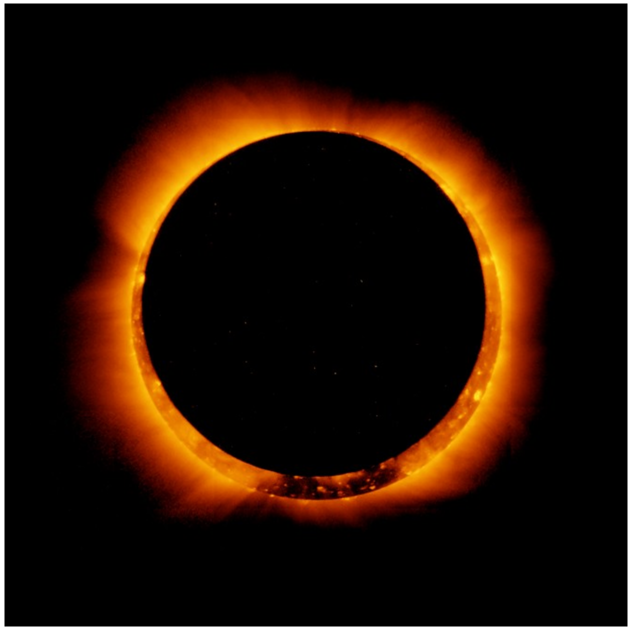
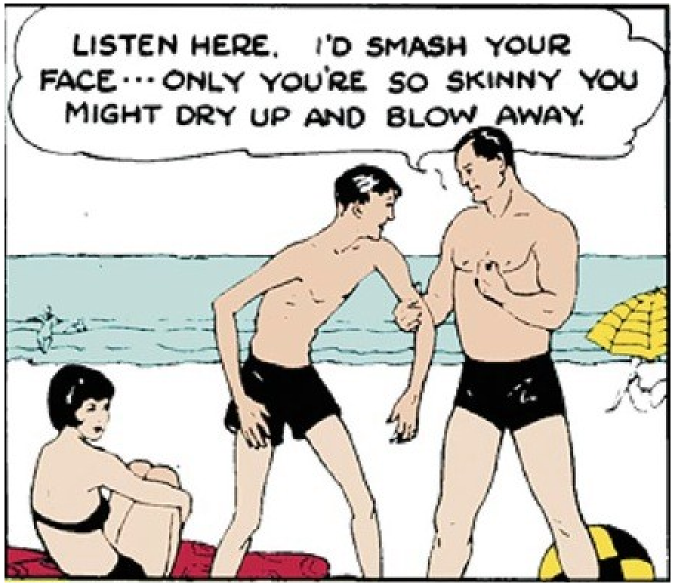
Alignment and Atlases - extra-special supplementary information



Atlas Conclusions and Questions

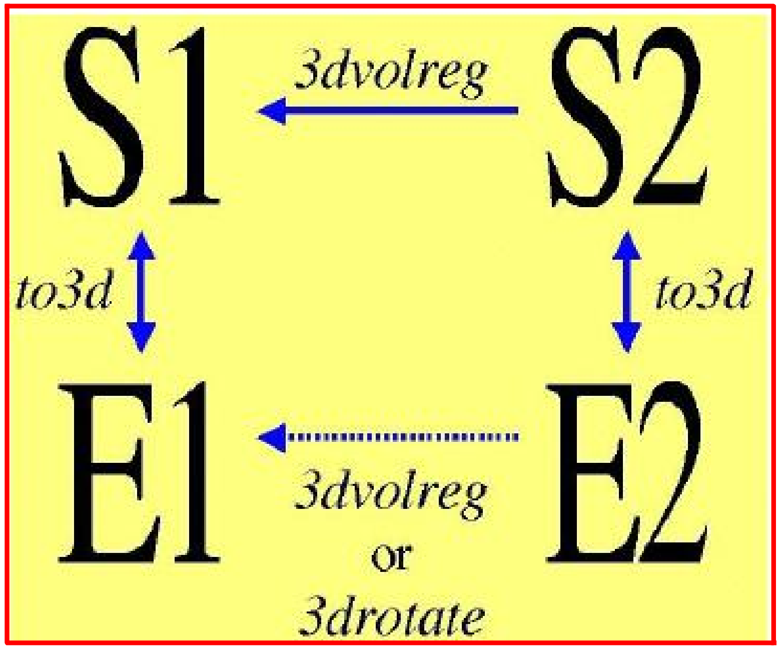

LISTEN HERE. I'D SMASH YOUR FACE··· ONLY YOU'RE SO SKINNY YOU MIGHT DRY UP AND BLOW AWAY.

Charles Atlas

Appendix A

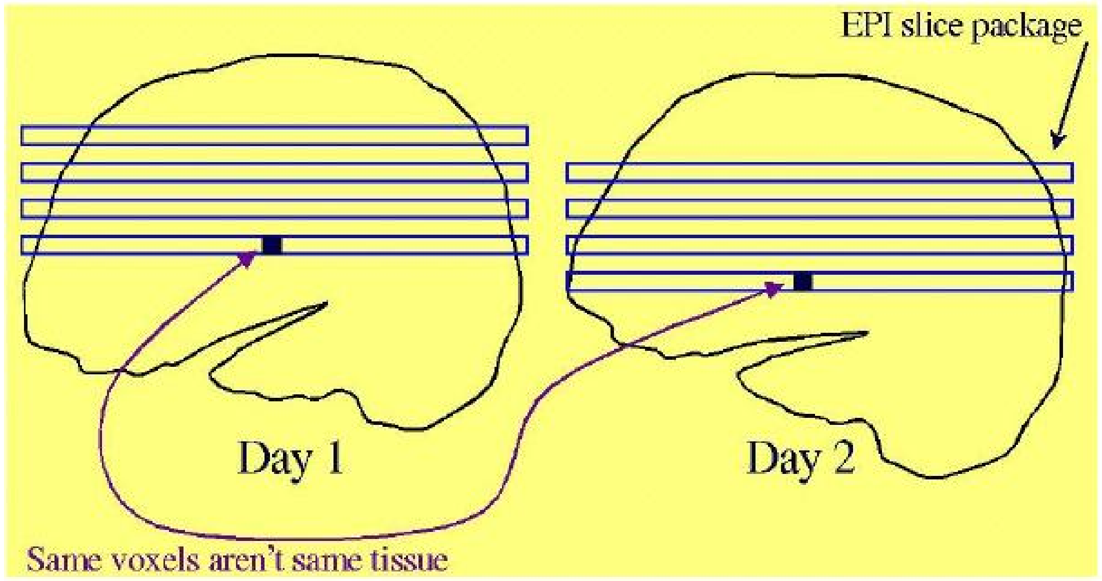
Inter-subject, inter-session registration

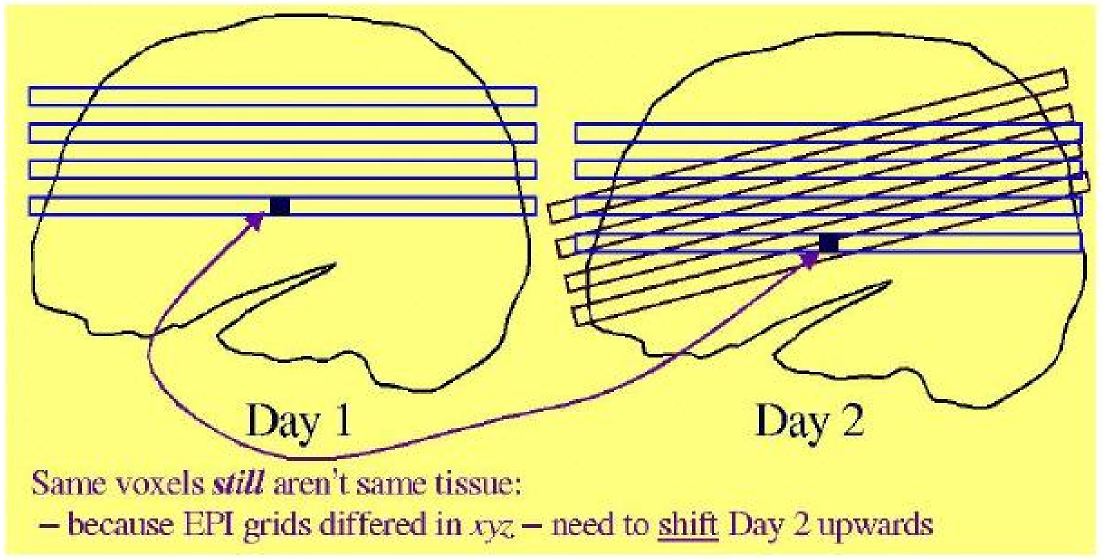
- Intra-subject, inter-session registration (for multi-day studies on same subject)
  - Longitudinal or learning studies; re-use of cortical surface models
  - Transformation between sessions is calculated by registering high-resolution anatomicals from each session
    - **`to3d`** defines defines relationship 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?)

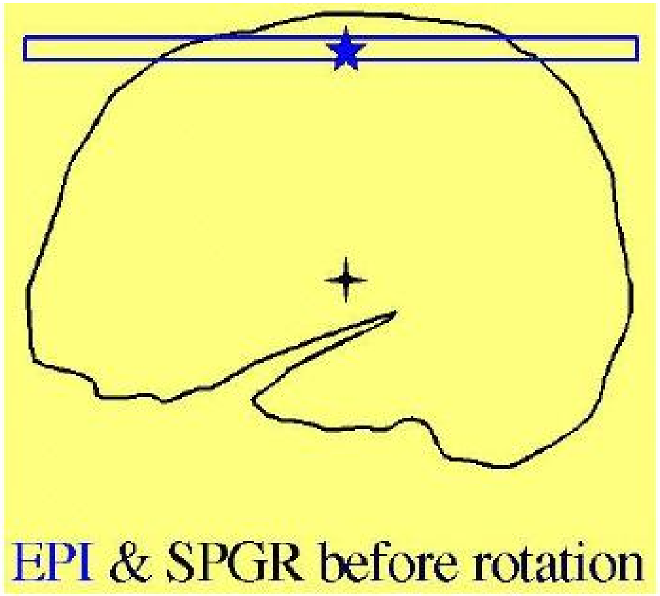
- Anatomical coverage differs



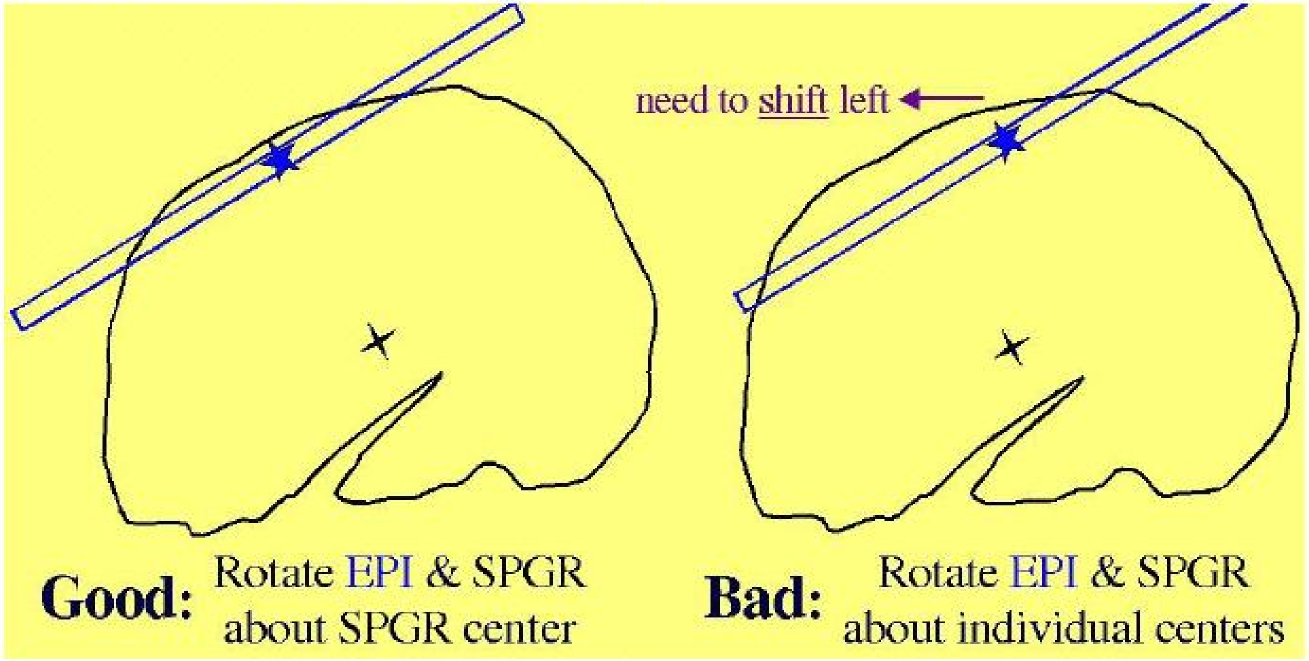

Day 1
Day 2
Same voxels *still* aren't same tissue:
– because EPI grids differed in *xyz* – need to shift Day 2 upwards

- ◇ At acquisition:
  Day 2 is rotated relative to Day 1

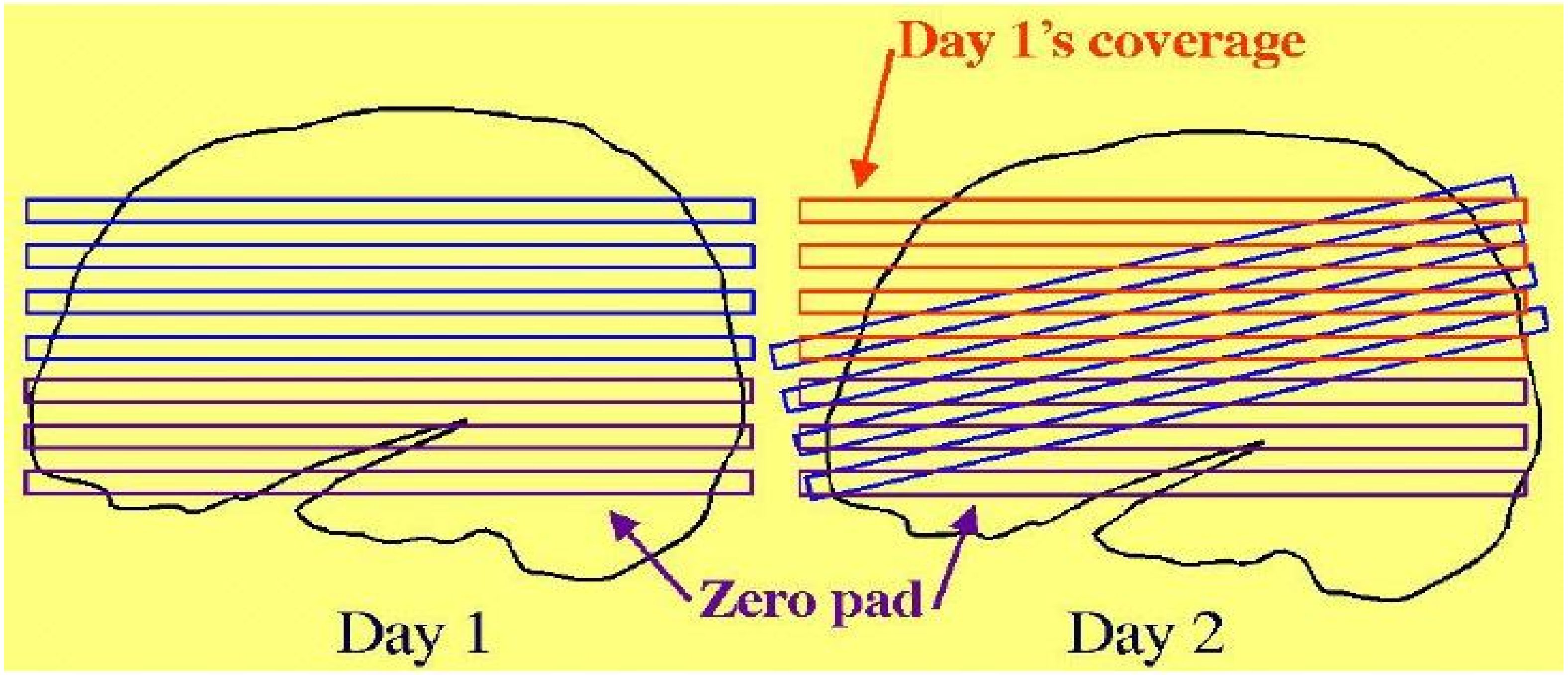
- ◇ After rotation to same orientation, then clipping to Day 2 xyz-grid



