3dDeconvolve

Advanced Features
Et cetera

Just in case you weren't confused enough already

Other Features of 3dDeconvolve - 1

- **`-input1D`** = used to process a single time series, rather than a dataset full of time series
  - e.g., test out a stimulus timing sequence on sample data
  - **`-nodata`** option can be used to check **X** matrix for collinearity
- **`-censor`** = used to turn off processing for some time points
  - for time points that are "bad" (e.g., too much movement; scanner problem)
  - **`-CENSORTR 2:37`** = newer way to specify omissions (e.g., run #2, index #37)
- **`-sresp`** = output standard deviation of HRF (β) estimates
  - can then plot error bands around HRF in AFNI graph viewer
- **`-errts`** = output residuals (difference between fitted model and data)
  - for statistical analysis of time series noise
- **`-TR_times dt`** = calculate **`-iresp`** and **`-sresp`** HRF results with time step **`dt`** (instead of input dataset TR)
  - Can be used to make HRF graphs look better
- **`-jobs`** ***N*** = run with independent threads — ***N*** of them
  - extra speed, if you have a dual-CPU system (or more)!

Other Features - 2

http://afni.nimh.nih.gov/pub/dist/doc/misc/Decon/DeconSummer2004.html
http://afni.nimh.nih.gov/pub/dist/doc/misc/Decon/DeconSpring2007.html

- Equation solver: Program computes **condition number** for **X** matrix (measures of how sensitive regression results are to changes in **X**)
  - If the condition number is "bad" (too big), then the program will not actually proceed to compute the results
  - You can use the **`-GOFORIT`** option on the command line to force the program to run despite **X** matrix warnings
    - But you should strive to understand why you are getting these warnings ***!!***
- Other matrix checks:
  - Duplicate stimulus filenames, duplicate regression matrix columns, all zero matrix columns

★Check the screen output for **`WARNING`**s and **`ERROR`**s★

- Such messages also saved into file **`3dDeconvolve.err`**

Other Features - 3

- ★ All-zero regressors *are* allowed (via **-allzero_OK** or **-GOFORIT**)
  - Will get zero weight in the solution
  - Example: task where subject makes a choice for each stimulus (e.g., male or female face?)
    - You want to analyze correct and incorrect trials as separate cases
    - What if some subject makes no mistakes? Hmmm…
      - Can keep the all-zero regressor (e.g., all **-stim_times** = *****)
      - Input files and output datasets for error-making and perfect-performing subjects will be organized the same way

- **3dDeconvolve_f** program can be used to compute linear regression results in single precision (7 decimal places) rather than double precision (16 places)
  - For better speed, but with lower numerical accuracy
  - Best to do at least one run ***both*** ways to check if results differ significantly (Equation solver *should* be safe, but …)

Other Features - 4

★ Default output format is 16-bit short integers, with a scaling factor for each sub-brick to convert it to floating point values

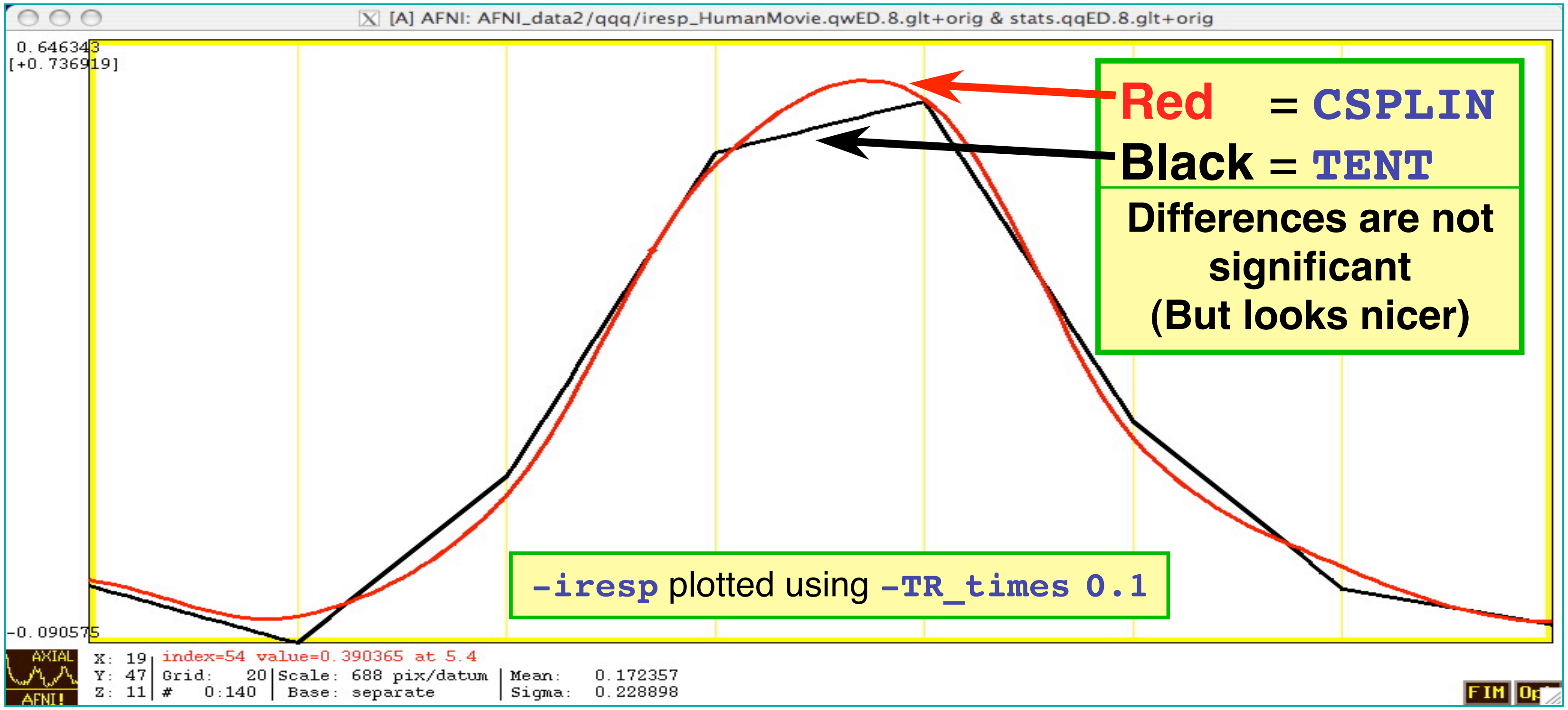
- `-float` option can be used to get 32-bit floating point format output — more precision, and more disk space

- `3dDeconvolve` recommends a `-polort` value, and prints that out as well as the value you chose (or defaulted to)
  - `-polort A` can be used to let the program set the detrending (AKA “high pass filtering”, since detrending removes low frequency content from data) level automatically

- `-stim_file` is used to input a column directly into **X** matrix
  - Motion parameters (as in previous examples)
  - If you create a stimulus+response model outside `3dDeconvolve` (e.g., using program `waver`)

Other Features - 5

- **`-stim_times`** has some other basis function options for the HRF model besides **`BLOCK`** and **`TENT`**
  - **`CSPLIN`** = cubic spline instead of **`TENT`** = linear spline
    - Same parameters: **`(start,stop,number of regressors)`**
    - Can be used as a "drop in" replacement for **`TENT`**



[A] AFNI: AFNI_data2/qqq/iresp_HumanMovie.qwED.8.glt+orig & stats.qqED.8.glt+orig

```
0.646343
[+0.736919]

-0.090575

X: 19| index=54 value=0.390365 at 5.4
Y: 47| Grid:    20|Scale: 688 pix/datum |Mean:    0.172357
Z: 11| #    0:140 | Base: separate     |Sigma:   0.228898
```

Red = **`CSPLIN`**
Black = **`TENT`**

Differences are not significant (But looks nicer)

`-iresp` plotted using **`-TR_times 0.1`**

Other Features - 6

- `-fitts` option is used to create a synthetic dataset
  - each voxel time series is full (signal+baseline) model as fitted to the data time series in the corresponding voxel location
- `3dSynthesize` program can be used to create synthetic datasets from *subsets* of the full model
  - Uses `-x1D` and `-cbucket` outputs from `3dDeconvolve`
    - `-cbucket` stores β coefficients for each **X** matrix column into dataset
    - `-x1D` stores the matrix columns (and `-stim_label`s, etc.)
  - Potential uses:
    - Baseline only dataset
      - `3dSynthesize -cbucket fred+orig -matrix fred.xmat.1D -select baseline -prefix fred_base`
      - Could subtract this dataset from original data (via `3dcalc`) to get signal+noise dataset that has no baseline component left
    - Just one stimulus class model (+ baseline) dataset
      - `3dSynthesize -cbucket fred+orig -matrix fred.xmat.1D -select baseline Faces -prefix fred_Faces`

Other Recent Small Changes

- Defaults are changed:
    - **`-nobout`** & **`-full_first`** & **`-bucket`** & **`-x1D`** are always implied
    - Names of statistics sub-bricks are slightly altered (to be more consistent)

- Checks if **`-stim_times`** inputs are out of range (AKA: the PSFB syndrome)
    - Prints **`WARNING`** message, but continues analysis

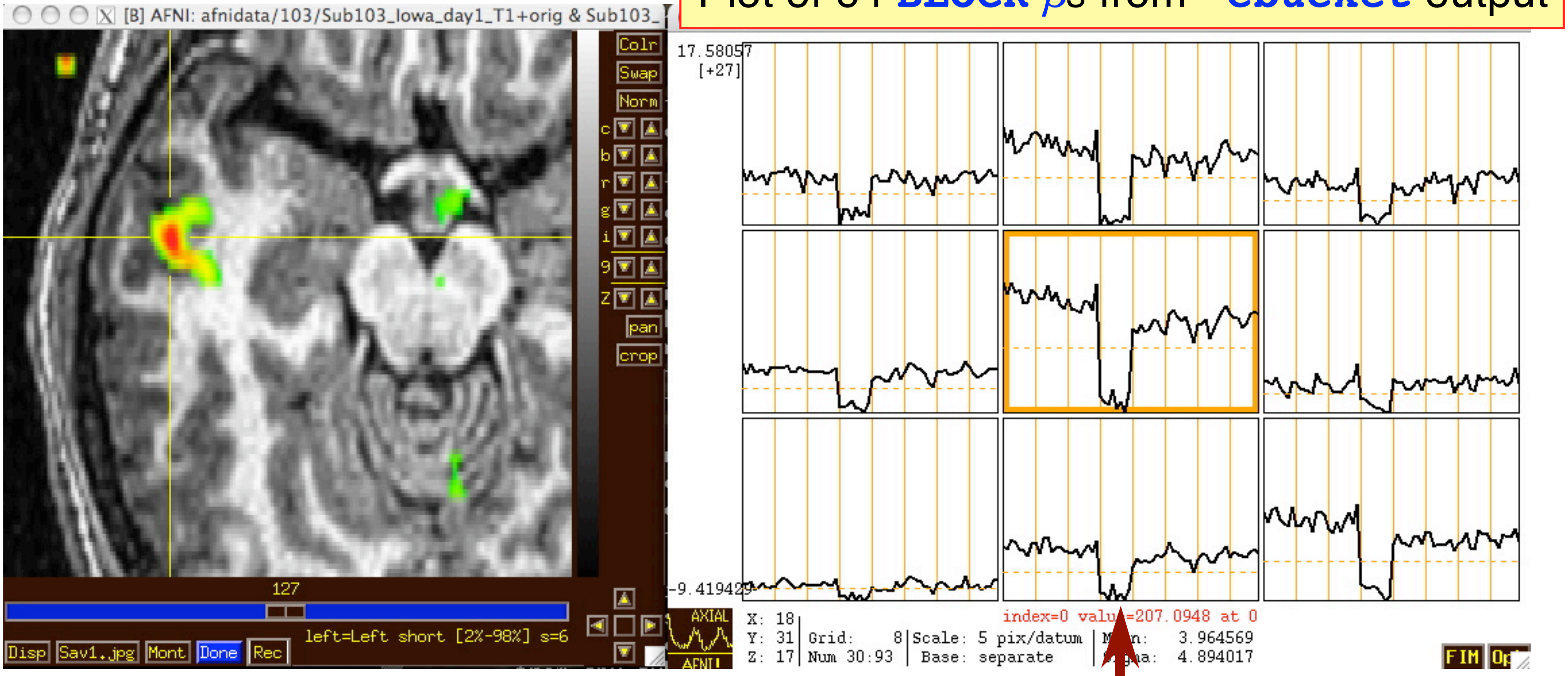
- When using **`-nodata`** with **`-stim_times`**, it is important to give the number of time points and the TR, as in **`-nodata 250 2.3`**
    - With **`-input1D`**, use **`-TR_1D 2.3`** to specify TR

