Time Series Analysis in AFNI

Outline: 6+ Hours of Edification

- Philosophy
- Sample FMRI data
- Theory underlying FMRI analyses: the HRF
- "Simple" or "Fixed Shape" regression analysis
  - ★ Theory and Hands-on examples
- "Deconvolution" or "Variable Shape" analysis
  - ★ Theory and Hands-on examples
- Advanced Topics

Goals: Conceptual Understanding + Prepare to Try It Yourself

Data Analysis Philosophy

- **Signal** = Measurable response to stimulus
- **Noise** = Components of measurement that interfere with detection of signal
- Statistical detection theory:
  - ★ Understand relationship between stimulus & signal
  - ★ Characterize noise statistically
  - ★ Can then devise methods to distinguish noise-only measurements from signal+noise measurements, and assess the methods' reliability
  - ★ Methods and usefulness depend strongly on the assumptions
    - o Some methods are "robust" against erroneous assumptions, and some are not

FMRI Philosopy: Signals and Noise

- FMRI Stimulus→Signal connection and noise statistics are both poorly characterized
- Result: there is no "**best**" way to analyze FMRI time series data: there are only "**reasonable**" analysis methods
- To deal with data, must make some assumptions about the signal and noise
- Assumptions will be wrong, but must do ***something***
- Different kinds of experiments require different kinds of analyses
  - ★ Since signal models and questions you ask about the signal will vary
  - ★ It is important to understand what is going on, so you can select and evaluate "reasonable" analyses

Meta-method for creating analysis methods

- Write down a mathematical model connecting stimulus (or “activation”) to signal
- Write down a statistical model for the noise
- Combine them to produce an equation for measurements given signal+noise
  - ⋆ Equation will have unknown parameters, which are to be estimated from the data
  - ⋆ N.B.: signal may have zero strength
- Use statistical detection theory to produce an algorithm for processing the measurements to assess signal presence and characteristics
  - ⋆ e.g., least squares fit of model parameters to data

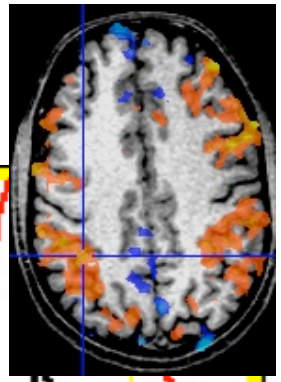
Time Series Analysis on Voxel Data

- Most common forms of FMRI analysis involve fitting an activation+BOLD model to each voxel's time series ***separately*** (AKA "univariate" analysis)
  - ★ Some pre-processing steps may do inter-voxel computations
    - o e.g., spatial smoothing to reduce noise
- Result of model fits is a set of parameters at each voxel, estimated from that voxel's data
  - ★ e.g., activation amplitude, delay, shape
  - ★ "SPM" = statistical parametric map
- Further analysis steps operate on individual SPMs
  - ★ e.g., combining/contrasting data among subjects

Some Sample FMRI Data Time Series

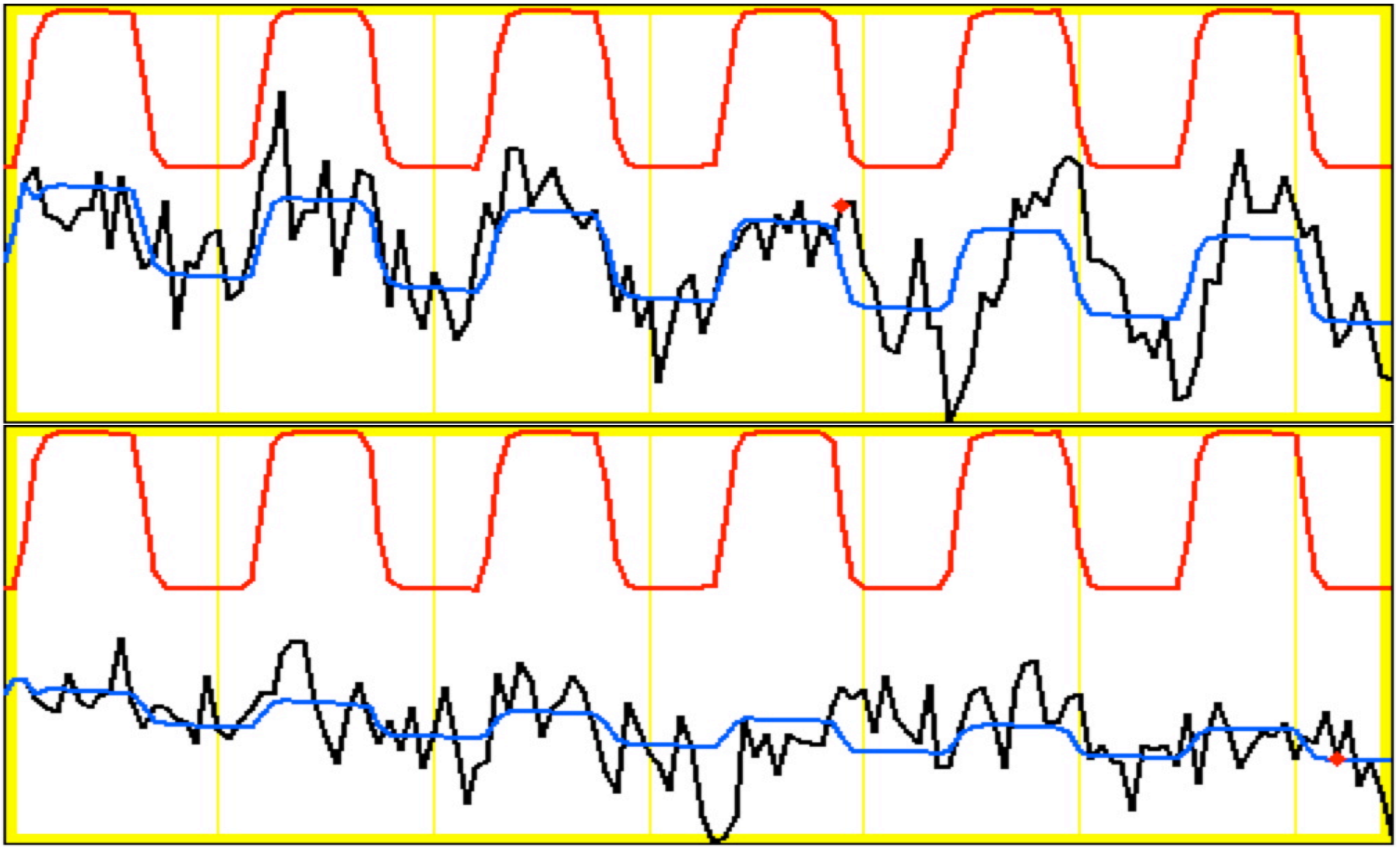
- First: Block-trial FMRI data
  - ★ "Activation" occurs over a sustained period of time (say, 10 s or longer), usually from more than one stimulation event, in rapid succession
  - ★ BOLD (hemodynamic) response accumulates from multiple close activations and is large
  - ★ BOLD response is often visible in time series
- Next 5 slides: same brain voxel in 9 imaging runs
  - ★ black curve (noisy) = data
  - ★ red curve (above data) = ideal model response
  - ★ blue curve (within data) = model fitted to data
  - ★ somatosensory task (finger being rubbed)

Same Voxel: Runs 1 and 2 (of 9)



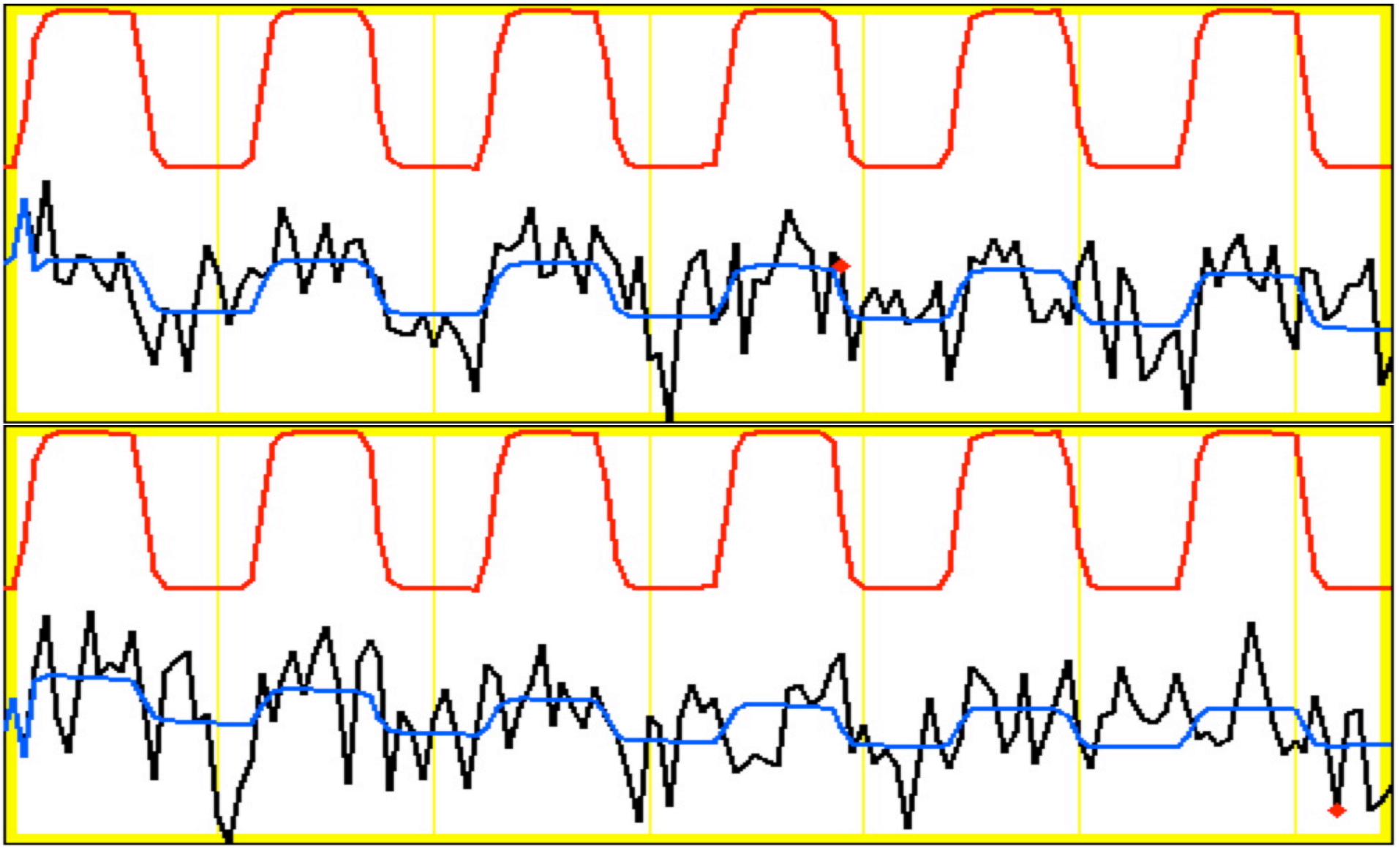
Block-trials: 27 s “on” / 27 s “off”; TR=2.5 s; 130 time points/run

Same Voxel: Runs 3 and 4



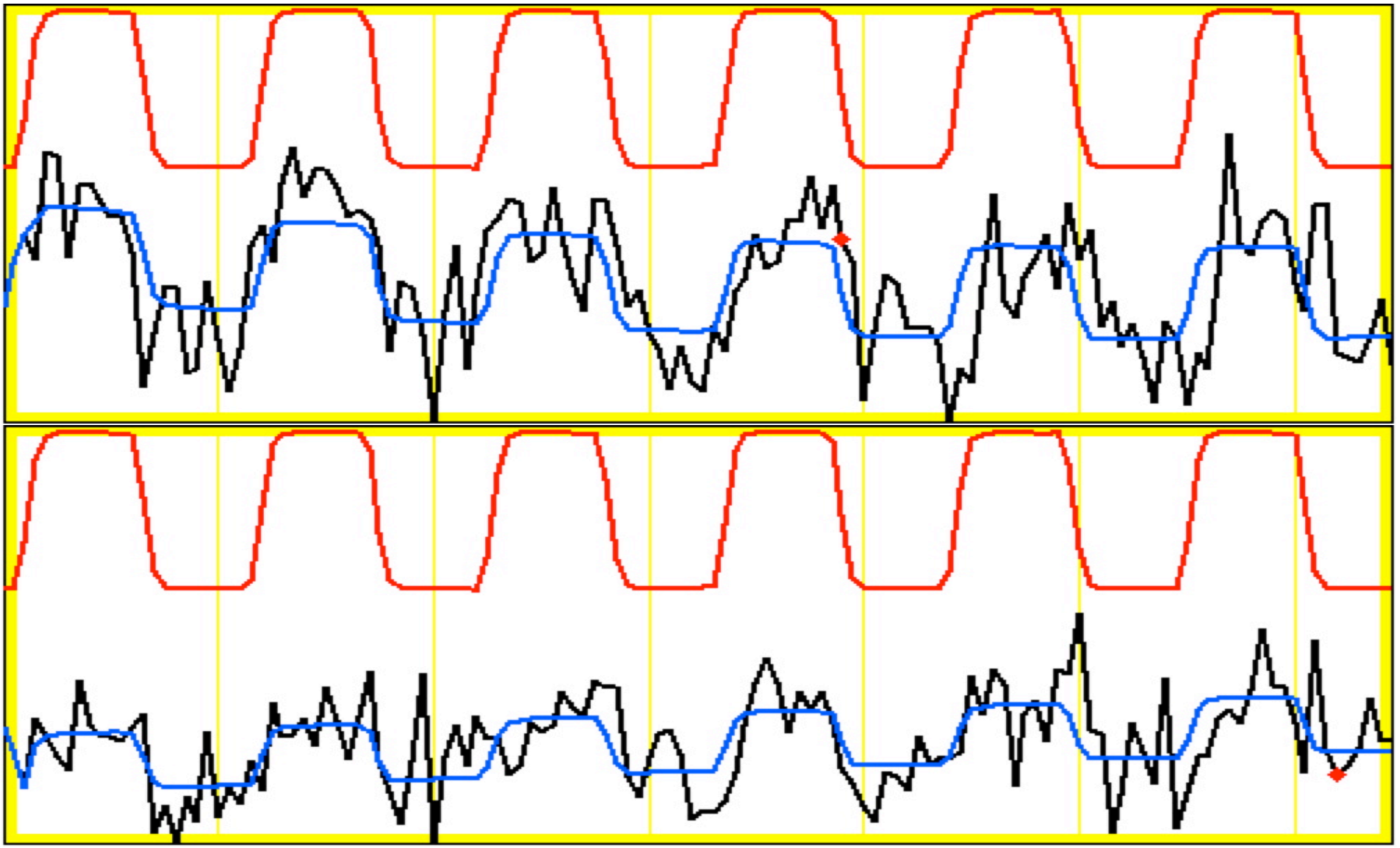
Block-trials: 27 s “on” / 27 s “off”; TR=2.5 s; 130 time points/run

Same Voxel: Runs 5 and 6



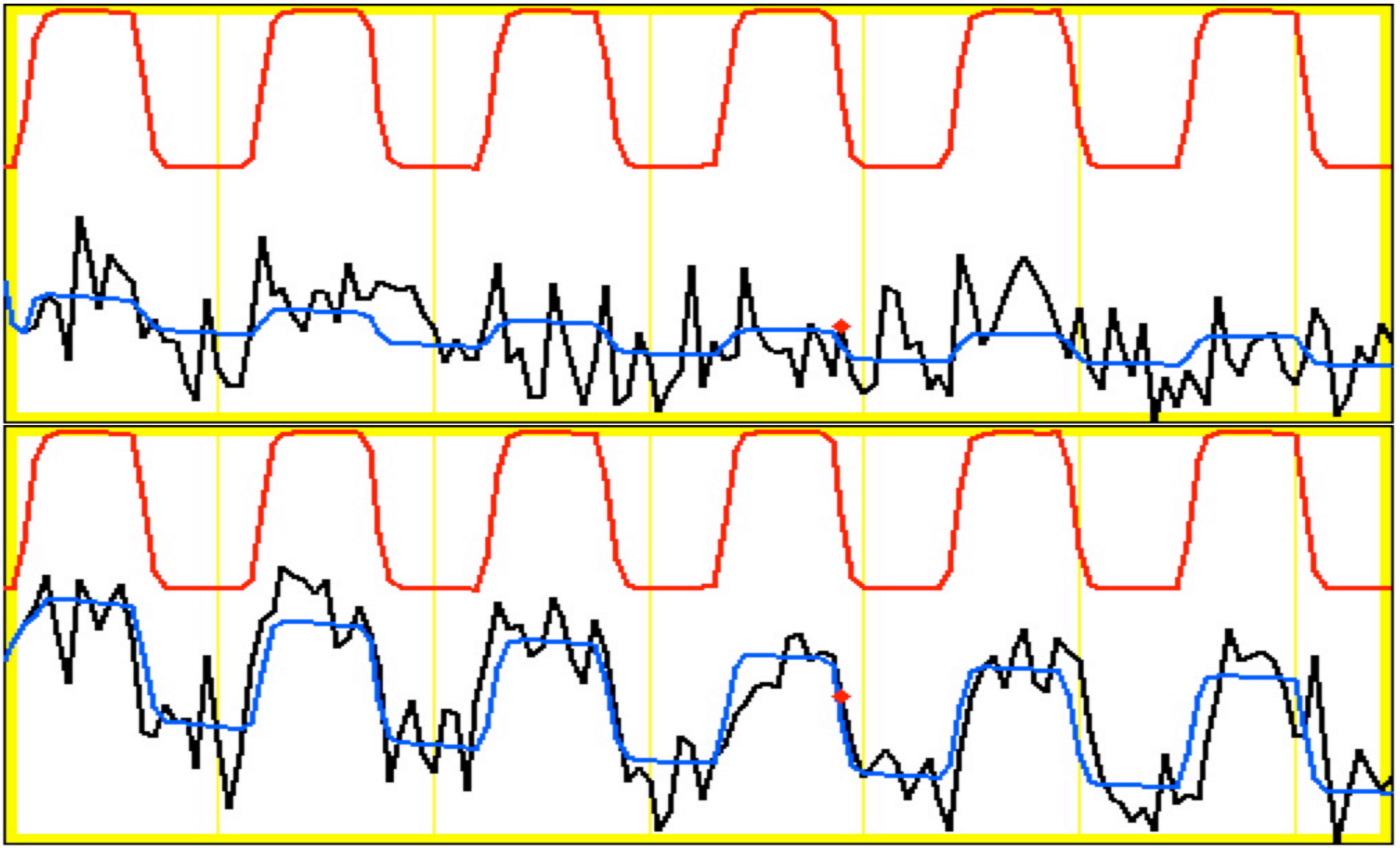
Block-trials: 27 s “on” / 27 s “off”; TR=2.5 s; 130 time points/run

Same Voxel: Runs 7 and 8



Block-trials: 27 s “on” / 27 s “off”; TR=2.5 s; 130 time points/run

Same Voxel: Run 9 and Average of all 9

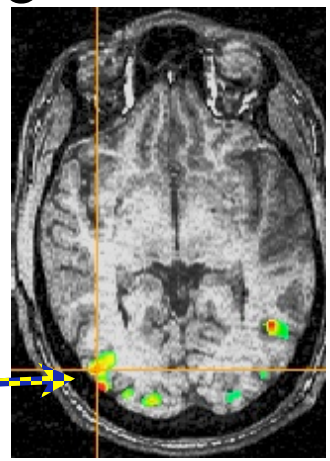


⇒ Activation amplitude and shape are variable! Why???

