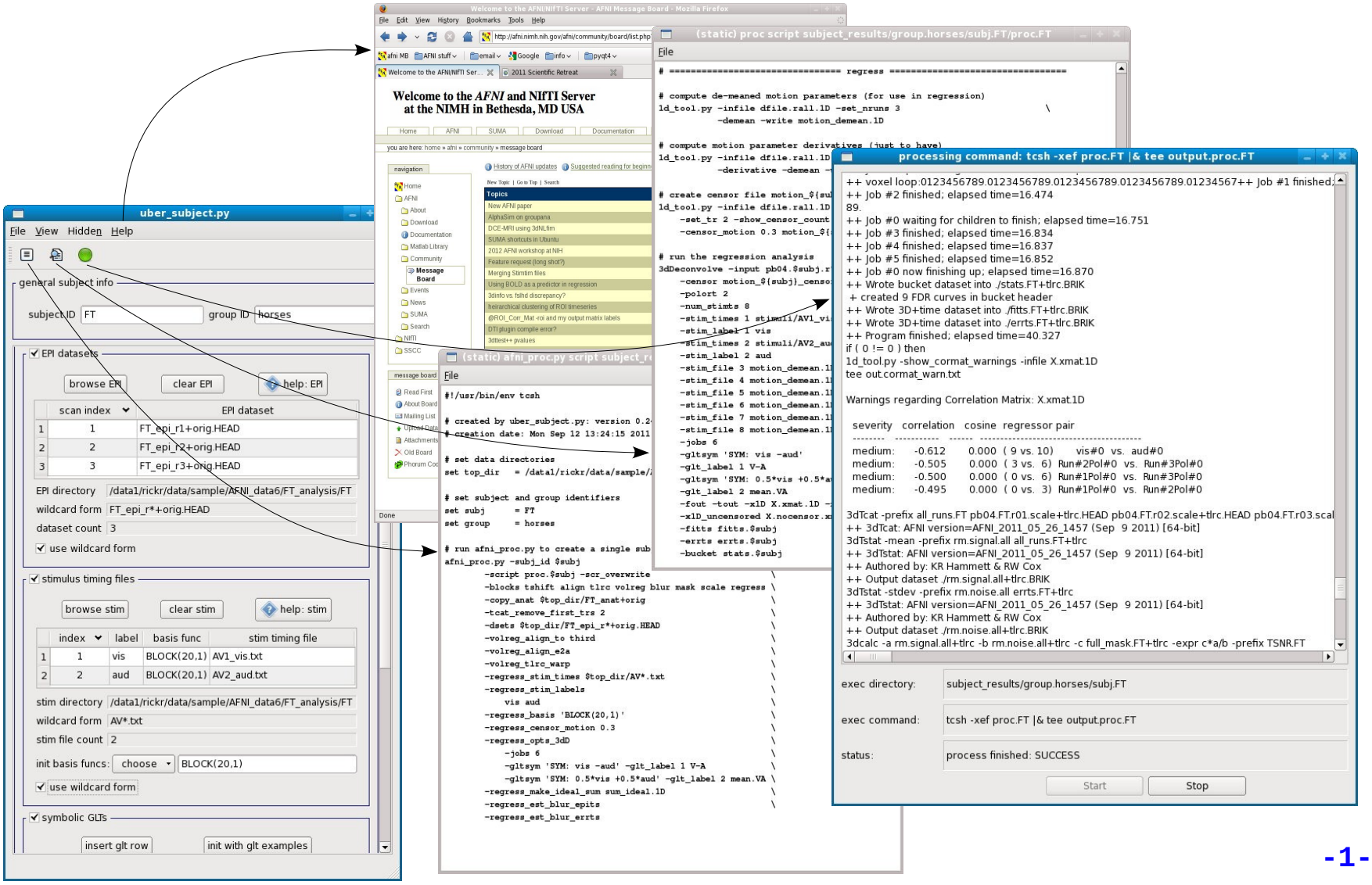
AFNI Start to Finish:
How to Analyze Data with AFNI