IM Regression - 1

- **IM** = **I**ndividual **M**odulation
  - Compute *separate* amplitude of response for each stimulus
    - Instead of computing average amplitude of responses to multiple stimuli in the same class
  - Response amplitudes (βs) for each individual block/event will be highly noisy
    - Can't use individual activation map for much
    - Must pool the computed βs in some further statistical analysis (t-test via **`3dttest`**? inter-voxel correlations in the βs? correlate βs with something else?)
  - Usage: **`-stim_times_IM k tname model`**
    - Like **`-stim_times`**, but creates a separate regression matrix column for each time given

IM Regression - 2

- Only application of IM thus far has been in checking some data we received from another institution
- Experiment: 64 blocks of sensorimotor task (8 runs each with 8 blocks)



```
[B] AFNI: afnidata/103/Sub103_Iowa_day1_T1+orig & Sub103_
```

```
17.58057
[+27]
-9.419429
```

```
                         index=0 valu =207.0948 at 0
X: 18 |                 |                          |M  n:    3.964569
Y: 31 |Grid:        8|Scale: 5 pix/datum |     a:   4.894017
Z: 17 |Num 30:93      | Base: separate
```

Plot of 64 **BLOCK** βs from **-cbucket** output

N.B.: sign reversal in run #4 = stimulus timing error!

IM Regression - 3

- IM works naturally with blocks, which only have 1 amplitude parameter per stimulus
- With event-related experiment and *deconvolution*, have multiple amplitude parameters per stimulus
  - Difficulty: each event in same class won't get the same shaped HRF this way
  - Desideratum: allow response shape to vary (that's deconvolution), but only allow amplitude to vary between responses in the same stimulus class
  - Problem: get unknowns that multiply each other (shape parameters × amplitude parameters) — and we step outside the realm of ***linear*** analysis
  - Possible solution: **semi-linear** regression (nonlinear in global shape parameters, linear in local amplitude params)

AM Regression - 1

- **AM** = **A**mplitude **M**odulated (or Modulation)
  - Have some extra data measured about each response to a stimulus, and *maybe* the BOLD response amplitude is modulated by this
  - Reaction time; Galvanic skin response; Pain level perception; Emotional valence (happy or sad or angry face?)
- Want to see if some brain activations vary proportionally to this **ABI** (**A**uxiliary **B**ehaviorial **I**nformation)
- Discrete levels (2 or maybe 3) of ABI:
  - Separate the stimuli into sub-classes that are determined by the ABI (“on” and “off”, maybe?)
  - Use a GLT to test if there is a difference between the FMRI responses in the sub-classes

```
3dDeconvolve ...                                                      \
 -stim_times 1 regressor_on.1D  'BLOCK(2,1)' -stim_label 1 'On'   \
 -stim_times 2 regressor_off.1D 'BLOCK(2,1)' -stim_label 2 'Off' \
 -gltsym 'SYM: +On | +Off' -glt_label 1 'On+Off'                  \
 -gltsym 'SYM: +On -Off'   -glt_label 2 'On-Off' ...
```

  - “`On+Off`” tests for any activation in *either* the “on” or “off” conditions
  - “`On-Off`” tests for differences in activation *between* “on” and “off” conditions
  - Can use **`3dcalc`** to threshold on ***both*** statistics at once to find a **conjunction**

AM Regression - 2

- Continuous (or several finely graded) ABI levels
  - Want to find active voxels whose activation level also depends on ABI
  - `3dDeconvolve` is a linear program, so must make the assumption that the change in FMRI signal as ABI changes is linearly proportional to the changes in the ABI values
- Need to make 2 separate regressors
  - One to find the mean FMRI response (the usual `-stim_times` analysis)
  - One to find the variations in the FMRI response as the ABI data varies
- The second regressor should have the form

$$r_{\mathrm{AM2}}(t) = \sum_{k=1}^{K} h(t-\tau_k)\cdot(a_k - \bar{a})$$

  - Where a_k=value of kth ABI value, and $\bar{a}$ is the average ABI value
- Response (β) for first regressor is standard activation map
- Statistics and β for second regressor make activation map of places whose BOLD response changes with changes in ABI
  - Using 2 regressors allows separation of voxels that are active but are *not* detectably modulated by the ABI from voxels that *are* ABI-sensitive

AM Regression - 3

- New feature of **3dDeconvolve**: **-stim_times_AM2**
- Use is very similar to standard **-stim_times**
    - **-stim_times_AM2 1 times_ABI.1D 'BLOCK(2,1)'**
    - The **times_ABI.1D** file has time entries that are “married” to ABI values:

```
10*5 23*4 27*2 39*5
17*2 32*5
*
16*2 24*3 37*5 41*4
```

    - Such files can be created from 2 standard ASCII .1D files using the new **1dMarry** program
        - The **-divorce** option can be used to split them up
- **3dDeconvolve** automatically creates the two regressors (unmodulated and amplitude modulated)
    - Use **-fout** option to get statistics for activation of pair of regressors (i.e., testing null hypothesis that *both* β weights are zero: that there is no ABI-independent *or* ABI-proportional signal change)
    - Use **-tout** option to test each β weight separately
    - Can **1dplot** **X** matrix columns to see each regressor

AM Regression - 4

- The **AM** feature is new, and so needs more practical user experiences before it can be considered “standard practice”
  - In particular: don’t know how much data or how many events are needed to get good ABI-dependent statistics
- If you want, **`-stim_times_AM1`** is also available
  - It only builds the regressor proportional to ABI data directly, with no mean removed: $r_{\text{AM1}}(t) = \sum_{k=1}^{K} h(t - \tau_k) \cdot a_k$
  - Can’t imagine what value this option has, but you never know … (if you can think of a good use, let me know)
- Future directions:
  - Allow more than one amplitude to be married to each stimulus time (insert obligatory polygamy/polyandry joke here) – this is done now
    - How many ABI types at once is too many? I don’t know.
  - How to deal with unknown nonlinearities in the BOLD response to ABI values? I don’t know. (Regress each event separately, then compute MI?)
  - Deconvolution with amplitude modulation? Requires more thought.

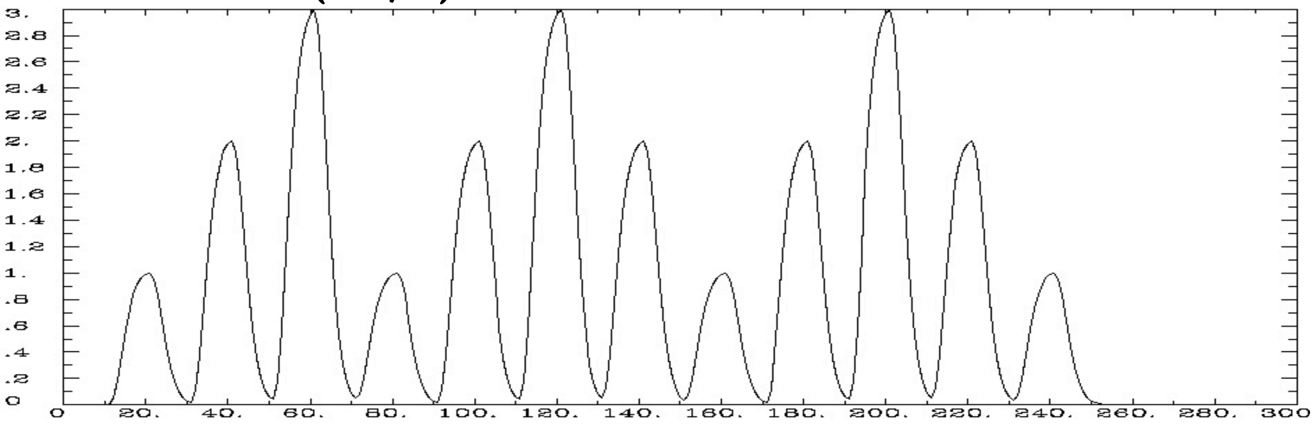
AM Regression - 5

Timing: `AM.1D` = `10*1 30*2 50*3 70*1 90*2 110*3 130*2 150*1 170*2 190*3 210*2 230*1`

-
  ```
  3dDeconvolve -nodata 300 1.0 -num_stimts 1 \
    -stim_times_AM1 1 AM.1D 'BLOCK(10,1)' -x1D AM1.x1D
  ```

-
  ```
  1dplot AM1.x1D'[2]'
  ```

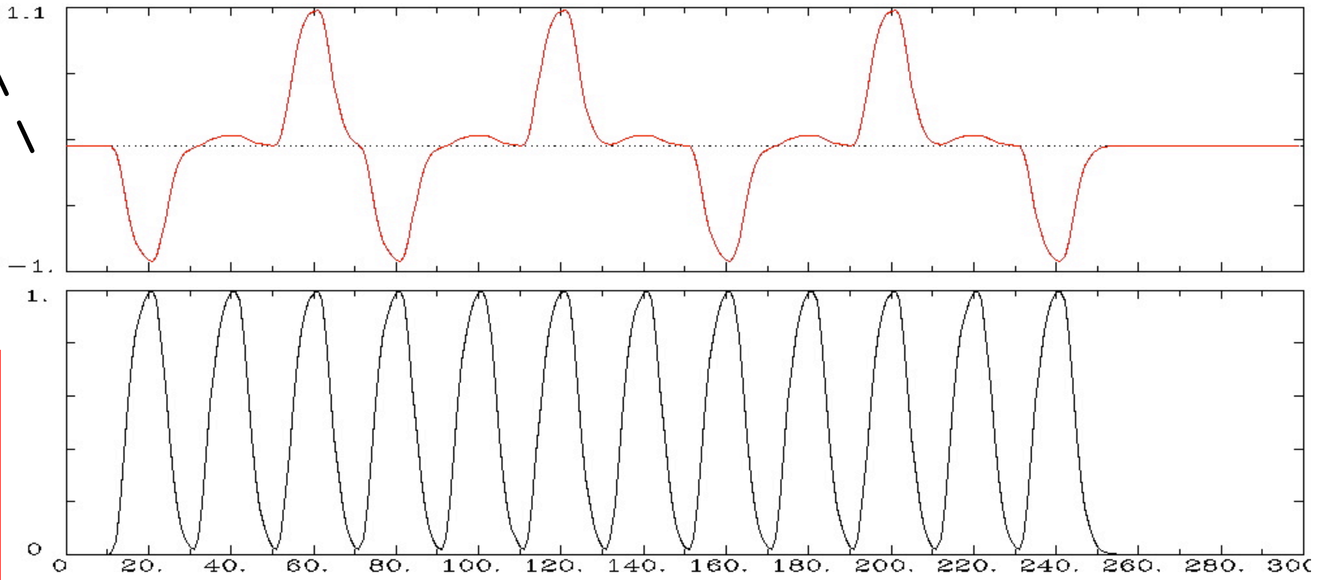
AM1 model of signal (modulation = ABI)



-
  ```
  3dDeconvolve -nodata 300 1.0 \
  -num_stimts 1      \
  -stim_times_AM2 1   \
  AM.1D 'BLOCK(10,1)'  \
  -x1D AM2.x1D
  ```

