  - Middle = volume-wise adjustment
  - Bottom = no alignment
- For **this** example, **2dImReg** appears to produce better results. This is because most of the motion is 'head nodding' and the acquisition is sagittal
- You should also use AFNI to scroll through the images (using the **Index** control) during the period of pronounced movement
  - Helps see if registration fixed problems

Real-Time 3D Image Registration

- The image alignment method using in **3dvolreg** is also built into the AFNI real-time image acquisition plugin
  - ✧ Invoke by command **afni -rt**
  - ✧ Then use **Define Datamode → Plugins → RT Options** to control the operation of real-time (RT) image acquisition
- Images (2D or 3D arrays of numbers) can be sent into AFNI through a TCP/IP socket
  - ✧ See the program **rtfeedme.c** for sample of how to connect to AFNI and send the data
    - ➥ Also see file **README.realtime** for lots of details
  - ✧ 2D images will be assembled into 3D volumes = AFNI sub-bricks
- Real-time plugin can also do 3D registration when each 3D volume is finished, and graph the movement parameters in real-time
  - ✧ Useful for seeing if the subject in the scanner is moving his head too much
    - ➥ If you see too much movement, telling the subject will usually help

- Realtime motion correction can easily be setup if DICOM images are made available on disk as the scanner is running.
- The script demo.realtime present in the AFNI_data1/EPI_manyruns directory demonstrates the usage:

```
#!/bin/tcsh

# demo real-time data acquisition and motion detection with afni

# use environment variables in lieu of running the RT Options plugin
setenv AFNI_REALTIME_Registration       3D:_realtime
setenv AFNI_REALTIME_Graph              Realtime

if ( ! -d afni ) mkdir afni
cd afni

afni -rt &

sleep 5

cd ..
echo ready to run Dimon
echo -n press enter to proceed...
set stuff = $<

Dimon -rt -use_imon -start_dir 001 -pause 200
```



- Screen capture from example of real-time image acquisition and registration
- Images and time series graphs can be viewed as data comes in
- Graphs of movement parameters