Irregular Stimulus Timing: Analysis with 3dDeconvolve

- 3dDeconvolve is set up to calculate response functions $h(t)$ when stimuli occur locked to image acquisition TR interval
  - ◇ The `-stim_nptr` option allows stimuli to occur at intervals of TR$/p$, where p is a small integer
  - ◇ But it seems like there is no way to do deconvolution for a completely irregular stimulus pattern
- The key is to write the response function $h(t)$ as a linear sum of a small number of basis functions, and then use 3dDeconvolve to calculate (and test) the weighting factors in this sum:

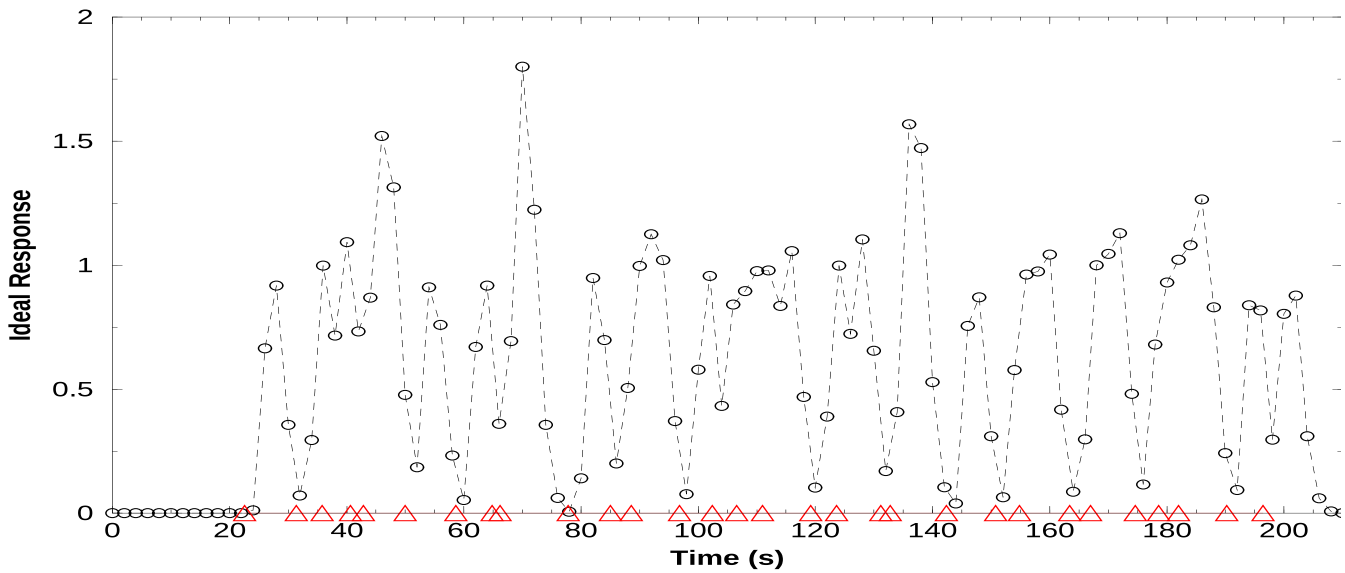
$$h(t) = \sum_{a=0}^{N_a} \lambda_a \cdot \psi_a(t)$$

where each $\psi_a(t)$ is a fixed function and the unknown amplitudes λ_a are to be determined from data

  - ◇ Must choose the $\psi_a(t)$ basis functions based on the duration and shape of response you expect

* Simulating a time series:
  - ◇ `waver -dt 2.0 -GAM -peak 1 -tstim 'cat tstim.1D' > ideal.1D`
  - ◇ File `tstim.1D` (30 stimulus times, averaging about 6 s apart):

    22.6 31.4 35.8 40.6 42.8 50.0 58.6 66.2 64.8 77.8 85.0 88.6 96.8 102.4 106.6 111.0
    119.2 123.6 131.2 132.8 142.4 150.8 154.8 163.4 167.0 174.6 178.6 182.0 190.2 196.4
  - ◇ Graph of response `ideal.1D` (circles/dashed lines) with triangles △ at `tstim.1D` stimulus times, for 106 volumes to be simulated at TR=2 s:



  - ◇ Despite appearances, this is <u>without noise</u>: fluctuations are just from different overlaps of gamma variate hemodynamic response, since stimuli are not evenly spaced in time
  - ◇ Goal of analysis is to retrieve hemodynamic response function (including amplitude) in each voxel, and test whether it is significantly different from zero