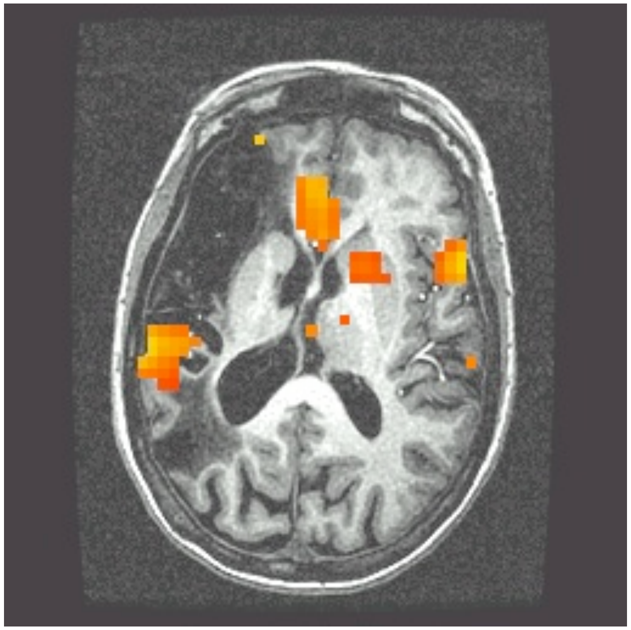
-
  ```
  1dplot -sepscl \
    AM2.x1D'[2,3]'
  ```

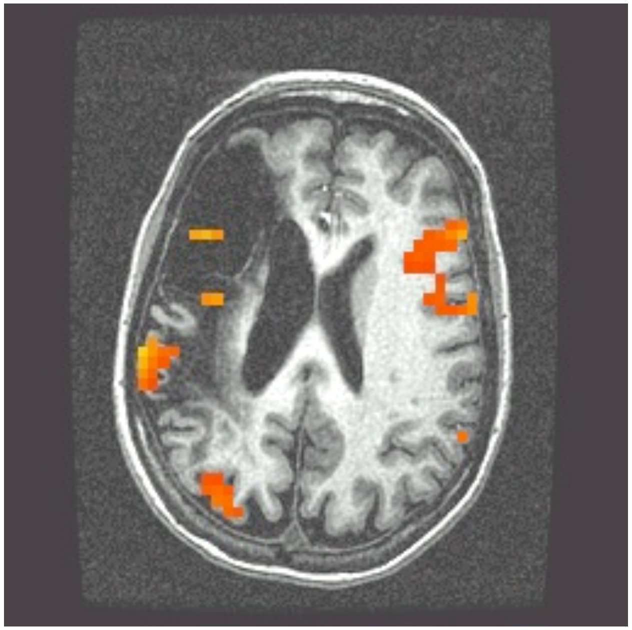
AM2 model of signal: is 2D sub-space spanned by these 2 time series



AM Regression - 6

- First actual user: Whitney Postman (formerly NIDCD; PI=Al Braun)
- Picture naming task in aphasic stroke patient
- ABI data = number of alternative names for each image (e.g., “balcony” & “porch” & “veranda”, vs. “strawberry”), from 1 to 18
  - 8 imaging runs, 144 stimulus events
- 2 slices showing activation map for BOLD responses proportional to ABI (β_{AM2})
  - What does this mean? Don’t ask me!





AM Regression - 7

- Alternative: use **IM** to get individual βs for each block/event and then do external regression statistics on those values
- Could do nonlinear fitting (to these βs) via **`3dNLfim`**, or inter-class contrasts via **`3dttest`**, **`3dLME`**, **`3dANOVA`**, or intra-class correlations via **`3dICC`**, etc.
- What is better: **AM** or **IM**+*something more* ?
  - We don't know – experience with these options is limited thus far – you can always try both!
  - If **AM** doesn't fit your models/ideas, then **IM**+ is clearly the way to go
  - Probably need to consult with SSCC to get some hints/advice

Other Advanced Topics in Regression

- Can have activations with multiple phases that are not always in the same time relationship to each other; e.g.:
  - a) subject gets cue #1
  - b) variable waiting time (“hold”)
  - c) subject gets cue #2, emits response
    - which depends on both cue #1 and #2

  timing of events is known

  - Cannot treat this as one event with one HRF, since the different waiting times will result in different overlaps in separate responses from cue #1 and cue #2
  - Solution is multiple HRFs: separate HRF (fixed shape or deconvolution) for cue #1 times and for cue #2 times
    - Must have significant variability in inter-cue waiting times, or will get a nearly-collinear model
      - impossible to tell tail end of HRF #1 from the start of HRF #2, if always locked together in same temporal relationship
    - How much variability is “significant”? Good question.

Slightly More Complicated Case

- Solving a visually presented puzzle:
  a) subject sees puzzle
  b) subject cogitates a while
  c) subject responds with solution

timing of events is measured

- The problem is that we expect some voxels to be significant in phase (b) as well as phases (a) and/or (c)
- Variable length of phase (b) means that shape for its response varies between trials
  - Which is contrary to the whole idea of averaging trials together to get decent statistics (which is basically what linear regression for the β weights does, in an elaborate sort of way)
- Could assume response ***amplitude*** in phase (b) is constant across trials, and response ***duration*** varies directly with time between phases (a) and (c)
  - Need three HRFs
  - Can't generate (b) HRF in **`3dDeconvolve`**

Yes we can*!*
`-dmBLOCK` model

Noise Issues

- “Noise” in FMRI is caused by several factors, not completely characterized
  - MR thermal noise (well understood, unremovable)
  - Cardiac and respiratory cycles (partly understood)
    - In principle, could measure these sources of noise separately and then try to regress them out
      - RETROICOR program underway (Rasmus Birn of FIM/NIMH)
  - Scanner fluctuations (e.g., thermal drift of hardware)
  - Small subject head movements (10-100 mm)
  - Very low frequency fluctuations (periods longer than 100 s)
- Data analysis should try to remove what can be removed and should allow for the statistical effects of what can’t be removed
  - “Serial correlation” in the noise time series affects the *t*- and *F*-statistics calculated by **`3dDeconvolve`**
  - Next slides: new AFNI program for dealing with this issue

Allowing for Serial Correlation

- *t*- and *F*-statistics denominators: estimates of noise variance
  - White noise estimate of variance:
    - N = number of time points
    - m = number of fit parameters
    - $N-m$ = degrees of freedom = how many equal-variance independent random values are left after time series is fit with m regressors

$$\hat{\sigma}^2 = \frac{1}{N-m}\sum_{i=0}^{N-1}[\text{data}_i - \text{fit}_i]^2$$

- **Problem**: if noise values at successive time points are correlated, this estimate of variance is biased to be too small, since there aren't really $N-m$ independent random values left
  - Denominator too small implies *t*- and *F*-statistics are too large!
  - And number of degrees of freedom is also too large.
  - So significance (*p*-value) of activations in individuals is overstated.
- **Solution #1**: estimate correlation structure of noise and then adjust statistics (downwards) appropriately
- **Solution #2**: estimate correlation structure of noise ***and*** also estimate β fit parameters using more efficient "generalized least squares", using this correlation, all at once (REML method)

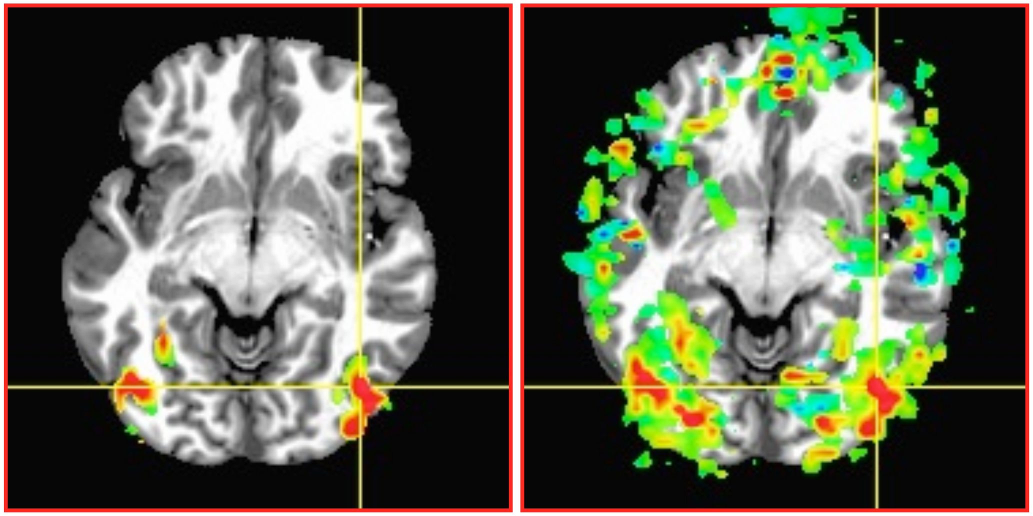
New Program: `3dREMLfit`

- Implements Solution #2
  - REML is a method for simultaneously estimating variance + correlation parameters ***and*** estimating regression fit parameters (βs)
  - Correlation structure of noise is ARMA(1,1)
    - 2 parameters ***a*** (AR) and ***b*** (MA) in each voxel
      - ***a*** describes how fast the noise de-correlates over time
      - ***b*** describes the short-range correlation in time (1 lag)
    - Unlike SPM and FSL, *each voxel* gets a separate estimate of its own correlation parameters
- Inputs to `3dREMLfit`
  - run `3dDeconvolve` first to setup `.xmat.1D` matrix file and GLTs (don't have to let `3dDeconvolve` finish analysis: `-x1D_stop`)
    - `3dDeconvolve` also outputs a command line to run `3dREMLfit`
  - then, input matrix file and 3D+time dataset to `3dREMLfit`
- Output datasets are similar to those in `3dDeconvolve`

Sample Outputs

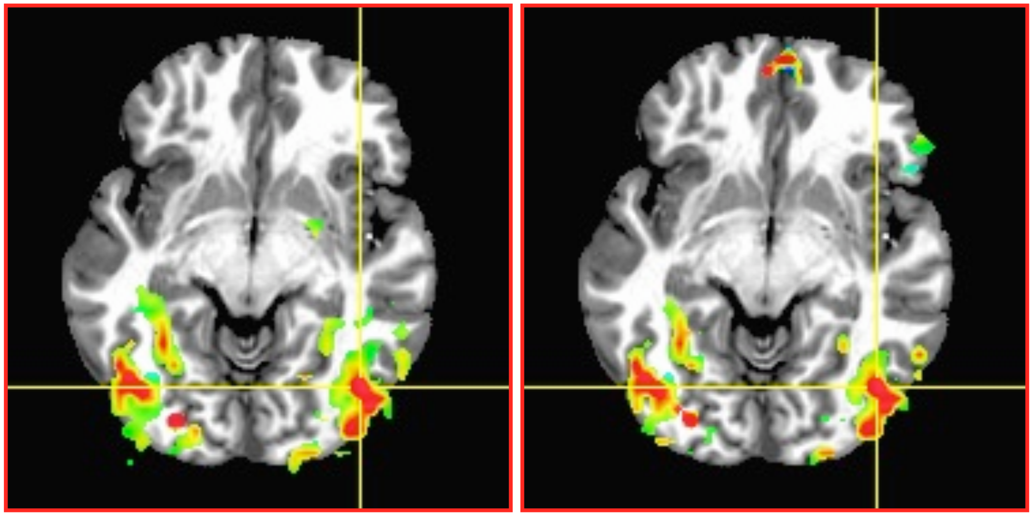
- Compare with **AFNI_data3/afni/rall_regress results**
-
  ```
  3dREMLfit -matrix rall_xmat.x1D  -input rall_vr+orig  -fout  -tout  \
            -Rvar rall_varR  -Rbuck rall_funcR  -Rfitts rall_fittsR  \
            -Obuck rall_funcO  -Ofitts rall_fittsO
  ```

REML
F=3.15
p=0.001



OLSQ
F=3.15
p=0.001

REML
F=1.825
p=0.061
▪ *F* = No activity outside brain!



