

## Where do AFNI Datasets Come From?

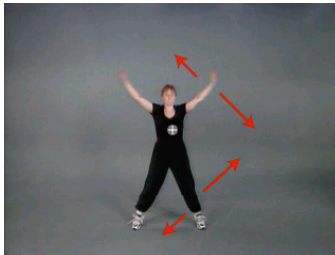
- Method 1: Create datasets with program to3d [principal subject of this talk]
  - ★ Input files are arrays of numbers --- i.e., image files
- Method 2: Realtime input from an external image source program (e.g., directly from scanner's reconstructed images)
  - ★ AFNI program Imon reads image files from GE realtime EPI output, checks them for various errors, sends them into AFNI for display and formatting --- while acquisition continues
  - ★ Sample program rtfeedme can be used to write your own image source program
- Method 3: AFNI programs can read other formats for display and analysis
  - ★ ANALYZE™ 7.5 format ⇒ **.hdr/.img** file pairs
    - ↳ Used by SPM and many other programs
    - ↳ Major drawback: lack of spatial orientation and position information in header
      - ↳ Can be difficult to overlay ANALYZE datasets with other datasets
  - ★ MINC format ⇒ **.mnc** files
    - ↳ Used by software from Montreal Neurological Institute --- mnitools
  - ★ CTF format ⇒ **.sv1** files
    - ↳ Generated from CTF MEG data analysis software package

- ★ Dataset stored as columns of ASCII-formatted numbers ⇒ **.1D** and **.3D** files
  - ↳ Used to store datasets when knowing where the data points are in space isn't important for the analysis
  - ↳ Example: node-wise analysis of group data on surfaces
  - ↳ Each column corresponds to one sub-brick
  - ↳ Each row corresponds to one voxel or node
  - ↳ .1D files: just columns of numbers
  - ↳ .3D files: contain an XML header with geometrical information
- ★ NIfTI-1 format ⇒ **.hdr/.img** file pairs or **.nii** files
  - ↳ New format, modified from ANALYZE 7.5 compatible programs
  - ↳ Supposed to be mostly compatible with ANALYZE 7.5 compatible programs
  - ↳ Format finalized late 2003; will be supported by SPM, AFNI, FSL, Brain Voyager
- Method 4: Output of most AFNI programs is AFNI-formatted datasets ⇒ **.HEAD/.BRICK** file pairs
  - ★ AFNI utility programs exist to re-write AFNI-formatted datasets into ANALYZE, MINC, and .3D formats
  - ★ In the future, AFNI programs will be able to write out NIfTI-1 **.nii** formatted datasets directly

## Creating AFNI Datasets with Program **to3d**

- **to3d** reads image files -- each containing 1 or more 2D slices -- and assembles them into AFNI datasets
- The collection of all the 2D slice data forms the **.BRIK** file
  - ★ An AFNI dataset can contain a single slice
- You must also provide **to3d** with some auxiliary data (for the **.HEAD** file):
  - ★ Orientation of slices in space
  - ★ Size of slices or of the voxels
  - ★ Slice offset -- where is the dataset volume located in space?
  - ★ For 3D+time datasets, you also need slice timing information
  - ★ **to3d** 'knows' how to get some of this auxiliary information from image file headers for some image file formats:
    - ➔ ANALYZE 7.5 **.hdr/.img** pairs contain voxel size information
    - ➔ Siemens **.ima** Files contain voxel size and orientation information
    - ➔ GE **I**. Files contain voxel size and orientation information
    - ➔ DICOM Files contain lots of relevant information
      - ◇ But manufacturers' variations on DICOM are frustrating

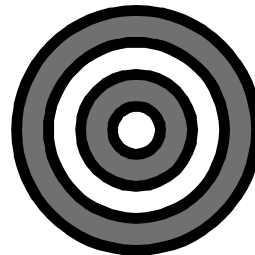
- **to3d** runs in two modes:
  - ★ *Command line mode*: you provide all auxiliary information on command line
  - ★ *Graphical interface (GUI) mode*: you provide auxiliary information by filling out an on-screen form
- **Sample Study**: data from NIH GE 3Tesla Scanner
  - ★ Files stored in directory [AFNI\\_data1/](#)
  - ★ Anatomical (SPGR) data  $\Rightarrow$  3D dataset (no time; 1 sub-brick)
    - ↳ 124 sagittal slices in subdirectory [SPGR\\_anat/](#)
  - ★ Functional (EPI) time series data  $\Rightarrow$  3D+time dataset (110 sub-bricks or time pts)
    - ↳ 2970 images (27 sagittal slices, 110 reps) in subdirectory [EPI\\_run1/](#)
    - ↳ Visual motion task: Videos of moving humans and tools (Beauchamp et al, 2002):



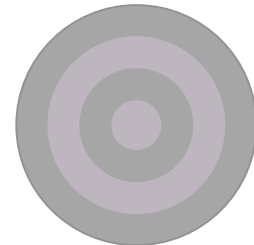
*Moving Human*



*Moving Tool*



*High Contrast  
Moving Grating*



*Low Contrast  
Moving Grating*

- Experiment log, taken at scanner:

**NIMH-LBC--Methods FMRI Research**

**RUN DATA**

Exp Code: DD

Date/Time: March 20, 2000 9 am

Investigator: Mike B.