- Another problem: rotation occurs around center of individual datasets


need to shift left
Good: Rotate EPI & SPGR about SPGR center
Bad: Rotate EPI & SPGR about individual centers

- 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
```

  1. Register E1 datasets within the session:

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

  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`**

  2. **`-rotparent`** tells where the inter-session transformation comes from
  3. **`-gridparent`** defines the output grid location/size of new dataset
     2. 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

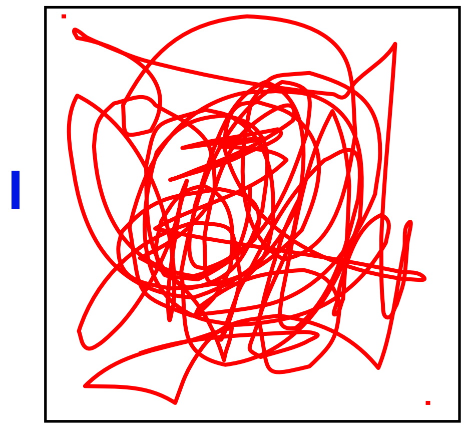
Algorithmic Features

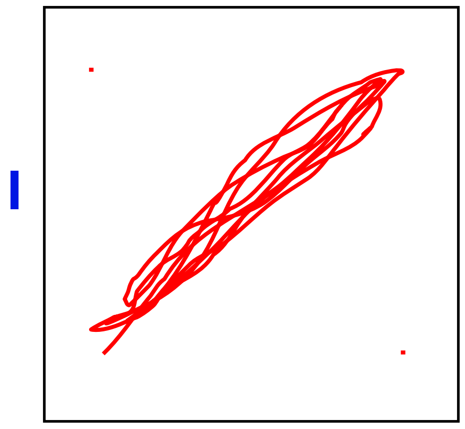
- Uses Powell's NEWUOA software for minimization of general cost function
- Lengthy search for initial transform parameters if two passes of registration are turned on [which is the default]
  - ◇ Random and grid search through hundreds of parameter sets for 15 good (low cost) parameter sets
  - ◇ Optimize 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

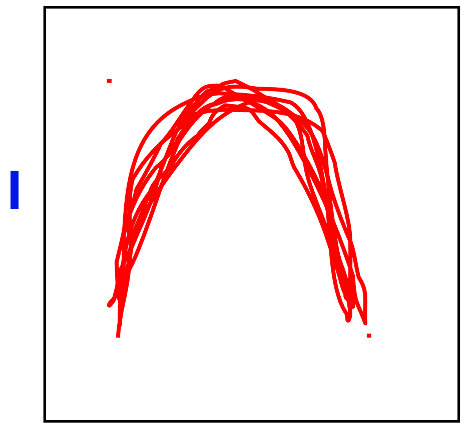
Algorithmic Features

- 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:





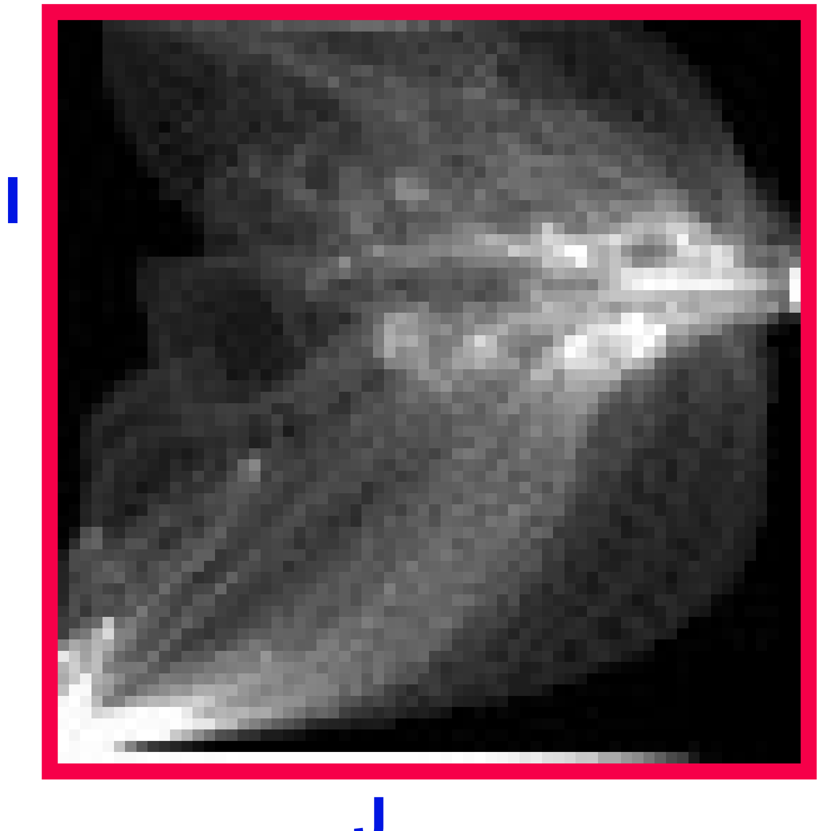


- J not useful in predicting I

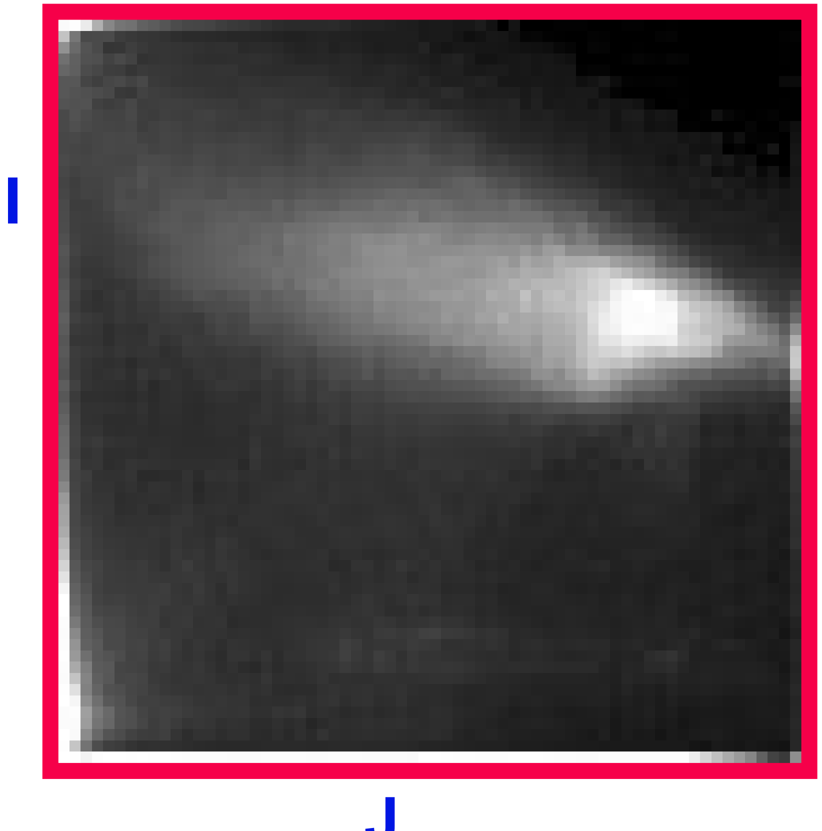
- I can be accurately predicted from J with a linear formula:
  `-leastsq` is OK

- 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

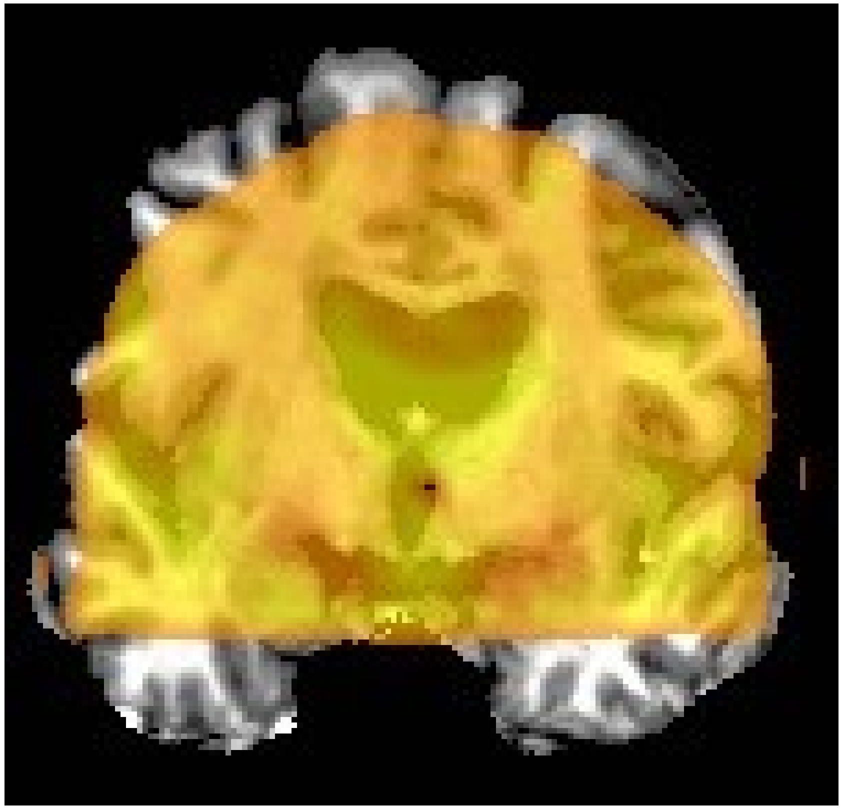


- Before alignment

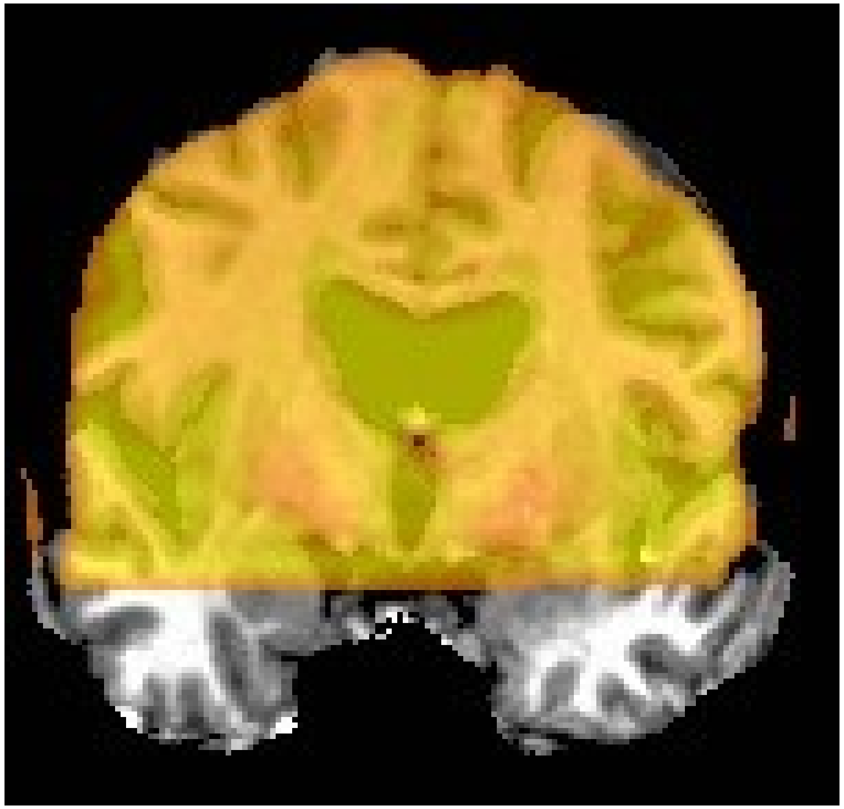


- After alignment (using **`-mutualinfo`**)

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



• Before alignment

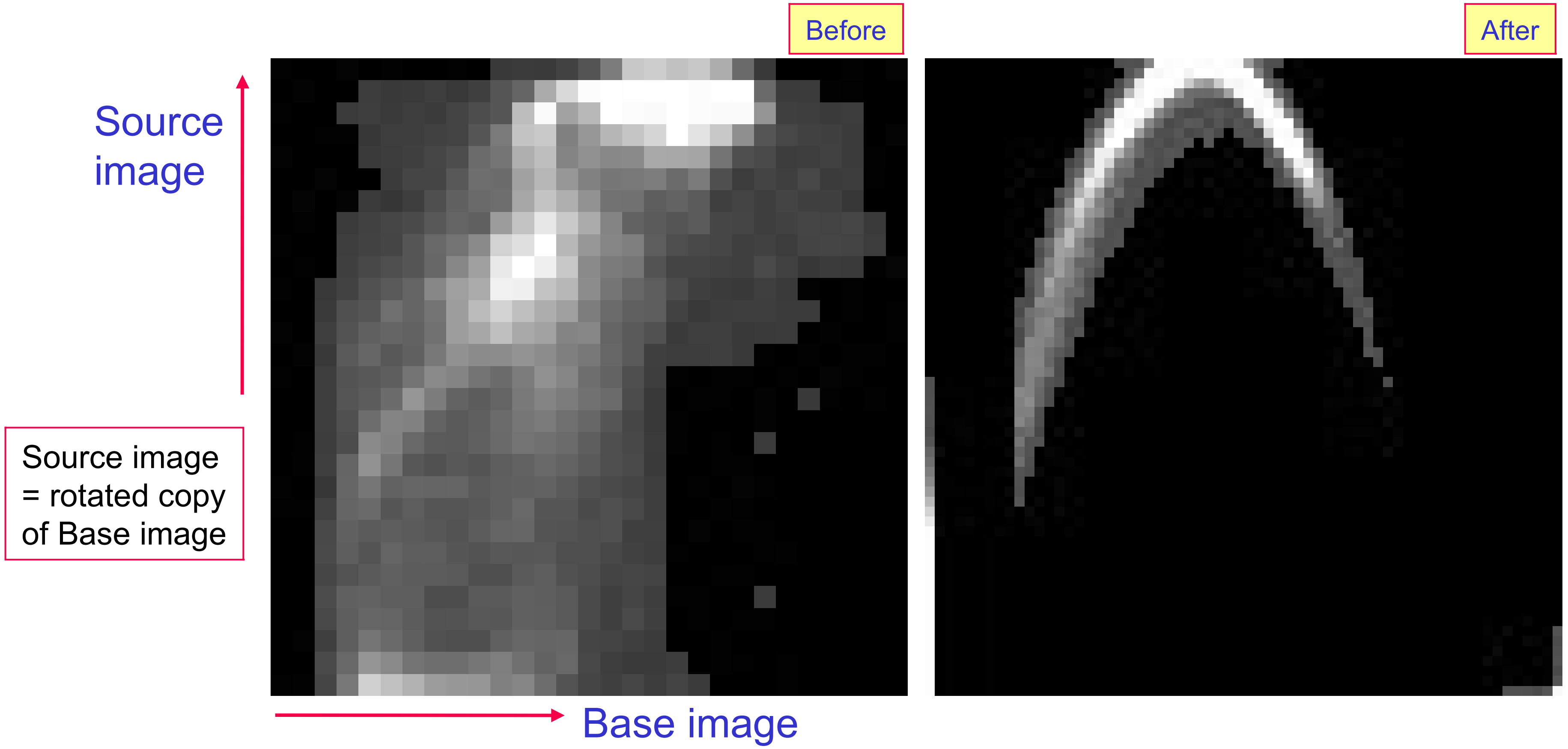


• After alignment (using **`-mutualinfo`**)

- 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

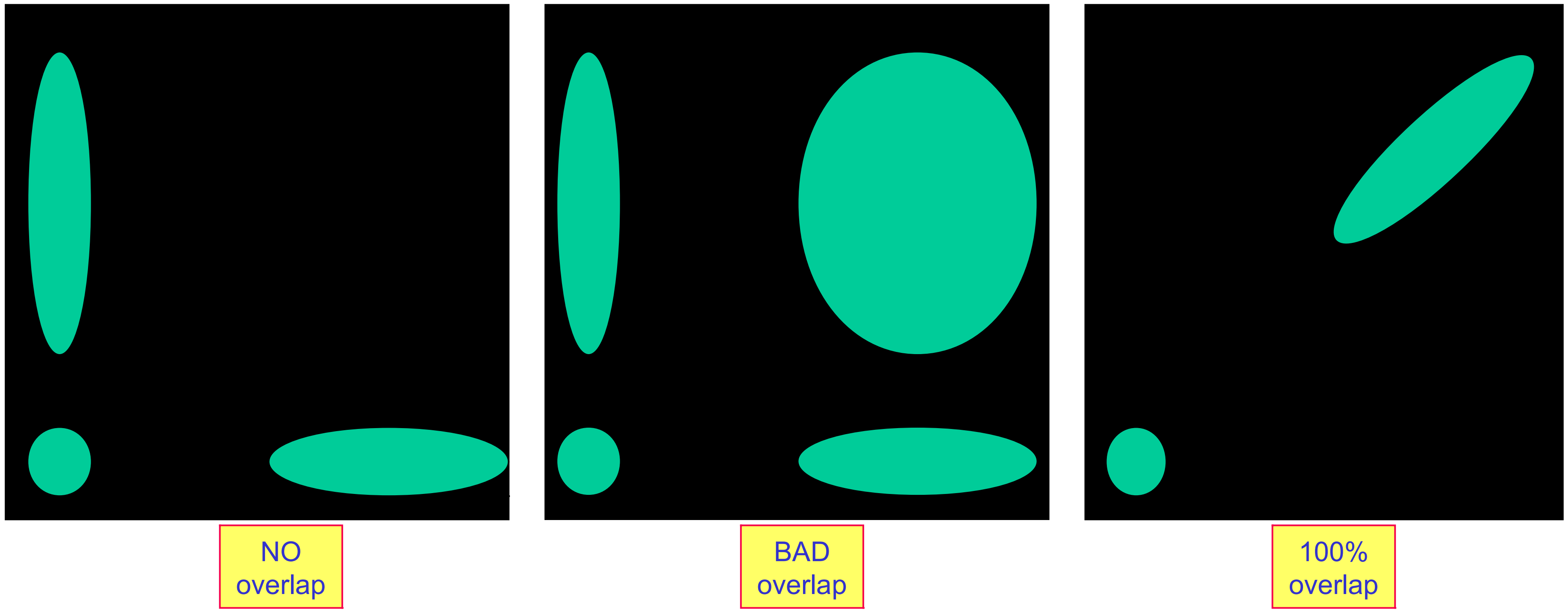
- Goal is to make 2D histogram become ‘simple’ in some sense, as a measurement of ‘predictability’ between $\mathbf{J}(\mathbf{x})$ and $\mathbf{I}(\mathbf{T}(\mathbf{x},\mathbf{w}))$
- Entropy H() of a histogram (finite number of bins):
  - $\{p_i\}$ = probabilities of index i occuring
  - $H(\{p_i\}) = -\Sigma_i \, p_i \log_2(p_i) > 0$
  - $H(\{p_i\})$ = 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
  - $H(\{r_{ij}\}) = -\sum_{ij} r_{ij} \log_2(r_{ij})$
  - 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`**

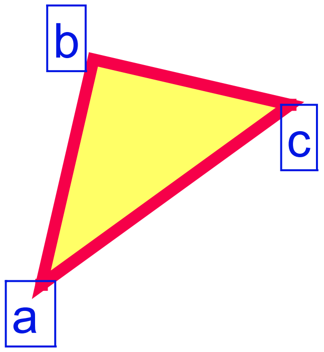