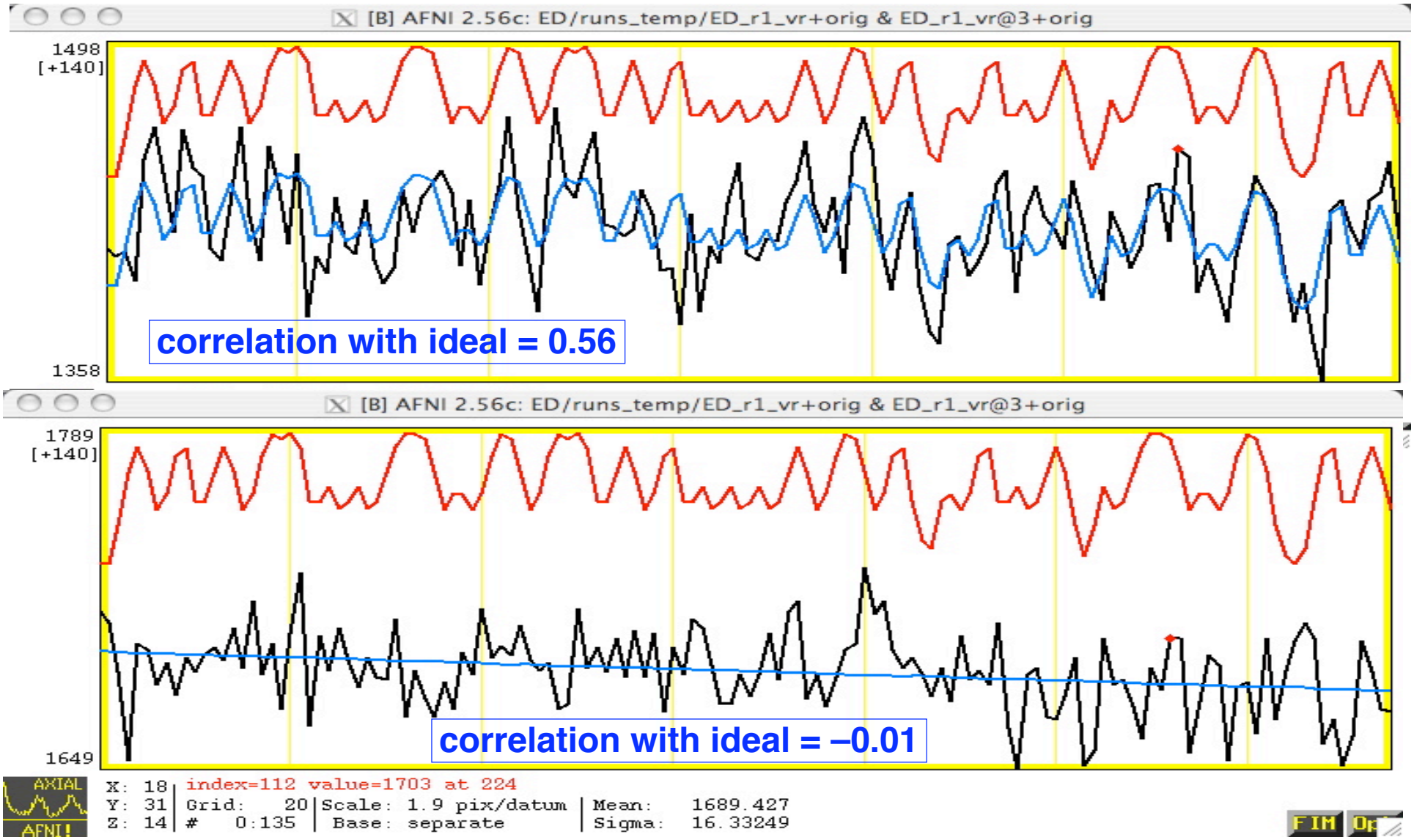
More Sample FMRI Data Time Series

- Second: Event-related FMRI
  - ★ "Activation" occurs in single relatively brief intervals
  - ★ "Events" can be randomly or regularly spaced in time
    - o If events are randomly spaced in time, signal model itself looks noise-like (to the human eye)
  - ★ BOLD response to stimulus tends to be weaker since fewer nearby-in-time "activations" have overlapping hemodynamic responses
- Next slide: Visual stimulation experiment

"Active" voxel shown in next slide



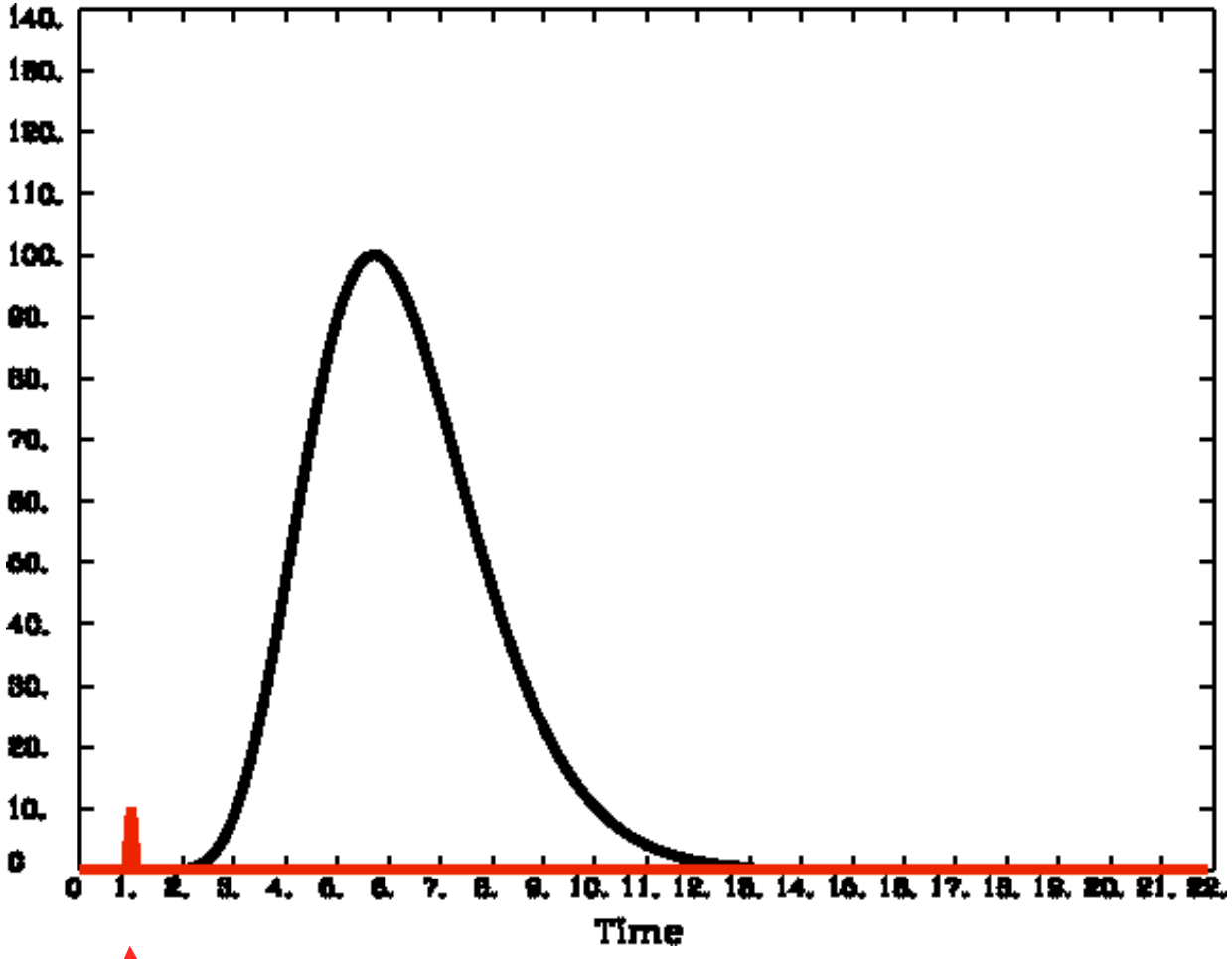
Two Voxel Time Series from Same Run



Lesson: ER-FMRI activation is not obvious via casual inspection

Hemodynamic Response Function (HRF)

- HRF is the idealization of measurable FMRI signal change responding to a single activation cycle (up and down) from a stimulus in a voxel



Response to brief activation (< 1 s):

- delay of 1-2 s
- rise time of 4-5 s
- fall time of 4-6 s
- model equation:

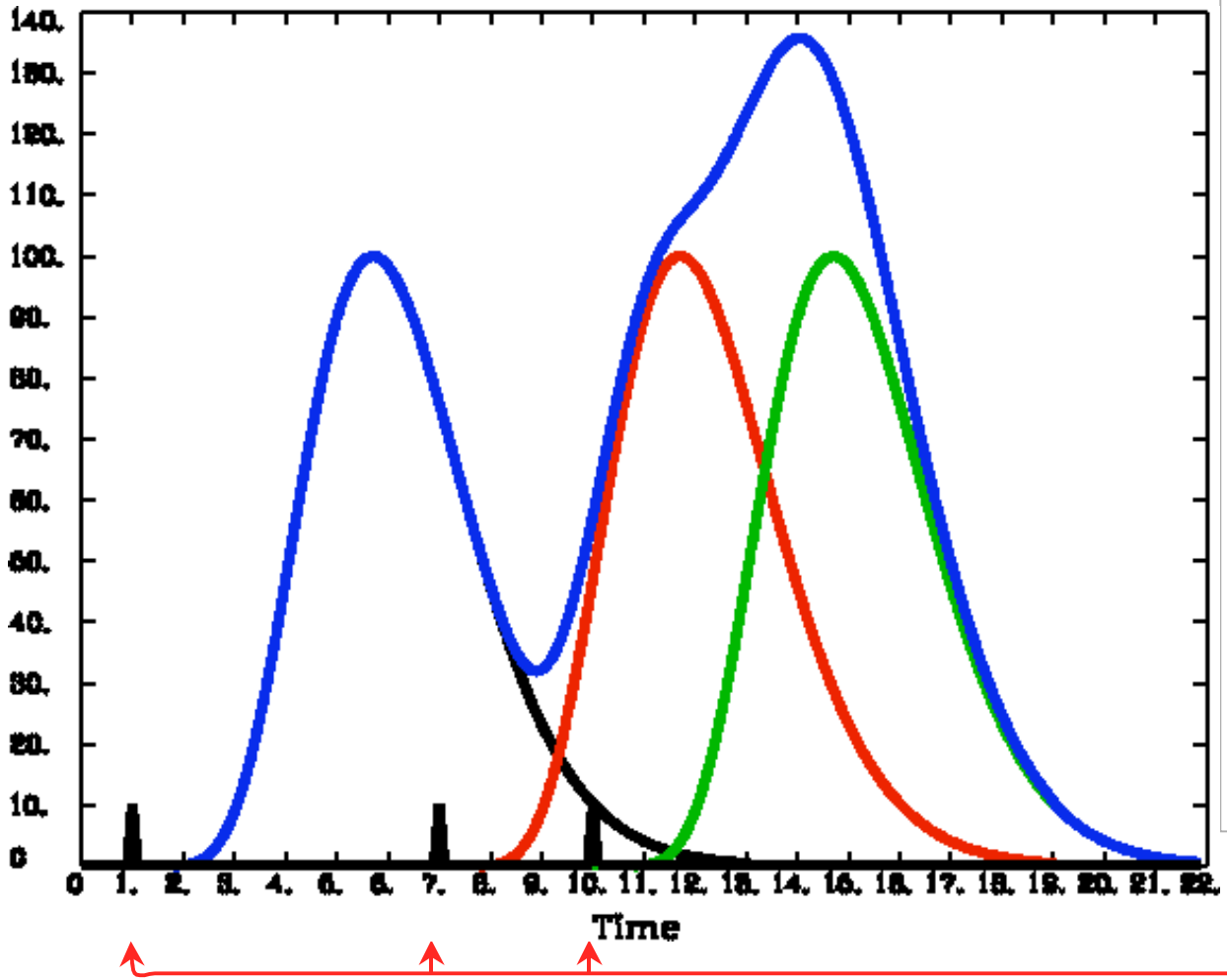
$$h(t) \propto t^b e^{-t/c}$$

- $h(t)$ is signal change t seconds ***after*** activation

1 Brief Activation

Linearity of HRF

- Multiple activation cycles in a voxel, closer in time than duration of HRF:
  - ★ Assume that overlapping responses add

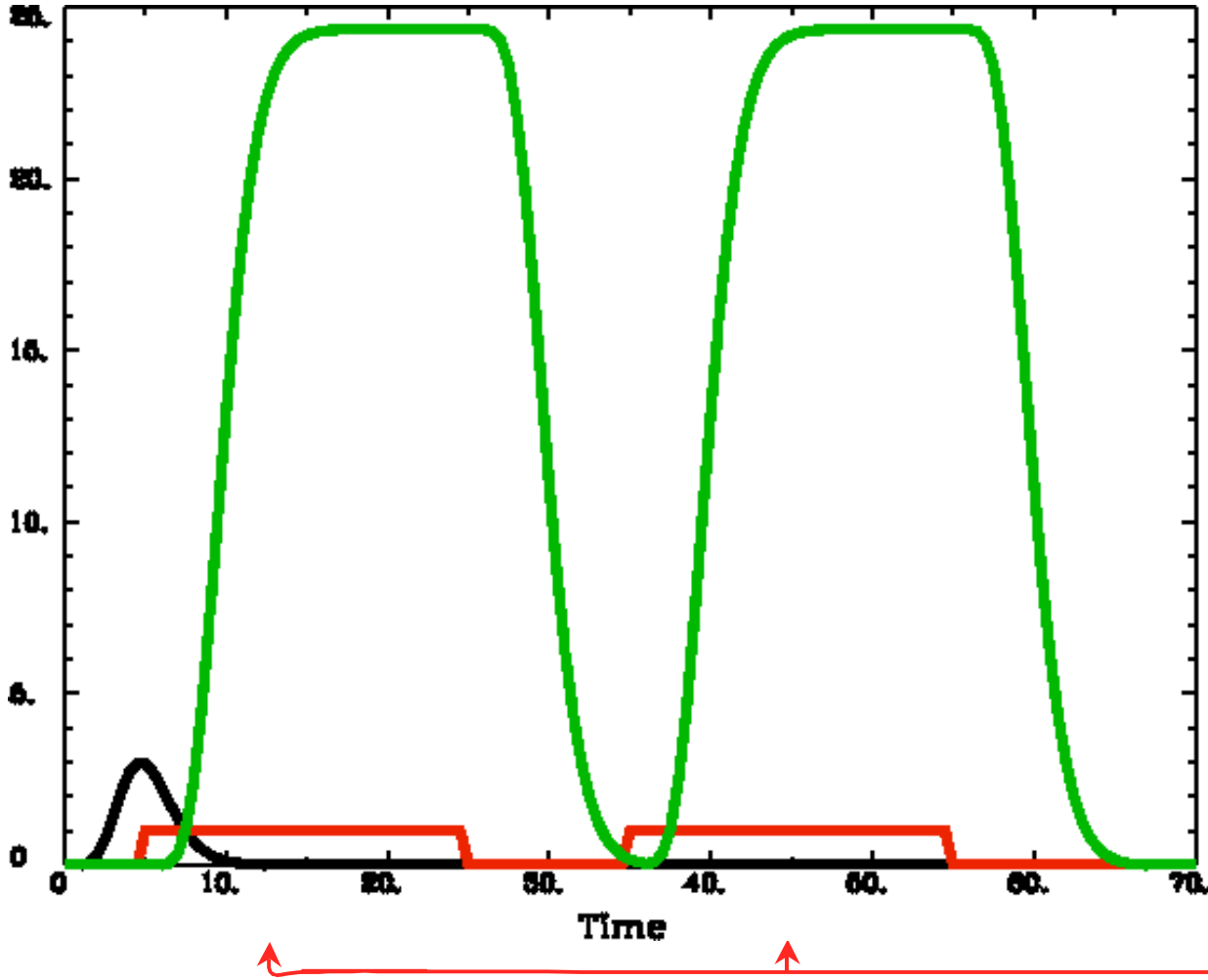


- Linearity is a pretty good assumption
- But not apparently perfect — about 90% correct
- Nevertheless, is widely taken to be true and is the basis for the "general linear model" (GLM) in FMRI

3 Brief Activations

Linearity and Extended Activation

- Extended activation, as in a block-trial experiment:
  - ★ HRF accumulates over its duration (≈ 10 s)

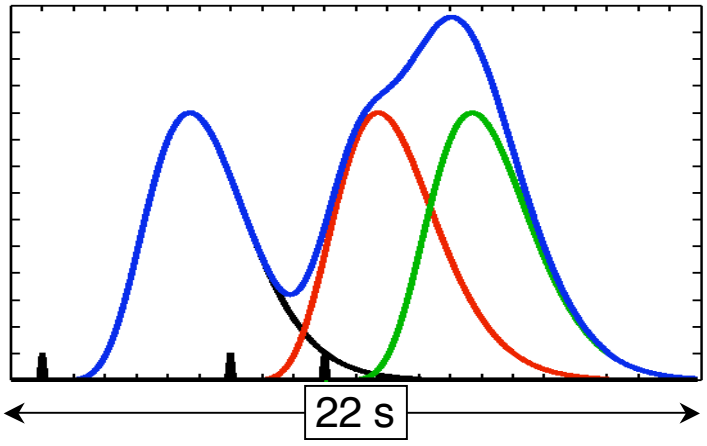


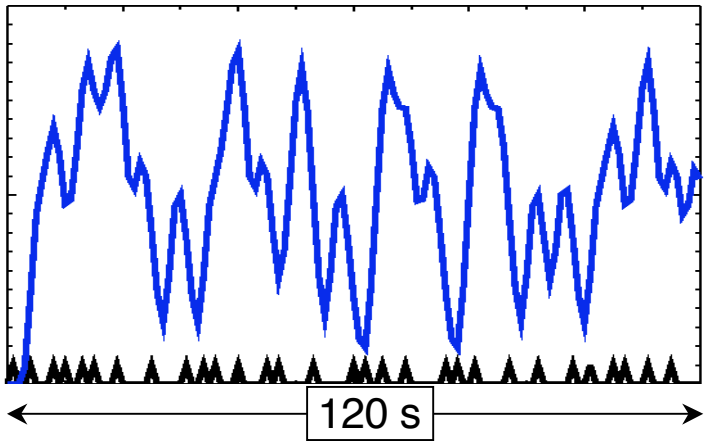
- **Black** curve = response to a single brief stimulus
- **Red** curve = activation intervals
- **Green** curve = summed up HRFs from activations
- Block-trials have larger BOLD signal changes than event-related experiments