Goal: run group analysis on single subject response magnitudes

- how do we get there?
  - create beta (response magnitude) maps for each subject
    - should be aligned, probably to a well known template
  - run group analysis program (e.g. **`3dttest++`**, **`3dMEMA`**, **`3dANOVA`***)
    - can use **`uber_ttest.py`** to run single group tests

- how do we create aligned beta maps?
  - write single subject processing script: pre-processing through regression
    - inputs: anat, EPI, stimulus timing
    - controls: processing decisions like blur size and alignment template
    - outputs: beta weights (and t-stats, contrasts, blur estimates, etc.)

- how do we write single subject processing scripts?
  - **`afni_proc.py`** can be used to generate processing scripts
  - **`uber_subject.py`** can be used to generate **`afni_proc.py`** command
    - can also run the **`afni_proc.py`** command (generates proc script)
    - can also run the proc script (i.e. actually analyze the data)

General suggestions

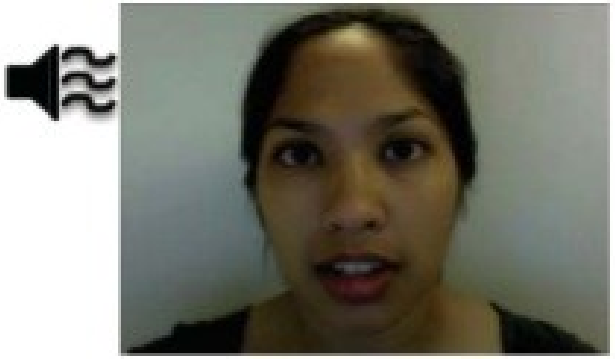
- picture this experiment as your own (i.e. feel responsibility)
  - decisions on processing were made by you (and your colleagues)
    - hopefully before acquiring any data
  - there is no single "correct" way to analyze data, just reasonable ways
- focus on understanding the processing steps
  - in light of your having chosen which steps to perform
- practice the good habit of reviewing results
  - do the initial images look good?
  - review each processing step along with data
  - are the EPI and anat well aligned by the end?
  - do the statistical results look reasonable?
- create scripts for any processing step
  - they are a record of how data was processed
  - easy to apply to any new subjects
  - easy to repeat
    - expect to re-analyze everything (mistake, new decision, etc.)
    - keep original data and all processing scripts

Review of stimulus conditions

- Speech Perception Task: Subjects were presented with audiovisual speech that was presented in a predominantly auditory or predominantly visual modality.
- A digital video system was used to capture auditory and visual speech from a female speaker.
- There were 2 types of stimulus conditions:

(1) **Auditory-Reliable**

Example: Subjects can clearly *hear* the word "cat," but the video of a woman mouthing the word is degraded.

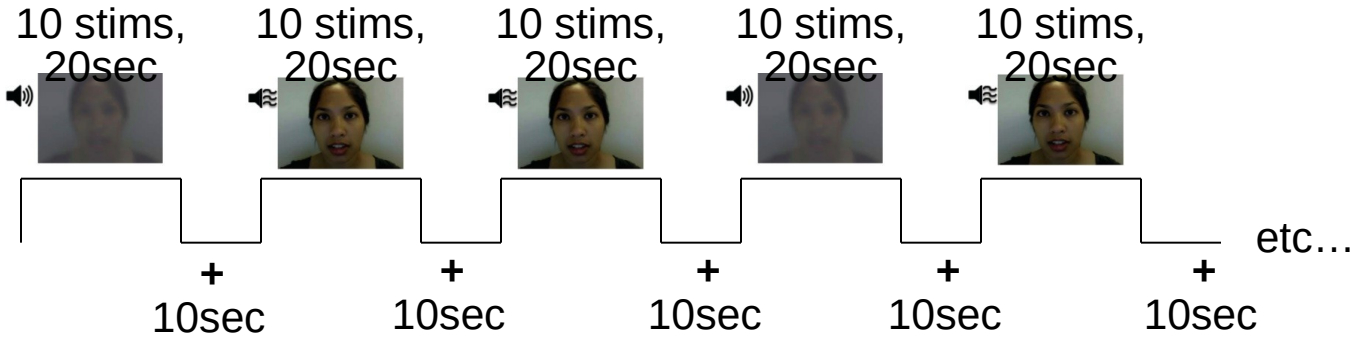


(2) **Visual-Reliable**

Example: Subjects can clearly *see* the video of a woman mouthing the word "cat," but the audio of the word is degraded.