Scanner: 3T

Coil: Wong/MAI/ GE

**ANAT Scan1:** Type: SPGR/FSE/MPIR TE(ms):\_\_\_ TR(ms):\_\_\_ Flip:\_\_\_ FOV(mm): 240

Matrix: 256x256 #slices: 124 Plane: Ax/Cor/Sag Thickness(mm): 1.2 First: 70.0 L

Last: 77.6 R

**EPI Scan:** GE-EPI /SE-EPI/ GE RT EPI TE(ms):\_\_\_ TR(ms): 2.5 sec Flip:\_\_\_

Plane: Ax/ Cor/ Sag FOV(mm): 240 Matrix: 64 Thickness(mm): 5.0

#slices: 27 (110 reps each) First: 69.0 R Last: 61.0 L

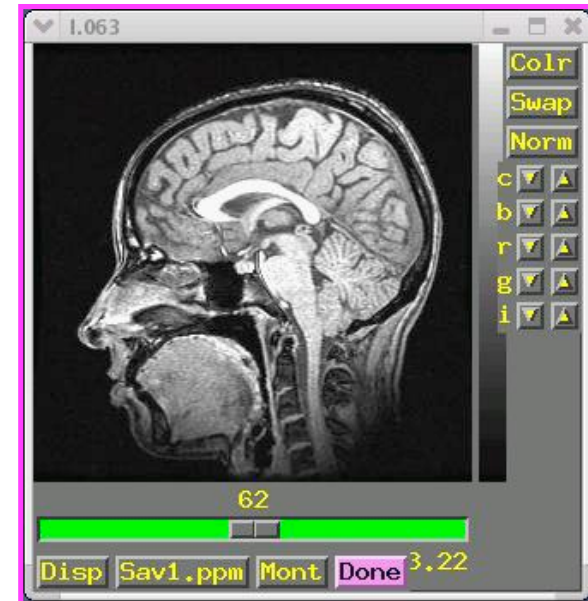
Run#	Time	Conditions	Stimulus File	Data File	Response File
1			001/1.001 ---> 041/1.972		(block design)
2			041/1.973 ---> 101/1.945		
3			101/1.946 ---> 161/1.918		
4			161/1.919 ---> 221/1.891		

- Using **to3d** to assemble the SPGR dataset:
  - ★ **cd AFNI\_data1/SPGR\_anat** ⇒ change directory, to get at images
  - ★ **ls** ⇒ to see what files are there (should see I.001 . . . I.124)
  - ★ **to3d I.\*** ⇒ run **to3d**, reading in all the images files --- GUI pops us:



- ↳ **to3d** understands **GE I.\*** files, and so has filled in some of the GUI
- ↳ Note: z-origin field **70.0 L** corresponds to experiment log

- ★ To check images that were just input, click the [[View Images](#)] button in the **to3d** GUI
  - ↳ Window is the same as the AFNI image viewer
  - ↳ Slider below image lets you move between slices



- ★ In this example, **to3d** has all the information needed from the **I.\*** headers
  - ↳ All you need to do is supply the data Prefix , then press [[Save Dataset](#)]
    - ↳ Look at the bottom right of the **to3d** GUI for these controls
    - ↳ I suggest the prefix anat
  - ↳ Dataset files **anat+orig.HEAD** and **anat+orig.BRIK** will be created
  - ↳ Then press [[quit](#)] button *twice* to exit **to3d** GUI
- ★ Script version (no GUI): **to3d -prefix anat I.\*** would create a dataset with no user intervention
- ★ Later: will give a more complicated example of assembling data from 'naked' image files, where no header information is available

- Using to3d to assemble the EPI 3D+time dataset:

- ★ `cd ../EPI_run1` ---> change directory to get at images

- ★ `ls` ---> to see what files are there (should see files I.0001 . . . I.2970)

- ★ We do not just do to3d I.\* to create a 3D+time dataset

- ★ For historical reasons, the time-axis information must be given on the `to3d` command line.

- ↳ Cannot be modified from the GUI

- ★ Command line: to3d -time:zt 27 110 0 alt+z I.\*

- ↳ -time:zt ⇒ slices usually presented in order of space (z), then time (t)

- ↳ -time:tz is needed at some sites

- ↳ If in doubt, do `to3d I.*` or `aiv I.*`, use viewer to look at slices and see their order [aiv = AFNI Image Viewer program]

- ↳ 27 110 ⇒ there are 27 slices in z and 110 in t (2970 total)

- ↳ 0 ⇒ the TR for volume acquisition will be read from the image headers

- ↳ If not available, could put 2.5s or 2500 instead of this 0

- ↳ alt+z ⇒ slices are gathered in alternating order in the +z direction

- ↳ Most EPI acquisitions are really 2D multislice, spread out through time

- ↳ AFNI header can contain information about slice timing offsets

- ↳ Other possible modes: zero (for 3D), @filename (to specify each slice)



to3d

x orientation (across screen) Superior-to-Inferior  
y orientation (down screen) Posterior-to-Anterior  
z orientation (slices 0,1,..) Right-to-Left

Datum: short [64x64]  
View: Original View

Field of view (mm) 240  
TR = 2.500 (s)  
NR = 110 Nz = 27

x voxel size (mm) 3.75  
y voxel size (mm) 3.75  
z voxel size (mm) 5

x origin (mm) [left edge] 118.125 S  
y origin (mm) [top edge] 118.125 P  
z origin (mm) [slice 0] 69 R

Field of view (mm) 240

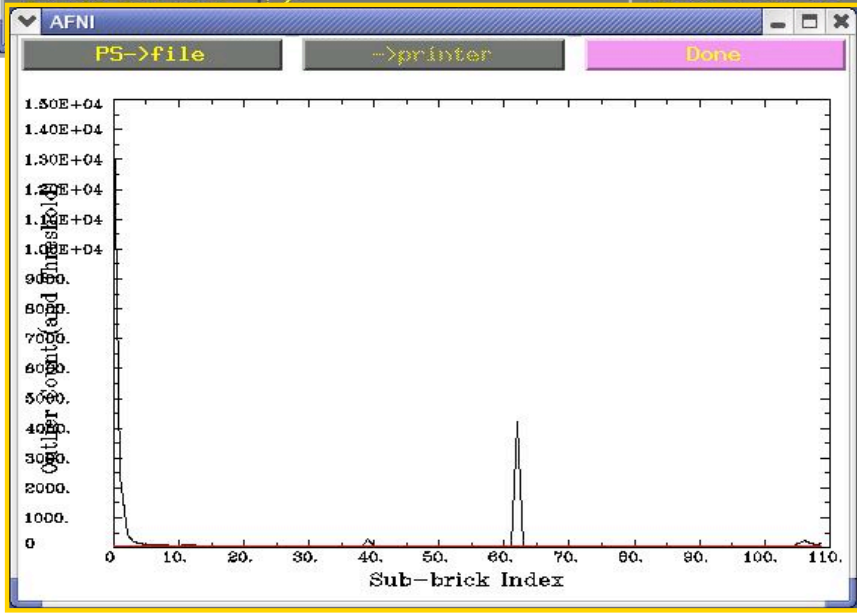
TR = 2.500 (s)  
NR = 110 Nz = 27

Copy geometry of this dataset  
Type of data in the images 3DIM\_HEAD\_ANAT  
Field below not applicable