Normalized MI

- 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



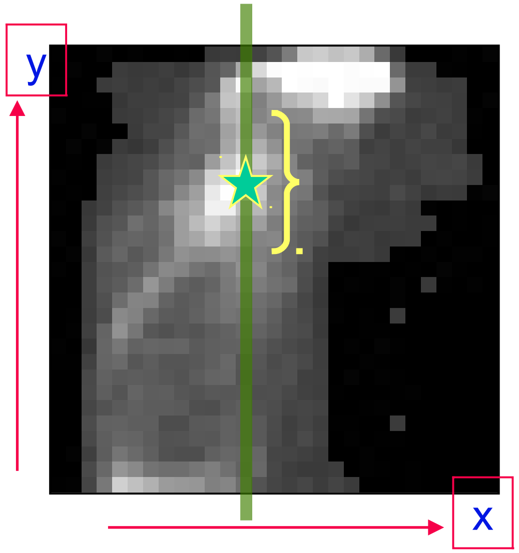
Hellinger Metric

- 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":
  - $\text{HM} = \Sigma_{ij} \left[\sqrt{r_{ij}} - \sqrt{(p_i \cdot q_j)} \right]^2$
  - `3dAllineate` minimizes –HM with `-cost hel`
  - This is the default cost function



Correlation Ratio

- Given 2 (non-independent) random variables x and y
  - ◇ Exp[y|x] is the expected value (mean) of y for a fixed value of x
    - ➥ Exp[a|b] ≡ Average value of ‘a’, given value of ‘b’
  - ◇ Var(y|x) is the variance of y when x is fixed = amount of uncertainty about value of y when we know x
    - ➥ v(x) ≡ Var(y|x) 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[y|x] is a good prediction of the value of y given the value of x
    - Does ***not*** say that Exp[x|y] is a good prediction of the x given y
  - CR(x,y) is a generalization of the Pearson correlation coefficient, which assumes that Exp[y|x] = $\alpha \cdot x+\beta$

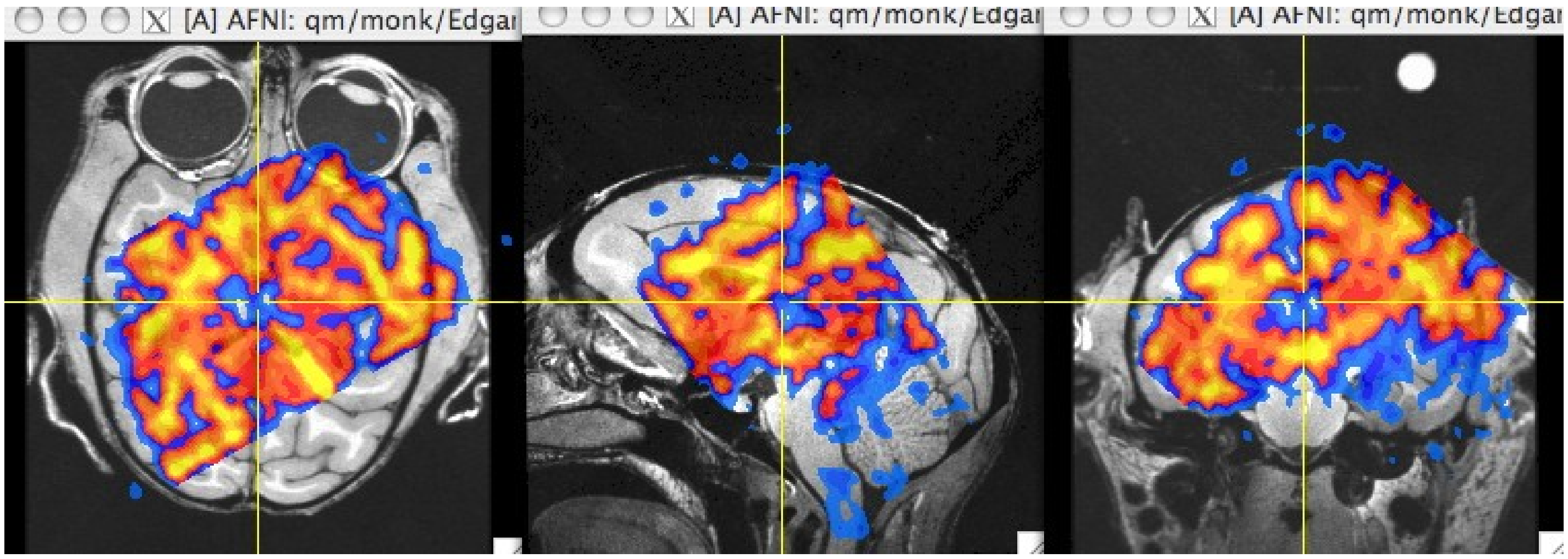
3dAllineate's Symmetrical CR

- First attempt to use CR in **3dAllineate** didn't give good results
- Note asymmetry: $CR(x,y) \neq CR(y,x)$
- **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:
  - $CRm(x,y) = 1 - [\, Exp(v(x)) \cdot Exp(v(y)) \,] \,/\, [\, Var(y) \cdot Var(x) \,]$

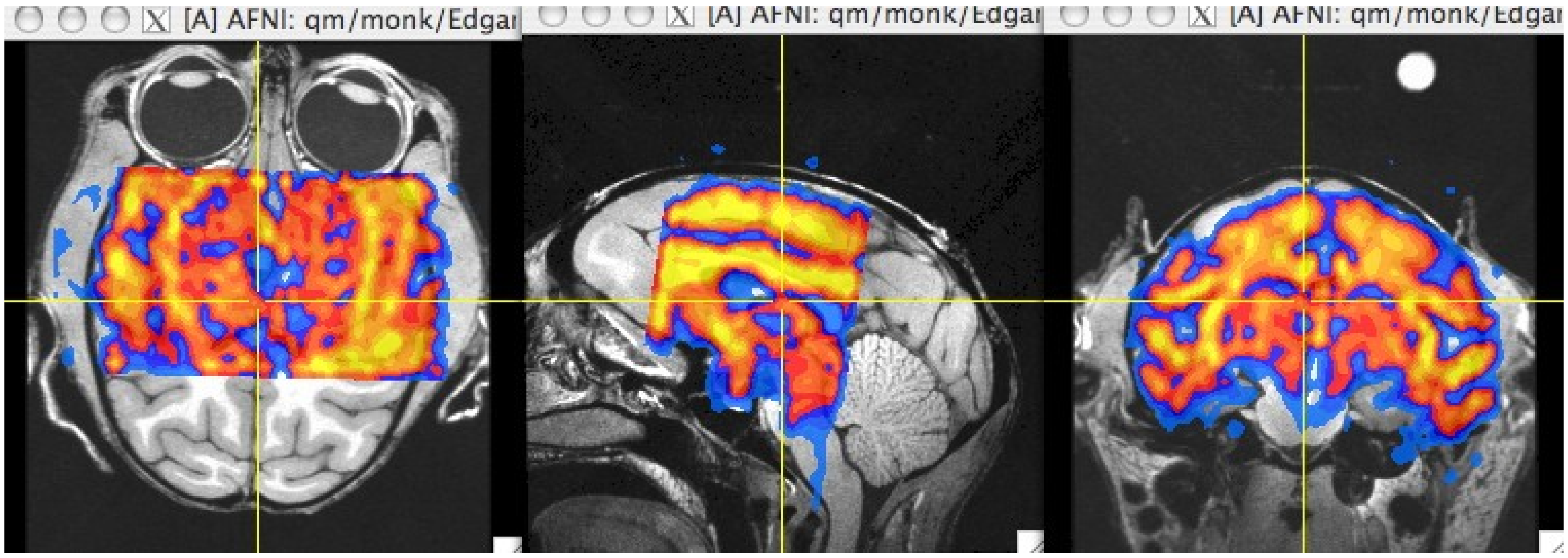
    $= CR(x,y) + CR(y,x) - CR(x,y) \cdot CR(y,x)$
  - $CRa(x,y) = 1 - {}^{1}\!/_{2}\, [\, Exp(v(x)) \,\square\, Var(y) \,] - {}^{1}\!/_{2}\, [\, Exp(v(y)) \,\square\, Var(x) \,]$

    $= [\, CR(x,y) + CR(y,x) \,] \,\square\, 2$
  - These work better than CR(**J**,**I**) in my test problems
- If Exp[y|x] can be used to predict y <u>and/or</u> 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

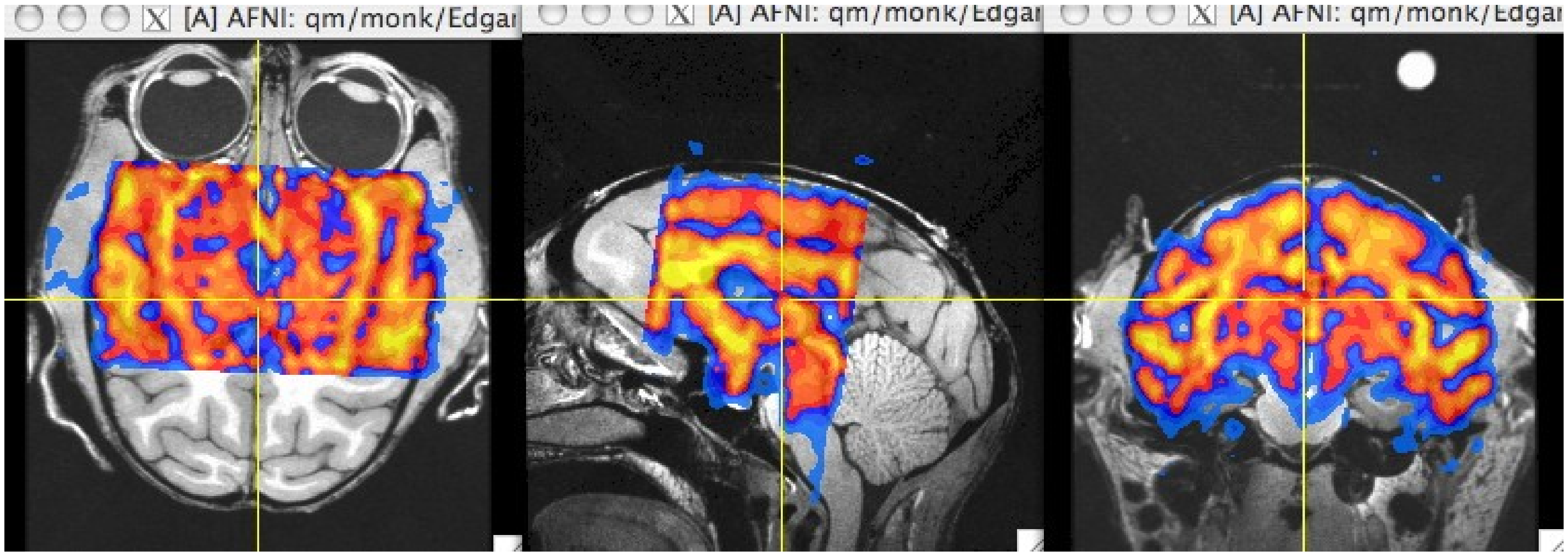


6 DOF
CRm

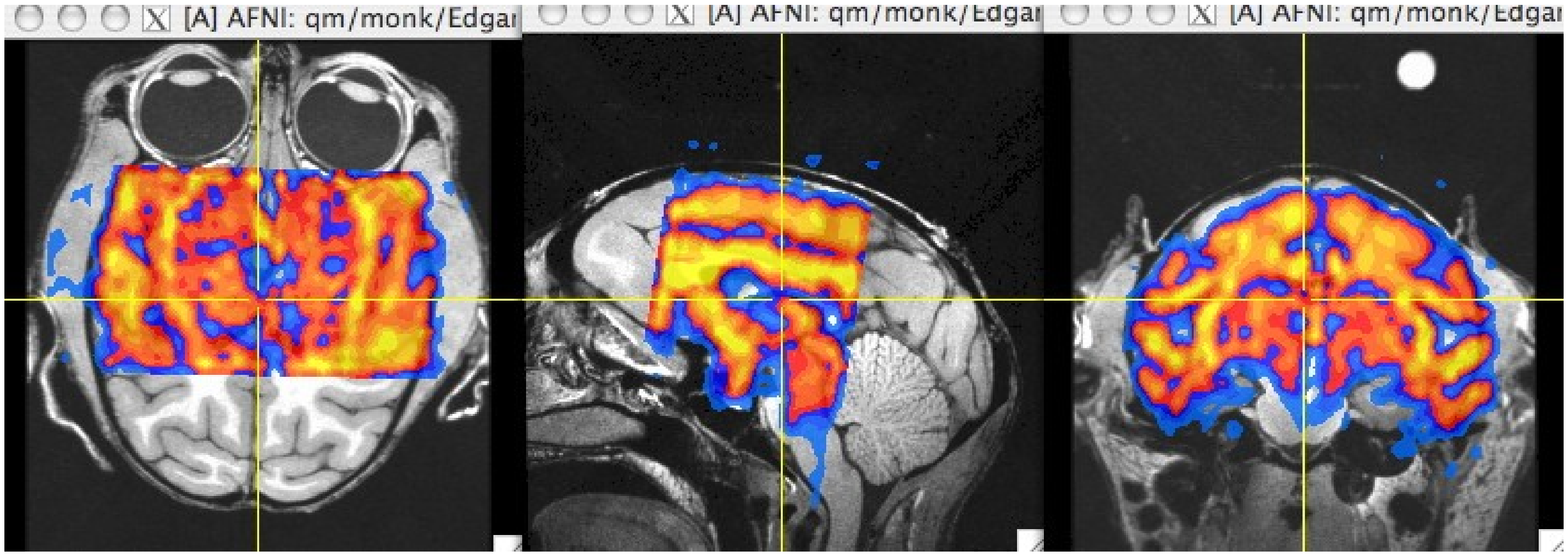


6 DOF
NMI

Test: Monkey EPI - Anat

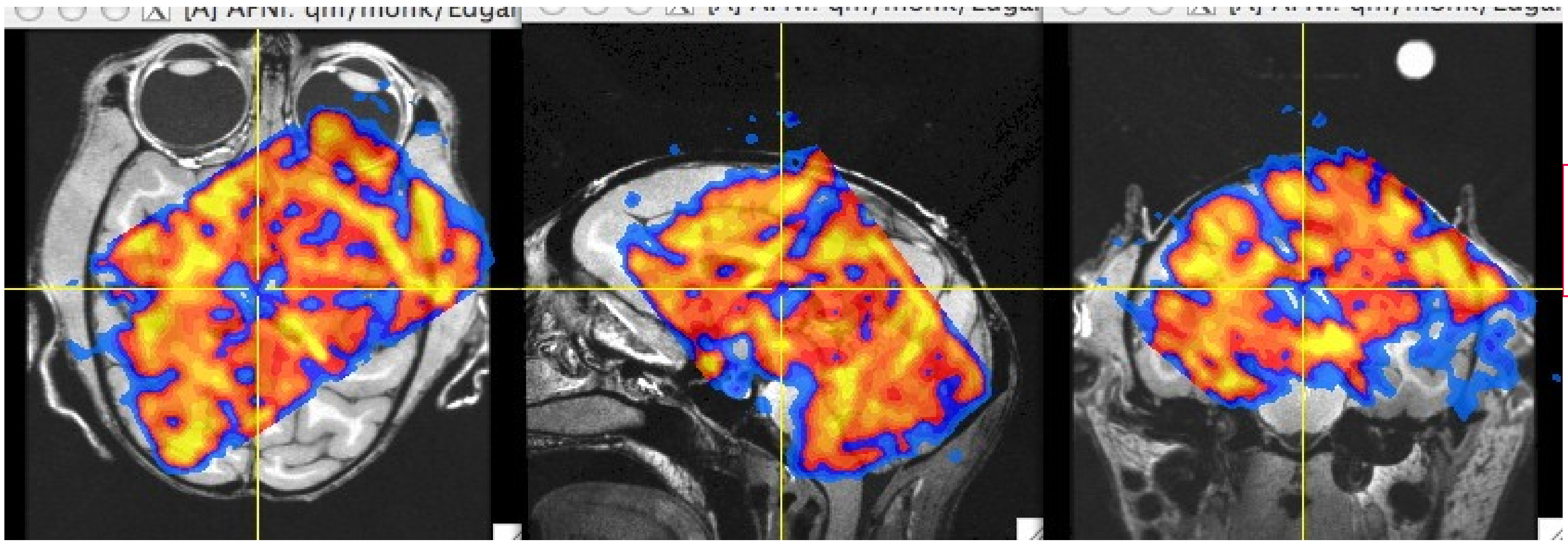


6 DOF
HEL

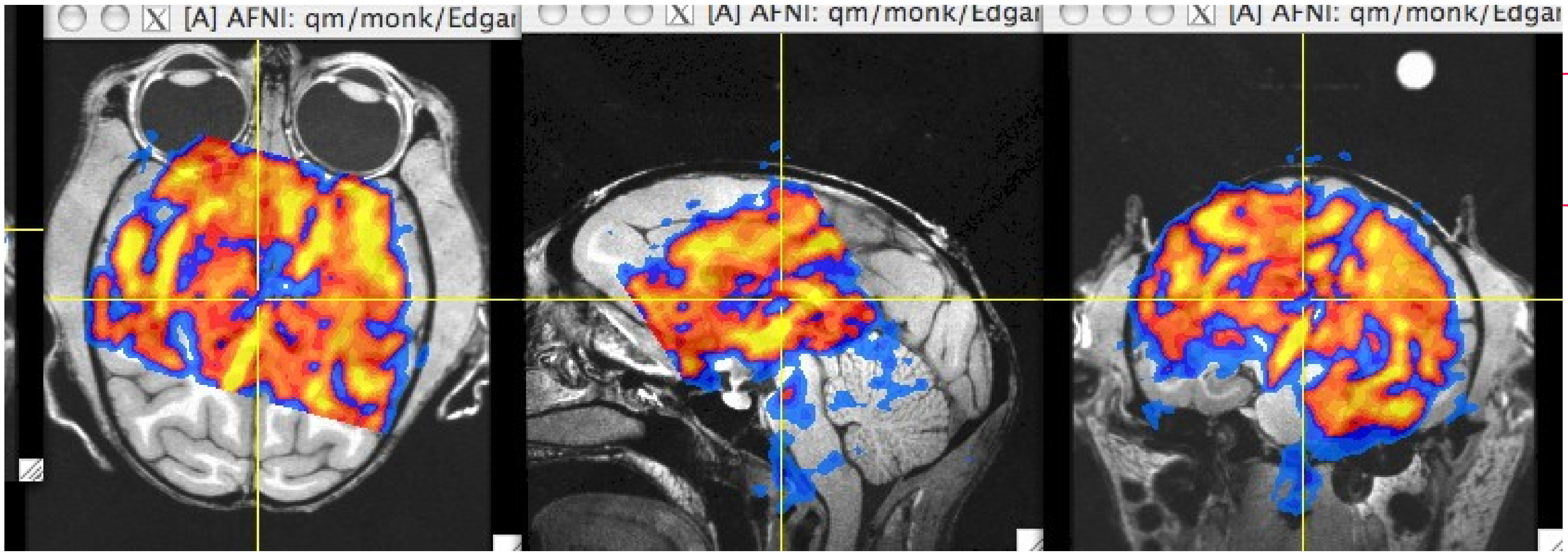


6 DOF
MI

Test: Monkey EPI - Anat

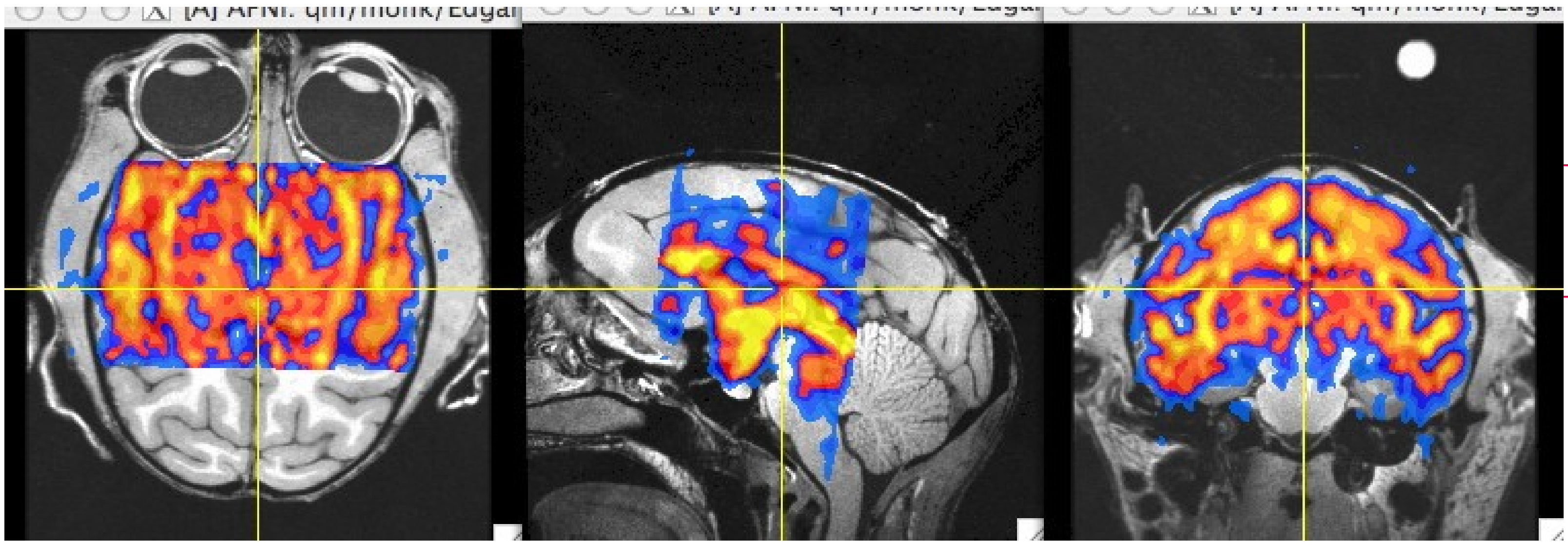


11 DOF
CRm

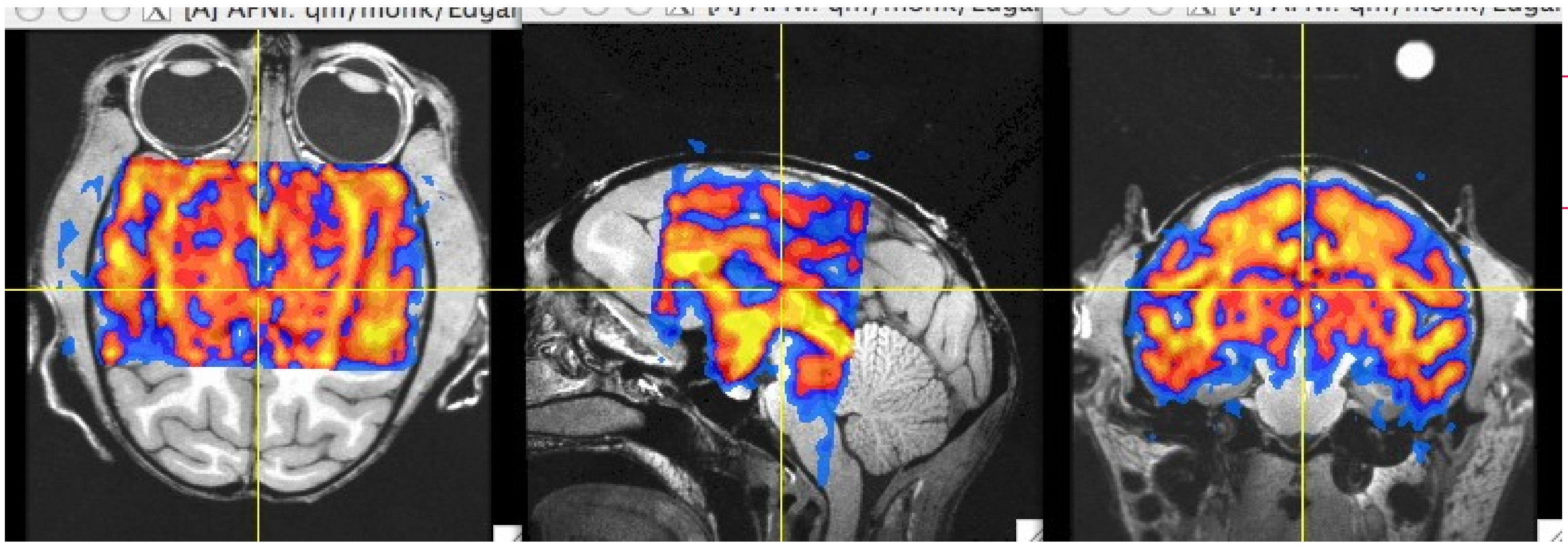


11 DOF
NMI

Test: Monkey EPI - Anat



11 DOF
HEL

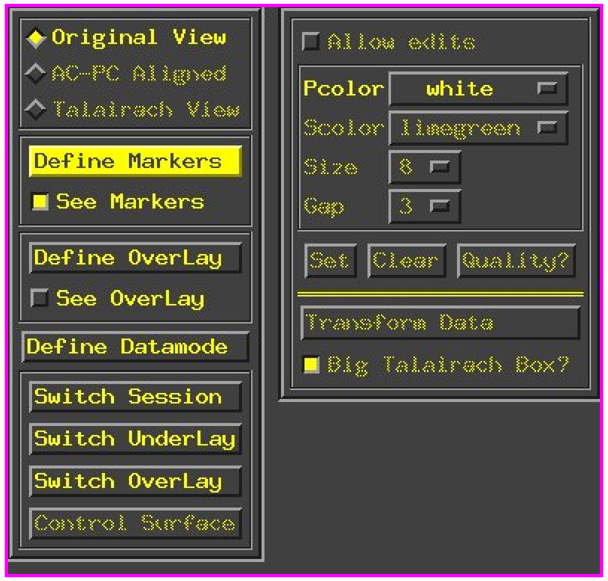


11 DOF
MI

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:

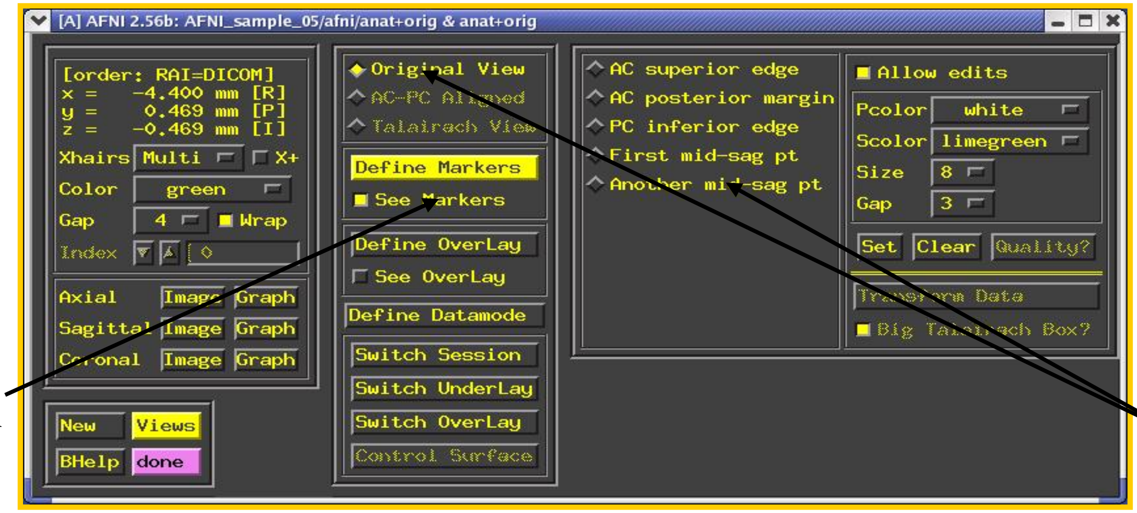


Gasp! Where did they go?

  - 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>
```