❖ Experiment Design:

- There were 3 runs in a scanning session.
- Each run consisted of 10 blocked trials:
  - 5 blocks contained Auditory-Reliable (*Arel*) stimuli, and
  - 5 blocks contained Visual-Reliable (*Vrel*) stimuli.
- Each block contained 10 trials of *Arel* stimuli OR 10 trials of *Vrel* stimuli.
  - Each block lasted for 20 seconds (1 second for stimulus presentation, followed by a 1-second inter-stimulus interval).
- Each baseline block consisted of a 10-second fixation point.



❖ Data Collected:

- 2 Anatomical datasets for each subject, collected at 3 tesla.
  - 175 sagittal slices
  - voxel dimensions = 0.938 x 0.938 x 1.0 mm

- 3 Time Series (EPI) datasets for each subject.
  - 33 axial slices x 152 volumes = 5016 slices per run
  - TR = 2 sec; voxel dimensions = 2.75 x 2.75 x 3.0 mm

- Sample size, n = 10 (all right-handed subjects)

afni_proc.py
uber_subject.py

- What is **afni_proc.py**?
  - a program used to generate processing scripts for single subject analysis
    - command-line program
  - generated scripts are in **tcsh** syntax
  - scripts are written to be easily read and modified

- What is **uber_subject.py**?
  - for running single subject analysis
  - a graphical user interface to **afni_proc.py**
  - quickly create processing scripts
  - can analyze all subjects from GUI
  - good for learning
    - FMRI processing, shell scripting, AFNI commands
    - can compare against manually generated scripts
      - for sanity, bug detection, quick evaluation

Overview of remaining steps

- **`cd AFNI_data6/FT_analysis`**
  - review directory contents and note subject data under directory **FT**
- **from the home directory**, run **`uber_subject.py`** and analyze subject **FT**
  - set subject ID, group ID
  - specify inputs: anat, EPI, stimulus timing files (all under **`FT_analysis/FT`**)
  - controls: BLOCK(20,1), init GLTs, remove first 2 TRs
  - peruse other options, e.g. multiple CPUs for 3dDeconvolve?
  - create afni_proc.py command
  - execute afni_proc.py command (to create proc script)
  - execute proc script (analyze subject data)
- briefly review processing script
- review proc script in modest detail, while viewing processed data
  - run **`afni`** from **`FT.results`** directory and follow script
  - run resulting **`@ss_review_driver`** script
- run group analysis (**`3dttest++, 3dMEMA or 3dANOVA2`**)
  - run **`uber_ttest.py`** on data under **`AFNI_data6/group_results`**
  - or run existing **`s1.3dANOVA2`** script

Note what is under **AFNI_data6/FT_analysis**

FT	- subject data directory
s01.ap.simple	- basic **afni_proc.py** script
s02.ap.align	- more advanced script
s09.cleanup	- remove analysis results
s11.proc.FT	- result of **s01.ap.simple**
s12.proc.FT.align	- result of **s01.ap.align**

under **FT**

AV1_vis.txt	- visual reliable timing
AV2_aud.txt	- auditory reliable timing
FT_anat+orig.BRIK/HEAD	- anatomical dataset
FT_epi_r1+orig.BRIK/HEAD	- EPI run 1
FT_epi_r2+orig.BRIK/HEAD	- EPI run 2
FT_epi_r3+orig.BRIK/HEAD	- EPI run 3