Anatomy parent is this dataset  
Type of anatomy in the images Echo Planar

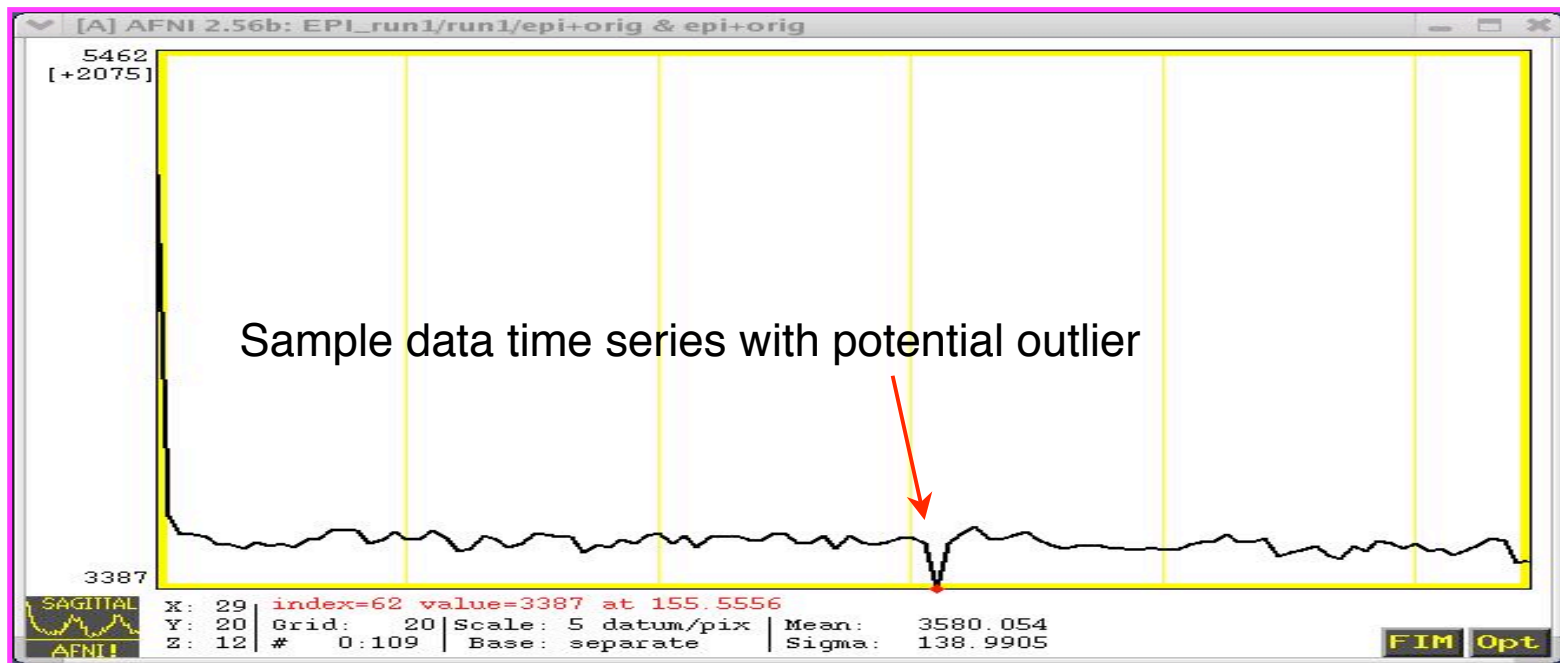
Byte Swap[2]  
button help  
View Images  
Save Dataset  
quit