OLSQ
F=5.358
p=5e-7
▪ *F* = No activity outside brain!

O h M y G O D ***!?!***

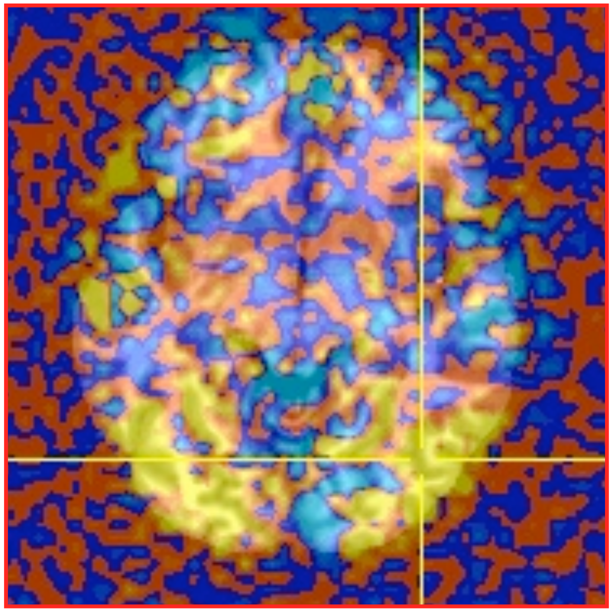
It's Not So Bad: *β* !

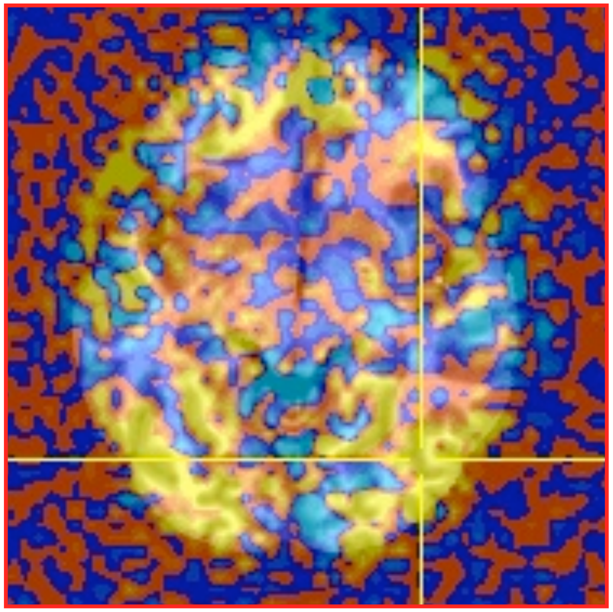
- For individual activation maps, **`3dREMLfit`**-ized *t*- and *F*-statistics are significantly different, and more accurate
- But … There are at present very few applications for such individual FMRI activation maps
  - pre-surgical planning; some longitudinal study?
- For standard group analysis, inputs are only *β* fit parameters
  - Which don't change so much between REML and OLSQ

Color Overlay = *β* weight from analysis on previous slide, no threshold

REML

CPU 500 s





OLSQ

CPU 156 s

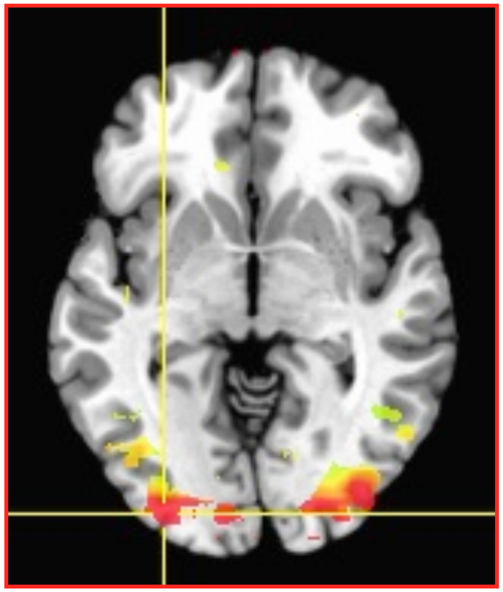
It’s Not So Bad At All: Group Analysis!

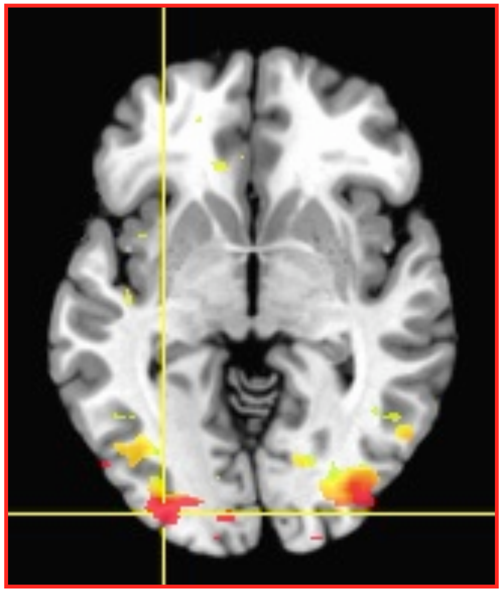
- Group analysis activation maps (`3dANOVA3`) from 16 subjects

REML

OLSQ

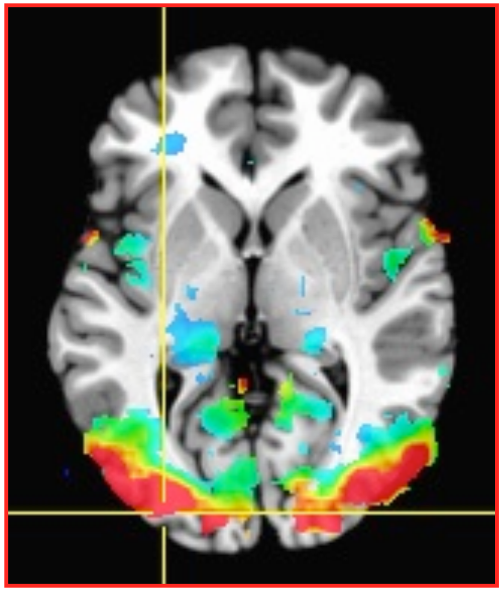
***F*-test for Affect condition**

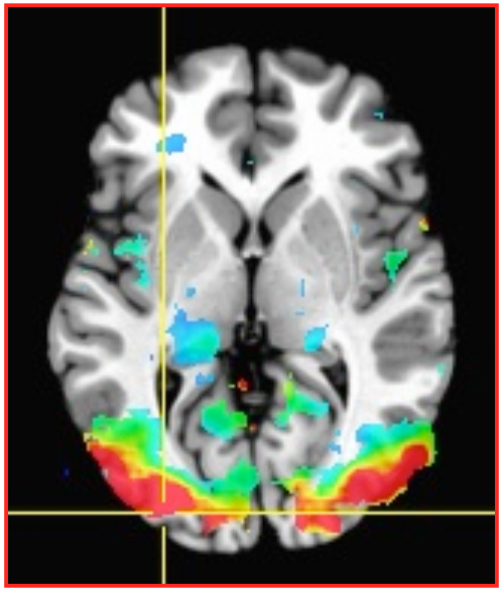




***F*-test for Affect condition**

***F*-test for Category condition**





***F*-test for Category condition**

Nonlinear Regression

- Linear models aren't the only possibility
  - e.g., could try to fit HRF of the form $h(t) = a \cdot t^b \cdot e^{-t/c}$
  - Unknowns b and c appear nonlinearly in this formula
- Program `3dNLfim` can do nonlinear regression (including nonlinear deconvolution)
  - User must provide a C function that computes the model time series, given a set of parameters (e.g., a, b, c)
    - We could help you develop this C model function
    - Several sample model functions in the AFNI source code distribution
  - Program then drives this C function repeatedly, searching for the set of parameters that best fit each voxel
  - Has been used to fit pharmacological wash-in/wash-out models (difference of two exponentials) to FMRI data acquired during pharmacological challenges
    - e.g., injection of nicotine, cocaine, ethanol, etc.
    - these are difficult experiments to do ***and*** to analyze

Deconvolution: The Other Direction

- Signal model: $Z(t) = H(t)*A(t)$ + baseline model + noise
- $H(t)$ = HRF = response magnitude t seconds after activation
  - $H(t)$ is **causal** = zero for $t < 0$
  - "$*$" is symbol for convolution, not multiplication!
- **`3dDeconvolve`**: find out something about $H(t)$ given $A(t)$
- Sometimes (PPI) want to solve the problem in the other direction: assume a model for $H(t)$ and find time series $A(t)$
  - Convolution is commutative: $H(t)*A(t) = A(t)*H(t)$
  - So the other direction looks to be the same problem
  - But isn't, since $H(t)$ is causal but $A(t)$ is not
    - Also, $H(t)*A(t)$ smooths out rough spots in $A(t)$, so undoing this deconvolution adds roughness — including noise, which is already rough — which must be controlled or output $A(t)$ will be junk
- Program **`3dTfitter`** solves this type of problem
  - Also can allow for *per voxel* baseline model components

Spatial Models of Activation

- Smooth data in space before analysis

- Average data across anatomically-selected regions of interest ROI (before or after analysis)
  - Labor intensive (*i.e.*, hire more students)

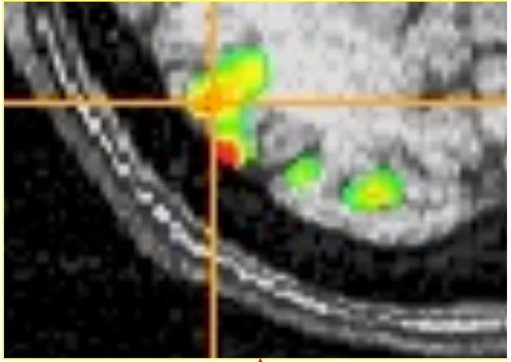
- Reject isolated small clusters of above-threshold voxels after analysis

Spatial Smoothing of Data

- Reduces number of comparisons
- Reduces noise (by averaging)
- Reduces spatial resolution
  - Blur it enough: Can make FMRI results look like low resolution (1990s) PET data
- Smart smoothing: average ***only*** over nearby brain or gray matter voxels
  - Uses resolution of FMRI cleverly
    - **`3dBlurToFWHM`** and **`3dBlurInMask`**
  - Or: average over selected ROIs
  - Or: cortical surface based smoothing

3dBlurToFWHM

- New program to smooth FMRI time series datasets to a specified smoothness (as estimated by FWHM of noise spatial correlation function)
  - Don't just add smoothness (à la **3dmerge**) but control it (locally and globally)
  - Goal: use datasets from diverse scanners