2 Extended Activations

Convolution Signal Model

- FMRI signal we look for in each voxel is taken to be sum of the individual trial HRFs
  - ⋆ Stimulus timing is assumed known (or measured)
  - ⋆ Resulting time series (**blue** curves) are called the ***convolution*** of the HRF with the stimulus timing
- Must also allow for baseline and baseline drifting
  - ⋆ Convolution models only the FMRI signal **changes**





- Real data starts at and returns to a nonzero, slowly drifting baseline

Simple Regression Models

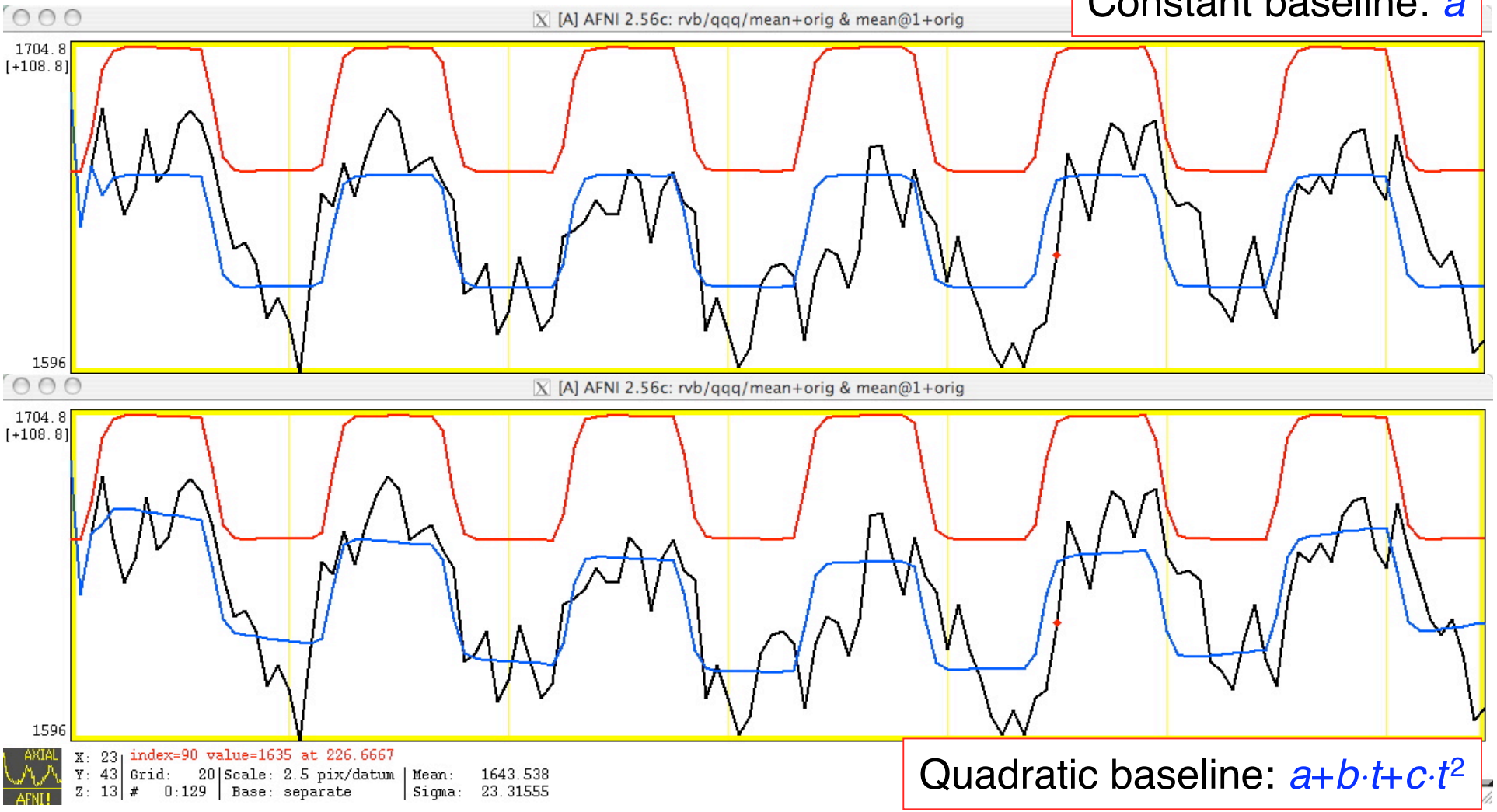
- Assume a fixed shape $h(t)$ for the HRF
  - ★ e.g., $h(t) = t^{8.6} \exp(-t/0.547)$ [MS Cohen, 1997]
  - ★ Convolved with stimulus timing (e.g., AFNI program **`waver`**), get ideal response function $r(t)$
- Assume a form for the baseline
  - ★ e.g., $a + b \cdot t$ for a constant plus a linear trend
- In each voxel, fit data $Z(t)$ to a curve of the form

$$Z(t) \approx a + b \cdot t + \beta \cdot r(t)$$

  - a, b, β are unknown parameters to be calculated in each voxel
  - a,b are "nuisance" parameters
  - β is amplitude of $r(t)$ in data = "how much" BOLD

Simple Regression: Example

Constant baseline: a



Quadratic baseline: $a+b \cdot t+c \cdot t^2$

- Necessary baseline model complexity depends on duration of **continuous** imaging — e.g., 1 parameter per 100 seconds

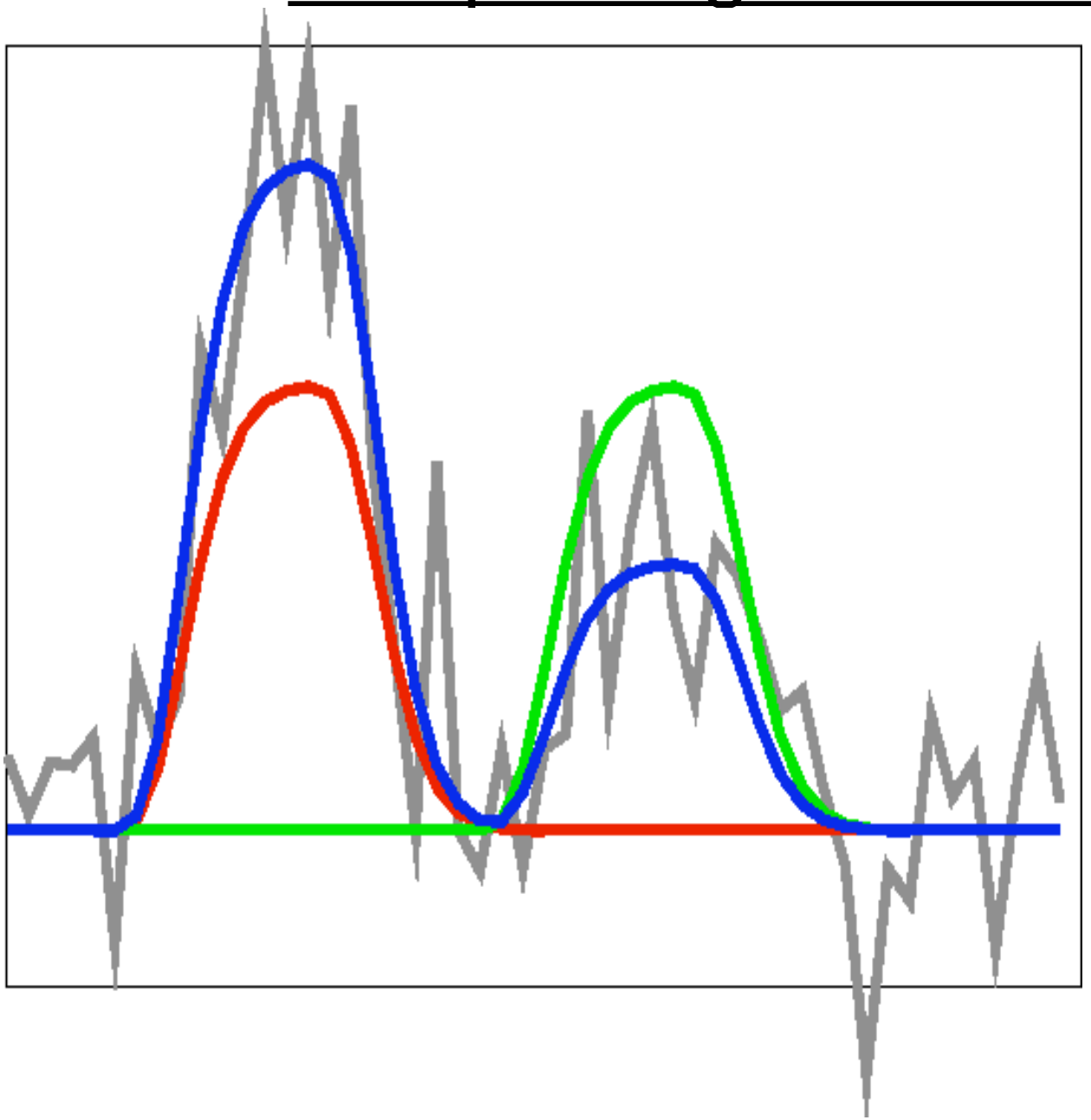
Multiple Stimuli = Multiple Regressors

- Usually have more than one class of stimulus or activation in an experiment
  - ★ e.g., want to see size of "face activation" vis-à-vis "house activation"; or, "what" vs. "where" activity
- Need to model each separate class of stimulus with a separate response function $r_1(t)$, $r_2(t)$, $r_3(t)$, ….
  - ★ Each $r_j(t)$ is based on the stimulus timing for activity in class number j
  - ★ Calculate a β_j amplitude = amount of $r_j(t)$ in voxel data time series $Z(t)$
  - ★ Contrast βs to see which voxels have differential activation levels under different stimulus conditions
    - o e.g., statistical test on the question $\beta_1 - \beta_2 = 0$?

Multiple Stimuli - Important Caveat

- You do ***not*** model the baseline condition
  - e.g., “rest”, visual fixation, high-low tone discrimination, or some other simple task
- FMRI can only measure ***changes*** in MR signal levels between tasks
  - So you need some simple-ish task to serve as a reference point
- The baseline model (e.g., $a + b \cdot t$) takes care of the signal level to which the MR signal returns when the “active” tasks are turned off
  - Modeling the reference task explicitly would be redundant (or “collinear”, to anticipate a forthcoming jargon word)

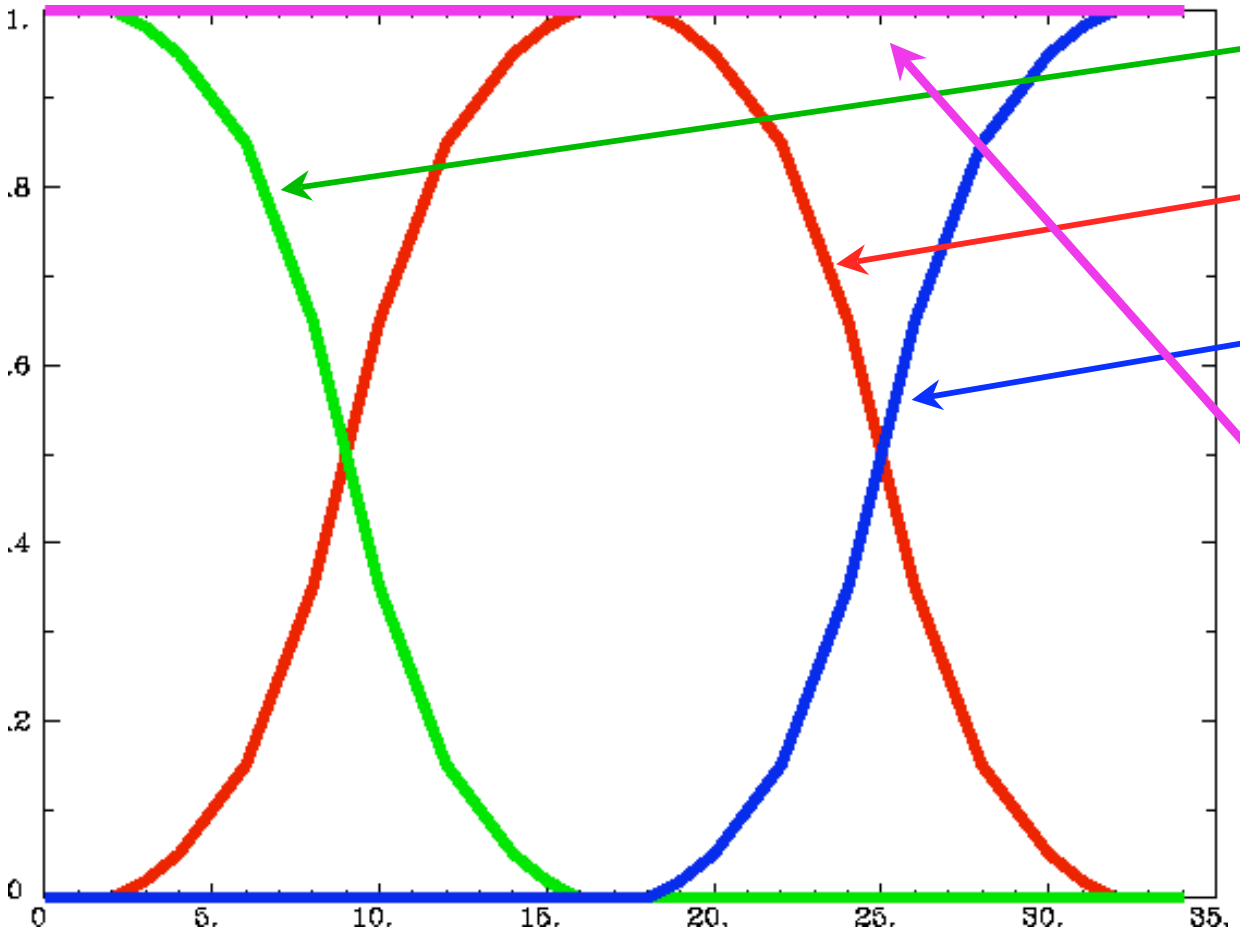
Multiple Regressors: Cartoon



- **Red** curve = signal model for class #1
- **Green** curve = signal model for #2
- **Blue** curve = $\beta_1 \cdot$#1$+\beta_2 \cdot$#2 where β_1 and β_2 vary from 0.1 to 1.7 in the animation

- Goal of regression is to find β_1 and β_2 that make the blue curve best fit the data time series
- **Gray** curve = **1.5**$\cdot$#1$+$**0.6**$\cdot$#2$+$noise = simulated data

Multiple Regressors: Collinearity!!



Green curve = signal model for #1

Red curve = signal model for class #2

Blue curve = signal model for #3

Purple curve = **#1** + **#2** + **#3** which is exactly = 1

• We cannot — ***in principle or in practice*** — distinguish sum of 3 signal models from constant baseline!!

No analysis can distinguish the cases

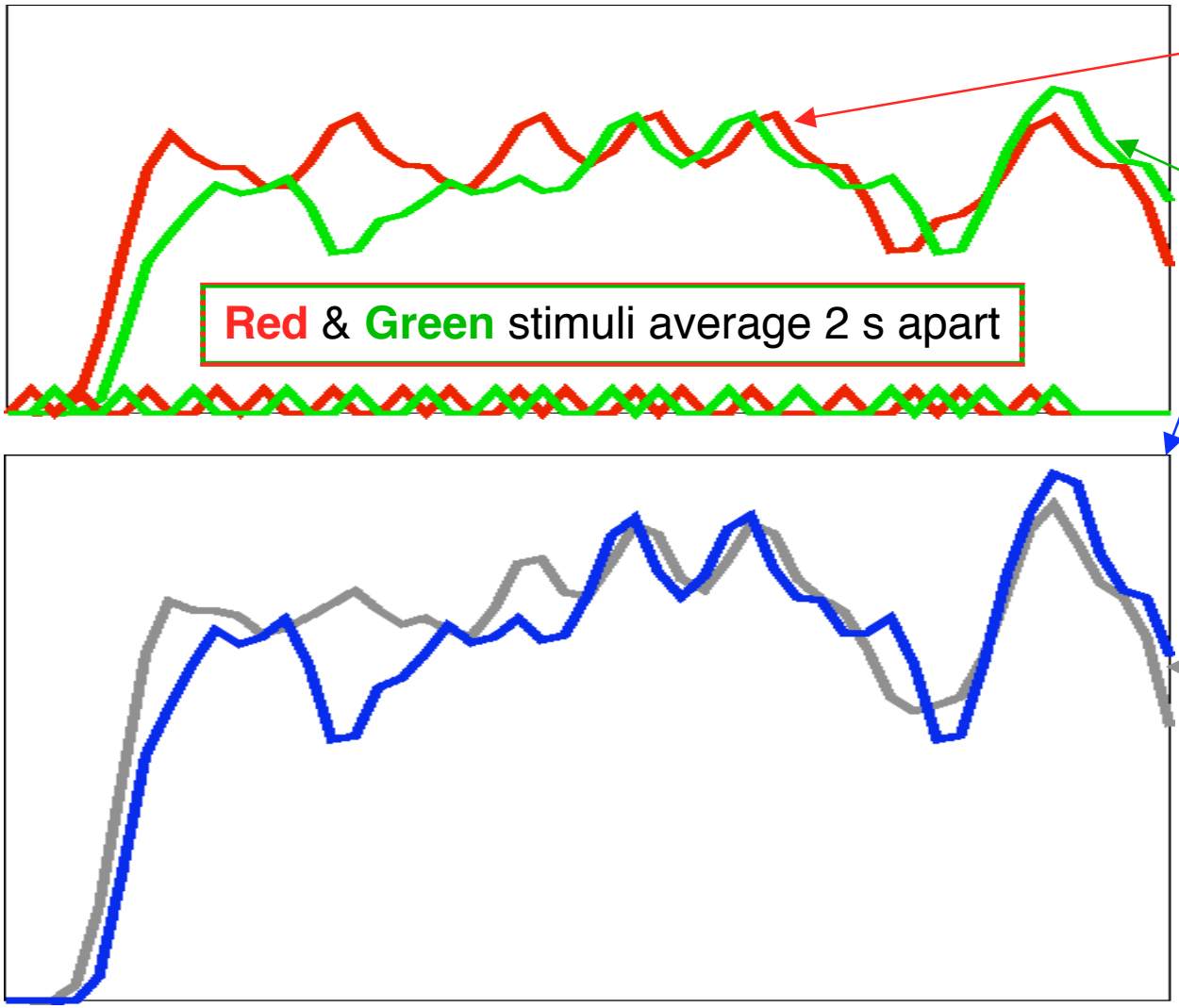
$Z(t)$=10+ 5·**#1** and

$Z(t)$= 0+15·**#1**+10·**#2**+10·**#3**

and an infinity of other possibilities

⟹ Collinear designs are **bad bad bad**!