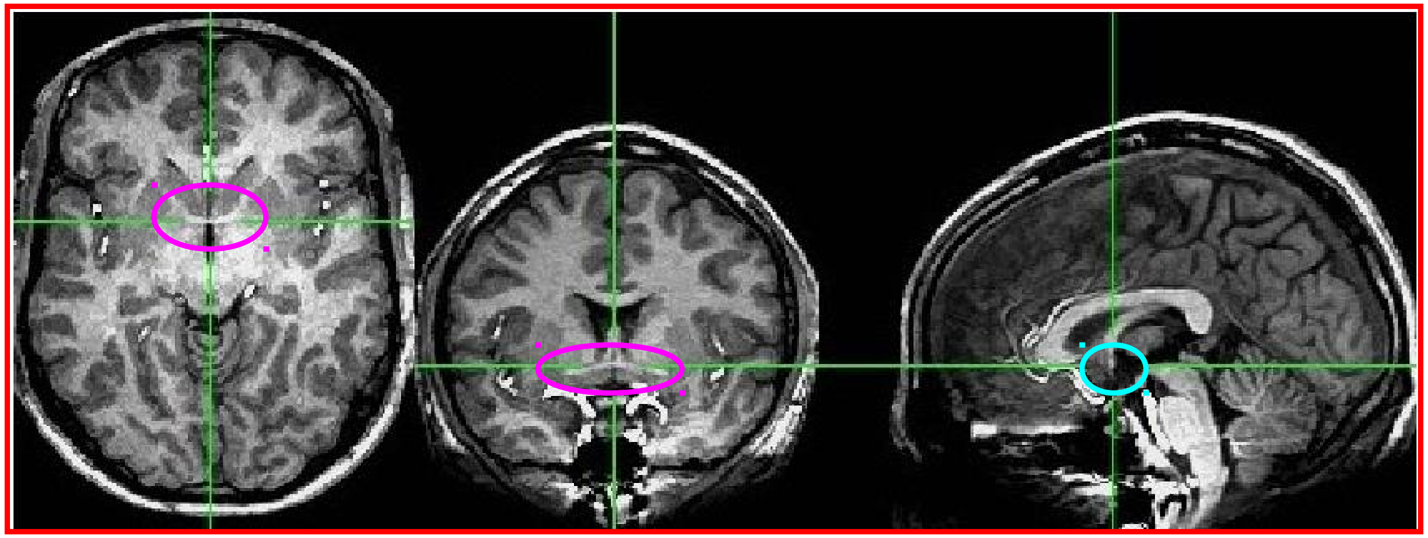
- **Class Example - Selecting the ac-pc markers:**
  - ◇ **`cd AFNI_data1/demo_tlrc`** ⇒ Descend into the `demo_tlrc/` subdirectory
  - ◇ **`afni &`** ⇒ This command launches the AFNI program
    - ➥ The "`&`" keeps the UNIX shell available in the background, so we can continue typing in commands as 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