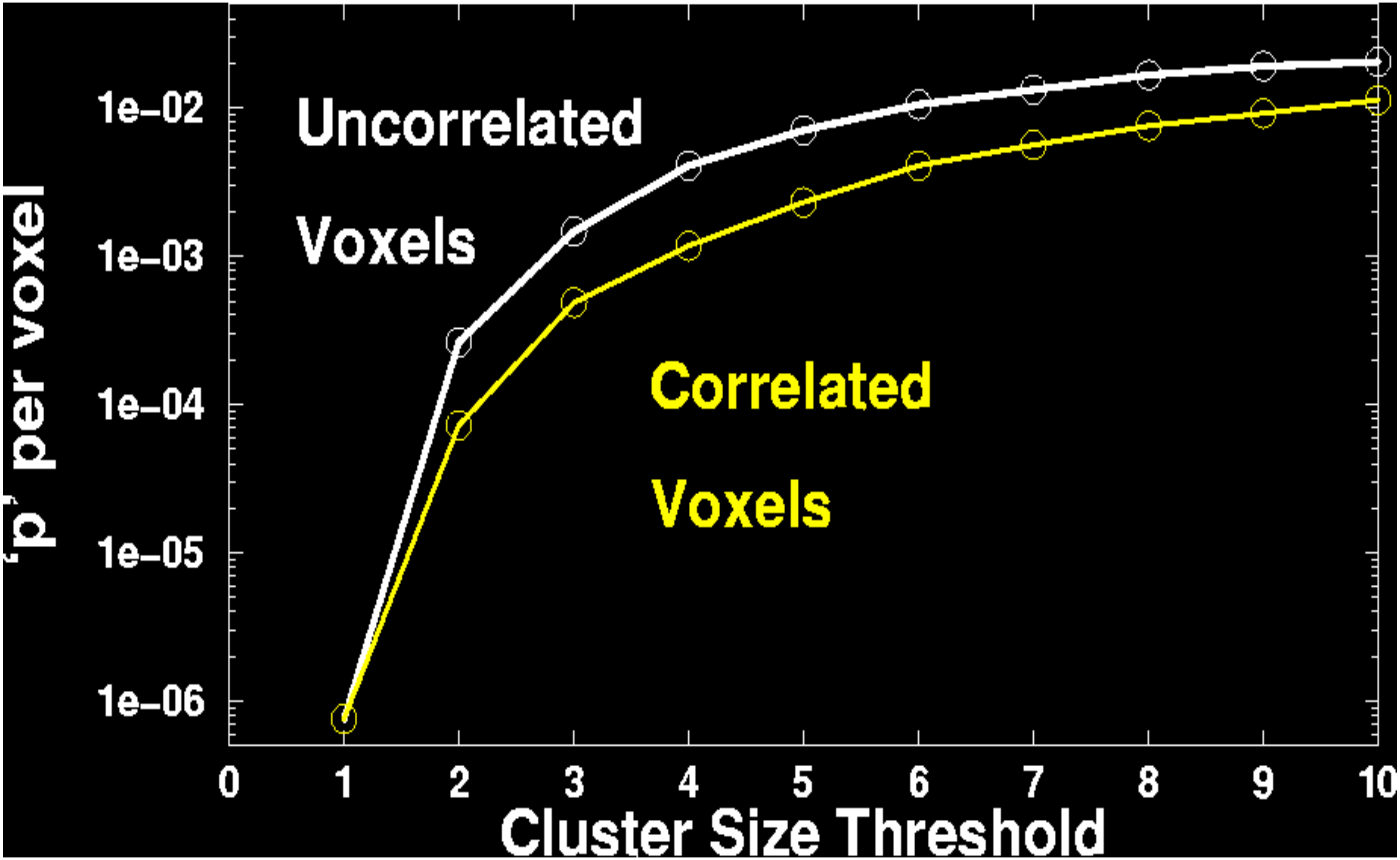


- Why blur FMRI time series?
  - Averaging neighbors will reduce noise
  - Activations are (usually) blob-ish (several voxels across)
  - Diminishes the multiple comparisons problem
- **3dBlurToFWHM** and **3dBlurInMask** blur only inside a mask region
  - To avoid mixing air (noise-only) and brain voxels
  - Partial Differential Equation (PDE) based blurring method
    - 2D (intra-slice) or 3D blurring

Spatial Clustering

- Analyze data, create statistical map (*e.g.*, ***t*** statistic in each voxel)
- Threshold map at a low ***t*** value, in each voxel separately
  - Will have many false positives
- Threshold map by rejecting clusters of voxels below a given size
- Can control false-positive rate by adjusting ***t*** threshold and cluster-size thresholds together

Cluster-Based Detection



What the World Needs Now

- Unified HRF/Deconvolution ⊕ Blob analysis
  - Time⊕Space patterns computed all at once, instead of arbitrary spatial smoothing
    - Increase statistical power by bringing data from multiple voxels together cleverly
  - Instead of time analysis followed by spatial analysis (described earlier)
  - Instead of component-style analyses (*e.g.*, ICA) that do not use stimulus timing

---

- Difficulty: models for spatial blobs
  - Little information *à priori* ⇒ must be adaptive

In the Thinking Stages

- “Area under curve” addition to **-gltsym** to allow testing of pieces of HRF models from **-stim_times**
- Slice- and/or voxel-dependent regressors
  - For physiological noise cancellation, etc.
  - To save memory? (Could process each slice separately)
    - One slice-at-a-time regression can be done in a Unix script, using 3dZcutup and 3dZcat programs

Very close now

Or 3dREMLfit

- Extend AM regression to allow for more than 1 piece of auxiliary information at each stimulus time
- Interactive tool to examine **-x1D** matrix for problems
  - and **3dDeconvolve** testing of GLT submatrices
- Semi-linear deconvolution program

Done!

Multi-Voxel Statistics

Spatial Clustering & False Discovery Rate:

“Correcting” the Significance

Basic Problem

- Usually have 20-100K FMRI voxels in the brain
- Have to make at least one decision about each one:
  - Is it “active”?
    - That is, does its time series match the temporal pattern of activity we expect?
  - Is it differentially active?
    - That is, is the BOLD signal change in task #1 different from task #2?
- Statistical analysis is designed to control the error rate of these decisions
  - Making ***lots*** of decisions: hard to get perfection in statistical testing

Multiple Testing Corrections

- **Two types of errors**
  - What is $\mathbf{H_0}$ in FMRI studies? H_0: no effect (activation, difference, …) at a voxel
  - Type I error = Prob(reject H_0 when H_0 is true) = false positive = p value

    Type II error = Prob(accept H_0 when H_1 is true) = false negative = β

    power = $1-\beta$ = probability of detecting true activation
  - Strategy: controlling type I error while increasing power (decreasing type II errors)
  - Significance level α (magic number 0.05) : $p < \alpha$

Justice System: Trial

	Hidden Truth	
	Defendant Innocent	Defendant Guilty
Reject Presumption of Innocence (Guilty Verdict)	**Type I Error (defendant very unhappy)**	**Correct**
Fail to Reject Presumption of Innocence (Not Guilty Verdict)	**Correct**	**Type II Error (defendant very happy)**

Statistics: Hypothesis Test

	Hidden Truth	
	H_0 True Not Activated	H_0 False Activated
Reject H_0 (decide voxel is activated)	**Type I Error (false positive)**	**Correct**
Don’t Reject H_0 (decide voxel isn’t activated)	**Correct**	**Type II Error (false negative)**

- **Family-Wise Error (FWE)**
  - Multiple testing problem: voxel-wise statistical analysis
    - With *N* voxels, what is the chance to make a false positive error (Type I) in one or more voxels?

      ***Family-Wise Error**:* $\alpha_{\text{FW}} = 1-(1-p)^N \rightarrow 1$ as N increases
    - For $N{\cdot}p$ small (compared to 1), $\alpha_{\text{FW}} \approx N{\cdot}p$
    - $N \approx 20{,}000+$ voxels in the brain
    - To keep probability of even one false positive $\alpha_{\text{FW}} < 0.05$ (the "corrected" *p*-value), need to have $p < 0.05 / 2{\times}10^4 = 2.5{\times}10^{-6}$
    - This constraint on the per-voxel ("uncorrected") *p*-value is so stringent that we'll end up rejecting a lot of true positives (Type II errors) also, just to be safe on the Type I error rate
- Multiple testing problem in FMRI
  - 3 occurrences of multiple tests: individual, group, and conjunction
  - Group analysis is the most severe situation (have the least data, considered as number of independent samples = subjects)

- **Two Approaches to the "Curse of Multiple Comparisons"**
  - Control FWE to keep expected total number of false positives below 1
    - Overall significance: α_{FW} = Prob(≥ one false positive voxel in the whole brain)
    - **Bonferroni correction**: $\alpha_{\text{FW}} = 1 - (1-p)^N \approx Np$, if $p << N^{-1}$
      - Use $p = \alpha / N$ as individual voxel significance level to achieve $\alpha_{\text{FW}} = \alpha$
      - Too stringent and overly conservative: $p = 10^{-8} \ldots 10^{-6}$
    - Something to rescue us from this hell of statistical super-conservatism?
      - Correlation: Voxels in the brain are not independent
        - Especially after we smooth them together!
        - Means that Bonferroni correction is *way way* too stringent
      - Contiguity: Structures in the brain activation map
        - We are looking for activated "blobs": the chance that pure noise (H_0) will give a set of seemingly-activated voxels next to each other is lower than getting false positives that are scattered around far apart
      - Control FWE based on spatial correlation (smoothness of image noise) **and** minimum cluster size we are willing to accept

---

  - Control false discovery rate (FDR)
    - FDR = expected proportion of false positive voxels among all detected voxels
      - Give up on the idea of having (almost) no false positives at all

Cluster Analysis: `AlphaSim`

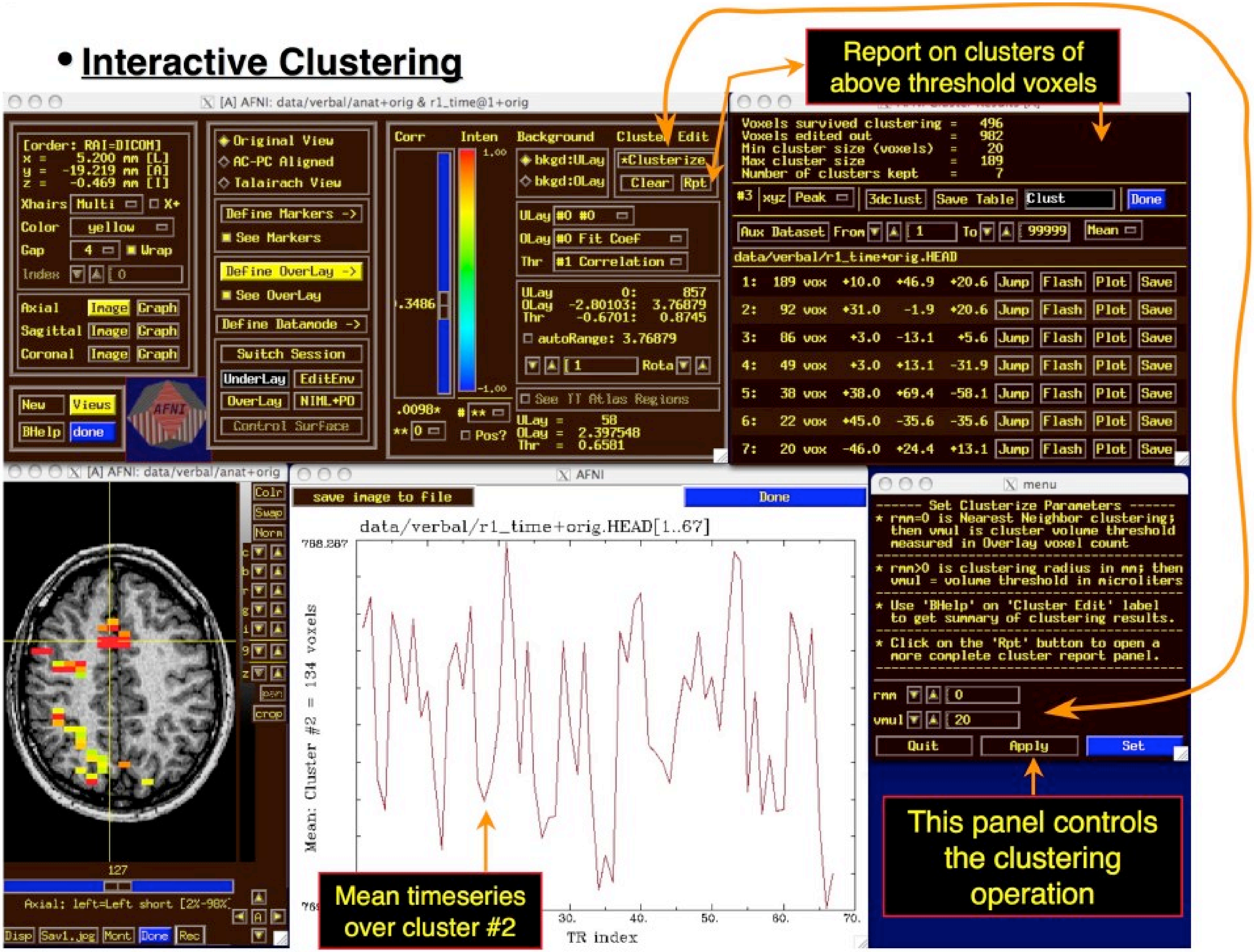
- **FWE control in AFNI**
  - Monte Carlo simulations with program `AlphaSim`
    - Named for a place where primary attractions are randomization experiments
    - Randomly generate some number (*e.g.*, 1000) of brain volumes with white noise (spatially uncorrelated)
      - That is, each "brain" volume is purely in H_0 = no activation
      - Noise images can be blurred to mimic the smoothness of real data
    - Count number of voxels that are false positives in each simulated volume
      - Including how many are false positives that are spatially together in clusters of various sizes (1, 2, 3, …)
    - Parameters to program
      - Size of dataset to simulate
      - Mask (e.g., to consider only brain-shaped regions in the 3D brick)
      - Spatial correlation FWHM: from `3dBlurToFWHM` or `3dFWHMx`
      - Connectivity radius: how to identify voxels belonging to a cluster?
        - Default = NN connection = touching faces
      - Individual voxel significance level = uncorrected *p*-value
    - Output
      - Simulated (estimated) overall significance level (corrected *p*-value)
      - Corresponding minimum cluster size at the input uncorrected *p*-value