Multiple Regressors: Near Collinearity



- **Red** curve = signal model for class #1
- **Green** curve = signal model for #2
- **Blue** curve = $\beta_1 \cdot$#1$+(1-\beta_1)\cdot$#2 where β_1 varies randomly from 0.0 to 1.0 in animation
- **Gray** curve = **0.66**·#1+**0.33**·#2 = simulated data *with no noise*
- Lots of different combinations of #1 and #2 are decent fits to gray curve

Stimuli are too close in time to distinguish response #1 from #2, considering noise

Equations: Notation

- Will generally follow notation of Doug Ward's manual for the AFNI program **3dDeconvolve**
- Time: continuous in reality, but in steps in the data
  - ★ Functions of continuous time are written like $f(t)$
  - ★ Functions of discrete time expressed like $f(\underbrace{n \cdot TR}_{=t_n})$ where n=0,1,2,… and TR=time step
  - ★ Usually use subscript notion f_n as shorthand
  - ★ Collection of numbers assembled in a column is a ***vector*** and is printed in boldface:

$$\left\{\begin{matrix}\text{vector of}\\ \text{length } N\end{matrix}\right\} = \begin{bmatrix} f_0 \\ f_1 \\ f_2 \\ \vdots \\ f_{N-1} \end{bmatrix} = \mathbf{f} \qquad \begin{bmatrix} A_{00} & A_{01} & \cdots & A_{0,N-1} \\ A_{10} & A_{11} & \cdots & A_{1,N-1} \\ \vdots & \vdots & \ddots & \vdots \\ A_{M-1,0} & A_{M-1,1} & \cdots & A_{M-1,N-1} \end{bmatrix} = \mathbf{A} = \left\{M \times N \text{ matrix}\right\}$$

Equations: Single Response Function

- In each voxel, fit data Z_n to a curve of the form

  $Z_n \approx a + b \cdot t_n + \beta \cdot r_n$ for $n=0,1,\ldots,N-1$ (N=# time pts)

  - a, b, β are unknown parameters to be calculated in each voxel
  - a,b are "nuisance" baseline parameters
  - β is amplitude of $r(t)$ in data = "how much" BOLD
  - Baseline model might be more complicated for long (> 150 s) continuous imaging runs:
    - $150 < T < 300$ s: $a+b \cdot t+c \cdot t^2$
    - Longer: $a+b \cdot t+c \cdot t^2 + \lceil T/200 \rceil$ low frequency components
    - Might also include as extra baseline components the estimated subject head movement time series, in order to remove residual contamination from such artifacts

Equations: Multiple Response Functions

- In each voxel, fit data Z_n to a curve of the form

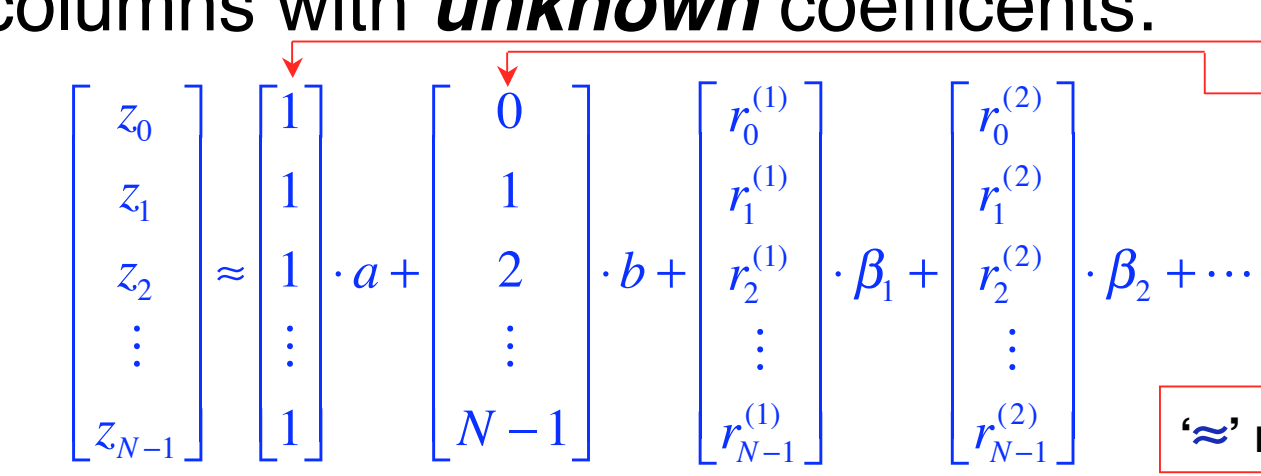
$$Z_n \approx [\text{baseline}]_n + \beta_1 \cdot r_n^{(1)} + \beta_2 \cdot r_n^{(2)} + \beta_3 \cdot r_n^{(3)} + \cdots$$

  - β_j is amplitude in data of $r_n^{(j)}=r_j(t_n)$; i.e., "how much" of j^{th} response function in in the data time series
  - In simple regression, each $r_j(t)$ is derived directly from stimulus timing ***and*** user-chosen HRF model
    - In terms of stimulus times: $r_n^{(j)} = \sum_{k=1}^{K_j} h(t_n - \tau_k^{(j)})$
    - If stimulus occurs on the imaging TR time-grid, stimulus can be represented as a 0-1 time series: $\left[s_0^{(j)} \quad s_1^{(j)} \quad s_2^{(j)} \quad s_3^{(j)} \quad \cdots \right]$ where $s_k^{(j)}=1$ if stimulus #j is on at time $t=k\cdot\text{TR}$, and $s_k^{(j)}=0$ if #j is off at that time:

$$r_n^{(j)} = h_0 s_n^{(j)} + h_1 s_{n-1}^{(j)} + h_2 s_{n-2}^{(j)} + h_3 s_{n-3}^{(j)} + \cdots = \sum_{q=0}^{p} h_q s_{n-q}^{(j)}$$

Equations: Matrix-Vector Form

- Express ***known*** data vector as a sum of ***known*** columns with ***unknown*** coefficents:



$$\begin{bmatrix} z_0 \\ z_1 \\ z_2 \\ \vdots \\ z_{N-1} \end{bmatrix} \approx \begin{bmatrix} 1 \\ 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} \cdot a + \begin{bmatrix} 0 \\ 1 \\ 2 \\ \vdots \\ N-1 \end{bmatrix} \cdot b + \begin{bmatrix} r_0^{(1)} \\ r_1^{(1)} \\ r_2^{(1)} \\ \vdots \\ r_{N-1}^{(1)} \end{bmatrix} \cdot \beta_1 + \begin{bmatrix} r_0^{(2)} \\ r_1^{(2)} \\ r_2^{(2)} \\ \vdots \\ r_{N-1}^{(2)} \end{bmatrix} \cdot \beta_2 + \cdots$$

or

$$\begin{bmatrix} z_0 \\ z_1 \\ z_2 \\ \vdots \\ z_{N-1} \end{bmatrix} \approx \begin{bmatrix} 1 & 0 & r_0^{(1)} & r_0^{(1)} & \cdots \\ 1 & 1 & r_1^{(1)} & r_1^{(1)} & \cdots \\ 1 & 2 & r_2^{(1)} & r_2^{(1)} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \\ 1 & N-1 & r_{N-1}^{(1)} & r_{N-1}^{(2)} & \cdots \end{bmatrix} \begin{bmatrix} a \\ b \\ \beta_1 \\ \beta_2 \\ \vdots \end{bmatrix}$$

or

'≈' means "least squares"

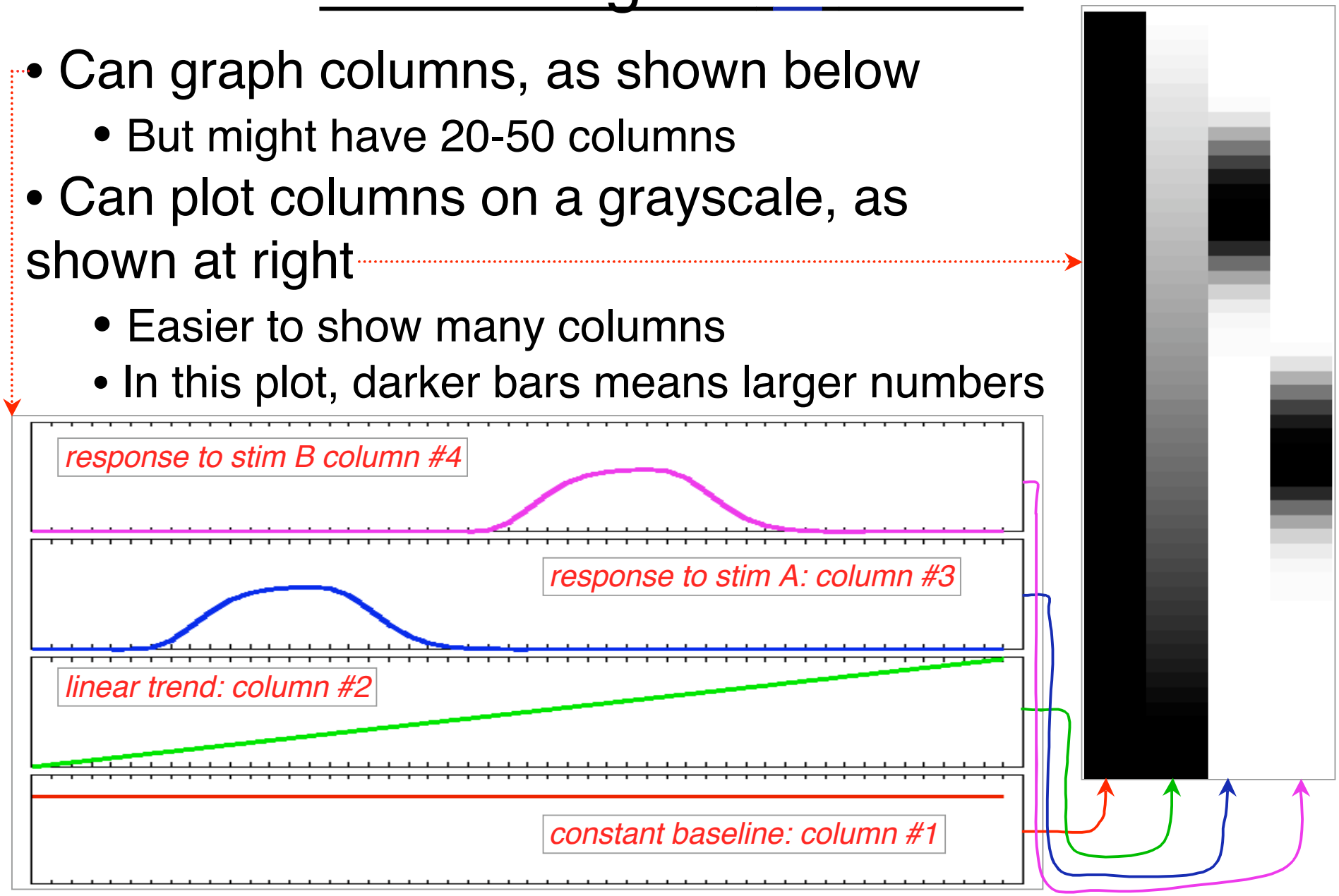
$$\underbrace{\mathbf{z}}_{\text{vector of data}} \approx \underbrace{\mathbf{R}}_{\text{matrix of columns}} \underbrace{\boldsymbol{\beta}}_{\text{vector of coeff}}$$

the "design" matrix

z depends on the voxel; **R** doesn't

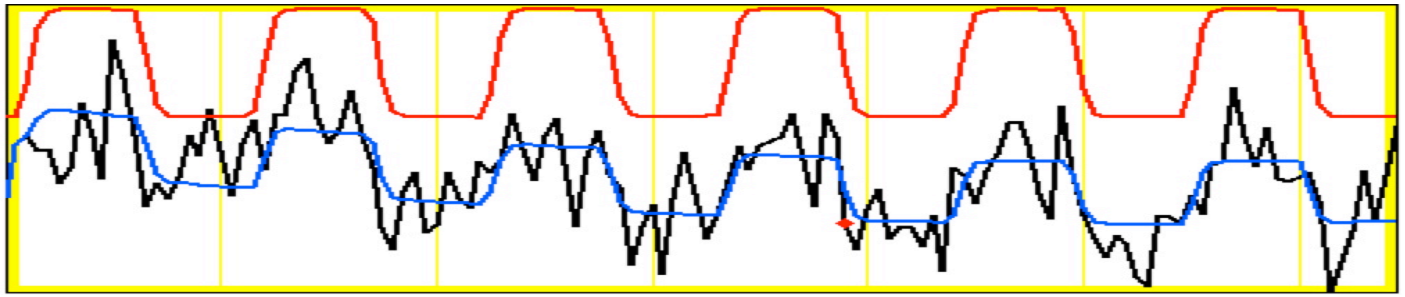
Visualizing the R Matrix

- Can graph columns, as shown below
  - But might have 20-50 columns
- Can plot columns on a grayscale, as shown at right
  - Easier to show many columns
  - In this plot, darker bars means larger numbers



Solving $\mathbf{z} \approx \mathbf{R}\beta$ for β

- Number of equations = number of time points
  - ★ 100s per run, but perhaps 1000s per subject
- Number of unknowns usually in range 5–50
- Least squares solution: $\hat{\beta} = \left[\mathbf{R}^T\mathbf{R}\right]^{-1}\mathbf{R}^T\mathbf{z}$
  - ★ $\hat{\beta}$ denotes an *estimate* of the true (unknown) β
  - ★ From $\hat{\beta}$, calculate $\hat{\mathbf{z}} = \mathbf{R}\hat{\beta}$ as the *fitted model*



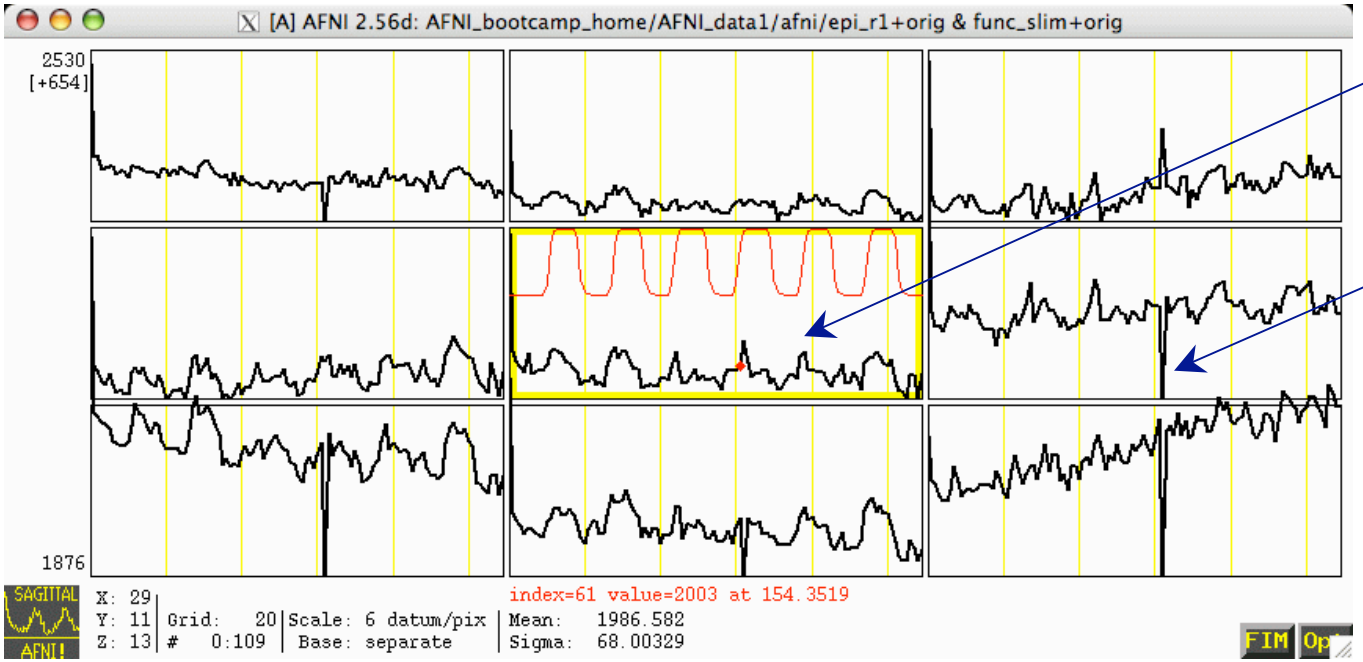
    - o $\mathbf{z} - \hat{\mathbf{z}}$ is the ***residual time series*** = noise (we hope)
- Collinearity: when matrix $\mathbf{R}^T\mathbf{R}$ can't be inverted
  - ★ Near collinearity: when inverse exists but is huge

Simple Regression: Recapitulation

- Choose HRF model $h(t)$ [AKA *fixed-model regression*]
- Build model responses $r_n(t)$ to each stimulus class
  - ⋆ Using $h(t)$ and the stimulus timing
- Choose baseline model time series
  - ⋆ Constant + linear + quadratic + movement?
- Assemble model and baseline time series into the columns of the $\mathbf{R}$ matrix
- For each voxel time series $\mathbf{z}$, solve $\mathbf{z} \approx \mathbf{R}\boldsymbol{\beta}$ for $\hat{\boldsymbol{\beta}}$
- **Individual subject maps**: Test the coefficients in $\hat{\boldsymbol{\beta}}$ that you care about for statistical significance
- **Group maps**: Transform the coefficients in $\hat{\boldsymbol{\beta}}$ that you care about to Talairach space, and perform statistics on these $\hat{\boldsymbol{\beta}}$ values

Sample Data Analysis: Simple Regression

- Enough theory (for now: more to come later!)
- To look at the data: type **`cd AFNI_data1/afni`** ; then **`afni`**
- **`Switch Underlay`** to dataset **`epi_r1`**
  - ★ Then Sagittal **`Image`** and **`Graph`**
  - ★ **`FIM→Pick Ideal`** ; then click **`afni/ideal_r1.1D`** ; then **`Set`**
  - ★ Right-click in image, **`Jump to (ijk)`**, then **`29 11 13`**, then **`Set`**



- Data clearly has activity in sync with reference
- Data also has a big spike, which is annoying
  - Subject head movement!