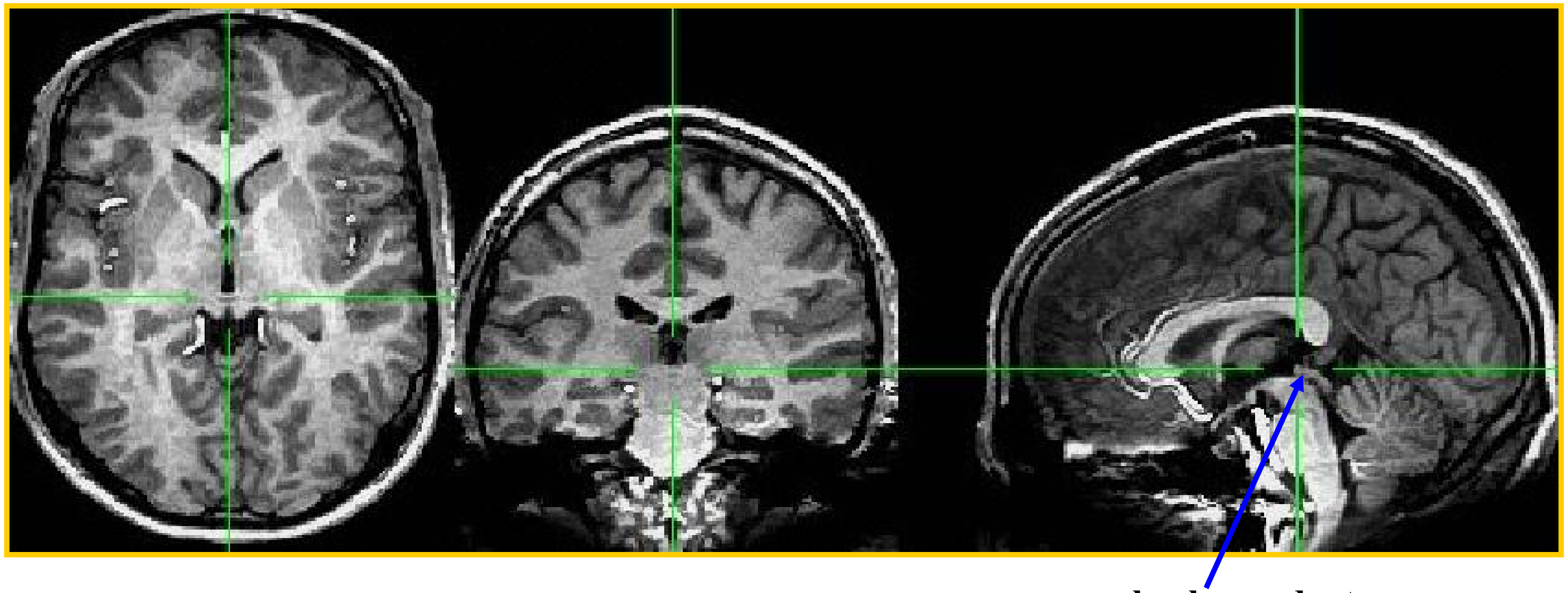
The AC-PC markers appear only when the orig view is highlighted

- ◇ 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



- <u>First goal is to mark top middle and rear middle of **AC**</u>
  - **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 [<u>AC superior edge</u>] 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 [<u>AC posterior margin</u>], then [Set]

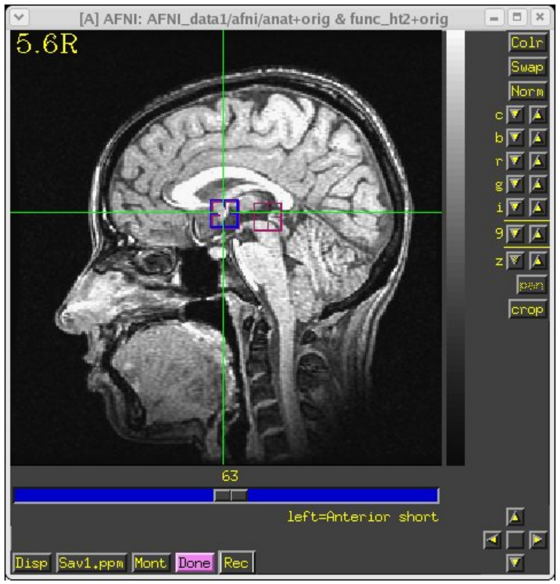
- <u>Second goal is to mark inferior edge of **PC**</u>
  - This is harder, since PC doesn’t show up well at 1 mm resolution
  - Fortunately, PC is always at the top of the cerebral aqueduct, which does show up well (at least, if CSF is properly suppressed by the MRI pulse sequence)



  - 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]`
- <u>Third goal is to mark **two inter-hemispheric points** (above corpus callosum)</u>
  - 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 **2°**

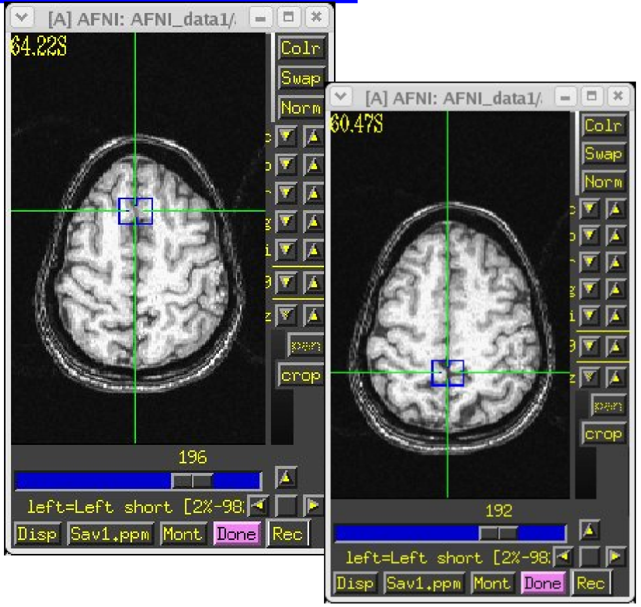
- AC-PC Markers Cheat Sheet
  - ◇ The AC-PC markers may take some time for the novice to master, so in the interest of time, we provide you with a little guide or "cheat sheet" to help you place markers on this example volume:

	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	

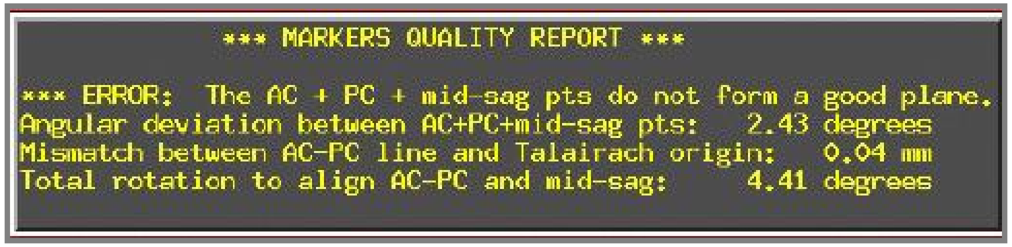


AC-PC markers

mid-sagittal markers

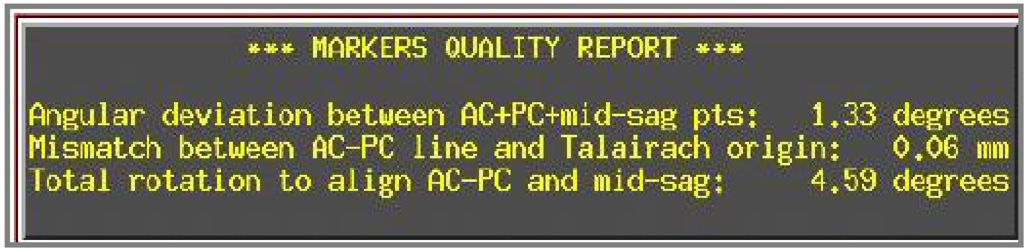


- Once all 5 markers have been set, the `[Quality?]` Button is ready
  - You can't `[Transform Data]` until `[Quality?]` Check is passed
  - In this case, quality check makes sure two planes from AC-PC line to mid-sagittal points are within 2^{o}
    - Sample below shows a 2.43^{o} deviation between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



```
                  *** MARKERS QUALITY REPORT ***

*** ERROR:  The AC + PC + mid-sag pts do not form a good plane.
Angular deviation between AC+PC+mid-sag pts:    2.43 degrees
Mismatch between AC-PC line and Talairach origin:    0.04 mm
Total rotation to align AC-PC and mid-sag:      4.41 degrees
```

    - Sample below shows a deviation between planes at less than 2^{o}. Quality check is passed



```
                  *** MARKERS QUALITY REPORT ***

Angular deviation between AC+PC+mid-sag pts:    1.33 degrees
Mismatch between AC-PC line and Talairach origin:    0.06 mm
Total rotation to align AC-PC and mid-sag:      4.59 degrees
```