- **Example:** `AlphaSim -nxyz 64 64 20 -dxyz 3 3 5 \`
  `-fwhm 5 -pthr 0.001 -iter 1000 -quiet -fast`
- Output is in 6 columns: focus on 1st and 6th columns (ignore others)
  - 1st column: cluster size in voxels
  - 6th column: alpha (α) = overall significance level = corrected *p*-value

```
Cl Size     Frequency    CumuProp     p/Voxel   Max Freq       Alpha
      1         47064    0.751113  0.00103719          0    1.000000
      2         11161    0.929236  0.00046268         13    1.000000
      3          2909    0.975662  0.00019020        209    0.987000
      4          1054    0.992483  0.00008367        400    0.778000
      5           297    0.997223  0.00003220        220    0.378000
      6           111    0.998995  0.00001407        100    0.158000
      7            32    0.999505  0.00000594         29    0.058000
      8            20    0.999825  0.00000321         19    0.029000
      9             8    0.999952  0.00000126          7    0.010000
     10             2    0.999984  0.00000038          2    0.003000
     11             1    1.000000  0.00000013          1    0.001000
```

  - At this uncorrected $p=0.001$, in this size volume, with noise of this smoothness: the chance of a cluster of size 8 *or larger* occurring by chance alone is 0.029
  - May have to run several times with different uncorrected *p*
    - uncorrected (`-pthr`) *p*↑ ⇔ required minimum cluster size↑
  - See detailed steps at **http://afni.nimh.nih.gov/sscc/gangc/mcc.html**

- **Interactive Clustering**



```
[order: RAI=DICOM]
x =    5.200 mm [L]
y =  -19.219 mm [A]
z =   -0.469 mm [I]
Xhairs Multi
Color  yellow
Gap    4    Wrap
Index  0
```

```
◆ bkgd:ULay
◇ bkgd:OLay
ULay #0 #0
OLay #0 Fit Coef
Thr  #1 Correlation
ULay          0:      857
OLay  -2.80103:  3.76879
Thr   -0.6701:   0.8745
□ autoRange: 3.76879
1    Rota
.3486
.0098*
# **
** 0
ULay =     58
OLay =  2.397548
Thr  =  0.6581
```

```
Voxels survived clustering  =   496
Voxels edited out           =   982
Min cluster size (voxels)   =    20
Max cluster size            =   189
Number of clusters kept     =     7
#3 xyz Peak   3dclust  Save Table  Clust  Done
Aux Dataset From 1 To 99999 Mean
data/verbal/r1_time+orig.HEAD
1:  189 vox  +10.0  +46.9  +20.6
2:   92 vox  +31.0   -1.9  +20.6
3:   86 vox   +3.0  -13.1   +5.6
4:   49 vox   +3.0  +13.1  -31.9
5:   38 vox  +38.0  +69.4  -58.1
6:   22 vox  +45.0  -35.6  -35.6
7:   20 vox  -46.0  +24.4  +13.1
```

```
127
Axial: left=Left short [2%-98%]
```

```
data/verbal/r1_time+orig.HEAD[1..67]
788.287
```

```
------ Set Clusterize Parameters ------
* rmm=0 is Nearest Neighbor clustering;
  then vmul is cluster volume threshold
  measured in Overlay voxel count
----------------------------------------
* rmm>0 is clustering radius in mm; then
  vmul = volume threshold in microliters
----------------------------------------
* Use 'BHelp' on 'Cluster Edit' label
  to get summary of clustering results.
----------------------------------------
* Click on the 'Rpt' button to open a
  more complete cluster report panel.
----------------------------------------
rmm  0
vmul 20
```

Report on clusters of above threshold voxels

Mean timeseries over cluster #2

This panel controls the clustering operation

False Discovery Rate in AFNI

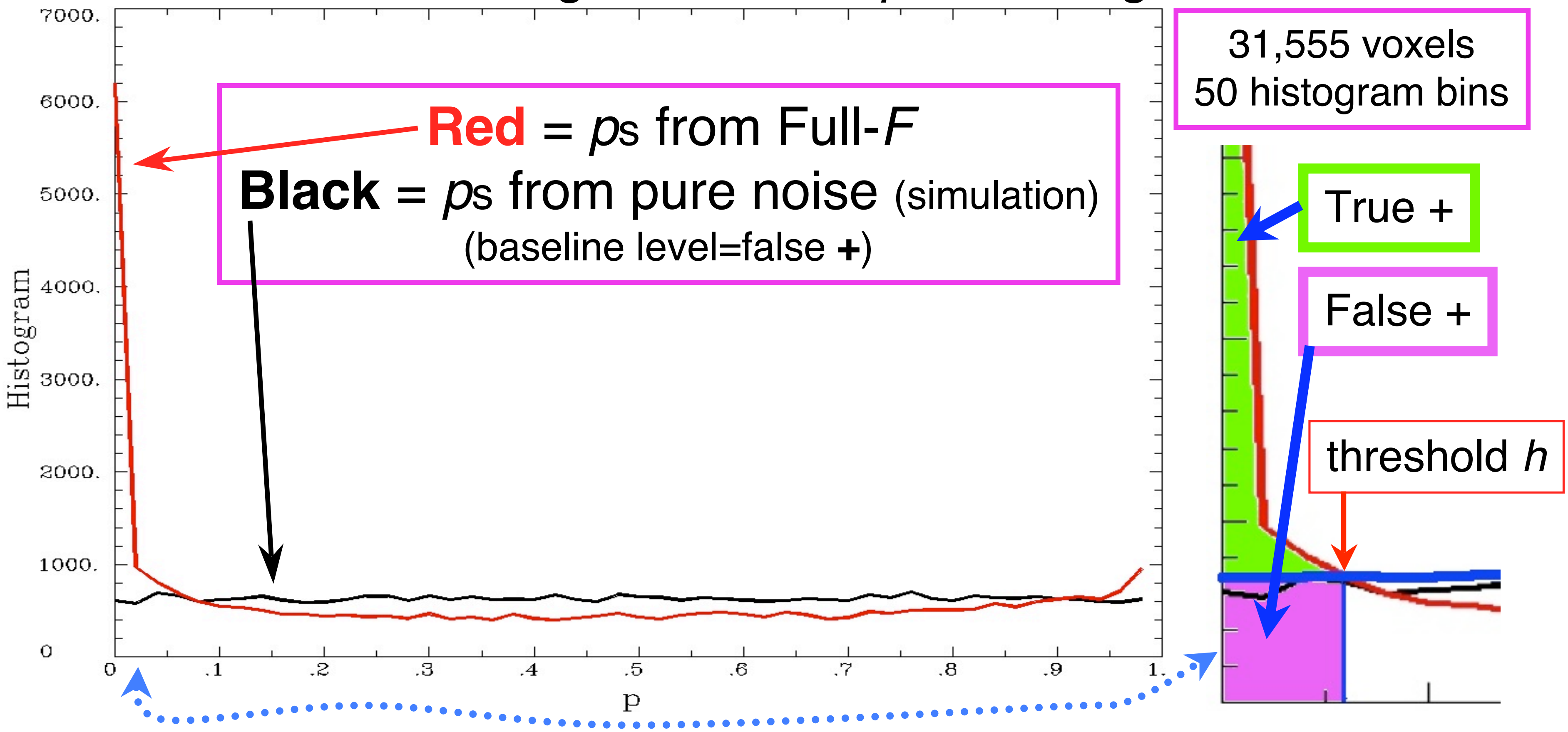
- Situation: making *many* statistical tests at once
  - e.g, Image voxels in FMRI; associating genes with disease
- Want to set threshold on statistic (e.g., *F*- or *t*-value) to control ***false positive*** error rate
- Traditionally: set threshold to control probability of making a ***single*** false positive detection
  - But if we are doing 1000s (or more) of tests at once, we have to be very stringent to keep this probability low
- **FDR**: accept the fact that there will be multiple erroneous detections when making lots of decisions
  - Control the ***fraction*** of positive detections that are wrong
    - Of course, no way to tell which individual detections are right!
  - Or at least: control the *expected value* of this fraction

FDR: *q* [and *z*(*q*)]

- Given some collection of statistics (say, *F*-values from `3dDeconvolve`), set a threshold *h*
- The **uncorrected *p*-value** of *h* is the probability that *F* > *h* when the null hypothesis is true (no activation)
  - "Uncorrected" means "per-voxel"
  - The "corrected" *p*-value is the probability that *any* voxel is above threshold in the case that they are all *un*activated
  - If have *N* voxels to test, $p_{\text{corrected}} = 1-(1-p)^N \approx Np$ (for small *p*)
    - Bonferroni: to keep $p_{\text{corrected}} < 0.05$, need $p < 0.05 / N$, which is very tiny
- The FDR ***q*-value** of *h* is the fraction of false positives expected when we set the threshold to *h*
  - Smaller *q* is "better" (more stringent = fewer false detections)
  - *z*(*q*) = conversion of *q* to Gaussian *z*-score: e.g, $z(0.05) \approx 1.95996$
    - So that larger is "better" (in the same sense): e.g, $z(0.01) \approx 2.57583$

Basic Ideas Behind FDR *q*

- ***If*** all the null hypotheses are true, ***then*** the statistical distribution of the *p*-values will be uniform
  - Deviations from uniformity at low *p*-values ⇒ true positives
  - Baseline of uniformity indicates how many true negatives are hidden amongst in the low *p*-value region



Red = *p*s from Full-*F*

Black = *p*s from pure noise (simulation)

(baseline level=false **+**)