Session direct. / Prefix for 3D

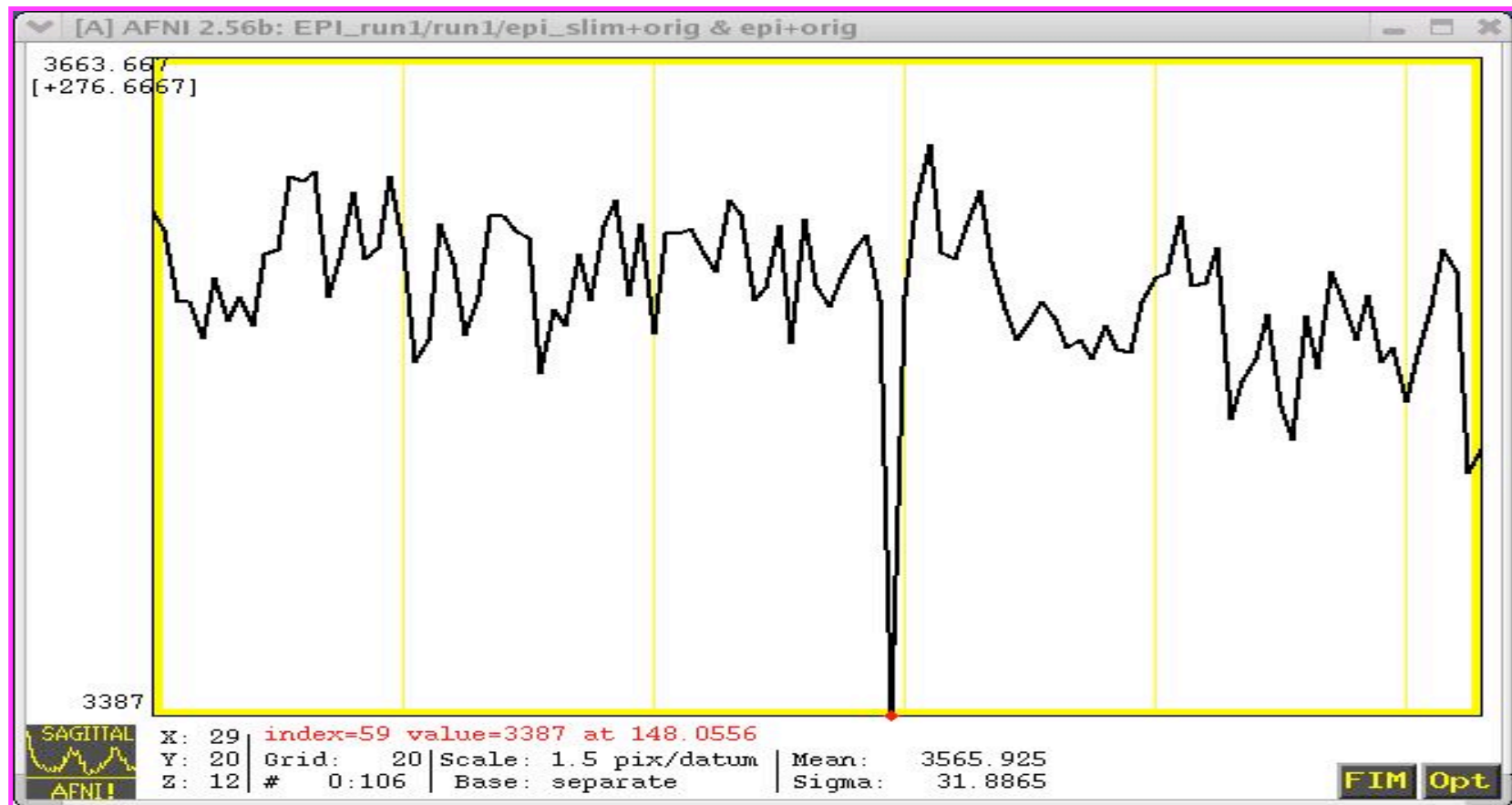


```
*****  
* Checking for *  
* time series *  
* outliers *  
*****  
to3d WARNING:  
Significant outliers detected in these sub-bricks:  
  0  1  2  3  4  5  6  7  8  9 10 11  
 12 39 62 73 105 106 107 108 109  
You should inspect the dataset for possible corruption.  
[Outliers are defined as in program 3dToutcount. ]  
[Outliers early in an EPI time series may be due to ]  
[the longitudinal magnetization equilibration effect. ]  
[Other causes are subject movement, scanner problems, ]  
[or anything that makes a time series look irregular. ]  
[ 3dToutcount -save outnam dataset | lplot -stdin ]  
[can be used to make a dataset 'outnam' that marks ]  
[outlier voxels; see 3dToutcount -help for details. ]
```

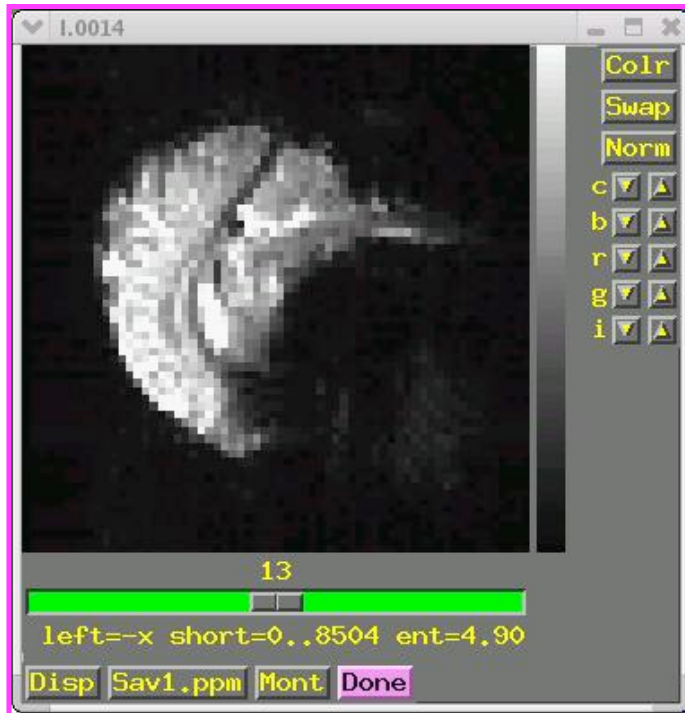
- ★ **Outliers** are data values that are very different from other values in the same time series
  - ↳ **to3d** reports sub-bricks (time points) that have a lot of outliers
  - ↳ You should use AFNI to look at these time points to see if there are major problems (e.g., head motion, scanner artifacts)
  - ↳ **to3d -skip\_outliers** option lets you skip the outlier detection step
  - ↳ Utility program **3dToutcount** can also report outliers and can even make a dataset with the 'outlier-ness' of each voxel value



- The outlier becomes much more obvious when the first three time points of the time series (which show possible scanner artifacts) are removed:



★ Again, fields in the GUI were filled from the `I.*` headers



- ★ In this example, the EPI and SPGR datasets are both sagittal slices. However, AFNI can work with SPGR/EPI datasets that have different planes (e.g., if SPGR is coronal and EPI is axial)
  - ↳ Programs 3dresample and 3daxialize can rewrite datasets in new orientations
- ★ Note slice thickness and slice offset (“z origin”)
  - ↳ Values match experiment log (that’s good)
- ★ Time information is displayed in GUI, but not editable
- ★ Have set “Type of Anatomy” to “Echo Planar”
  - ↳ Just acts as a reminder to user

★ Script version: `to3d -time:zt 27 110 0 alt+z -prefix epi_r1 I.*`

★ Program `3drefit` can be used to change some header items in an AFNI dataset after it is created