- Generate a simulated dataset with this ideal response time series in every voxel (with varying amplitude), a baseline, and some noise (with varying standard deviation):
  - ◇ Make a 1 slice dataset from something handy, just to use as a template:
    `3dZcutup -prefix zcut -keep 10 10 epi07+orig`
  - ◇ Calculate a dataset with this geometry ($64\times64\times1$) with the `ideal.1D` function intensity and noise variance varying in 8×8 blocks across the image:

```
3dcalc -a 'zcut+orig[0]' \
       -b ideal.1D       \
       -datum float      \
       -prefix sim:time  \
       -expr '100+(int(i/8)+1)*0.25*b+int(j/8)*0.25*gran(0,1)'
```

  - ↪ `3dcalc` is a voxel-by-voxel dataset calculator program
  - ↪ `-a 'zcut+orig[0]'` means to read in the #0 sub-brick of this dataset and call its voxel values by the symbol `a`
  - ↪ `-b ideal.1D` means to read in this time series file and call its values by the symbol `b` (since this has no spatial dimension, each spatial voxel from `b` at a given time index will have the same value, unlike from `a`)
    - ▷ Since the `-a` dataset doesn't have a time axis, the `-b` time series will be used to provide a time axis in the output of 106 points (same length as `ideal.1D`)

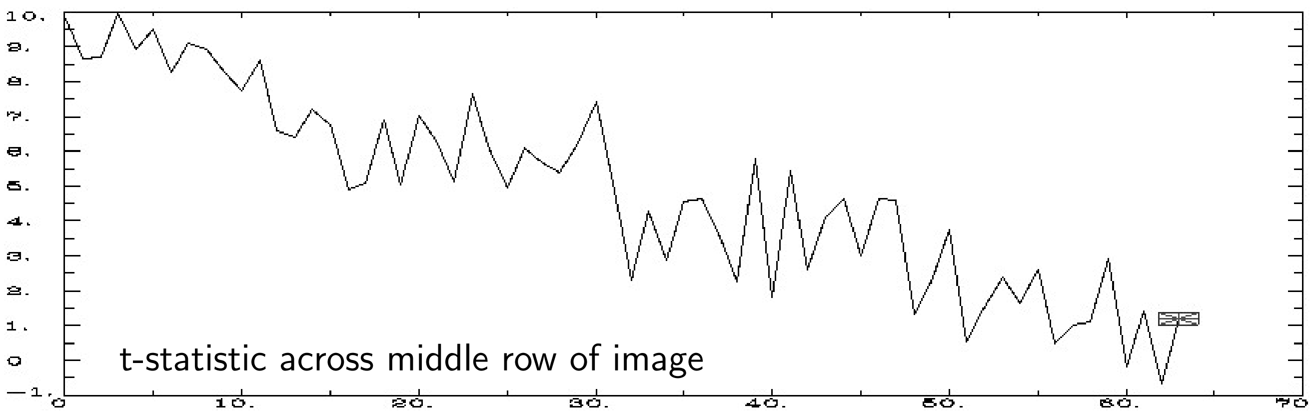
- ↪ `-datum float` means to write the output dataset in floating point format (don't need to deal with scaling issues then)
- ↪ `-prefix sim:time` is the prefix part of the output dataset filename
- ↪ `-expr '100+(int(i/8)+1)*0.25*b+int(j/8)*0.25*gran(0,1)'` is the mathematical expression that determines the output at each voxel
  - ▷ Symbols `b`, `i`, `j` are used
  - ▷ Dataset `a` was only read in to act as a template for the output dataset (i.e., to set up its spatial axes and dimensions)
  - ▷ Since there was no `-i` or `-j` option, 3dcalc assigns the x-axis voxel index to the symbol `i` (range=0..63) and the y-axis voxel index to the symbol `j` (range=0..63)
  - ▷ `100` is the baseline
  - ▷ `(int(i/8)+1)` ranges 1..8 across image horizontally, in blocks 8 voxels wide
    - • `(int(i/8)+1)*0.25*b` gives signal from 0.25 to 2 times `b`
  - ▷ `int(j/8)` ranges 0..7 across image vertically, in blocks 8 voxels wide
    - • `int(j/8)*0.25*gran(0,1)` gives noise increasing down image, in blocks
    - • `gran(0,1)`=Gaussian deviate with mean=0, standard deviation=1
- ◇ Result is spatially square 3D+time dataset with varying amounts of signal and noise that we can use for testing