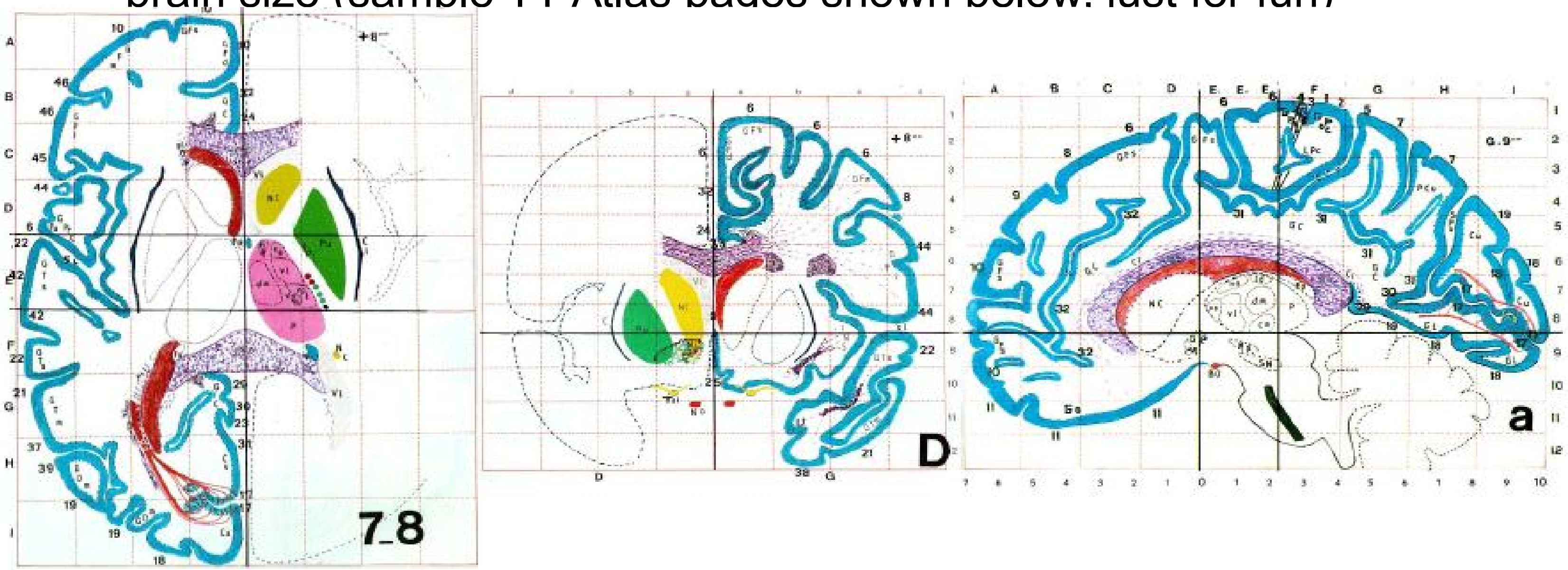
- We can now save the marker locations into the dataset header

- 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)

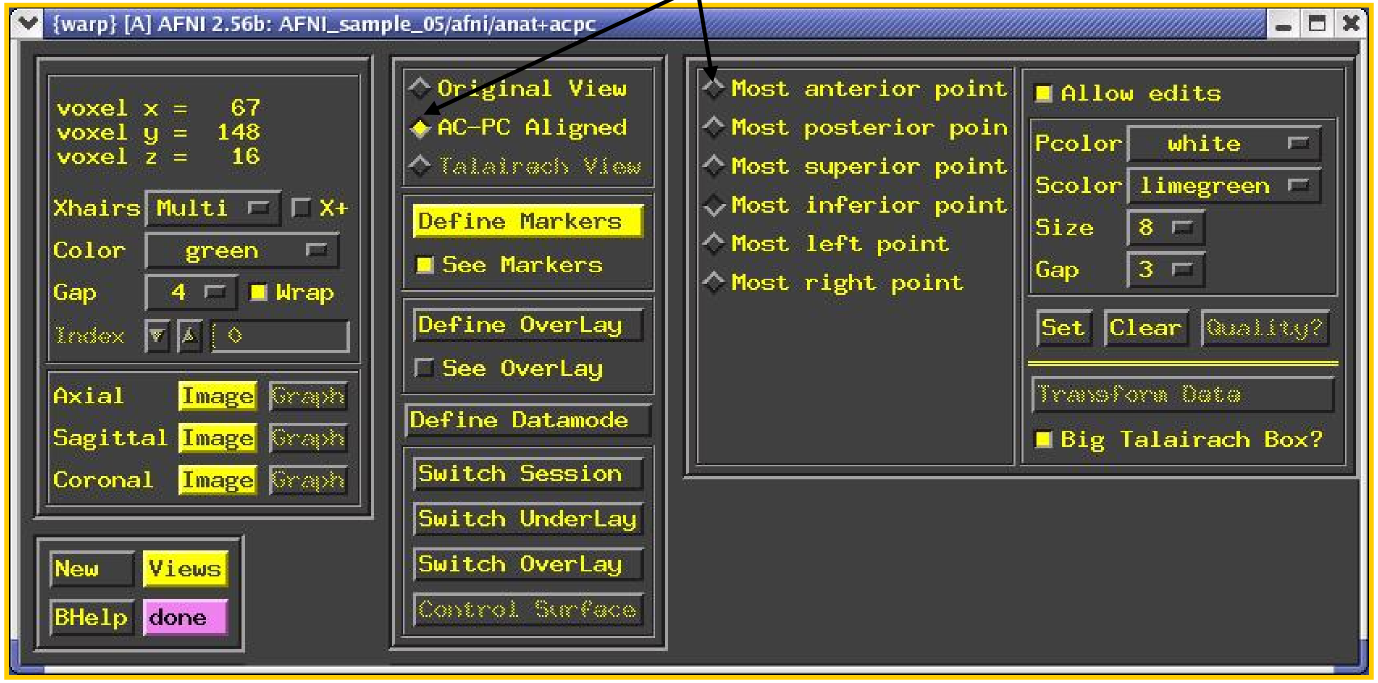


Most anterior to AC	70 mm
AC to PC	23 mm
PC to most posterior	79 mm
Most inferior to AC	42 mm
AC to most superior	74 mm
AC to left (or right)	68 mm

Length of cerebrum	172 mm
Height of cerebrum	116 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:

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

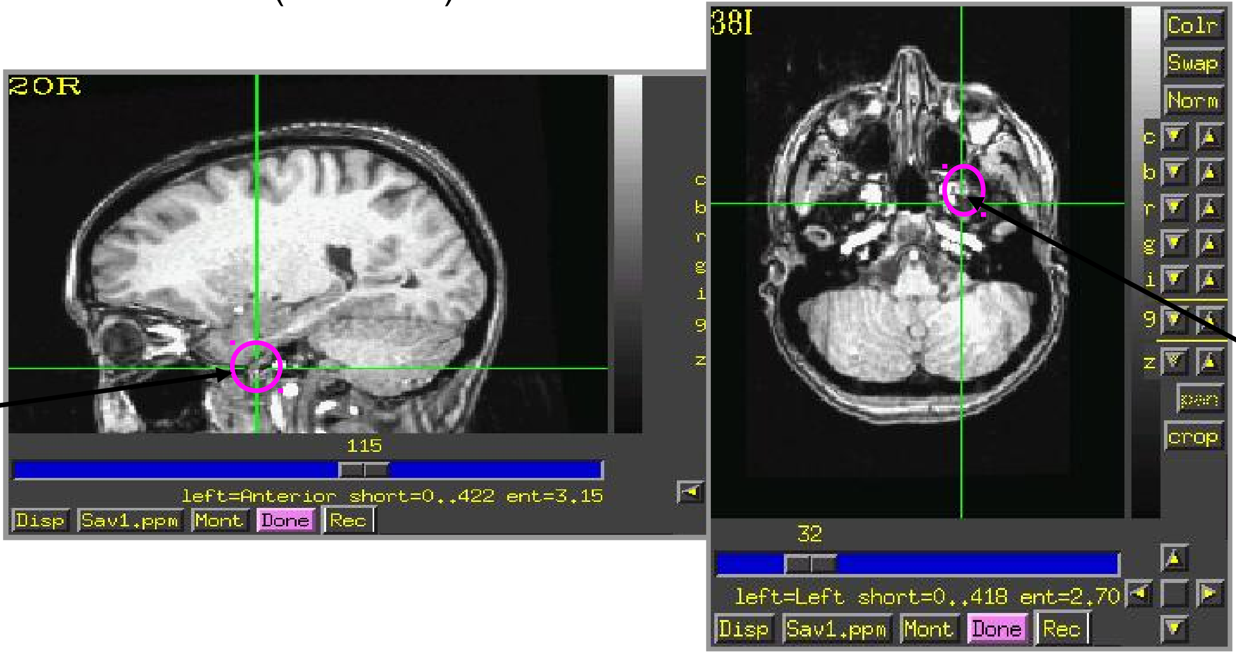

{warp} [A] AFNI 2.56b: AFNI_sample_05/afni/anat+acpc

```
voxel x =   67
voxel y =  148
voxel z =   16
Xhairs  Multi   X+
Color   green
Gap     4       Wrap
Index           0
```

```
Pcolor  white
Scolor  limegreen
Size    8
Gap     3
```

- 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


20R
115
left=Anterior short=0..422 ent=3.15
Disp
Sav1.ppm
Mont
Done
Rec
38I
Colr
Swap
Norm
32
left=Left short=0..418 ent=2.70
crop

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`

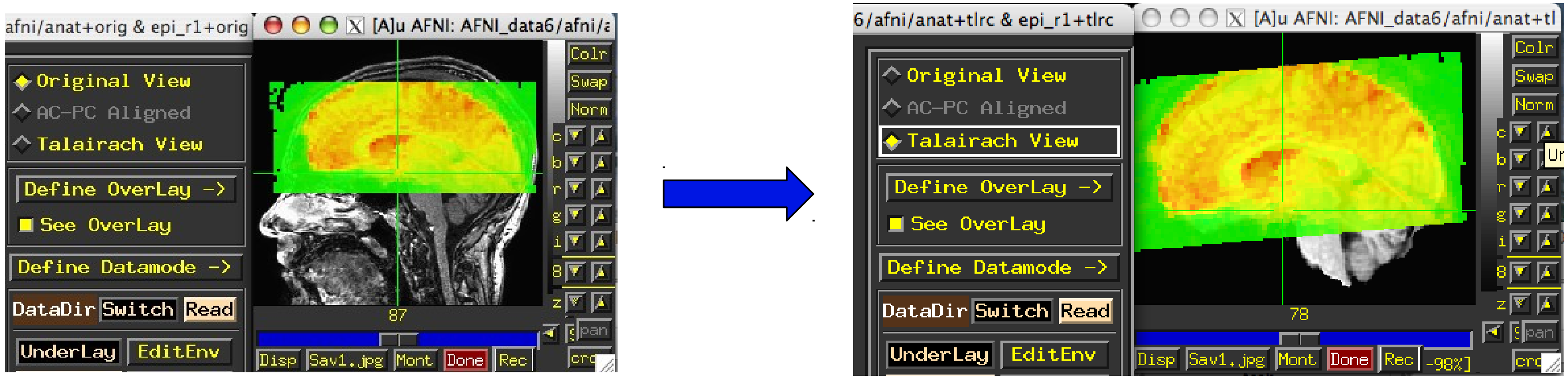
  - In addition, the following functional dataset (which I -- the instructor -- created earlier) should be stored in the `demo_tlrc/` directory:

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

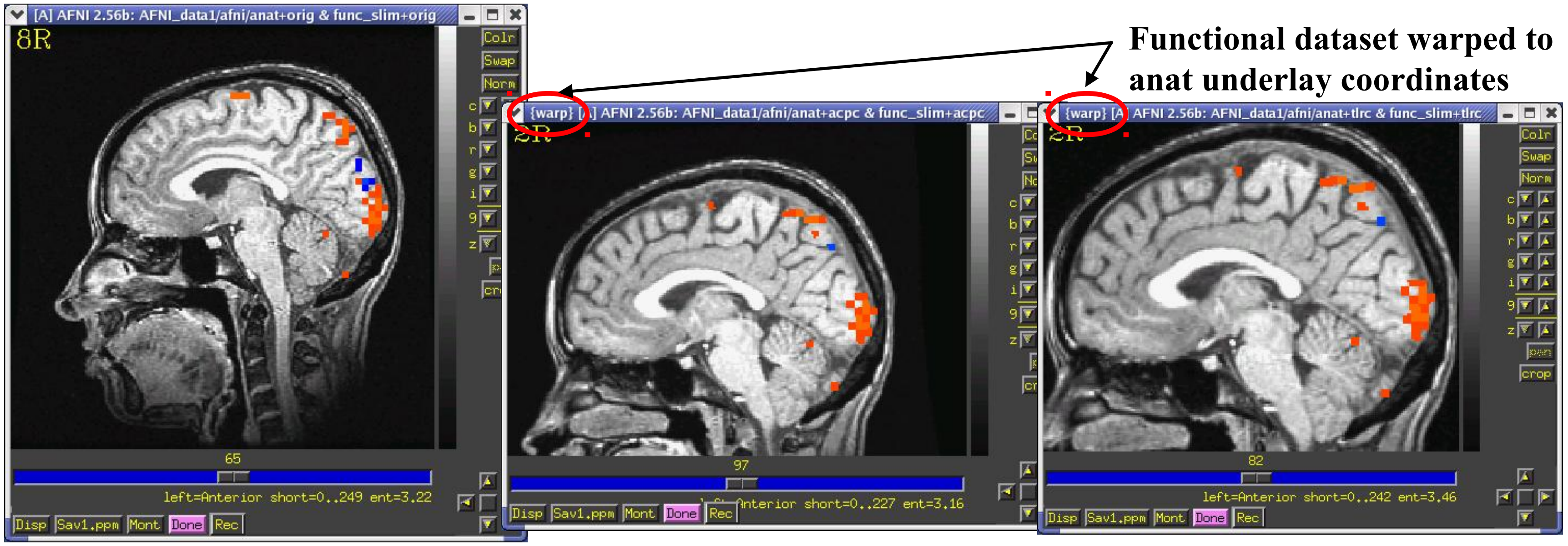
    - Note that this functional dataset is in the `+orig` format (not `+acpc` or `+tlrc`)

- **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):

$$\begin{tikzcd}
\texttt{anat+orig} \arrow[r] & \texttt{anat+acpc} \arrow[r] \arrow[d] & \texttt{anat+tlrc} \arrow[d] \\
\texttt{func+orig} \arrow[u] & \texttt{func+acpc} & \texttt{func+tlrc}
\end{tikzcd}$$



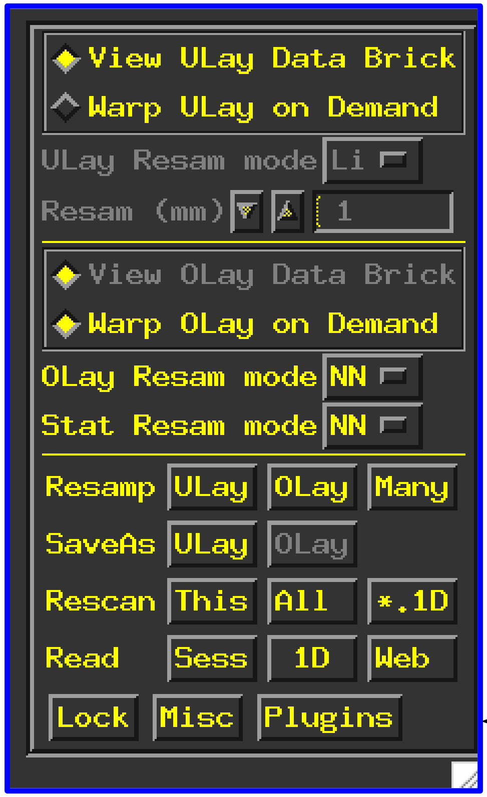
- How does AFNI actually create these follower datsets?
  - After [**Transform Data**] creates **anat+acpc**, other datasets in the same directory are scanned
    - AFNI defines the geometrical transformation ("warp") from **func_slim+orig** using the **to3d**-defined relationship between **func_slim+orig** 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:


[A] AFNI 2.56b: AFNI_data1/afni/anat+orig & func_slim+orig
{warp} [A] AFNI 2.56b: AFNI_data1/afni/anat+acpc & func_slim+acpc
{warp} [A] AFNI 2.56b: AFNI_data1/afni/anat+tlrc & func_slim+tlrc
8R
65
97
82
left=Anterior short=0..249 ent=3.22
Anterior short=0..227 ent=3.16
left=Anterior short=0..242 ent=3.46
Colr
Swap
Norm
crop
Disp
Sav1.ppm
Mont
Done
Rec
Functional dataset warped to anat underlay coordinates

func_slim+orig ⟶ **"func_slim+acpc"** ⟶ **"func_slim+tlrc"**

- 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

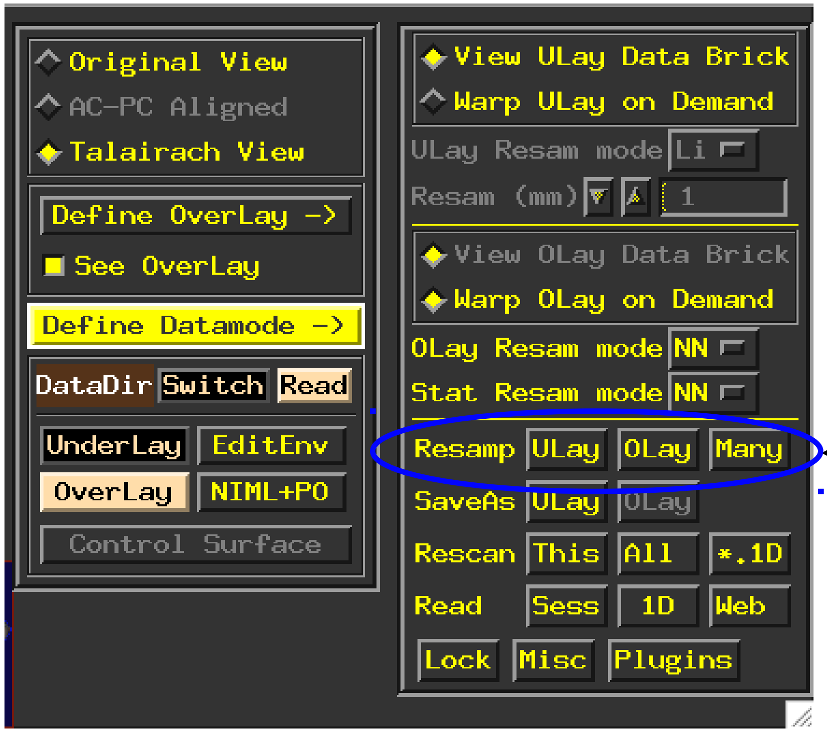
Read new: session directory, 1D file, dataset from Web address

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:
    - How do we write our anat **.BRIK** files to disk?
    - 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.HEADanat+acpc.BRIK
anat+tlrc.HEADanat+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**

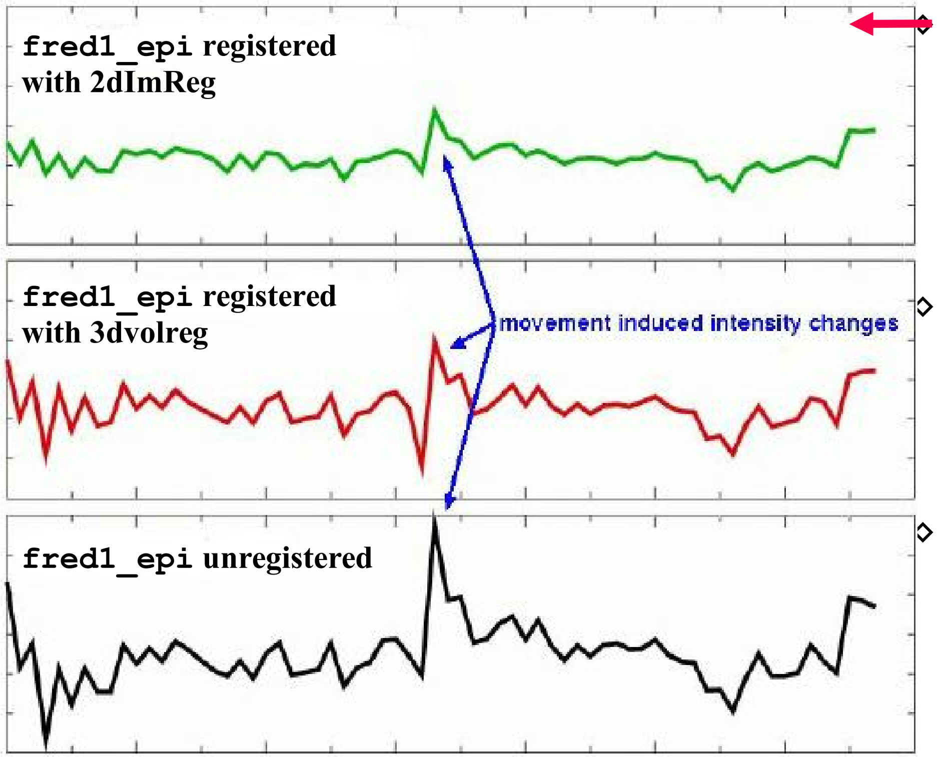
- Why bother saving transformed datasets to disk anyway?
  - Datasets without .BRIK files are of limited use:
    - You can't display 2D slice images from such a dataset
    - You can't use such datasets to graph time series, do volume rendering, compute statistics, run any command line analysis program, 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

Appendix D

Miscellaneous alignment and atlas topics

- 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
```



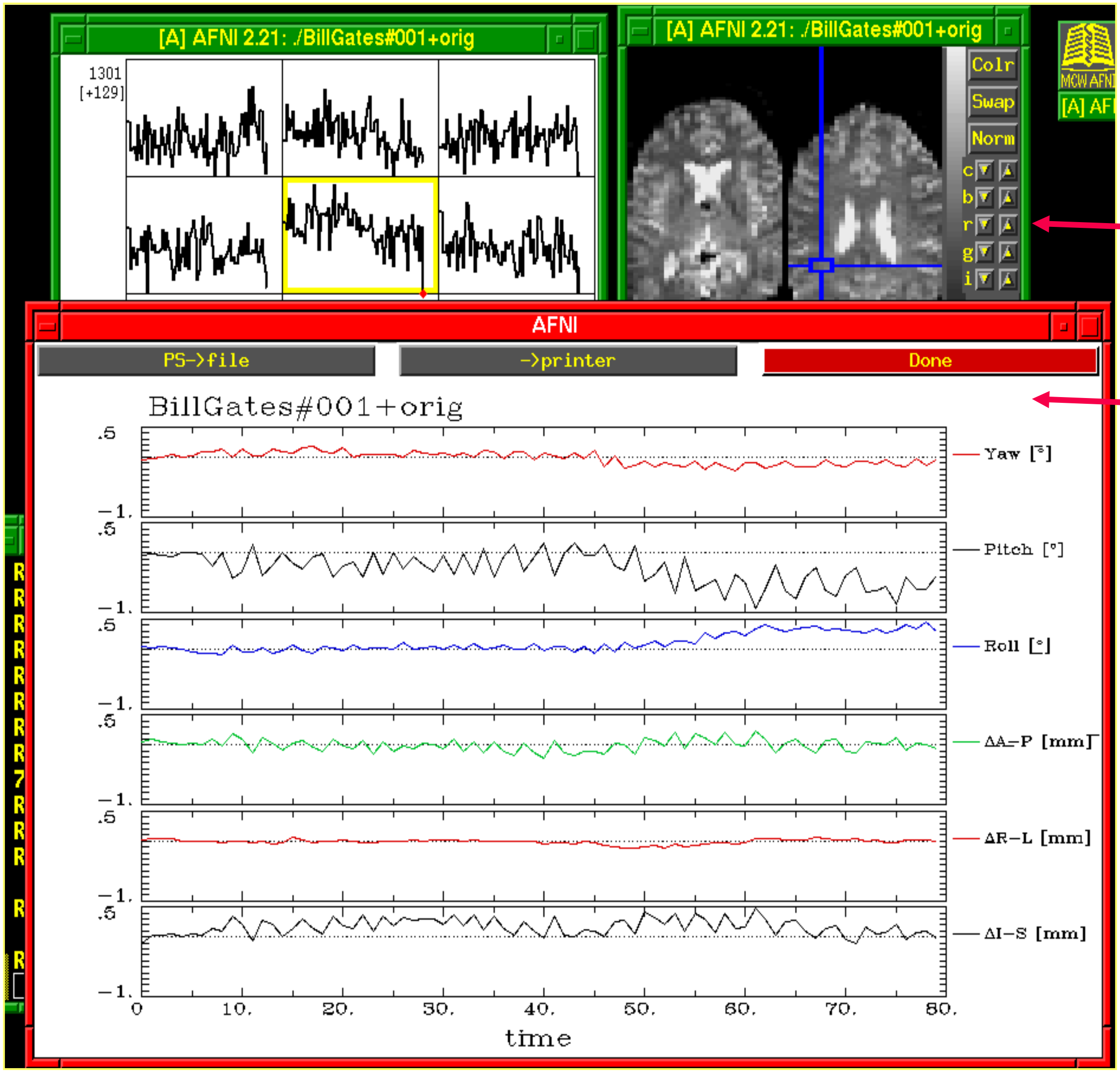
- Graphs of a single voxel time series near the edge of the brain:
  - Top = slice-wise alignment
  - Middle = volume-wise adjustment
  - Bottom = no alignment
- For **<u>this</u>** example, **`2dImReg`** appears to produce better results. This is because most of the motion is 'head nodding' and the 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

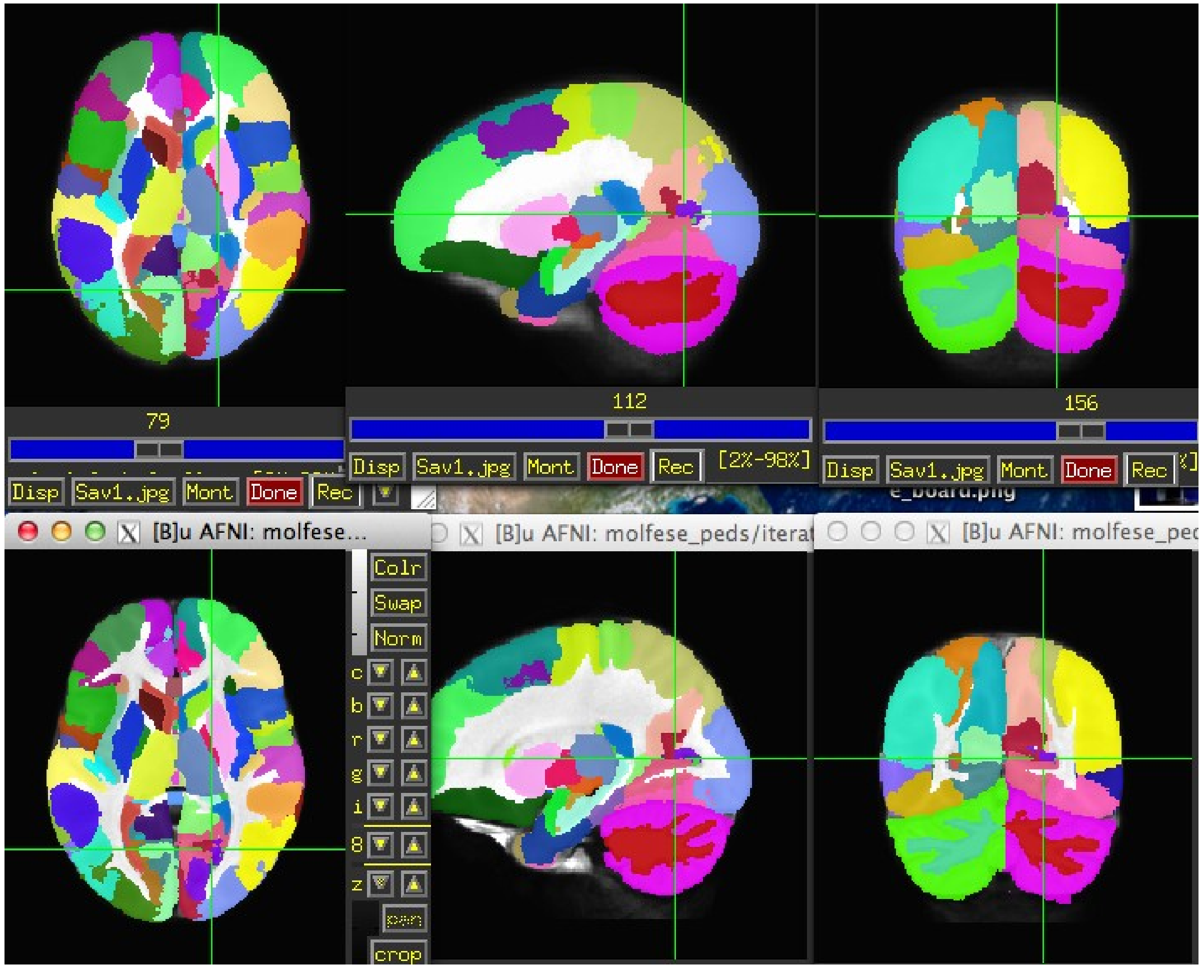
- Realtime motion correction can easily be setup if DICOM images are made available on disk as the scanner is running.
- The script demo.realtime present in the AFNI_data1/EPI_manyruns directory demonstrates the usage:

```
#!/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

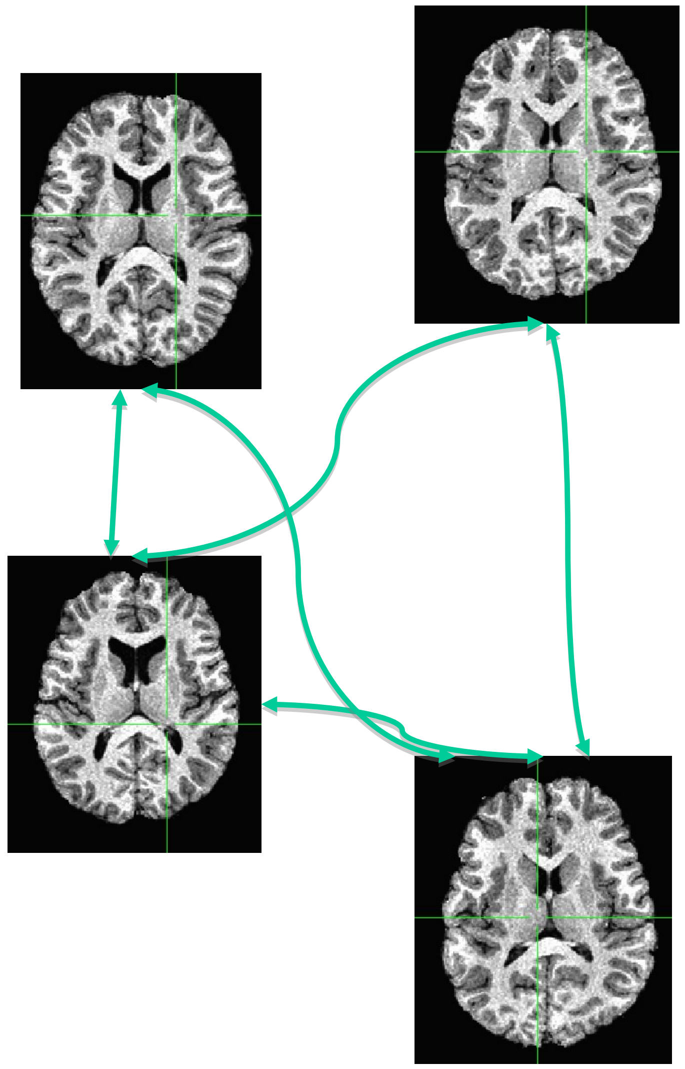
Haskins Pediatric Atlas – MPM atlases



Affine Group

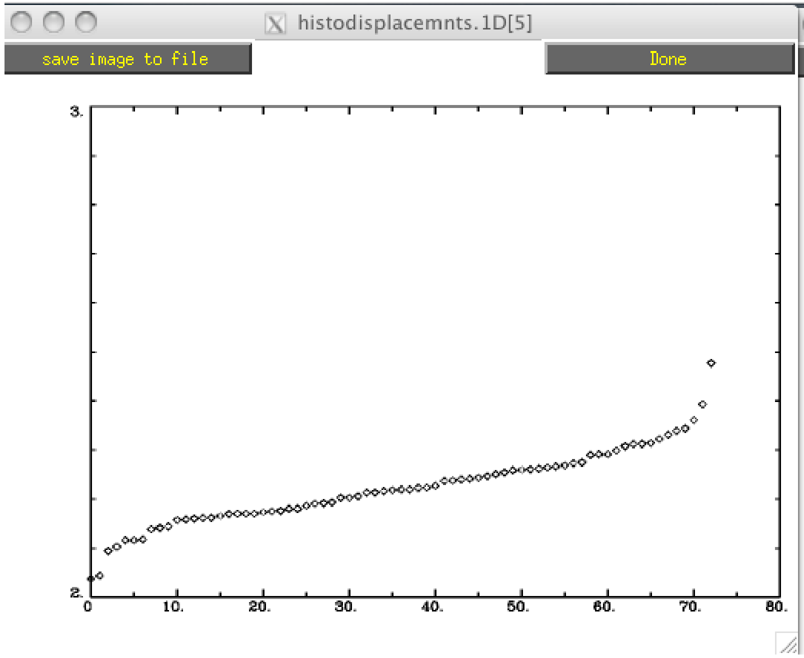
Nonlinear Group I – iterative

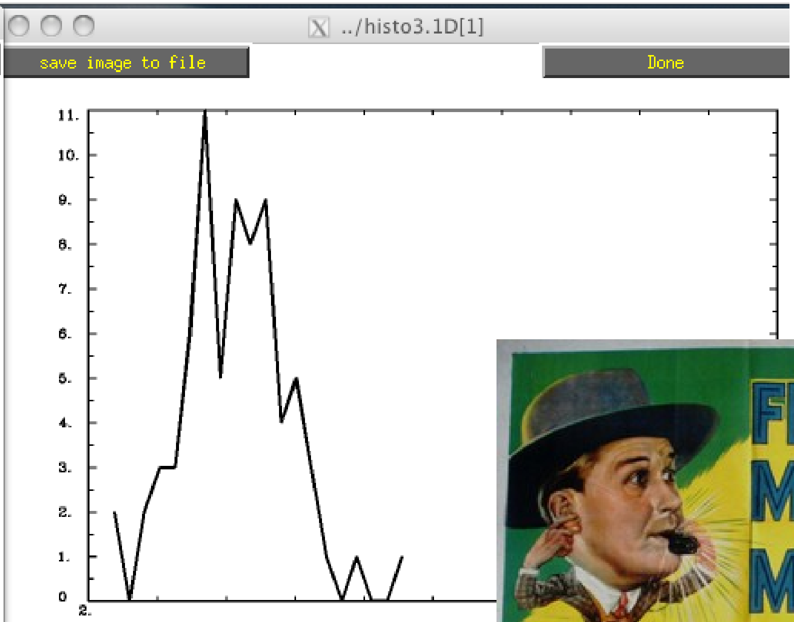
Haskins Pediatric Atlas – Finding the "typical" kid's brain



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 displacement
6. Minimum displacement – Winner! Typical subject
7. Maximum displacement – QC? Atypical subjects

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

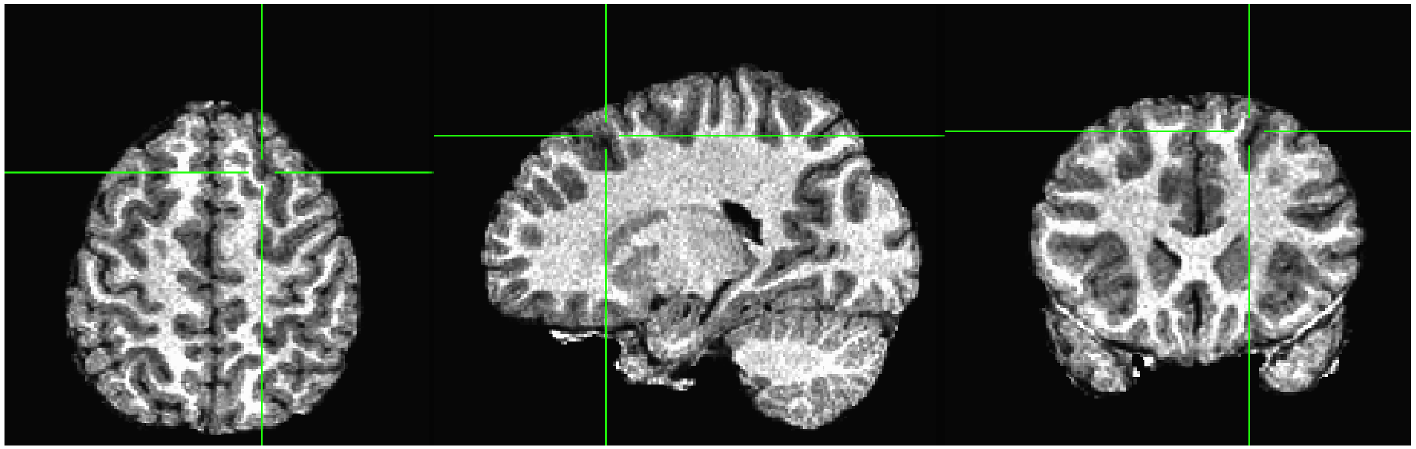




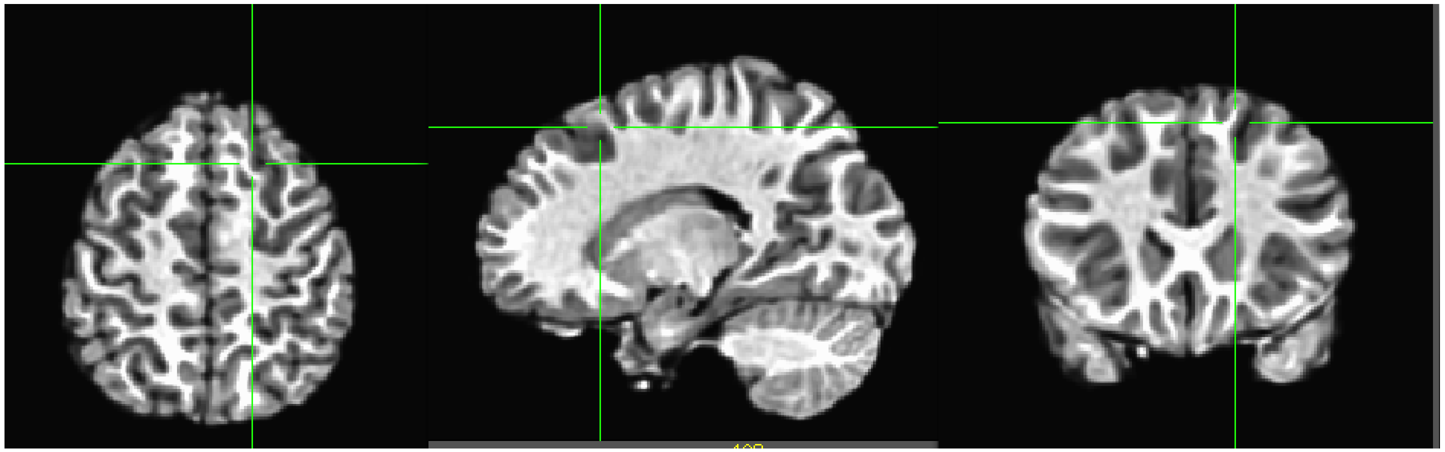
FIBBER McGEE and MOLLY
IN
Heavenly Days
with
EUGENE PALLETTE · GORDON OLIVER
RAYMOND WALBURN · BARBARA HALE
DON DOUGLAS · FRIEDA INESCORT
IRVING BACON · AND THE KING'S MEN
PRODUCED BY ROBERT FELLOWS
DIRECTED BY HOWARD ESTABROOK

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

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

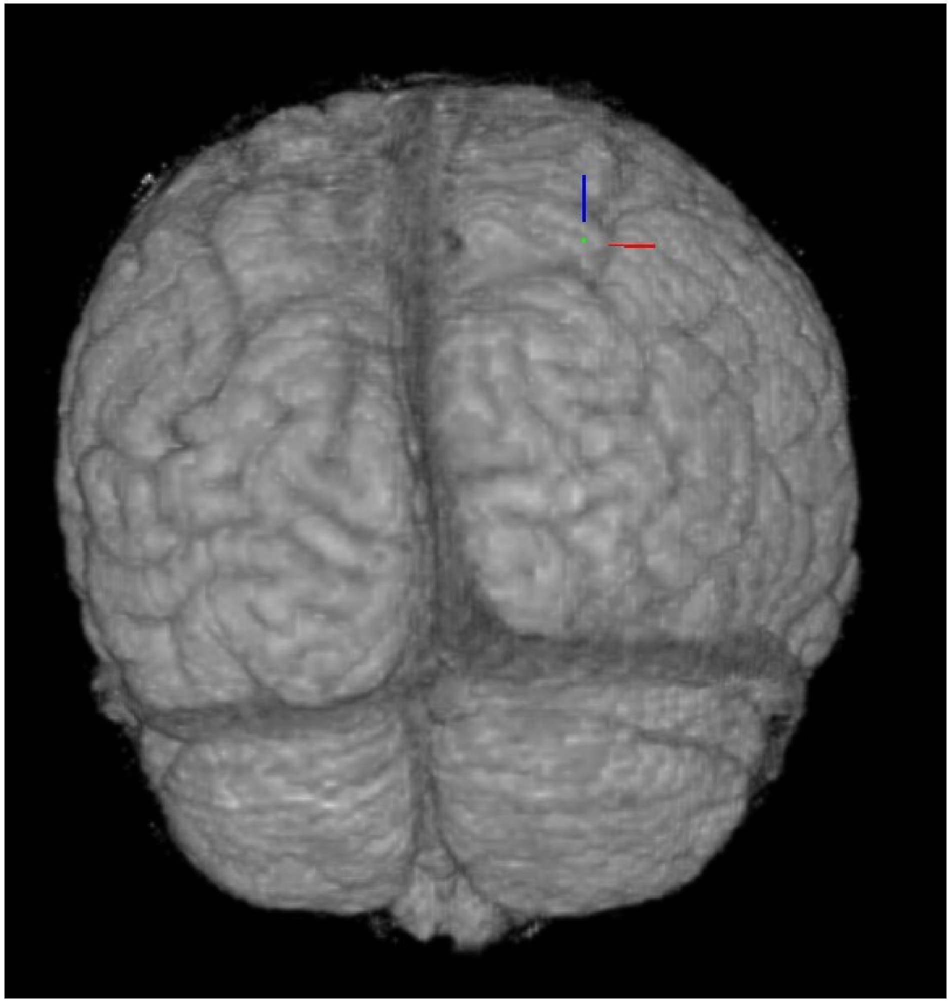


sn_xxx



anisotropically
smoothed

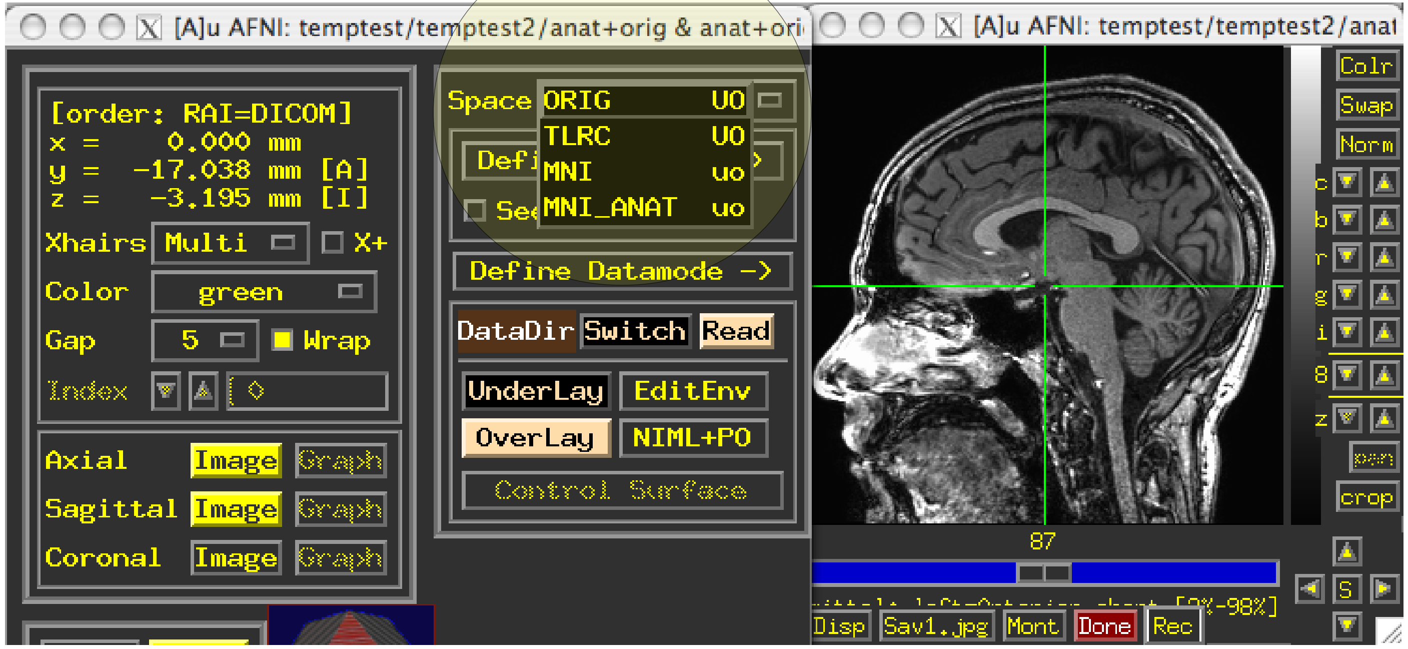
Haskins Pediatric Atlas – Finding the "atypical" kid's brain



Petalia – torque asymmetry

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

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


[A]u AFNI: temptest/temptest2/anat+orig & anat+ori

[A]u AFNI: temptest/temptest2/anat

```
[order: RAI=DICOM]
x =     0.000 mm
y =   -17.038 mm [A]
z =    -3.195 mm [I]
```

Xhairs Multi

Color green

Gap 5

Index 0

Space ORIG

ORIG	UO
TLRC	UO
MNI	uo
MNI_ANAT	uo