↳ Example: `3drefit -TR 1s epi_r1+orig` will change the TR of the dataset to 1 second

## Creating New AFNI 3D+time Datasets with Program **Imon**

- **Imon** can be run during a scanning session on a **GE scanner**, to monitor the collection of time series `I.*` files. The user is notified of any missing or out-of-sequence slices
- **Imon** can also be run separate from scanning, either to verify the integrity of `I.*` files, or to create AFNI 3D+time datasets by using the **-GERT\_Reco2** option
  - ★ **Imon** is run in command line mode
    - ↳ The **-GERT\_Reco2** option is added to the command line so that `I.*` files examined by **Imon** can then be assembled into an AFNI 3D+time dataset
    - ↳ When not being used in real-time mode, the **-quit** option is added so that **Imon** will terminate after processing all of the `I.*` files

- Why not use **to3d** directly to create AFNI datasets?
  - ★ EPI images collected using GE's real time EPI sequence are saved in a peculiar fashion
    - ↳ Only 999 image files can be stored in a single directory
    - ↳ If a run consists of 110 volumes of 27 slices each, we have 2970 image files
    - ↳ With a limit of 999 `I.*` files per directory, a run made up of 2970 images would have to be saved in 3 separate directories (numbered **001/**, **021/**, and **041/**):
      - ⇒ E.g., **001/I.028...I.999** + **021/I.001...I.999** + **041/I.001...I.999**  
= 2970 `I.*` files total
    - ↳ The second run would be stored in directories **061/I.001** ⇒ **101/I.973**, the third run in **101/I.974** ⇒ **161/I.946**, and the nightmare continues...
    - ↳ This setup already makes it difficult to delineate between runs. Now imagine what happens if the scanner hiccups, if you stop a scan in the middle and start a new one, or start collecting scans with a different number of slices!
- **Imon** attempts to identify complete scans from the images in those directories. It also monitors missing or out-of-order images, and generates the commands necessary to turn them into AFNI bricks using the script **GERT\_Reco2**

- Using **Imon** to assemble the EPI 3D+time datasets

- ★ **cd ../EPI\_manyruns** ⇒ change directory to get at GE subdirectories containing images

- ★ **ls** ⇒ to view the GE subdirectories containing 4 runs worth of **I.\*** files

- ↳ Directories are numbered in multiples of 20 (default naming system used by the GE scanner): 001/ 021/ 041/ 061/. . . 201/ 221/

- ★ Command line: **Imon -start\_dir 001 -GERT\_Reco2 -quit**

- ↳ **-start\_dir** specifies the starting directory where **Imon** will begin monitoring the images. In this example, our start directory is 001/

- ↳ **-GERT\_Reco2** will create a script called 'GERT\_Reco2', similar to the one that program **Ifile** creates (for more info on **Ifile**, type **Ifile -help**).

- ↳ The **GERT\_Reco2** script may be run to create the AFNI datasets corresponding to the **I.\*** files

- ↳ **-quit** will terminate **Imon**, after all image files have been examined,

- ↳ If **-quit** is not used, the program will forever wait for more images, until **<ctrl-c>** is used to terminate the program

- ★ For a full list of **Imon** options, type **Imon -help**

- Output from **Imon** command:

```
File Edit View Terminal Go Help
[morwen EPI_manyruns]$ Imon -start_dir 001 -GERT_Reco2 -quit

Imon running, use <ctrl-c> to quit...

-- scanning for first volume

-- first volume found
-- scanning for additional volumes...
-- run 1: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 2
7 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53
54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80
81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100 101 102 103 104 10
5 106 107 108 109 110
-- run 2: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 2
7 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53
54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80
81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100 101 102 103 104 10
5 106 107 108 109 110
-- run 3: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 2
7 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53
54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80
81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100 101 102 103 104 10
5 106 107 108 109 110
-- run 4: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 2
7 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53
54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80
81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100 101 102 103 104 10
5 106 107 108 109 110

final run statistics:
volume info :
  slices : 27
  z_first : 69.0000
  z_last : -61.0000
  z_delta : -5.0000

run # 1 : volumes = 110, first file = 001/I.028
run # 2 : volumes = 110, first file = 061/I.001
run # 3 : volumes = 110, first file = 101/I.973
run # 4 : volumes = 110, first file = 161/I.946

[morwen EPI_manyruns]$
```

- **Imon** will search for missing or out-of-order images.
- Any errors will be noted on the screen
- The starting point for each run is reported at the end of **Imon**'s examination

```
run# 1: volumes 110, first file = 001/I.028
run# 2: volumes 110, first file = 061/I.001
run# 3: volumes 110, first file = 101/I.973
run# 4: volumes 110, first file = 161/I.946
```

```
run # 1 : volumes = 110, first file = 001/I.028
run # 2 : volumes = 110, first file = 061/I.001
run # 3 : volumes = 110, first file = 101/I.973
run # 4 : volumes = 110, first file = 161/I.946
```



- **ls EPI\_manyruns** ⇒ to view the newly created **GERT\_Reco2** script
  - ★ This script contains the commands that will automatically create bricks from the complete scans and store them in a newly created subdirectory called **afni/**
  - ★ To run the script, type **./GERT\_Reco2**
- **cd afni** ⇒ to get at datasets
- **ls** ⇒ to view the AFNI 3D+time datasets:

**Outbrick\_r1+orig.HEAD**

**Outbrick\_r2+orig.HEAD**

**Outbrick\_r3+orig.HEAD**

**Outbrick\_r4+orig.HEAD**

**Outbrick\_r1+orig.BRIK**

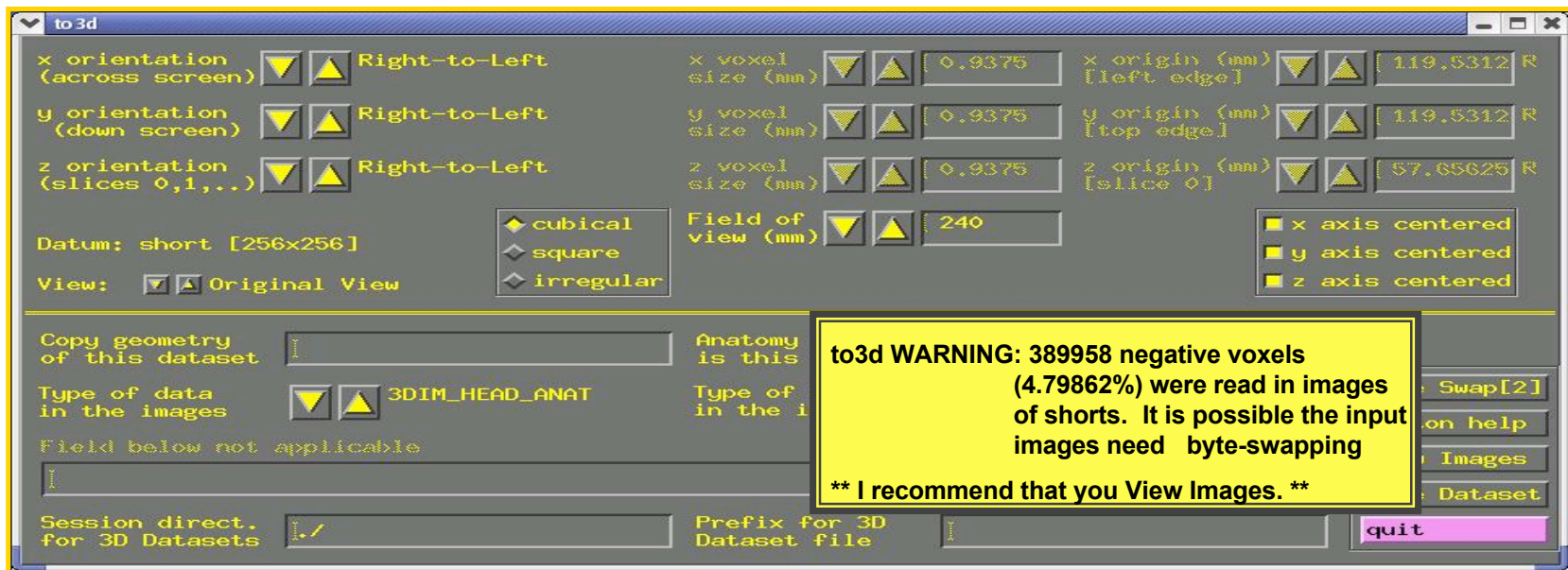
**Outbrick\_r2+orig.BRIK**

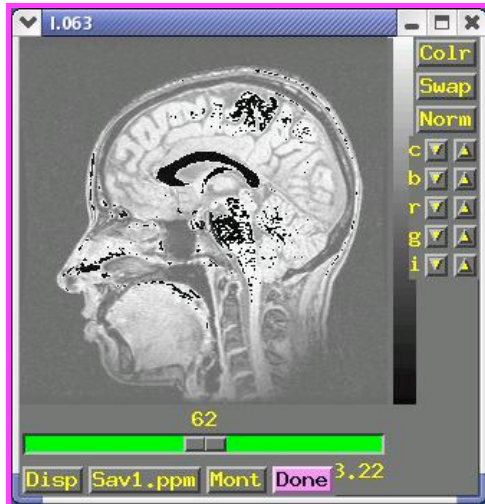
**Outbrick\_r3+orig.BRIK**

**Outbrick\_r4+orig.BRIK**

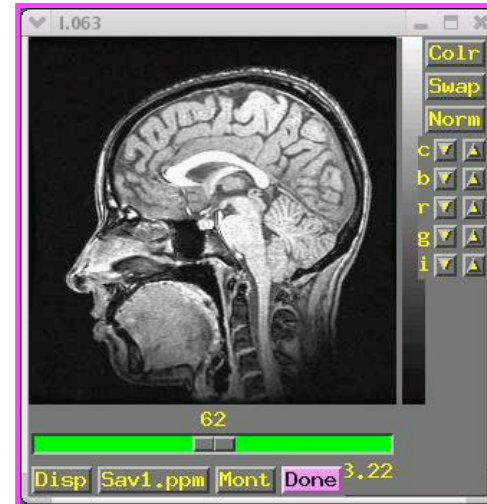
# Assembling 'Naked' Images into AFNI Datasets

- 'Naked' image  $\Rightarrow$  image file without header data that AFNI understands
- User must supply geometrical information to **to3d**
  - ★ This is when the written experiment log is critical!
- The **SPGR\_naked/** directory contains the same SPGR images as before, but stripped of all header information
  - ★ Each file has 131072 bytes = 256 x 256 16-bit integers ('shorts')
  - ★ **cd ../.. /SPGR\_naked** (to get at images N.001...N.110), then **to3d N.\***