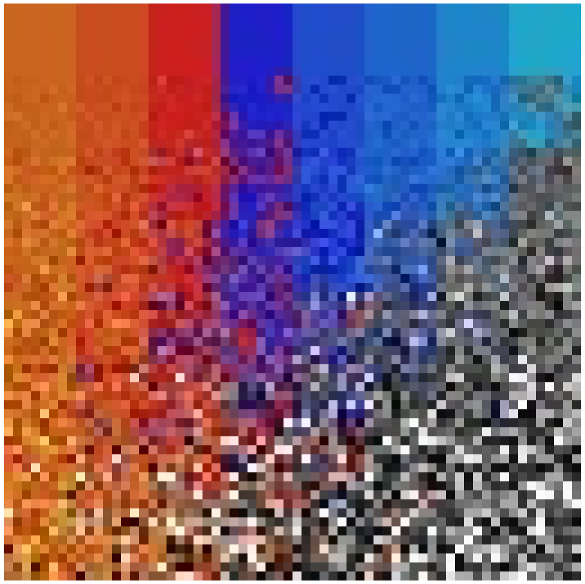
◇ Simple regression analysis with 3dDeconvolve:

```
3dDeconvolve -input sim:time+orig                          \
             -num_stimts 1                                 \
             -stim_file  1 ideal.1D                        \
             -stim_label 1 Response                        \
             -stim_minlag 1 0  -stim_maxlag 1 0            \
             -fitts sim_regress_fitts                      \
             -tout -bucket sim_regress_stats
```

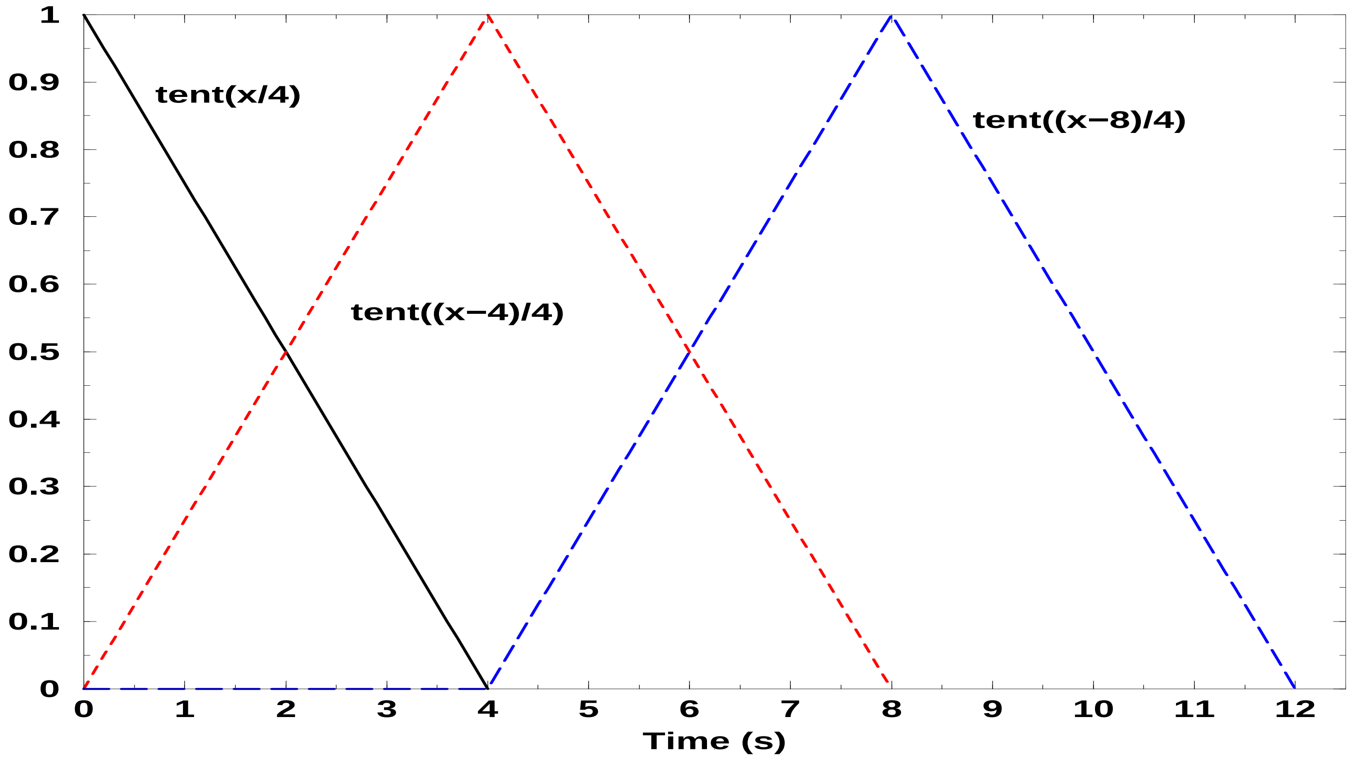
↪ In AFNI:

- Set Function→Func to sub-brick #4 Response[0] Coef
- Set Function→Thr to sub-brick #5 Response[0] t-tstat
- Set Function→** to 1; slider to p ≈1.0-4 (t≈4)
- Set Function→Pos ON; pbar # to 11
- Set Datamode→Misc→Voxel Coords ON
- Turn on See Function, open Axial Image
- Noise increases downwards; signal leftwards





- However, our goal is deconvolution analysis, not simple regression
  - ◇ Assume hemodynamic response to individual stimulus lasts no more than 12 s (depends on type of stimulus)
  - ↪ Actual `waver -GAM` function is significant only from 2..10 s post-stimulus
  - ◇ Model it as the sum of 3 "tent" functions:

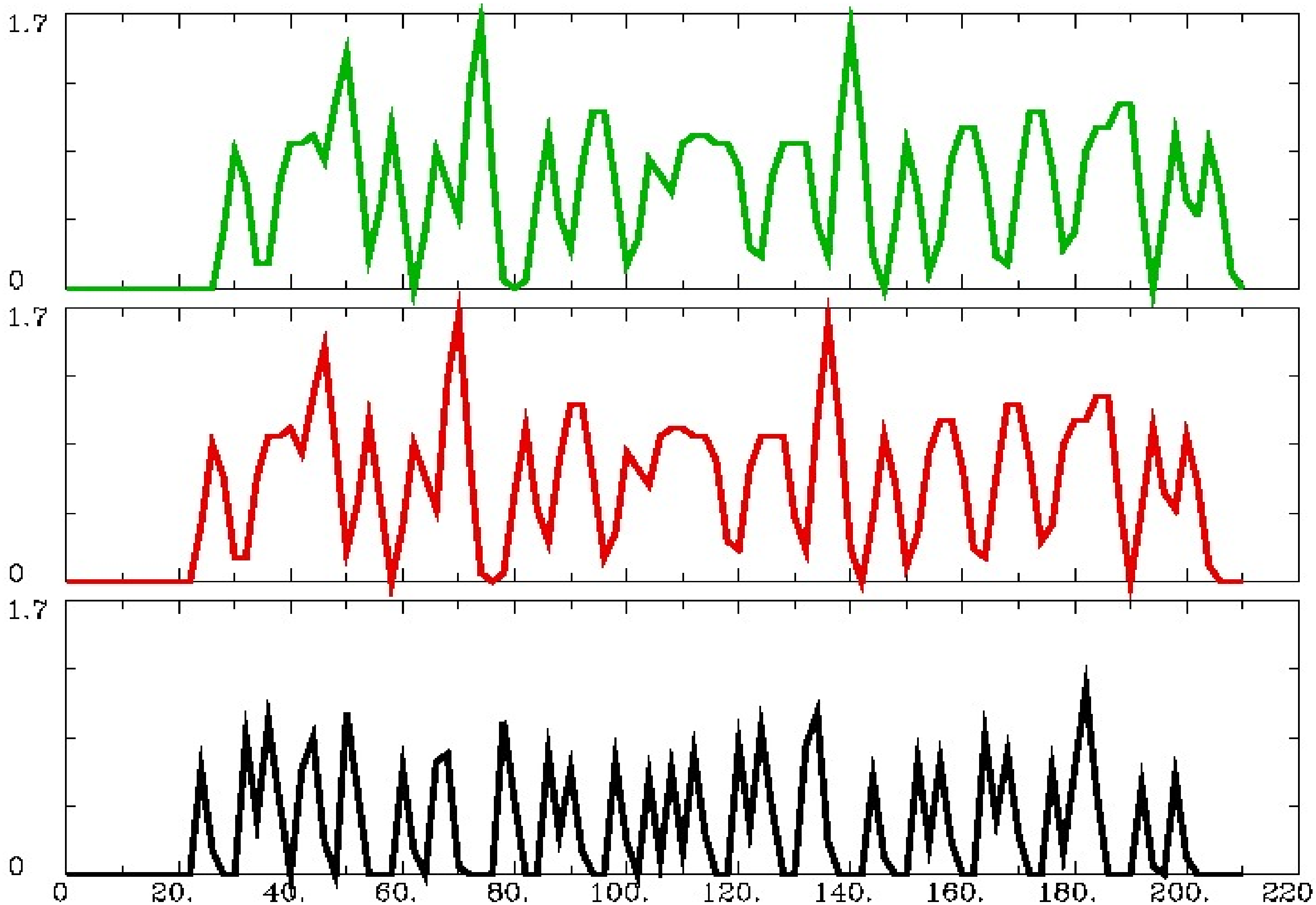


↪ $\texttt{tent}(x) = \begin{cases} 1-|x| & \text{if } -1 < x < 1 \\ 0 & \text{if } |x| \geq 1 \end{cases}$

↪ Built in function in 3dcalc, waver, etc.