31,555 voxels

50 histogram bins

True +

False +

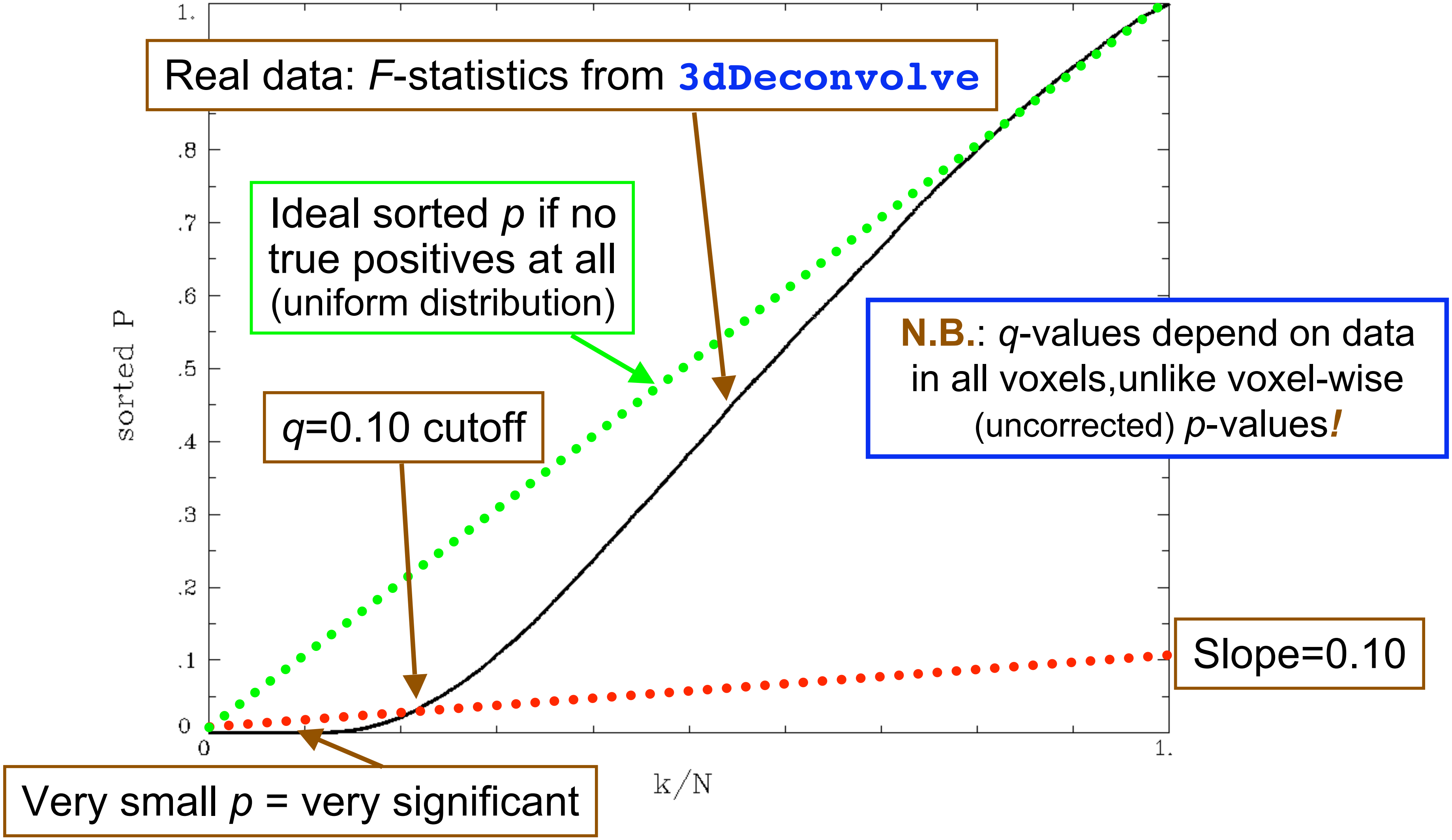
threshold *h*

How *q* is Calculated from Data

- Compute *p*-values of each statistic: P_1, P_2, P_3, $\cdots$, P_N
- Sort these: $P_{(1)} \leq P_{(2)} \leq P_{(3)} \leq \cdots \leq P_{(N)}$ {subscript$_{()}$ ≡ sorted}
- For $k = 1..N$, $q_{(k)} = \min_{m \geq k} [\, N{\cdot}P_{(m)}/m]$
  - Easily computed from sorted *p*-values by looping downwards from $k = N$ to $k = 1$
- By keeping track of voxel each $P_{(k)}$ came from: can put *q*-values (or $z(q)$ values) back into image
  - This is exactly how program **`3dFDR`** works
- By keeping track of statistic value (*t* or *F*) each $P_{(k)}$ came from: can create curve of threshold *h* vs. $z(q)$
- **N.B.**: *q*-values depend on the data in ***all*** voxels, unlike these voxel-wise (uncorrected) *p*-values***!***
  - Which is why it’s important to mask brain properly

Graphical Calculation of *q*

- Graph sorted *p*-values of voxel #*k* vs. κ=*k*/*N* (the cumulative histogram of *p*, flipped sideways) and draw some lines from origin



Real data: *F*-statistics from `3dDeconvolve`

Ideal sorted *p* if no true positives at all (uniform distribution)

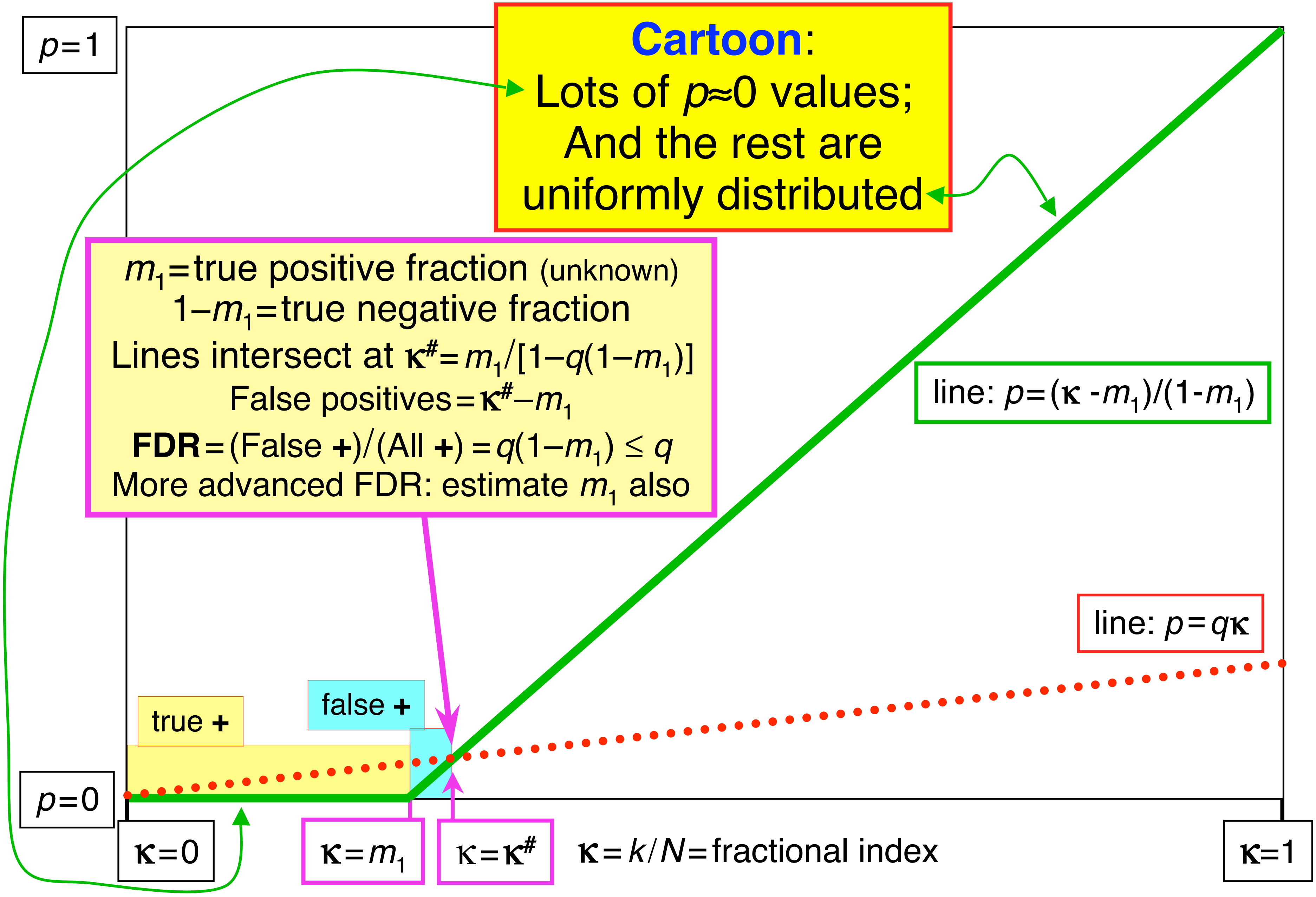
q=0.10 cutoff

N.B.: *q*-values depend on data in all voxels,unlike voxel-wise (uncorrected) *p*-values*!*

Slope=0.10

Very small *p* = very significant

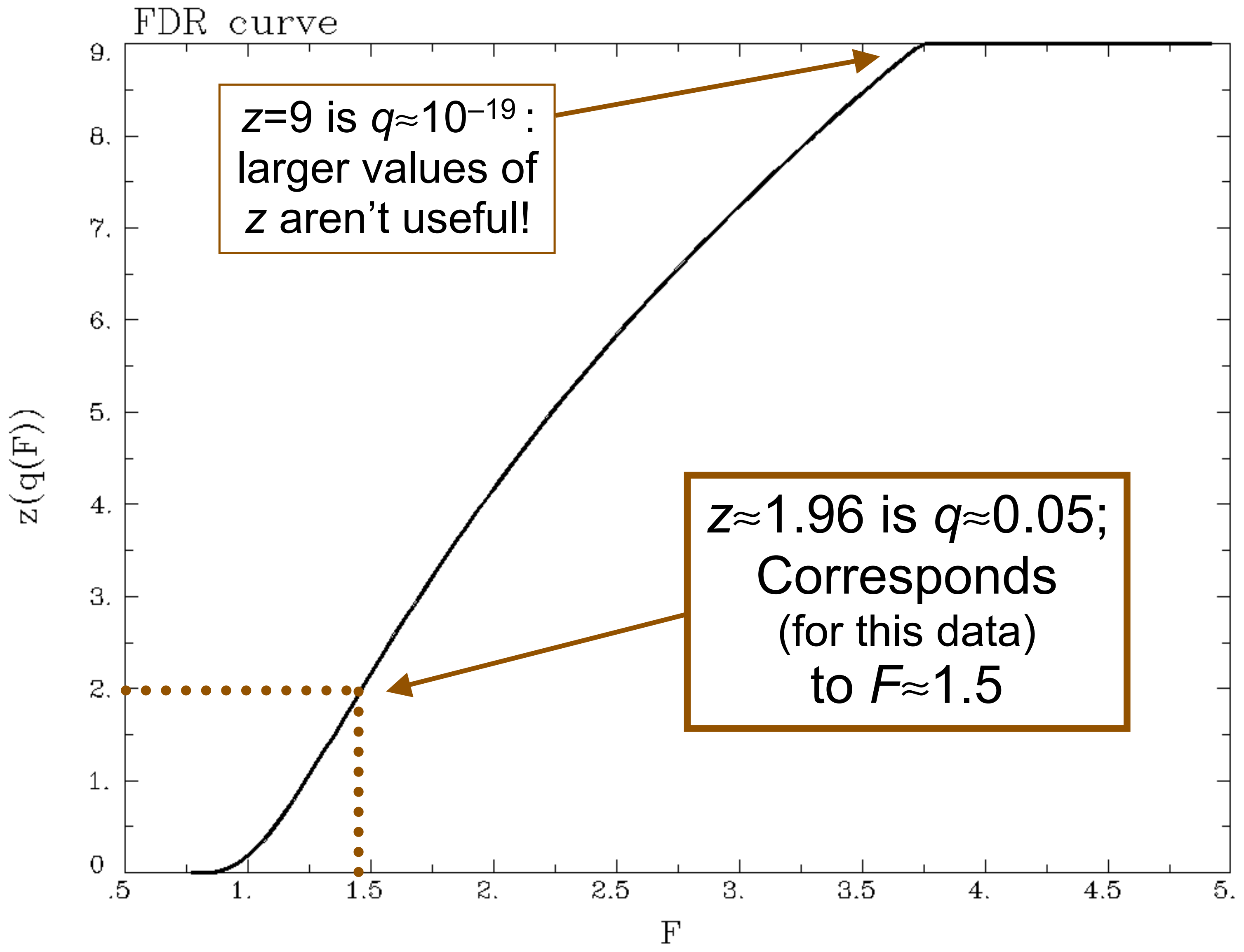
Why This Line-Drawing Works



Cartoon:
Lots of $p \approx 0$ values;
And the rest are
uniformly distributed

m_1=true positive fraction (unknown)
$1-m_1$=true negative fraction
Lines intersect at $\kappa^\# = m_1/[1-q(1-m_1)]$
False positives$=\kappa^\# - m_1$
FDR$=$(False **+**)/(All **+**)$= q(1-m_1) \leq q$
More advanced FDR: estimate m_1 also

Same Data: threshold *F* vs. *z*(*q*)



z=9 is $q \approx 10^{-19}$: larger values of z aren't useful!