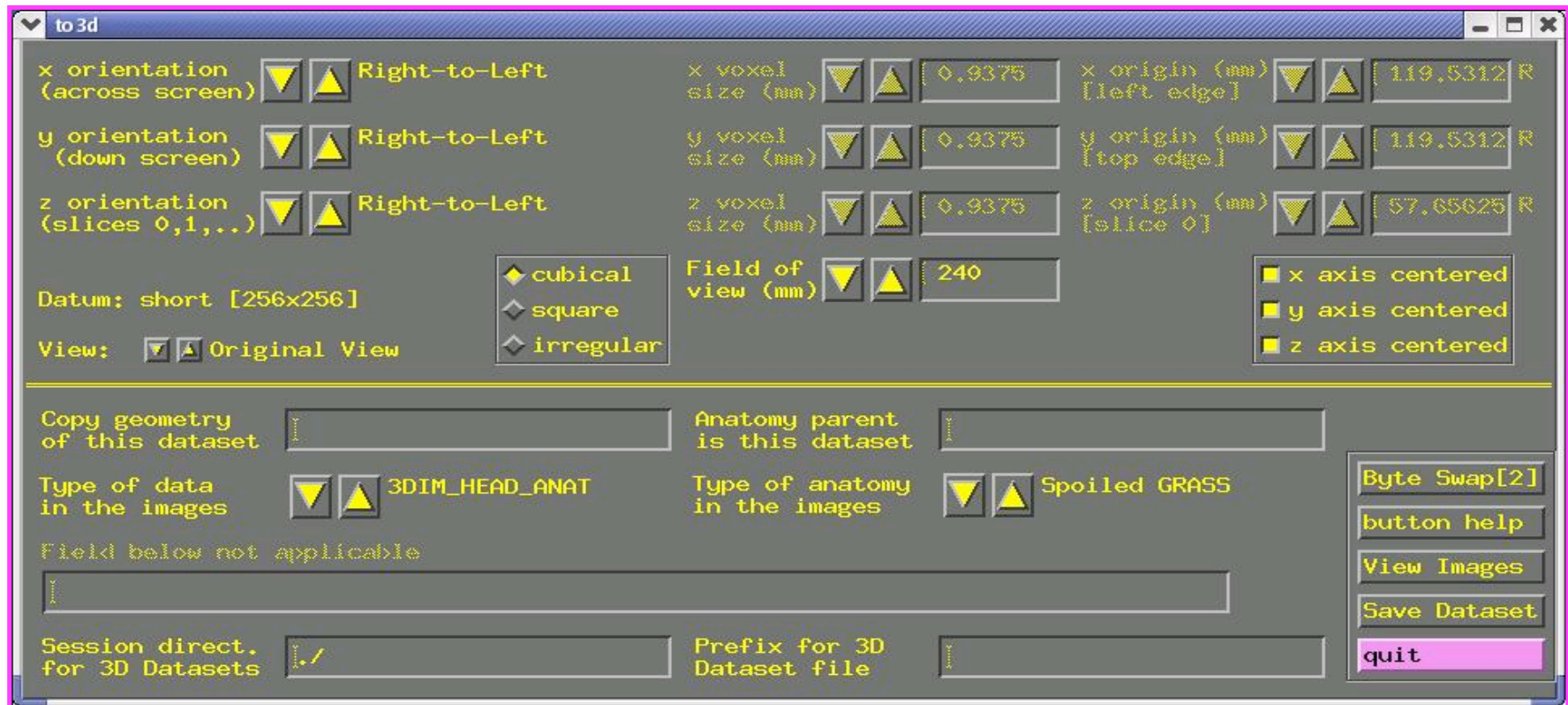
*Linux/Intel computers*



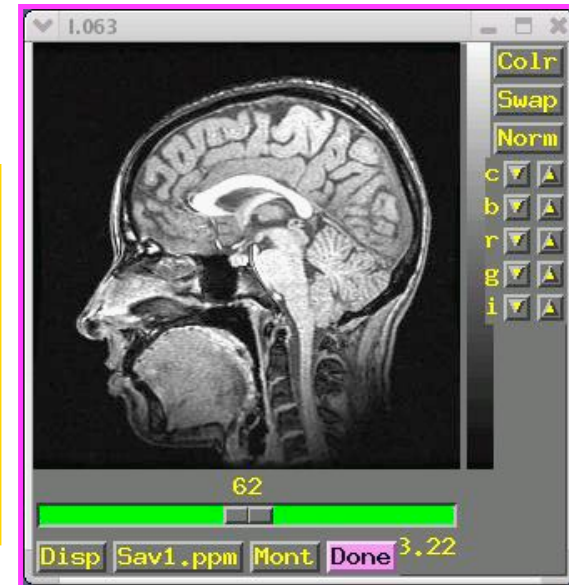
*SGI/Sun/etc. computers*

- On Linux/Intel computers: the peculiar appearance of images shows that something is wrong:
  - ★ MR images from scanners that are stored as shorts: 2 bytes per number
  - ★ Like a 2-digit decimal number: “93” means “9 x 10 + 3”
    - ↳ By universal custom, we write the “9” first
    - ↳ Could also write the same number as “39” (if we had a different custom)
  - ★ Customs for computers are not so universal
    - ↳ Sun and SGI systems store 2 byte numbers in reverse order from Intel
    - ↳ Result is that numbers are mangled (and some show up as negative)
    - ↳ Solution: press `to3d`'s [[Byte Swap\[2\]](#)] button and images are fixed!

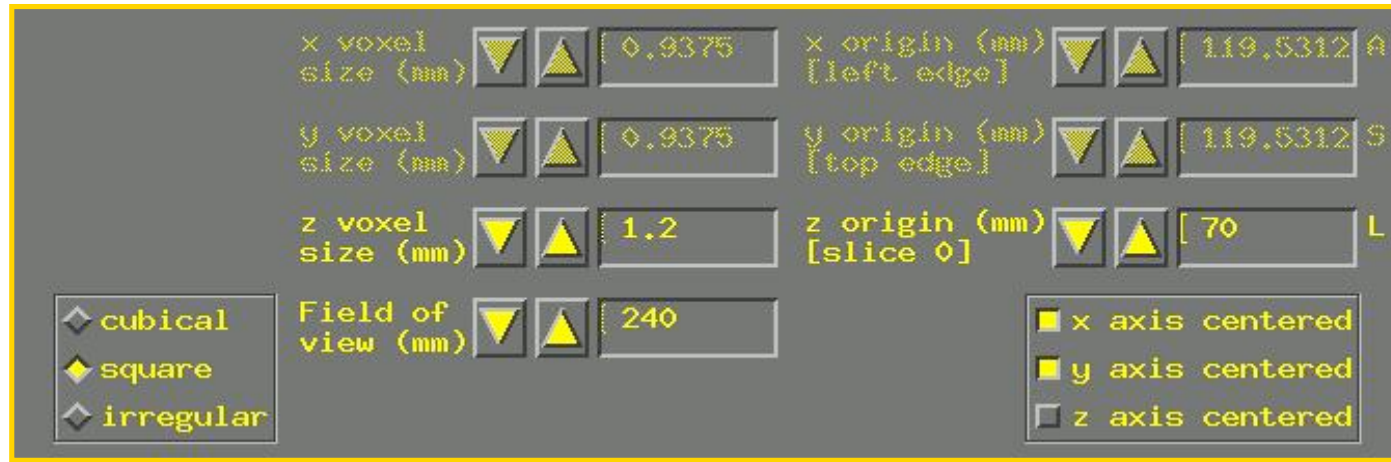
- Same `to3d` control panel (without negative voxel warning):