Preparing Data for Analysis

- Six preparatory steps are possible:
  - ★ Image registration (realignment): program **3dvolreg**
  - ★ Image smoothing: program **3dmerge**
  - ★ Image masking: program **3dClipLevel** or **3dAutomask**
  - ★ Conversion to percentile: programs **3dTstat** and **3dcalc**
  - ★ Censoring out time points that are bad: program **3dToutcount** or **3dTqual**
  - ★ Catenating multiple imaging runs into 1 big dataset: program **3dTcat**

- Not all steps are necessary or desirable in any given case
- In this first example, will only do registration, since the data obviously needs this correction

Data Analysis Script

- In file **epi_r1_decon**:

```
waver -GAM                                  \
      -input epi_r1_stim.1D                 \
      -TR 2.5                               \
      > epi_r1_ideal.1D

3dvolreg -base 2                            \
         -prefix epi_r1_reg                 \
         -1Dfile epi_r1_mot.1D              \
         -verb                              \
         epi_r1+orig

3dDeconvolve                                \
    -input epi_r1_reg+orig                  \
    -nfirst     2                           \
    -num_stimts 1                           \
    -stim_file  1 epi_r1_ideal.1D           \
    -stim_label 1 AllStim                   \
    -tout                                   \
    -bucket epi_r1_func                     \
    -fitts  epi_r1_fitts
```

- **waver** creates model time series from input stimulus timing in file **epi_r1_stim.1D**
- Plot a 1D file to screen with
  1dplot epi_r1_ideal.1D

3dvolreg (3D image registration) will be covered in a later presentation

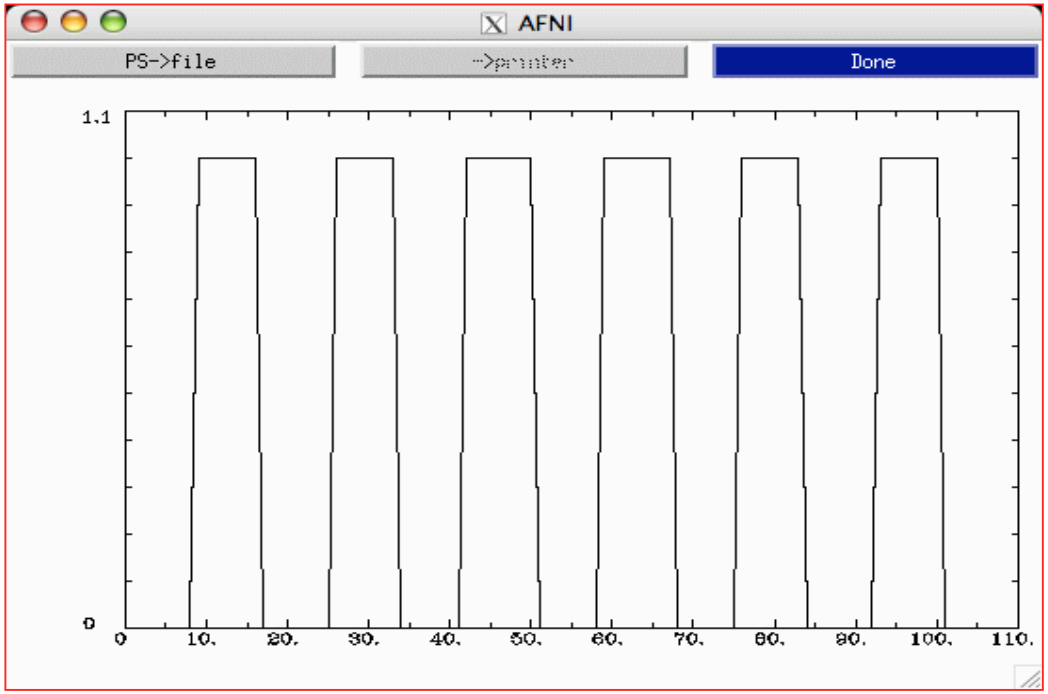
- **3dDeconvolve** = regression code
- ← Name of input dataset
- ← Index of first sub-brick to process
- ← Number of input model time series
- ← Name of first input model time series file
- ← Name for results in AFNI menus
- ← Indicates to output *t*-statistic for β weights
- ← Name of output "bucket" dataset (statistics)
- ← Name of output model fit dataset

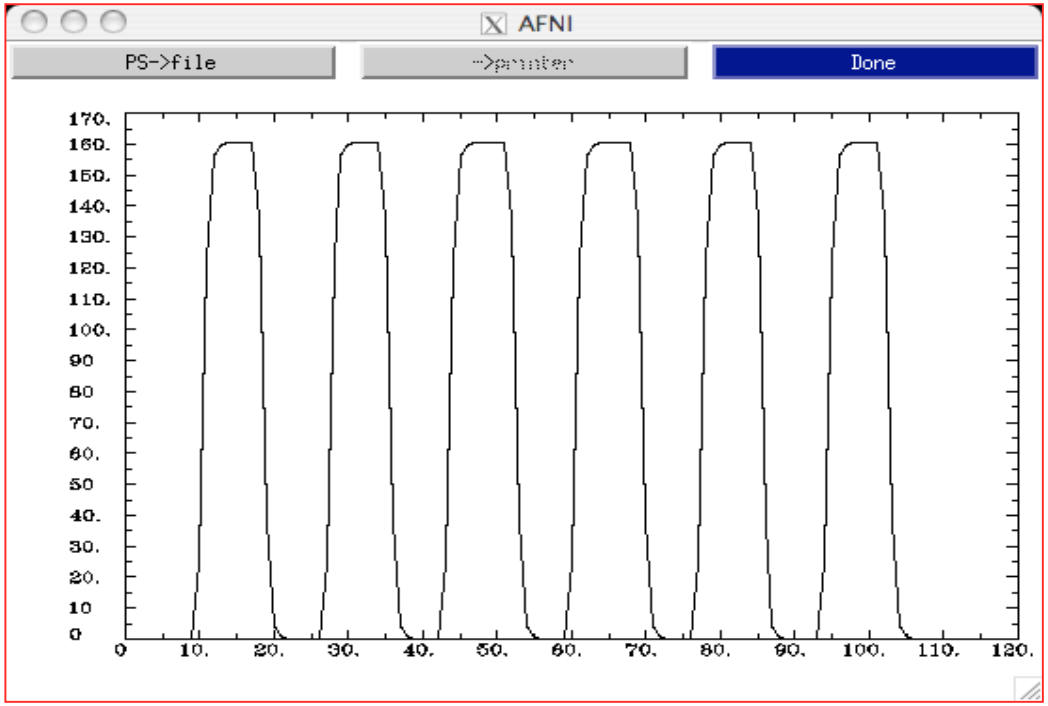
Contents of.1D files

epi_r1_stim.1D **epi_r1_ideal.1D**

```
0                0
0                0
0                0
0                0
0                0
0                0
0                0
0                0
0                0
1                0
1          24.4876
1          122.869
1          156.166
1          160.258
1          160.547
1          160.547
1          160.547
0          160.547
0          136.059
0          37.6781
0          4.38121
0          0.288748
0                0
0                0
...              ...
```

- 1 line per time point
- TR=2.5 s
- 0=stim OFF
- 1=stim ON
- Note that "ideal" is delayed from stimulus
- Graphs at right created with **1dplot**





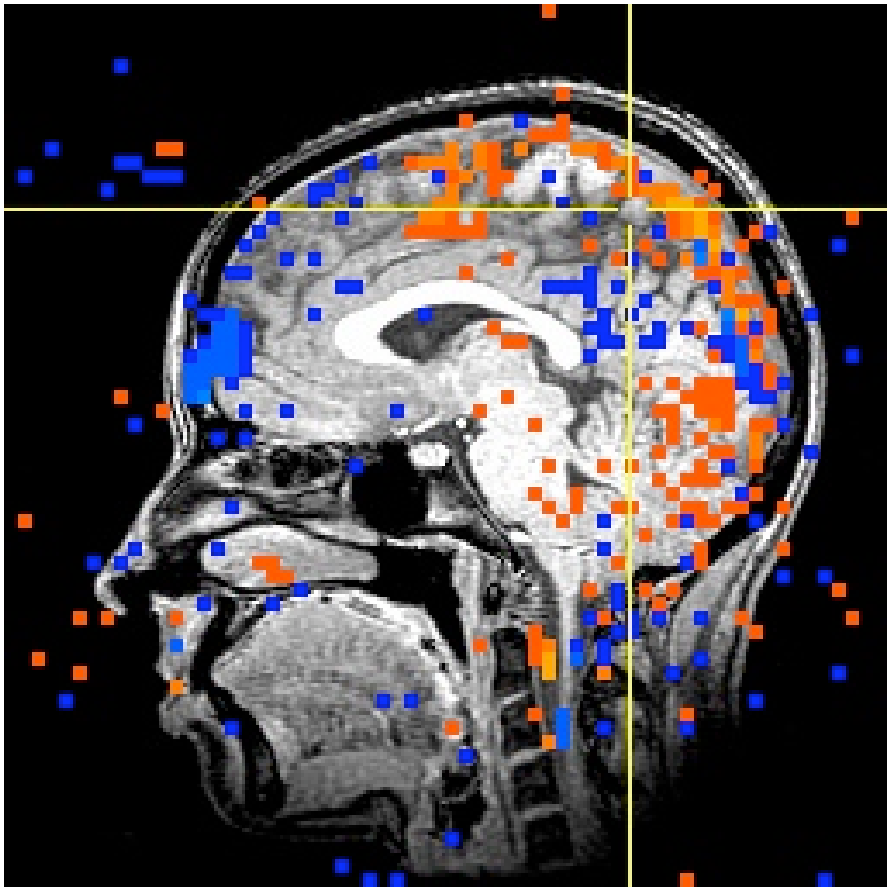
To Run Script and View Results

- type **source epi_r1_decon** ; then wait for programs to run
- type **afni** to view what we've got
  - ★ **Switch Underlay** to epi_r1_reg (output from **3dvolreg**)
  - ★ **Switch Overlay** to epi_r1_func (output from **3dDeconvolve**)
  - ★ **Sagittal Image** and **Graph** viewers
  - ★ **FIM→Ignore→2** to have graph viewer not plot 1st 2 time pts
  - ★ **FIM→Pick Ideal** ; pick **epi_r1_ideal.1D** (output from **waver**)
- Define Overlay to set up functional coloring
  - **Olay→Allstim[0] Coef** (sets coloring to be from model fit β)
  - **Thr→Allstim[0] t-s** (sets threshold to be model fit *t*-statistic)
  - **See Overlay** (otherwise won't see the function!)
  - Play with threshold slider to get a meaningful activation map (e.g., t=4 is a decent threshold)

More Viewing the Results

- Graph viewer: **Opt→Tran 1D→Dataset #N** to plot the model fit dataset output by **3dDeconvolve**
  - Will open the control panel for the **Dataset #N** plugin
  - Click first **Input** on ; then choose **Dataset epi_r1_fitts+orig**
  - Also choose **Color dk-blue** to get a pleasing plot
  - Then click on **Set+Close** (to close the plugin panel)
  - Should now see fitted time series in the graph viewer instead of data time series
  - Graph viewer: click **Opt→Double Plot→Overlay** on to make the fitted time series appear as an overlay curve
  - This tool lets you visualize the quality of the data fit
- Can also now overlay function on MP-RAGE anatomical by using **Switch Underlay** to **anat+orig** dataset
  - Probably won't want to graph the **anat+orig** dataset!

Stimulus Correlated Movement?



- **3dvolreg** saved the motion parameters estimates into file **epi_r1_mot.1D**
- For fun: **1dplot epi_r1_mot.1D**

- Extensive “activation” (i.e., correlation of data time series with model time series) along the top of the brain is an indicator of stimulus correlated motion artifact
- Can remain even after registration, due to errors in registration, magnetic field inhomogeneities, etc.
- Can be partially removed by using the estimated movement history (from **3dvolreg**) as additional baseline model functions

Removing Residual Motion Artifacts

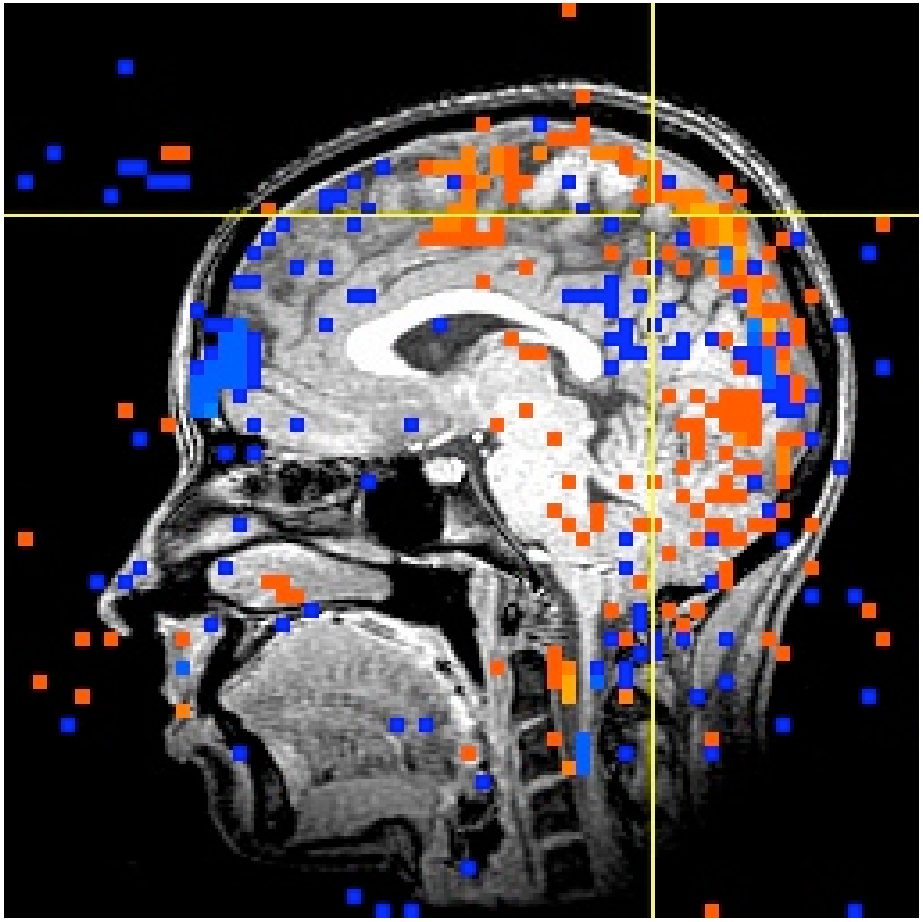
- Last part of script **epi_r1_decon**:

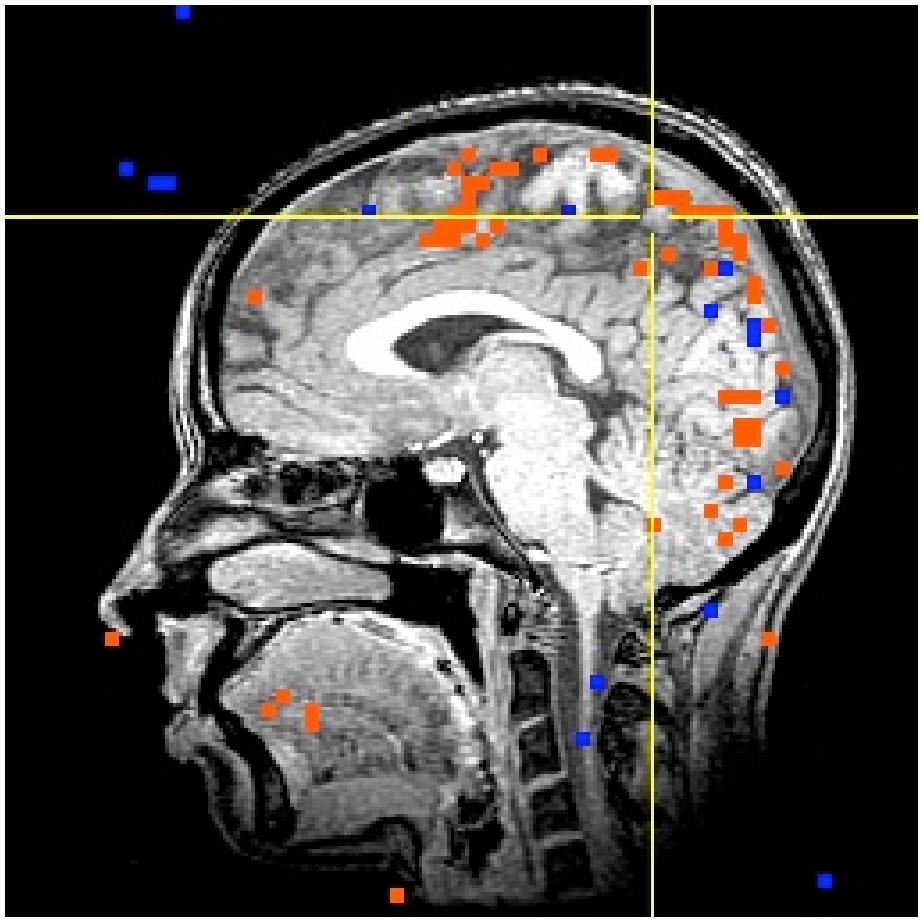
```
3dDeconvolve                                \
    -input epi_r1_reg+orig                  \
    -nfirst     2                           \
    -num_stimts 7                           \
    -stim_file  1 epi_r1_ideal.1D           \
    -stim_label 1 AllStim                   \
    -stim_file  2 epi_r1_mot.1D'[0]'        \
    -stim_base  2                           \
    -stim_file  3 epi_r1_mot.1D'[1]'        \
    -stim_base  3                           \
    -stim_file  4 epi_r1_mot.1D'[2]'        \
    -stim_base  4                           \
    -stim_file  5 epi_r1_mot.1D'[3]'        \
    -stim_base  5                           \
    -stim_file  6 epi_r1_mot.1D'[4]'        \
    -stim_base  6                           \
    -stim_file  7 epi_r1_mot.1D'[5]'        \
    -stim_base  7                           \
    -tout                                   \
    -bucket epi_r1_func_mot                 \
    -fitts  epi_r1_fitts_mot
```

These new lines add 6 regressors to the model and assign them to the baseline (**-stim_base** option)

Output files: take a moment to look at results

Some Results: Before and After





No: movement parameters are not in baseline model

Yes: movement parameters are in baseline model

t-statistic threshold set to a *p*-value of 10^{-4} in both images

Multiple Stimulus Classes

- The experiment analyzed here in fact is more complicated
  - ★ There are 4 related visual stimulus types
  - ★ One goal is to find areas that are differentially activated between these different types of stimuli
  - ★ We have 4 imaging runs, 108 useful time points each (skipping first 2 in each run) that we will analyze together
    - o Already registered and put together into dataset **rall_vr+orig**
  - ★ Stimulus timing files are in subdirectory **stim_files/**
  - ★ Script file **waver_ht2** will create HRF models for regression:

```
cd stim_files
waver -dt 2.5 -GAM -input scan1to4a.1D > scan1to4a_hrf.1D
waver -dt 2.5 -GAM -input scan1to4t.1D > scan1to4t_hrf.1D
waver -dt 2.5 -GAM -input scan1to4h.1D > scan1to4h_hrf.1D
waver -dt 2.5 -GAM -input scan1to4l.1D > scan1to4l_hrf.1D
cd ..
```

  - ★ Type **source waver_ht2** to run this script
    - o Might also use **1dplot** to check if things are reasonable

Regression with Multiple Model Files

- Script file **decon_ht2** does the job:

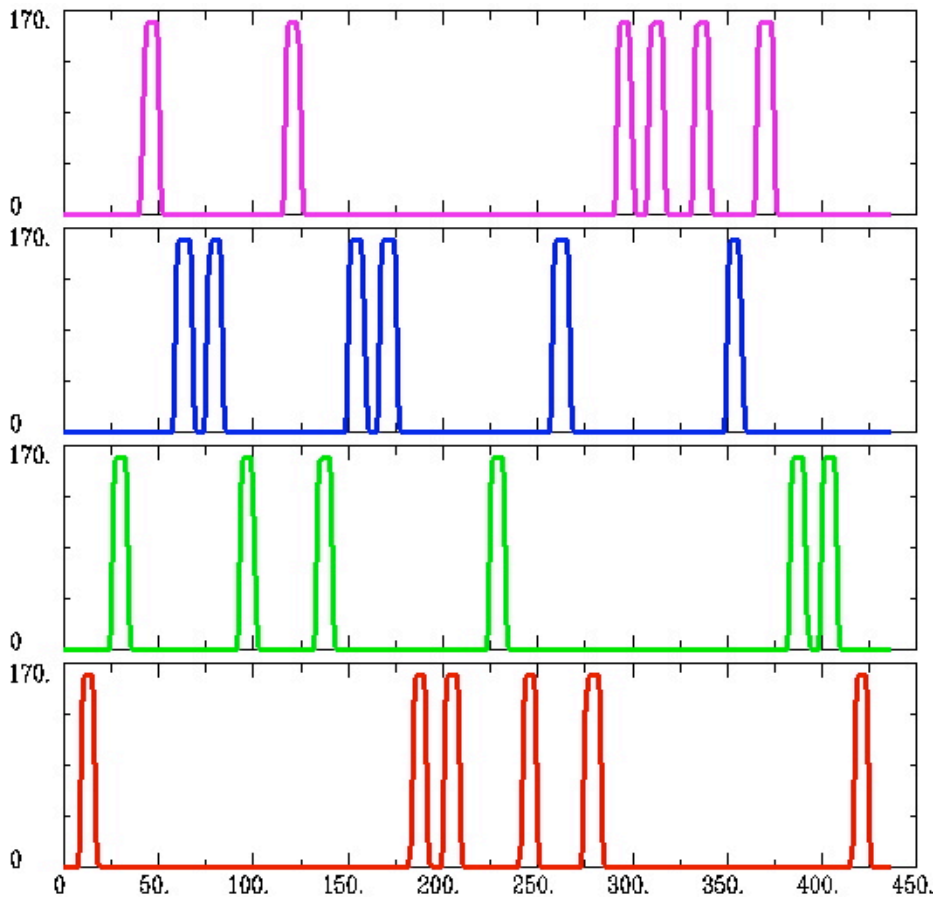
```
3dDeconvolve -xout -input rall_vr+orig                                    \
    -num_stimts 4                                                         \
    -stim_file 1 stim_files/scan1to4a_hrf.1D  -stim_label 1 Actions       \
    -stim_file 2 stim_files/scan1to4t_hrf.1D  -stim_label 2 Tool          \
    -stim_file 3 stim_files/scan1to4h_hrf.1D  -stim_label 3 HighC         \
    -stim_file 4 stim_files/scan1to4l_hrf.1D  -stim_label 4 LowC          \
    -concat contrasts/runs.1D                                             \
    -glt 1 contrasts/contr_AvsT.txt   -glt_label 1 AvsT                   \
    -glt 1 contrasts/contr_HvsL.txt   -glt_label 2 HvsL                   \
    -glt 1 contrasts/contr_ATvsHL.txt -glt_label 3 ATvsHL                 \
    -full_first -fout -tout                                               \
    -bucket func_ht2
```

- Run this script by typing **source decon_ht2** (takes a few minutes)

- Stim #1 = visual presentation of active movements
- Stim #2 = visual presentation of simple (tool-like) movements
- Stims #3 and #4 = high and low contrast gratings

Regressors for This Script

via **1dgrayplot**



via **1dplot**

Extra Features of 3dDeconvolve - 1

`-concat contrasts/runs.1D` = file that indicates where new imaging runs start

```
0
108
216
324
```

`-full_first` = put **full model** statistic first in output file, not last

`-fout -tout` = output both *F*- and *t*-statistics

- The full model statistic is an *F*-statistic that shows how well the sum of all 4 input model time series fits voxel time series data
- The individual models also will get individual *F*- and *t*-statistics indicating the significance of their individual contributions to the time series fit
  - ★ i.e., F_{Actions} tells if model (**Actions+HighC+LowC+Tools+baseline**) explains more of the data variability than model (**HighC+LowC+Tools+baseline**)

Extra Features of 3dDeconvolve - 2

```
-glt 1 contrasts/contr_AvsT.txt   -glt_label 1 AvsT
-glt 1 contrasts/contr_HvsL.txt   -glt_label 2 HvsL
-glt 1 contrasts/contr_ATvsHL.txt -glt_label 3 ATvsHL
```

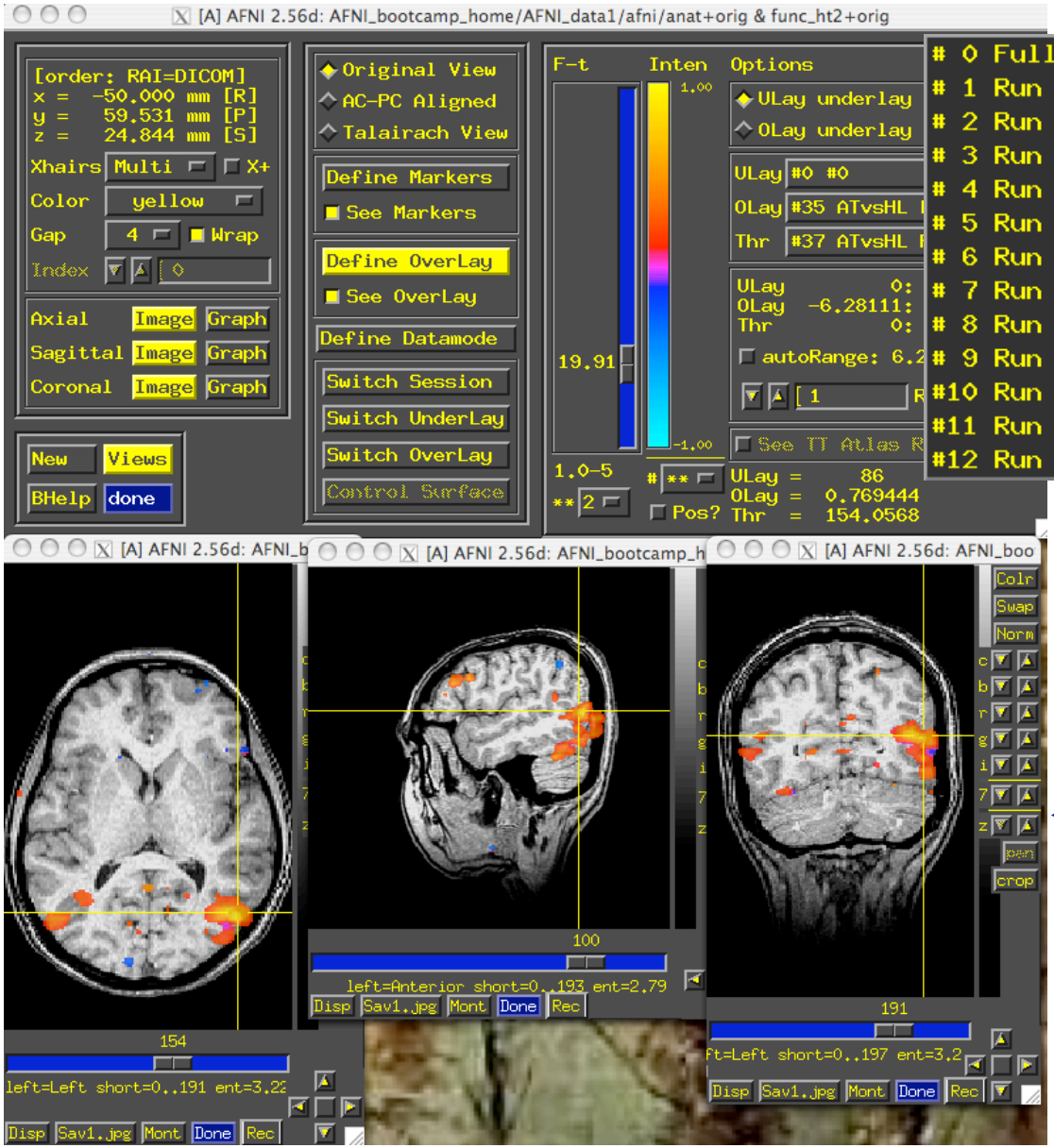
- **GLT**s are General Linear Tests
- **3dDeconvolve** provides tests for each regressor separately, but if you want to test combinations or contrasts of the β weights in each voxel, you need the **-glt** option
- File **contrasts/contr_AvsT.txt** = **0 0 0 0 0 0 0 0 1 -1 0 0** (one line with 12 numbers)

- Goal is to test a linear combination of the β weights
  - ★ In this data, we have 12 β weights: 8 baseline parameters (2 per imaging run), which are first in the β vector, and 4 regressor magnitudes, which are from **-stim_file** options
  - ★ This particular test contrasts the Actions and Tool βs
    - tests if $\beta_{Actions} - \beta_{Tool} \neq 0$

Extra Features of 3dDeconvolve - 3

- File `contrasts/contr_HvsL.txt` = **0 0 0 0 0 0 0 0 0 0 1 -1**
  - Goal is to test if $\beta_{HighC} - \beta_{LowC} \neq 0$
- File `contrasts/contr_ATvsHL.txt` = **0 0 0 0 0 0 0 0 1 1 -1 -1**
  - Goal is to test if $(\beta_{Actions} + \beta_{Tool}) - (\beta_{HighC} + \beta_{LowC}) \neq 0$
  - Regions where this statistic is significant will have had different amounts of BOLD signal change in the activity viewing tasks versus the grating viewing tasks
    - This is a way to factor out primary visual cortex
- `-glt_label 3 ATvsHL` option is used to attach a meaningful label to the resulting statistics sub-bricks

Results of **decon_ht2** Script



- Menu showing labels from **3dDeconvolve** run
- Images showing results from third contrast: **ATvsHL**
- Play with this yourself to get a feel for it

Statistics from 3dDeconvolve

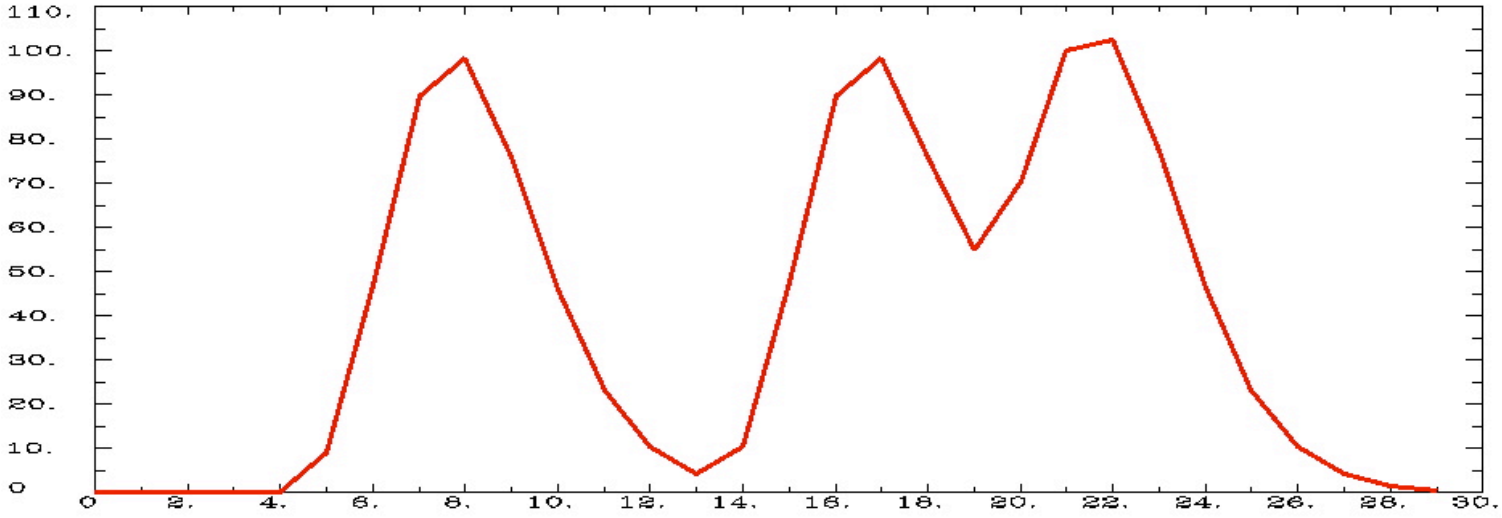
- An *F*-statistic measures significance of how much a model component reduced the variance of the time series data
- Full *F* measures how much the signal regressors reduced the variance over just the baseline regressors (**sub-brick #0 below**)
- Individual partial-model *F*s measures how much each individual signal regressor reduced data variance over the full model with that regressor excluded (**sub-bricks #19, #22, #25, and #28 below**)

- The **Coef** sub-bricks are the β weights (e.g., **#17, #20, #23, #26**)
- A *t*-statistic sub-brick measure impact of one coefficient

```
# 0 Full F-stat     #13 Run #4 t^0 Coe #26 LowC[0] Coef
# 1 Run #1 t^0 Coe  #14 Run #4 t^0 t-s #27 LowC[0] t-st
# 2 Run #1 t^0 t-s  #15 Run #4 t^1 Coe #28 LowC F-stat
# 3 Run #1 t^1 Coe  #16 Run #4 t^1 t-s #29 AvsT LC[0] coe
# 4 Run #1 t^1 t-s  #17 Actions[0] Coe #30 AvsT LC[0] t-s
# 5 Run #2 t^0 Coe  #18 Actions[0] t-s #31 AvsT F-stat
# 6 Run #2 t^0 t-s  #19 Actions F-stat #32 HvsL LC[0] coe
# 7 Run #2 t^1 Coe  #20 Tool[0] Coef   #33 HvsL LC[0] t-s
# 8 Run #2 t^1 t-s  #21 Tool[0] t-st   #34 HvsL F-stat
# 9 Run #3 t^0 Coe  #22 Tool F-stat    #35 ATvsHL LC[0] c
#10 Run #3 t^0 t-s  #23 HighC[0] Coef  #36 ATvsHL LC[0] t
#11 Run #3 t^1 Coe  #24 HighC[0] t-st  #37 ATvsHL F-stat
#12 Run #3 t^1 t-s  #25 HighC F-stat
```

Alternative Way to Run waver

- Instead of giving stimulus timing on the TR-grid as a set of 0s and 1s
- Can give the actual stimulus times (in seconds) using the **`-tstim`** option
  - ★ **`waver -dt 1.0 -GAM -tstim 3 12 17 | 1dplot -stdin`**



- If times are in a file, can use **`-tstim` `` `cat filename` ``** to place them on the command line after **`-tstim`** option
  - ★ This is most useful for event-related experiments

Note backward single quotes

Deconvolution Signal Models

- Simple or Fixed-shape regression:
  - ★ We fixed the shape of the HRF
  - ★ Used waver to generate the signal model from the stimulus timing
  - ★ Found the amplitude of the signal model in each voxel
- Deconvolution or Variable-shape regression:
  - ★ We allow the shape of the HRF to vary in each voxel, for each stimulus class
  - ★ Appropriate when you don't want to over-constrain the solution by assuming an HRF shape
  - ★ ***Caveat***: need to have enough time points during the HRF in order to resolve its shape

Deconvolution: Pros and Cons

+ Letting HRF shape varies allows for subject and regional variability in hemodynamics
+ Can test HRF estimate for different shapes; e.g., are later time points more “active” than earlier?
– Need to estimate more parameters for each stimulus class than a fixed-shape model (e.g., 4-15 *vs.* 1 parameter=amplitude of HRF)
– Which means you need more data to get the same statistical power (assuming that the fixed-shape model you would otherwise use was in fact “correct”)
– Freedom to get any shape in HRF results can give weird shapes that are difficult to interpret

Expressing HRF via Regression Unknowns

- The tool for expressing an unknown function as a finite set of numbers that can be fit via linear regression is an **expansion in basis functions**

$$h(t) = \beta_0 \psi_0(t) + \beta_1 \psi_1(t) + \beta_2 \psi_2(t) + \cdots = \sum_{q=0}^{q=p} \beta_q \psi_q(t)$$

  - ★ The basis functions $\psi_q(t)$ are known, as is the expansion order $\boldsymbol{p}$
  - ★ The unknowns to be found (in each voxel) comprises the set of weights $\boldsymbol{\beta_q}$ for each $\psi_q(t)$

- Since β weights appear only by multiplying known values, and HRF only appears in final signal model by linear convolution, resulting signal model is still solvable by linear regression

Basis Function: “Sticks”

- The set of basis functions you use determines the range of possible HRFs that you can compute
- “Stick” (or Dirac delta) functions are very flexible
  - ★ But they come with a strict limitation
- $\delta(t)$ is 1 at t=0 and is 0 at all other values of t
- $\psi_q(t) = \delta(t - q \cdot \mathrm{TR})$ for q=0,1,2,…,p

$\Rightarrow h(0) = \beta_0$

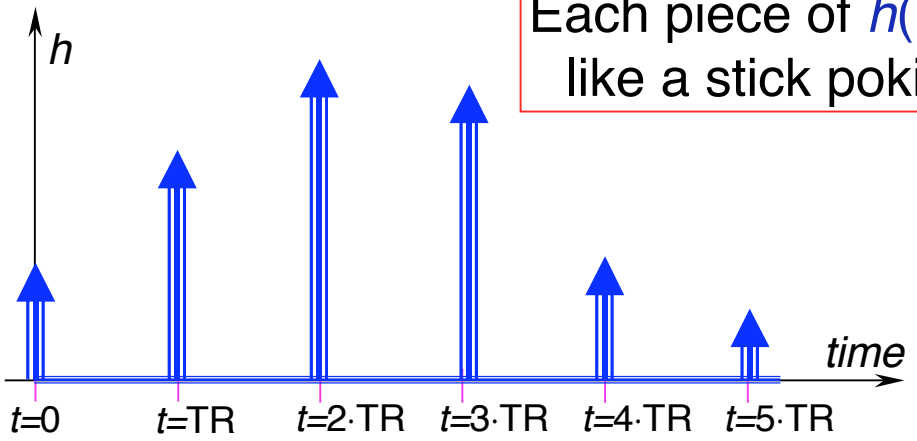
$\Rightarrow h(\mathrm{TR}) = \beta_1$

$\Rightarrow h(2 \cdot \mathrm{TR}) = \beta_2$

$\Rightarrow h(3 \cdot \mathrm{TR}) = \beta_3$

$\Rightarrow$ et cetera

$\Rightarrow h(t) = 0$ for any t not on the TR grid



Each piece of $h(t)$ looks like a stick poking up

Sticks: Good Points

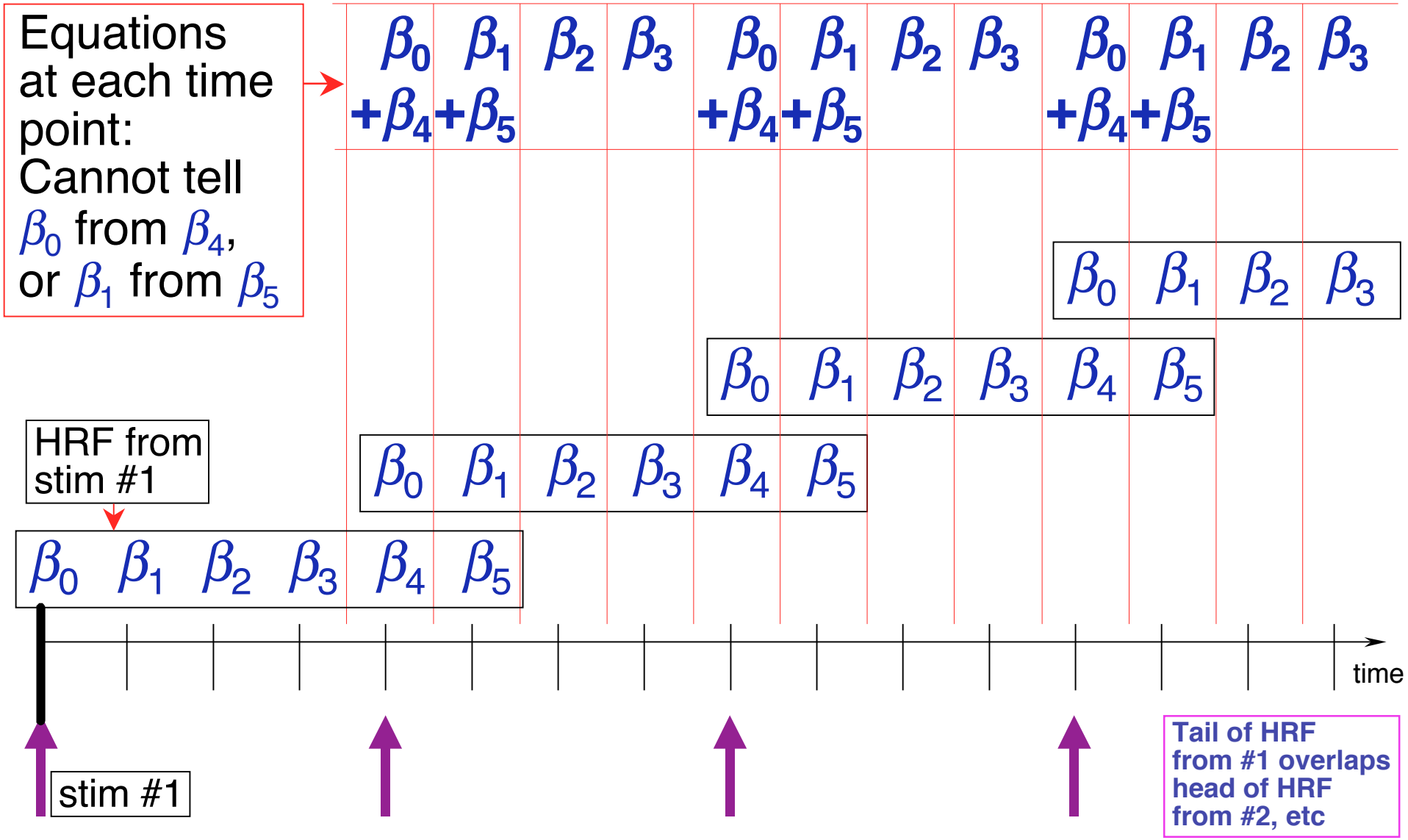
- Can represent arbitrary shapes of the HRF, up and down, with ease
- Meaning of each β_q is completely obvious
  - ⋆ Value of HRF at time lag $q \cdot \mathrm{TR}$ after activation
- **3dDeconvolve** is set up to deal with stick functions for representing HRF, so using them is very easy
  - What is called p here is given by command line option `-stim_maxlag` in the program
  - When choosing p, rule is to estimate longest duration of neural activation after stimulus onset, then add 10-12 seconds to allow for slowness of hemodynamic response