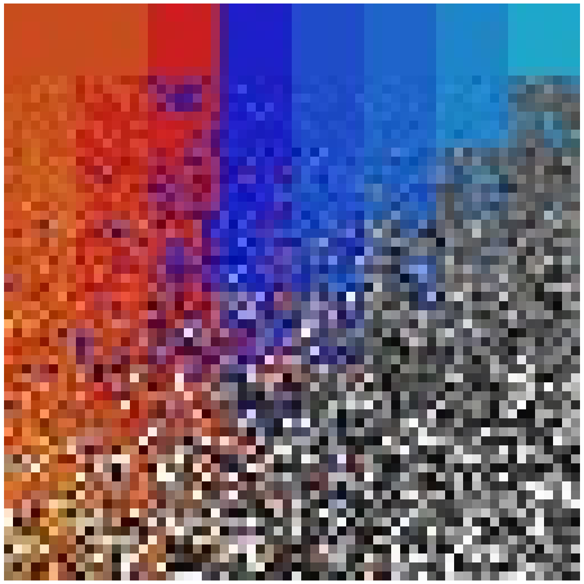
◇ We generate response time series for each of the 3 basis functions `tent(t/4)`, `tent((t-4)/4)`, and `tent((t-8)/8)`:

```
waver -dt 2.0 -EXPR 'tent(t/4)' -peak 1 -tstim `cat tstim.1D` > tent0.1D
waver -dt 2.0 -EXPR 'tent((t-4)/4)' -peak 1 -tstim `cat tstim.1D` > tent4.1D
waver -dt 2.0 -EXPR 'tent((t-8)/4)' -peak 1 -tstim `cat tstim.1D` > tent8.1D
```



◇ Then we run 3dDeconvolve with these 3 regressors:

```
3dDeconvolve -input sim:time+orig.HEAD          \
             -num_stimts 3                      \
             -stim_file 1 tent0.1D              \
             -stim_file 2 tent4.1D              \
             -stim_file 3 tent8.1D              \
             -stim_label 1 tent0                \
             -stim_label 2 tent4                \
             -stim_label 3 tent8                \
             -stim_minlag 1 0  -stim_maxlag 1 0 \
             -stim_minlag 2 0  -stim_maxlag 2 0 \
             -stim_minlag 3 0  -stim_maxlag 3 0 \
             -fitts sim_decon_fitts             \
             -fout -bucket sim_decon_stats
```



- Function→Func = #4 tent4[0] Coef
- Function→Thr = #8 Full F-stat
- Function→** = 1; slider p ≈1.0-4 (F≈7.8)
- Function→Pos ON; # = 11

◇ To calculate hemodynamic response function in each voxel, need to combine basis functions with amplitudes from output dataset `sim_decon_stats+orig`:

```
3dcalc -a 'sim_decon_stats+orig[2]'  \
       -b 'sim_decon_stats+orig[4]'  \
       -c 'sim_decon_stats+orig[6]'  \
       -d 121junk.1D                 \
       -dt 0.1                       \
       -datum float                  \
       -prefix sim:hrf_fit           \
       -expr 'a*tent(t/4)+b*tent((t-4)/4)+c*tent((t-8)/4)'
3drefit -epan sim:hrf_fit+orig
```

↪ Symbols `a`, `b`, `c` are for 3D (no time) volumes

↪ `-d 121junk.1D` is just to input a time series of length 121 points (contents don't matter, since symbol `d` isn't using in expression)

↪ `-dt 0.1` sets TR for manufactured 3D+time dataset to 0.1 s

↪ Symbol `t` is time in `-expr`

↪ `3drefit` command is to make output an Anat type dataset (so can graph it in AFNI)

◇ Also make a 1D file with the true gamma deviate hemodynamic response function, for comparison: `waver -GAM -dt 0.1 > gamma.1D`

◇ In AFNI:

↪ Switch Anatomy to sim:hrf_fit; open Axial Graph

↪ FIM→Pick Ideal and choose gamma.1D timeseries

↪ Opt→Colors, Etc.→Ideal: Use Thick Lines ON

↪ Opt→Colors, Etc.→Grid color to lt-blue2 (say)

↪ Opt→Baseline→Global ON; Opt→Baseline→Set Global to −1

↪ Press h key to get horizontal line at y=0; press a to autoscale