AV1_vis.txt:

```
60 90 120 180 240
120 150 180 210 270
0 60 120 150 240
```

Single Subject Analysis: FT

- from home directory (use the **cd** command), run **uber_subject.py &**
  - subject ID ***FT***, group ID ***horses***
  - browse anat: choose ***AFNI_data6/FT_anaysis/FT/FT_anat+orig.HEAD***
  - browse EPI: choose ***FT_epi_r1+orig.HEAD*** (and ***epi_r2*** and ***epi_r3***)
    - select all 3 data files with ctrl or shift keys, then hit ***Open***
  - browse stim: choose ***AV1_vis.txt*** and ***AV2_aud.txt***
  - stimuli: set both basis functions to ***BLOCK(20,1)***
    - init basis funcs to ***BLOCK(5,1)***, change 5 to 20 and hit Enter
  - symbolic GLTs: click to view option frame and ***init with glt examples***
  - expected options: removed first ***2*** TRs per run
  - (optional) extra regress options: ***2*** CPUs (or 12 or whatever is appropriate)
  - ***generate afni_proc.py command*** (left action button)
  - ***generate proc script*** (middle action button)
  - ***process this subject*** (right action button)
    - scripts/results are under **`subject_results/group.horses/subj.FT`**
  - review the **`proc.FT`** script while looking at the results under **`FT.results`**
    - **`cd subject_results/group.horses/subj.FT`**
    - **`cd FT.results ; afni &`**
  - after script and data review, run **`./@ss_review_driver`**
    - **minimal** data review (should run on each subject)

Group Analysis: paired t-test (Vrel-Arel)

- from home directory, run **`uber_ttest.py &`**
  - using data from under **`AFNI_data6/group_results`**
  - program ***3dttest++***, script ***script.V-A***, prefix ***ttest.V-A***, mask dset: ***skip***
  - choose ***paired test***
  - fill "datasets A" table with datasets for **`Vrel`** betas
    - get subj dsets
      - ✓ choose file : **`OLSQ.FP.betas+tlrc.HEAD`**
      - ✓ alter name into wildcard pattern: replace "***FP***" with "*******"
      - ✓ press Enter or ***apply pattern*** button (should have 10 datasets)
      - ✓ press ***OK*** (at bottom)
    - set name ***Vrel***, data index ***Vrel#0_Coef*** (or index 0), t-stat index: ***skip***
  - fill "datasets B" table with datasets for **Arel** betas
    - identical table of datasets, so choose ***copy other table***
      - ✓ set ***Arel***, data index ***Arel#0_Coef***, t-stat index: ***skip***
  - ***generate processing script***: press left action button (note hint)
  - ***execute processing script***: press green action button (note hint)
- script/output/results are under: **group_results/test.001.3dttest++**
- script is **script.V-A**, results are under **ttest.results**
- **cd group_results/test.001.3dttest++/ttest.results** (practice **<tab>** key)
- **afni &**

Additional comments

- inputs for subject **FT** are under **AFNI_data6/FT_analysis/FT**

- results from **uber_*.py** go where the program was run
  - **uber_subject.py**: **subject_results/group.GROUP/subj.SUBJECT**
  - **uber_ttest.py**: **group_results/test.INDEX.PROGRAM**
  - so in class, these directory trees should end up under the home directory

- it is not necessary to master all shell script details
  - but want to understand processing steps

- when analyzing data, run **@ss_review_driver** for every subject
  - script represents the **minimum** of what to look at for each subject
  - for the first few subjects analyzed, look at all results in detail
    - in more detail than the level of this class
    - before acquiring many subjects

AFNI Start to Finish
(the horror continues...)

- To continue reviewing the data on your own, please see the corresponding tutorial that continues under the data directory:
  - **`AFNI_data6/FT_analysis/tutorial`**
- Alternatively, this can be viewed from the AFNI web site:

  http://afni.nimh.nih.gov/pub/dist/edu/data/CD.expanded/AFNI_data6/FT_analysis/tutorial
- or from the Help menu of **`uber_subject.py`**
  - Help --> Browse --> web: tutorial-single subject analysis