Sticks and TR-locked Stimuli

- **$h(t) = 0$ for any t not on the TR grid**
- This limitation means that, using stick functions as our basis set, we can only model stimuli that are "locked" to the TR grid
  - ★ That is, stimuli/activations don't occur at fully general times, but only occur at integer multiples of TR
- For example, suppose an activation is at $t = 1.7 \cdot \mathrm{TR}$
  - ★ We need to model the response at later times, such as $2 \cdot \mathrm{TR}$, $3 \cdot \mathrm{TR}$, etc., so need to model $h(t)$ at times such as $t = (2 - 1.7) \cdot \mathrm{TR} = 0.3 \cdot \mathrm{TR}$, $t = 1.3 \cdot \mathrm{TR}$, etc., after the stimulus
  - But the stick function model doesn't allow for such intermediate times
    - ***or***, can allow Δt for sticks to be a fraction of TR for data
    - e.g., $\Delta t = \mathrm{TR}/2$, which implies twice as many β_q parameters to cover the same time interval (time interval needed is set by hemodynamics)
    - then would allow stimuli that occur on TR-grid or halfway in-between

Deconvolution and Collinearity

- Regular stimulus timing can lead to collinearity!



3dDeconvolve with Stick Functions

- Instead of inputting a signal model time series (e.g., created with **`waver`** and stimulus timing), you input the stimulus timing directly
  - ★ Format: a text file with 0s and 1s, 0 at TR-grid times with no stimulus, 1 at time with stimulus
- Must specify the maximum lag (in units of TR) that we expect HRF to last after each stimulus
  - ★ This requires you to make a judgment about the activation — brief or long?
- **`3dDeconvolve`** returns estimated values for each β_q for each stimulus class
  - ★ Usually then use a GLT to test the HRF (or pieces of it) for significance

Extra Features of 3dDeconvolve - 4

- **`-stim_maxlag k p`** = option to set the maximum lag to **`p`** for stimulus timing file #**`k`** for **`k=0,1,2,...`**
  - ★ Stimulus timing file input using command line option **`-stim_file k filename`** as before
  - ★ Can also use **`-stim_minlag k m`** option to set the minimum lag if you want a value **`m`** different from **`0`**
  - ★ In which case there are **`p-m+1`** parameters in this HRF
- **`-stim_nptr k r`** = option to specify that there are **`r`** stimulus subintervals per TR, rather than just 1
  - ★ This feature can be used to get a finer grained HRF, at the cost of adding more parameters that need to be estimated
  - Need to make sure that the input stimulus timing file (from **`-stim_file`**) has **`r`** entries per TR
  - TR for **`-stim_file`** and for output HRF is data TR **`÷ r`**

Script for Deconvolution - The Data

- **`cd AFNI_data2`**
  - ★ data is in **`ED/`** subdirectory (10 runs of 136 images, TR=2 s)
  - ★ script in file **`@analyze_ht05`**
    - o stimuli timing and GLT contrast files in **`misc_files/`**
  - ★ start script ***now*** by typing **`source @analyze_ht05`**
    - o will discuss details of script while it runs
- This is an event-related study from Mike Beauchamp (LBC/NIMH)
  - ★ Four classes of stimuli (short videos)
    - o Tools moving (e.g., a hammer pounding) - **TM**
    - o People moving (e.g., jumping jacks) - **HM**
    - o Points outlining tools moving (no objects, just points) - **TP**
    - o Points outlining people moving - **HP**
  - ★ Goal is to find if there is an area that distinguishes natural motions (HM and HP) from simpler rigid motions (TM and TP)

Script for Deconvolution - Outline

- Registration of each imaging run (there are 10): `3dvolreg`
- Smooth each volume in space (136 sub-bricks per run): `3dmerge`
- Create a brain mask: `3dAutomask` and `3dcalc`
- Rescale each voxel time series in each imaging run so that its average through time is 100: `3dTstat` and `3dcalc`
  - ⋆ If baseline is 100, then a β_q of 5 (say) indicates a 5% signal change in that voxel at time laq #q after stimulus
- Catenate all imaging runs together into one big dataset (1360 time points): `3dTcat`
- Compute HRFs and statistics: `3dDeconvolve`
  - ⋆ Each HRF will have 15 output points (lags from 0 to 14) with a TR of 1.0 s, since the input data has a TR of 2.0 s and we will be using the `-stim_nptr k r` option with `r=2`
- Average together central points 4..9 of each separate HRF to get peak % change in each voxel: `3dTstat`

Script for Deconvolution - 1

```
#!/bin/tcsh

if ( $#argv > 0 ) then
    set subjects = ( $argv )
else
    set subjects = ED
endif
```

This script is designed to run analyses on a lot of subjects at once. We will only analyze the ED data here. The other subjects will be included in the Group Analysis presentation.

```
#=================================================================
# Above command will run script for all our subjects - ED, EE, EF - one after
# the other if, when we execute the script, we type: ./@analyze_ht05 ED EE EF.
# If we type ./@analyze_ht05 or tcsh @analyze_ht05, it'll run the script only
# for subject ED.  The user will then have to go back and edit the script so
# that 'set subjects' = EE and then EF, and then run the script for each subj.
#=================================================================

foreach subj ($subjects)

    cd $subj
```

Loop over subjects

First step is to change to the directory that has this subject's data

Script for Deconvolution - 2

```
#=======================================================
# volume register and time shift our datasets, and remove the first
# two time points
#=======================================================

foreach run ( `count -digits 1 1 10` )
   3dvolreg -verbose                              \
       -base {$subj}_r{$run}+orig'[2]' \
       -tshift 0                                  \
       -prefix  {$subj}_r{$run}_vr      \
       {$subj}_r{$run}+orig'[2..137]'

# will store run data in runs_orig directory

#=======================================================
# smooth data with 3dmerge.
#=======================================================

   3dmerge -1blur_rms 4                          \
           -doall                                \
           -prefix {$subj}_r{$run}_vr_bl \
           {$subj}_r{$run}_vr+orig
end
```

Loop over imaging runs 1..10

Image registration of each run to its #2 sub-brick

Lightly blur each dataset to reduce noise and increase functional overlap between runs and subjects

End of loop over imaging runs

Script for Deconvolution - 3

```
#=========================================================
# create masks for each run using 3dAutomask
#=========================================================

foreach run ( `count -digits 1 1 10` )
    3dAutomask -prefix mask_r{$run} {$subj}_r{$run}_vr_bl+orig
end

#=========================================================
# create a mask enveloping masks of the individual runs
#=========================================================

3dcalc -a mask_r1+orig -b mask_r2+orig -c mask_r3+orig    \
       -d mask_r4+orig -e mask_r5+orig -f mask_r6+orig    \
       -g mask_r7+orig -h mask_r8+orig -i mask_r9+orig    \
       -j mask_r10+orig                                   \
       -expr 'step(a+b+c+d+e+f+g+h+i+j)'                  \
       -prefix full_mask
```

Loop over imaging runs 1..10

This mask dataset will be 1 inside the largest contiguous high intensity EPI region, and 0 outside that region — this makes a brain mask

Script for Deconvolution - 4

```
#=========================================================================
# re-scale each run's baseline to 100.
# If baseline is 100, and result of 3dcalc on one voxel is 106, then
# we can say that at that voxel shows a 6% increase is signal activity
# relative to baseline.
# Use full_mask to remove non-brain
#=========================================================================

foreach run ( `count -digits 1 1 10` )
    3dTstat -prefix mean_r{$run} {$subj}_r{$run}_vr_bl+orig

    3dcalc -a {$subj}_r{$run}_vr_bl+orig \
           -b mean_r{$run}+orig          \
           -c full_mask+orig             \
           -expr "(a/b * 100) * c"       \
           -prefix scaled_r{$run}

    /bin/rm mean_r{$run}+orig*
end
```

Mean of the runth dataset, through time: run=1..10

- Divide each voxel value (‘**a**’) by its temporal mean (‘**b**’) and scale by 100
- Result will have temporal mean of 100
- Voxels not in the mask will be set to 0 (by ‘**c**’)

Script for Deconvolution - 5

```
#=========================================================================
# Now we can concatenate our 10 normalized runs with 3dTcat.
#=========================================================================

 3dTcat -prefix {$subj}_all_runs      \
     scaled_r1+orig scaled_r2+orig    \
     scaled_r3+orig scaled_r4+orig    \
     scaled_r5+orig scaled_r6+orig    \
     scaled_r7+orig scaled_r8+orig    \
     scaled_r9+orig scaled_r10+orig

#=========================================================================
# move unneeded run data into separate directories
#=========================================================================

mkdir runs_orig runs_temp

mv {$subj}_r*_vr* scaled* runs_temp
mv {$subj}_r* runs_orig
```

"Gluing" the runs together, since `3dDeconvolve` only operates on one input dataset at a time

Gets this stuff out of the way so that we don't see it when we run AFNI later

Script for Deconvolution - 6

```
#=======================================================================
# run deconvolution analysis
#=======================================================================

3dDeconvolve -input {$subj}_all_runs+orig -num_stimts 4                  \
   -stim_file 1 ../misc_files/all_stims.1D'[0]' -stim_label 1 ToolMov \
            -stim_minlag 1 0 -stim_maxlag 1 14 -stim_nptr 1 2           \
   -stim_file 2 ../misc_files/all_stims.1D'[1]' -stim_label 2 HumanMov\
            -stim_minlag 2 0 -stim_maxlag 2 14 -stim_nptr 2 2           \
   -stim_file 3 ../misc_files/all_stims.1D'[2]' -stim_label 3 ToolPnt \
            -stim_minlag 3 0 -stim_maxlag 3 14 -stim_nptr 3 2           \
   -stim_file 4 ../misc_files/all_stims.1D'[3]' -stim_label 4 HumanPnt\
            -stim_minlag 4 0 -stim_maxlag 4 14 -stim_nptr 4 2           \
   -glt 4 ../misc_files/contrast1.1D  -glt_label 1 FullF               \
   -glt 1 ../misc_files/contrast2.1D  -glt_label 2 HvsT                \
   -glt 1 ../misc_files/contrast3.1D  -glt_label 3 MvsP                \
   -glt 1 ../misc_files/contrast4.1D  -glt_label 4 HMvsHP              \
   -glt 1 ../misc_files/contrast5.1D  -glt_label 5 TMvsTP              \
   -glt 1 ../misc_files/contrast6.1D  -glt_label 6 HPvsTP              \
   -glt 1 ../misc_files/contrast7.1D  -glt_label 7 HMvsTM              \
   -iresp 1 TMirf -iresp 2 HMirf -iresp 3 TPirf -iresp 4 HPirf         \
   -full_first -fout -tout -nobout -polort 2                           \
   -concat ../misc_files/runs.1D                                       \
   -progress 1000                                                      \
   -bucket {$subj}_func
```

Script for Deconvolution - 6a

```
3dDeconvolve -input {$subj}_all_runs+orig -num_stimts 4                    \
   -stim_file 1 ../misc_files/all_stims.1D'[0]' -stim_label 1 ToolMov \
            -stim_minlag 1 0 -stim_maxlag 1 14 -stim_nptr 1 2              \
   -stim_file 2 ../misc_files/all_stims.1D'[1]' -stim_label 2 HumanMov\
            -stim_minlag 2 0 -stim_maxlag 2 14 -stim_nptr 2 2              \
   -stim_file 3 ../misc_files/all_stims.1D'[2]' -stim_label 3 ToolPnt \
            -stim_minlag 3 0 -stim_maxlag 3 14 -stim_nptr 3 2              \
   -stim_file 4 ../misc_files/all_stims.1D'[3]' -stim_label 4 HumanPnt\
            -stim_minlag 4 0 -stim_maxlag 4 14 -stim_nptr 4 2              \
```

- Input dataset is the catenated thing created earlier
- There are 4 time series models
- All stimuli time series are in one file with 4 columns: **`../misc_files/all_stims.1D`**
  - The selectors like '**`[2]`**' pick out a particular column
  - Each stimulus and HRF will be sampled at TR/2 = 1.0 s, due to the use of **`-stim_nptr k 2`** for each **`k`**
  - Lag from 0 to 14 is about right for hemodynamic response to a brief stimulus

Script for Deconvolution - 6b

```
-glt 4 ../misc_files/contrast1.1D  -glt_label 1 FullF          \
-glt 1 ../misc_files/contrast2.1D  -glt_label 2 HvsT           \
-glt 1 ../misc_files/contrast3.1D  -glt_label 3 MvsP           \
-glt 1 ../misc_files/contrast4.1D  -glt_label 4 HMvsHP         \
-glt 1 ../misc_files/contrast5.1D  -glt_label 5 TMvsTP         \
-glt 1 ../misc_files/contrast6.1D  -glt_label 6 HPvsTP         \
-glt 1 ../misc_files/contrast7.1D  -glt_label 7 HMvsTM         \
```

- Run many GLTs to contrast various pairs and quads of cases
  - Each case has 15 points in its HRF, so each GLT needs 60 inputs indicating how to combine all these β weights
  - Plus 3 zero inputs per imaging run (30 more inputs) to skip over the β weights for the baseline parameters
- One example: HvsT (**contrast2.1D**) - all one line in the file!

```
 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 \
 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 \
-0 -0 -0 -1 -1 -1 -1 -1 -1 -1 -0 -0 -0 -0 -0 \
 0  0  0  1  1  1  1  1  1  1  0  0  0  0  0 \
-0 -0 -0 -1 -1 -1 -1 -1 -1 -1 -0 -0 -0 -0 -0 \
 0  0  0  1  1  1  1  1  1  1  0  0  0  0  0
```

skip 30 baseline parameters

-TM: 3..9 seconds
+HM: 3..9 seconds
-TP: 3..9 seconds
+HP: 3..9 seconds

Script for Deconvolution - 6c

```
-iresp 1 TMirf -iresp 2 HMirf -iresp 3 TPirf -iresp 4 HPirf        \
-full_first -fout -tout -nobout -polort 2                          \
-concat ../misc_files/runs.1D                                      \
-progress 1000                                                     \
-bucket {$subj}_func
```

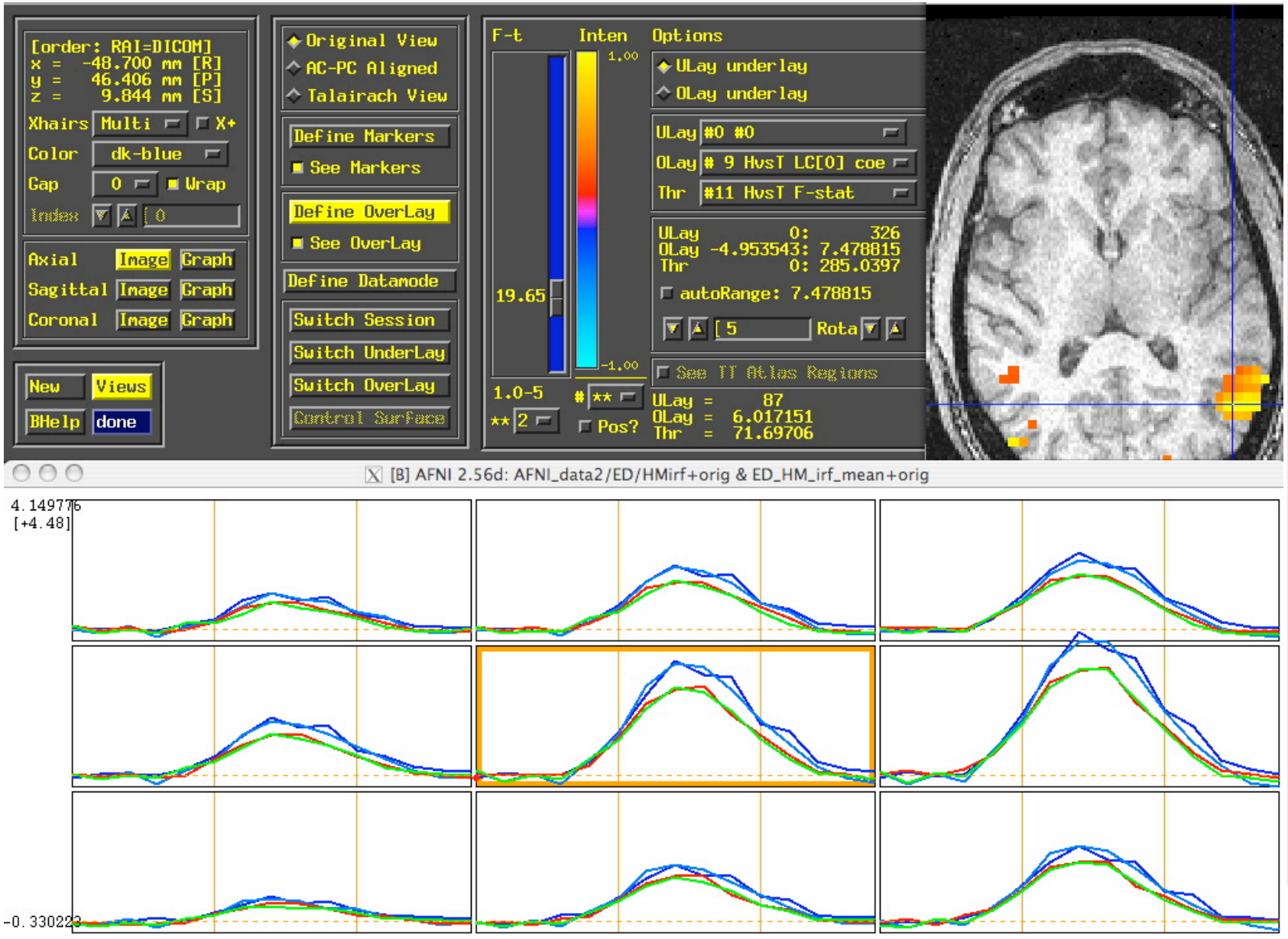
- Output HRF (**-iresp**) 3D+time dataset for each stimulus class
  - Each of these datasets will have TR=1.0 s and have 15 time points (lags 0..14)
  - Can plot them atop each other using **Dataset#N** plugin
- **-nobout** = don't output statistics of baseline parameters
- **-polort 2** = use a quadratic polynomial (3 parameters) for the baseline in each run
- **-concat** … = use this file to indicate when each run starts
- **-progress 1000** = display some results every 1000th voxel
- **-bucket** … = save statistics into dataset with this prefix