- Above the double line: must fill out 3 types of geometry information
  - ★ Left column: orientation of the dataset axes
  - ★ Middle column: size of the dataset images or voxels
  - ★ Right column: offset of the first slice



- Screen shot above shows correct orientation for this dataset
  - ★ Use the image viewing window to judge how images are laid out
  - ★ Click the arrows to scroll through the 6 possible options for each orientation to set correct values
  - ★ “x orientation” of dataset is across the screen (Anterior to Posterior)
  - ★ “y” orientation of dataset is down the screen (Superior to Inferior)
  - ★ “z” orientation” of dataset is in increasing slice order (from Left to Right)
    - Must know subject’s right from left (from experiment log sheet or vitamin E tablet placed on one side of the head)
    - Determine this by using the slider at the bottom of image window



- To set dataset geometrical size/location, experiment log sheet is essential
- Screen shot above shows setting slice thickness to **1.2 mm**
  - ★ Default Field of view (FOV) of **240 mm** is correct for these images
  - ★ The default voxel geometry is “cubical”, which is incorrect for this example
  - ★ Must set geometry to “**square**” (x size = y size, z size different)
  - ★ Then set “z voxel size” to correct value (by typing in box)
- Screen shot shows setting of first slice to **70.0 mm in Left (L)** direction
  - ★ Default is that slices are centered in the magnet
  - ★ This default is usually not the case in the z direction
  - ★ Click “z axis centered” off
  - ★ Enter offset (here 70.0 mm) into the “z origin” box

- Final required steps:
  - ★ Enter prefix for new dataset into [**Prefix**] text box at lower right of **to3d** control window
    - ↳ Choosing a good prefix is important for keeping datasets organized
  - ★ Press [**Save Dataset**] button
  - ★ Press [**quit**] (twice) to exit **to3d**
  - ★ The new dataset files should show up when you use command **ls**
  - ★ For organizational purposes, you may want to move your datasets to some other directory

- Geometry parent lets you copy the geometry data from a pre-existing dataset and apply it to the dataset now under construction
  - ★ Enter name of pre-existing dataset into [[Copy geometry of this dataset](#)] field
    - ↳ If in another directory, you must include that in the filename
  - ★ When you press 'Enter' or move the cursor from the text-entry field, **to3d** tries to read geometry parent dataset header
  - ★ If geometry parent has same spatial dimensions as current dataset, all geometry fields will be filled out
    - ↳ Does not affect the time fields, which must still be set using **-time:zt** or **-time:tz** on the command line
  - ★ Geometry parent very useful when constructing multiple EPI datasets from a single scanning session
- Using **to3d** in command line mode
  - ★ You can specify all needed inputs to to3d by using command line options
    - ↳ For a full list of options, type [to3d -help](#)
  - ★ If enough information is present on command line to define a dataset, then the GUI will not be opened, and the dataset will be written to disk
    - ↳ If the command line is incomplete, then the GUI will be opened



- ★ For the SPGR dataset example ('naked' image files):

```
to3d -xFOV 120A-P -yFOV 120S-I -zSLAB 70.0L-77.6R \  
-prefix anatNaked -2swap -spgr N.*
```

- ➔ -xFOV 120A-P says that the x axis of the images runs from 120 mm Anterior to 120 mm Posterior
- ➔ -yFOV 120S-I says that the y axis of the images runs from 120 mm Superior to 120 mm Inferior
- ➔ -zSLAB 70.0L-77.6R says that the z axis of the slices runs from 70 mm Left to 77.6 mm Right
  - ⇨ FOV refers to the coordinates of the outer edge of the first voxel to the outer edge of the last voxel along the relevant axis (x and y, in most cases)
  - ⇨ SLAB refers to coordinates of the center of the outermost voxels (z=slice direction, in most cases)
- ➔ -prefix anatNaked gives the prefix for output dataset filenames (in this case, **anatNaked+orig.HEAD** and **anatNaked+orig.BRIK**)
- ➔ -2swap means to byte-swap the images while reading them
- ➔ -spgr means to label this data as being of SPGR (SPoiled GRass) type
- ➔ N.\* means to read the images from the files whose names start with string "N." and end with anything ("\*" is a wildcard)

- ★ For the EPI dataset example (if image files were ‘naked’):

```
to3d -xFOV 120S-I -yFOV 120P-A -zSLAB 69.0R-61.0L -2swap \  
-time:zt 27 110 2500 alt+z -prefix epiRun1 -epan I.*
```

(this is all on one command line)

- ↳ Options (with their arguments) can appear in any order
- ↳ Input image filenames always appear last (i.e., I.\* )

- Conclusion

- ★ With practice, command line usage for `to3d` becomes more useful than the GUI
  - ↳ Usually need to create many datasets at once
  - ↳ Can put commands in a script file and execute them
  - ↳ Then edit the file to change a few things, and run it again
- ★ Just create the file with your favorite UNIX text editor (emacs, nedit, vi), typing each command on a separate line
  - ↳ Long commands can be split across multiple lines by ending all but the last line with the “\” character
  - ↳ There must not be a blank after the “\”!!!
- ★ You can execute a script file by typing a command like `tcsh <filename>`, which just means to read commands from “filename”
- ★ As time goes on, you build up a set of scripts that automate various tasks for you, and ensure you do things the same way each time