$z \approx 1.96$ is $q \approx 0.05$; Corresponds (for this data) to $F \approx 1.5$

Recent Changes to `3dFDR`

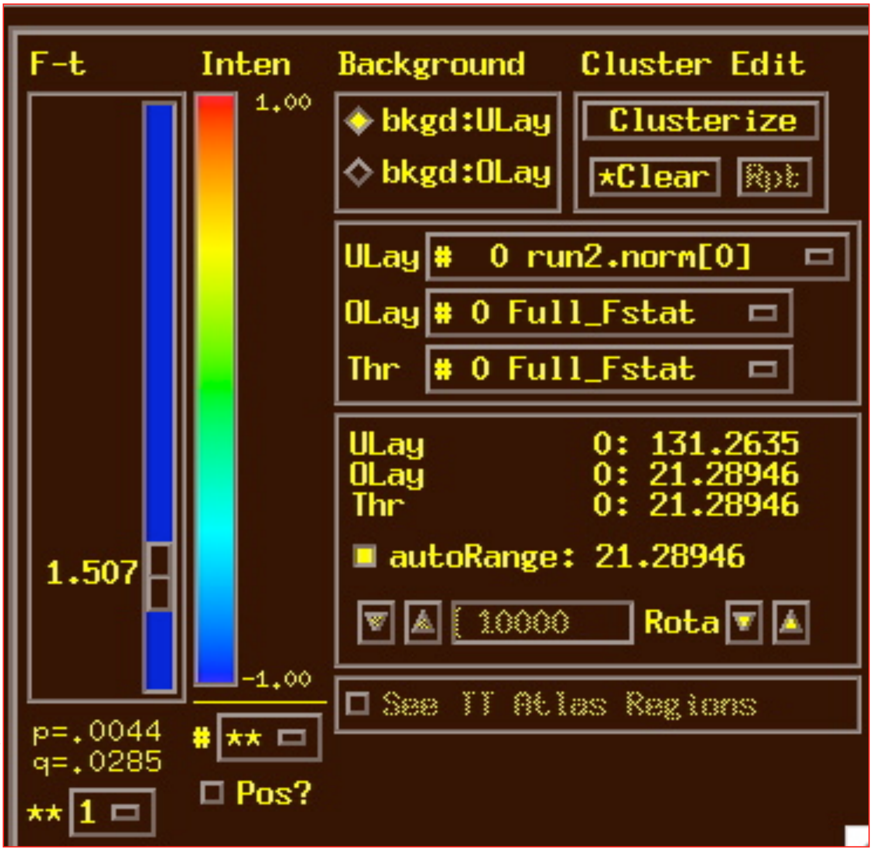
- Don't include voxels with p=1 (e.g., F=0), even if they are in the `–mask` supplied on the command line
  - This changes decreases N, which will decrease q and so increase $z(q)$: recall that $q_{(k)} = \min_{m \geq k} [\ N \cdot P_{(m)} / m]$
- Sort with Quicksort algorithm
  - Faster than the bin-based sorting in the original code
  - Makes a big speed difference on large 1 mm^3 datasets
    - Not much speed difference on small 3 mm^3 grids, since there aren't so many voxels to sort
- Default mode of operation is '`-new`' method
  - Prints a warning message to let user know things have changed from the olden days
  - User can use '`-old`' method if desired

FDR curves: *h* vs. *z*(*q*)

- **3dDeconvolve**, **3dANOVAx**, **3dttest**, and **3dNLfim** now compute FDR curves for all statistical sub-bricks and store them in output header

- **3drefit -addFDR** does same for other datasets
  - **3drefit -unFDR** can be used to delete such info

- **AFNI** now shows *p*- **and** *q*-values below the threshold slider bar
  - Interpolates FDR curve from header (threshold→*z*→*q*)
    - Can be used to adjust threshold by "eyeball"



ULay # 0 run2.norm[0]
OLay # 0 Full_Fstat
Thr # 0 Full_Fstat

ULay 0: 131.2635
OLay 0: 21.28946
Thr 0: 21.28946

■ autoRange: 21.28946

1.507

p=.0044
q=.0285

** 1

q = N/A means it's not available

MDF hint = "missed detection fraction"

FDR Statistical Issues

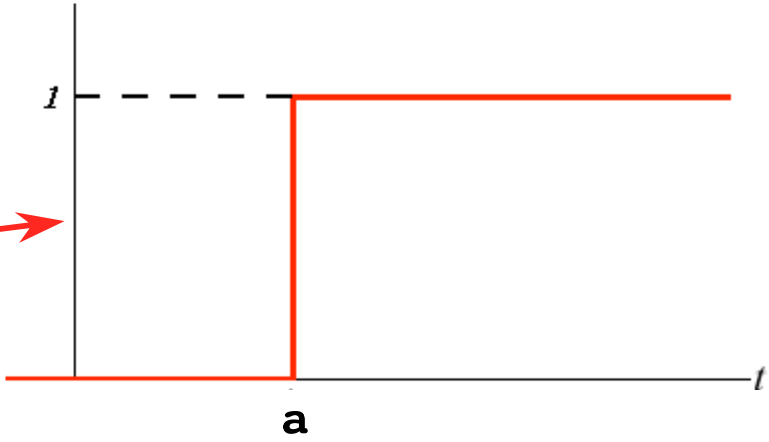
- FDR is conservative (*q*-values are too large) when voxels are positively correlated (e.g., from spatially smoothing)
  - Correcting for this is not so easy, since *q* depends on data (including true positives), so a simulation like **`AlphaSim`** is hard to conceptualize
  - At present, FDR is an alternative way of controlling false positives, vs. **`AlphaSim`** (clustering)
    - Thinking about how to combine FDR and clustering
- Accuracy of FDR calculation depends on *p*-values being uniformly distributed under the null hypothesis
  - Statistic-to-*p* conversion should be accurate, which means that null *F*-distribution (say) should be correctly estimated
  - Serial correlation in FMRI time series means that **`3dDeconvolve`** denominator DOF is too large
  - ⇒ *p*-values will be too small, so *q*-values will be too small
    - **`3dREMLfit`** can ride to the rescue***!***

FWE or FDR?

- These 2 methods control Type I error in different sense
  - FWE: α_{FW} = Prob (≥ one false positive voxel/cluster in the whole brain)
    - Frequentist's perspective: Probability among **many** hypothetical activation maps gathered under identical conditions
    - Advantage: can directly incorporate smoothness into estimate of α_{FW}
  - FDR = expected fraction of false positive voxels among all detected voxels
    - Focus: controlling false positives among detected voxels in **one** activation map, as given by the experiment at hand
    - Advantage: not afraid of making a few Type I errors in a large field of true positives
  - Concrete example
    - Individual voxel $p = 0.001$ for a brain of 25,000 EPI voxels
    - Uncorrected → ≈25 false positive voxels in the brain
    - FWE: corrected $p = 0.05$ → ≈5% of the time would expect one or more false positive clusters in the entire volume of interest
    - FDR: $q = 0.05$ → ≈5% of voxels among those positively labeled ones are false positive
- What if your favorite blob fails to survive correction?
  - Tricks (don't tell anyone we told you about these)
    - One-tail t-test?
    - ROI-based statistics – e.g., grey matter mask, or whatever regions you focus on
  - Analysis on surface; or, Use better group analysis tool (**`3dLME`**, **`3dMEMA`**, etc.)

Conjunction Analysis

- **Conjunction**
  - Dictionary: “a compound proposition that is true if and only if all of its component propositions are true”
  - FMRI: areas that are active under 2 or more conditions (**`AND`** logic)
    - e.g, in a visual language task and in an auditory language task
  - Can also be used to mean analysis to find areas that are exclusively activated in one task but not another (**`XOR`** logic) or areas that are active in either task (non-exclusive **`OR`** logic)
  - If have *n* different tasks, have 2^n possible combinations of activation overlaps in each voxel (ranging from nothing there to complete overlap)
  - Tool: **`3dcalc`** applied to statistical maps
    - Heaviside step function defines a *On/Off* logic
    - **`step(t-a)`** = 0 if **`t < a`**
= 1 if **`t > a`**
    - Can be used to apply more than one threshold at a time



- Example of forming all possible conjunctions
  - 3 contrasts/tasks A, B, and C, each with a *t*-stat from **3dDeconvolve**
  - Assign each a number, based on binary positional notation:
    - A: $001_2 = 2^0 = \mathbf{1}$; B: $010_2 = 2^1 = \mathbf{2}$; C: $100_2 = 2^2 = \mathbf{4}$
  - Create a mask using 3 sub-bricks of *t* (*e.g.*, threshold = 4.2)

```
3dcalc -a ContrA+tlrc -b ContrB+tlrc -c ContrC+tlrc \
-expr '1*step(a-4.2)+2*step(b-4.2)+4*step(c-4.2)'    \
-prefix ConjAna
```

  - Interpret output, which has 8 possible ($=2^3$) scenarios:

$000_2 =$ **0**: none are active at this voxel

$001_2 =$ **1**: A is active, but no others

$010_2 =$ **2**: B, but no others

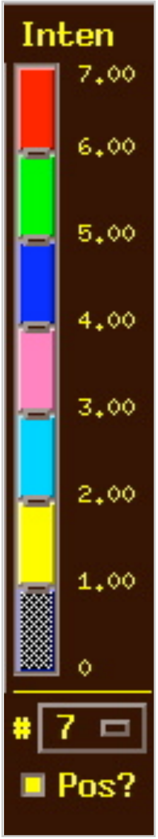
$011_2 =$ **3**: A and B, but not C

$100_2 =$ **4**: C but no others

$101_2 =$ **5**: A and C, but not B

$110_2 =$ **6**: B and C, but not A

$111_2 =$ **7**: A, B, and C are all active at this voxel



Can display each combination with a different color and so make pretty pictures that *might* even mean something!

- **Multiple testing correction issue**
  - How to calculate the *p*-value for the conjunction map?
  - No problem, ***if*** each entity was corrected (e.g., cluster-size thresholded at *t*=4.2) before conjunction analysis, via **`AlphaSim`**
  - But that may be too stringent (conservative) and over-corrected
  - With 2 or 3 entities, analytical calculation of conjunction p_{conj} is possible
    - Each individual test can have different uncorrected (per-voxel) *p*
    - Double or triple integral of tails of non-spherical (correlated) Gaussian distributions — not available in simple analytical formulae
  - With more than 3 entities, may have to resort to simulations
    - Monte Carlo simulations? (AKA: Buy a fast computer)
    - Will Gang Chen write such a program? Only time will tell!