Script for Deconvolution - 7

```
#=========================================================================
# make slim dataset.  Too many sub-bricks in our bucket dataset.
# Use 3dbucket to slim it down and include sub-bricks of interest only.
#=========================================================================

3dbucket -prefix {$subj}_func_slim -fbuc {$subj}_func+orig'[125..151]'

#=========================================================================
# Remember IRF datasets created by 3dDeconvolve?
# There are 15 time lags in each voxel.  Remove lags 0-3 and 10-15 b/c not
# interesting.  Then find mean percent signal change for lags 4-9 in each
# voxel with '3dTstat'.
# Then transform to Talairach coordinates with 'adwarp'.
#=========================================================================

  foreach cond (TM HM TP HP)
    3dTstat -prefix {$subj}_{$cond}_irf_mean  {$cond}irf+orig'[4..9]'

    adwarp  -apar {$subj}spgr+tlrc  -dpar {$subj}_{$cond}_irf_mean+orig
  end

cd ..
end
#=========================================================================
# End of script!
# Take the {$subj}_{$cond}_irf_mean+tlrc datasets and input into 3dANOVA2.
#=========================================================================
```

End of loop over subjects; go back to upper directory whence we started

Results: Humans vs. Tools



- Color overlay is HvsT contrast
- **Blue** curves are Human HRFs
- **Red** & **Green** curves are Tool HRFs

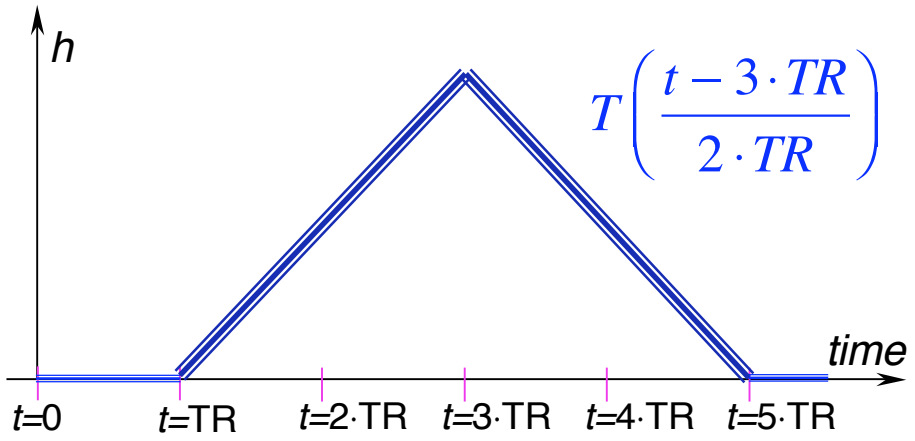
More Fun 3dDeconvolve Options

- **`-mask`** = used to turn off processing for some voxels
  - ★ speed up the program by not processing non-brain voxels
- **`-input1D`** = used to process a single time series, rather than a dataset full of time series
  - ★ test out a stimulus timing sequence
  - ★ **`-nodata`** option can be used to check for collinearity
- **`-censor`** = used to turn off processing for some time points
  - ★ for time points that are “bad” (e.g., too much movement)
- **`-sresp`** = output standard deviation of HRF estimates
  - ★ can plot error bands around HRF in AFNI graph viewer
- **`-errts`** = output residuals (i.e., difference between fitted model and data)
  - ★ for statistical analysis of time series noise
- **`-jobs`** *N* = run with multiple CPUS — *N* of them
  - ★ extra speed, if you have a dual- or quad-processor system!

3dDeconvolve with Free Timing

- The fixed-TR stick function approach doesn't fit with arbitrary timing of stimuli
  - ★ When subject actions/reactions are self-initiated, timing of activations cannot be controlled
- If you want to do deconvolution, then must adopt a different basis function expansion approach
  - ★ One that has a finite number of parameters but also allows for calculation of $h(t)$ at any arbitrary point in time
- Simplest set of such functions are closely related to stick functions: **tent functions**

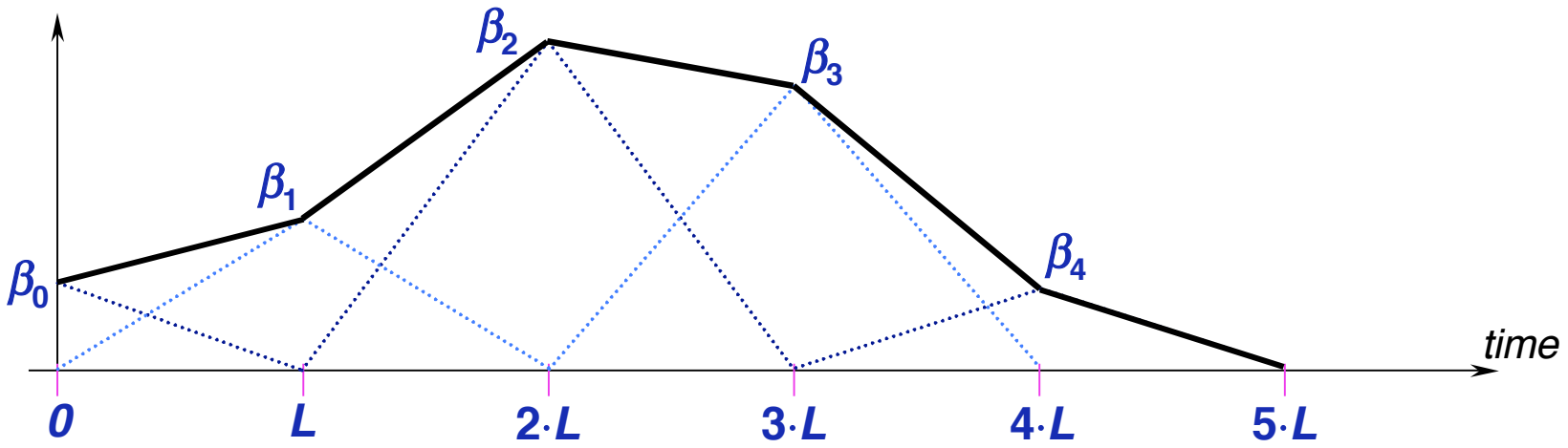
$$T(x) = \begin{cases} 1-|x| & \text{for } -1 < x < 1 \\ 0 & \text{for } |x| > 1 \end{cases}$$



Tent Functions = Linear Interpolation

- Expansion in a set of spaced-apart tent functions is the same as linear interpolation

$$\beta_0 \cdot T\left(\frac{t}{L}\right) + \beta_1 \cdot T\left(\frac{t-L}{L}\right) + \beta_2 \cdot T\left(\frac{t-2\cdot L}{L}\right) + \beta_3 \cdot T\left(\frac{t-3\cdot L}{L}\right) + \cdots$$



- Tent function parameters are also easily interpreted as function values
- User must decide on relationship of tent function spacing *L* and time grid TR

At This Point …

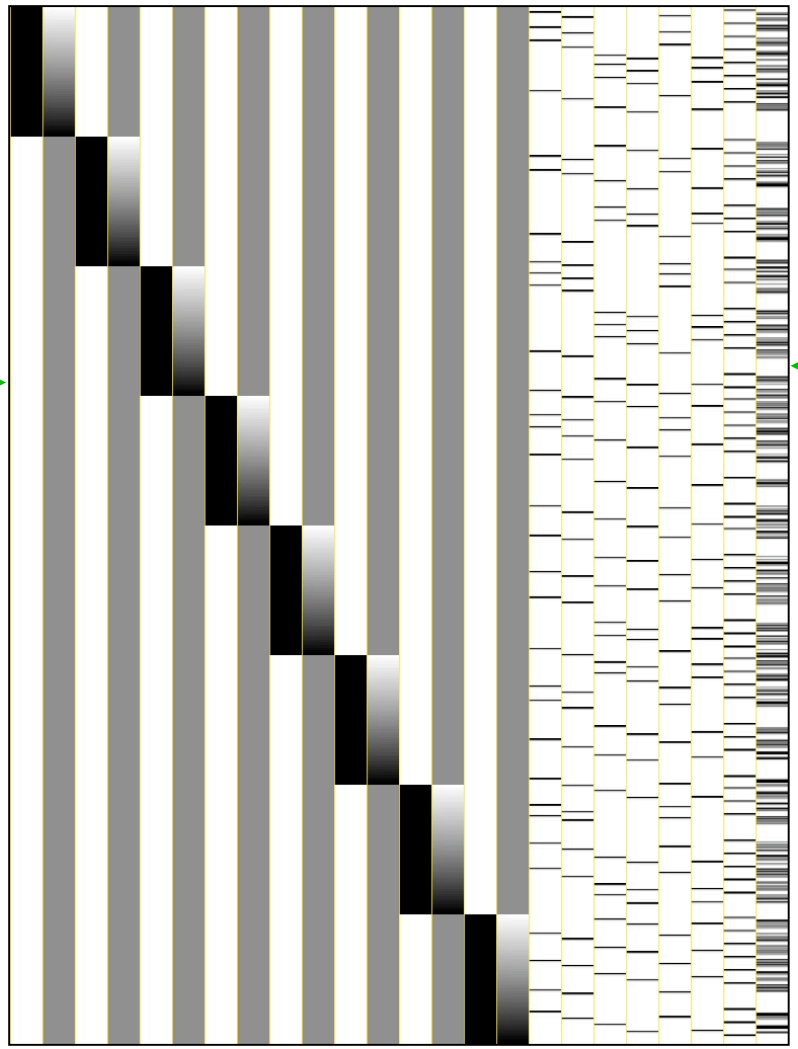
- `3dDeconvolve` is not set up to use tent functions directly
- In the past, we have now explained in grotesque detail how to set up a combination of `waver`, `3dcalc`, and `3dDeconvolve` to "trick" the system into doing deconvolution with tent functions (or other basis sets)
- However, you are saved from this excruciation
- At this moment, we have an interactive Matlab script that will set up the details for you
- In the near future, we will put tent functions directly into `3dDeconvolve`, allowing the direct use of non-TR locked stimulus timing
  - ★ Date of this promise: **13 July 2004** (AFNI Summer Bootcamp)

Upgrades Since July 2004

- See http://afni.nimh.nih.gov/doc/misc/3dDeconvolveSummer2004/
- Equation solver: Gaussian elimination to compute **R** matrix pseudo-inverse was replaced by SVD (like principal components)
  - ★ Advantage: smaller sensitivity to computational errors
  - ★ "Condition number" and "inverse error" values are printed at program startup, as measures of accuracy of pseudo-inverse
  - ★ Condition number < 1000 is good
  - ★ Inverse error < 1.0e-10 is good
- `3dDeconvolve_f` program can be used to compute in single precision (7 decimal places) rather than double precision (16)
  - ★ For better speed, but with lower numerical accuracy
  - ★ Best to do at least one run both ways to check if results differ significantly

Recent Upgrades - 2

- New **`-xjpeg xxx.jpg`** option will save a JPEG image file of the columns of the **R** matrix into file **`xxx.jpg`** (and an image of the pseudo-inverse of **R** into file **`xxx_psinv.jpg`**)



Constant and linear baselines for each run (**`-polort 1`**)

Simple regression functions created by **`waver`** and input by **`-stim_file`**

Why 'x' instead of 'R'? Because SPM calls this the 'X' matrix, not the 'R' matrix.

Recent Upgrades - 3

- Matrix inputs for **-glt** option can now indicate lots of zero entries using a notation like **30@0 1 -1 0 0** to indicate that 30 zeros precede the rest of the input line
  - ★ Example: 10 imaging runs and **-polort 2** for baseline
  - ★ Can put comments into matrix and .1D files, using lines that start with '**#**' or '**//**'
  - ★ Can use '****' at end of line to specify continuation
- Matrix input for GLTs can also be expressed symbolically, using the names given with the **-stim_label** options:

```
-stim_label 1 Ear -stim_maxlag 1 4
-stim_label 2 Wax -stim_maxlag 2 4
```

  - ★ Old style GLT might be

Sum of Ear – Sum of Wax (lags 2..4)

```
{zeros for baseline} 0 0 1 1 1 0 0 -1 -1 -1
```

  - ★ New style (via **-gltsym** option) is

```
Ear[2..4] -Wax[2..4]
```

Recent Upgrades - 4

- New **-xsave** option saves the **R** matrix (and other info) into a file that can be used later with the **-xrestore** option to calculate some extra GLTs, without re-doing the entire analysis (goal: save some time)
- **-input** option now allows multiple 3D+time datasets to be specified to automatically catenate individual runs into one file 'on the fly'
  - ★ Avoids having to use program **3dTcat**
  - ★ User must still supply full-length **.1D** files for the various input time series (e.g., **-stim_file**) options
  - ★ **-concat** option will be ignored if this option is used
    - o Break points between runs will be taken as the break points between the various **-input** datasets
- **-polort** option now uses Legendre polynomials instead of simple 1, t, t^2, t^3, … basis functions (more numerical accuracy)

Recent Upgrades - 5

- **3dDeconvolve** now checks for duplicate **-stim_file** names and for duplicate matrix columns, and prints warnings
  - ★ These are not fatal errors
    - o If the same regressor is given twice, each copy will only get half the amplitude (the “beta weight”) in the solution
- All zero regressors are now allowed
  - ★ Will get zero weight in the solution
    - o A warning message will be printed to the screen
  - ★ Example: task where subject makes a choice for each stimulus
    - o You want to analyze correct and incorrect trials a separate cases
    - o What if a subject makes no mistakes?

Recent Upgrades - 6

- Direct input of stimulus timing plus a response model, using new **`-stim_times`** option
  - ★ **`-stim_times k tname rtype`**
  - ★ **`k`** = stimulus index (from 1 to **`-num_stimts`** value)
  - ★ **`tname`** = name of **`.1D`** file containing stimulus times in units of seconds (***important***: TR value in dataset header must be correct!)
  - ★ **`rtype`** = name of response model to use for each stimulus read from **`tname`** file
    - o **`GAM`** = gamma variate function from waver
    - o **`TENT(b,c,n)`** = tent function deconvolution, ranging from time **`s+b`** to **`s+c`** after each stimulus time **`s`**, with **`n`** basis functions
    - o several other **`rtype`** options available

Recent Upgrades - 7

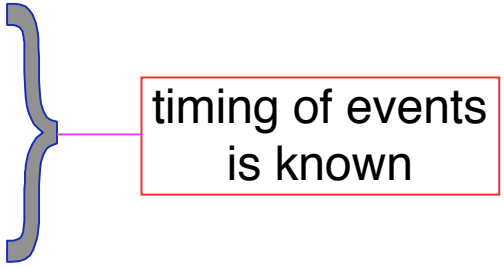
- Recall: **`-iresp`** option outputs the HRF model for one stimulus
  - ★ When used with **`-stim_times`**, values are usually output using the dataset TR time spacing
  - ★ Can changes to a different grid via new **`-TR_times dt`** option, which sets the output grid spacing for **`-iresp`** to **`dt`** for HRF models computed via **`-stim_times`**
    - o Will be useful for producing nice smooth pictures of HRF
- **Difficulty**: using GLTs with results from **`-stim_times`**
  - ★ GLTs operate on regression coefficients
  - ★ For most **`rtype`** models, regression coefficients don't correspond directly to HRF amplitudes
    - o Exceptions: **`GAM`**, **`TENT`**, **`BLOCK`**
    - o Planned solution: ***see next slide***

Upgrades – *Planned* or *Dreamed of*

- Automatic baseline normalization of input time series
- Automatic mask generation (à la `3dAutomask` program)
- Spatial blur (à la `3dmerge -1blur)`
- Time shift input before analysis (à la `3dTshift` program)
- Negative lags for `-stim_file` method of deconvolution
  - ★ for pre-stimulus cognition/anticipation
  - ★ `-stim_times` already allows pre-stimulus response
- 'Area under curve' addition to `-gltsym` to allow testing of pieces of HRF models from `-stim_times`
- Slice-dependent regressors
  - ★ For physiological noise cancellation
- Floating point output format
  - ★ Currently is shorts + scale factor

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 ("rest")
  c) subject gets cue #2, emits response
     ➥ which depends on both cue #1 and #2



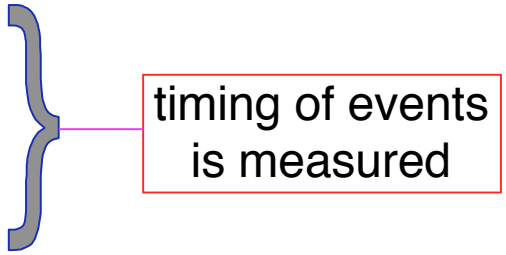
★ Cannot treat this as one event with one HRF, since the different waiting times will result in different overlaps in separate responses from cue #1 and cue #2

★ 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.

More Complicated Case

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



- 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 amounts to)
- 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; phase (b)'s is a little tricky to generate using waver, but it could be done

Noise Issues

- "Noise" in FMRI is caused by several factors, not completely characterized
  - ★ MR thermal noise (well understood)
  - ★ Cardiac and respiratory cycles (partly understood)
    - o In principle, could measure these sources of noise separately and then try to regress them out
      - ➥ RETROICOR program in progress (Rasmus Birn of FIM)
  - ★ Scanner fluctuations (e.g., thermal drift of hardware)
  - ★ Small subject head movements (10-100 μm)
  - ★ Very low frequency fluctuations (periods longer than 100 s)
- Data analysis should try to remove what can be removed and 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`**
  - ★ At present, nothing is done to correct for this effect (by us)

Nonlinear Regression

- Linear models aren't everything
  - ★ 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***)
  - ★ 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
    - o e.g., injection of nicotine, cocaine, etc.