optseq

Comments or questions: analysis-bugs@nmr.mgh.harvard.edu\\
$Id: optseq.tex,v 1.1 2005/05/04 17:00:49 greve Exp $

1 Introduction

When stimuli are presented so closely together that their hemodynamic responses overlap, it becomes necessary to "jitter" the stimulus onsets in order to compute the hemodynamic response without assuming a shape. This jitter is realized by inserting random amounts of *null* events between task events; ie, the stimulus onset asynchrony (SOA) is randomized. NOTE: the null event (also known as "fixation") is not to be considered an event. When designing such an experiment, one must determine the order of stimulus event type as well as how much null to put between each. This combination is referred to as the *schedule*. Even for simple designs, there is a huge number of possibilities, and not all of them are the same.

optseq is a program for optimizing schedules given the various parameters of the experimental design and analysis, including TR, temporal estimation resolution (TER), number of time points (or scans) to be collected, the time window over which the hemodynamic response will be estimated, and the number of presentations per run for each condition. Note that optseq will not determine the optimum average length of the null event (ie, average SOA).

The optimization criterion is based on the following forward model:

$$y = MXh + n, \tag{1}$$

where y is the measured BOLD signal, M is a subsampling matrix (may be the identity), X is the stimulus convolution matrix dependent only upon the stimulus sequence, h is the hemodynamic response, and n is noise. If the noise is white, then the expected RMS error is $e_{RMS} = \sigma_n^2 trace((X^t X)^{-1})$. Note that under these assumptions the error is dependent only on the stimulus sequence X and noise variance σ_n^2. The noise variance is beyond the experimenter's control; however, the RMS error can still be manipulated by chaning the stimulus schedule X. We define the *efficiency* as $E = \frac{1}{trace((X^t X)^{-1})}$. Optseq searches candidate sequences for the ones that maximize the efficiency (and so minimize the expected RMS error).

The output sequences are stored in the *paradigm file format*. This is a text file with (at least) two columns. The first column indicates the time at which a stimulus is to be presented, where time zero is implicitly assumed to be the onset of the first data acquisition. The second column indicates the identification number of the type of stimulus presented. Each event type has its own unique number which is assigned by optseq, with identification number 0 being reserved for the null condition and increasing contiguously for other conditions. For **optseq** output, a third column is added indicating the duration that the stimulus is to be presented. This is redundant with the first column but can be convenient for converting to a file used by a stimulus presentation program.

2 Usage

Typing optseq at the command line without any options will give the following message:

```
USAGE: optseq [-options] -ntp ntp -npercond n1 <n2 ...> -o outprefix
   -npercond n1 <n2 ...> : number of stimuli for each non-null condition in a session
   -tpercond t1 <t2 ...> : duration of each stimulus type
   -o outprefix          : prefix of output paradigm files
   -ntp  ntp             : Number of time points to be collected for each run
Options:
   -label l1 <l2 ...> : label for each non-null condition
   -nsessions <int> : number of sessions (1)
   -nruns <int>     : number of runs in each session (1)
   -TR    <float>   : TR to use (2 sec)
   -TER   <float>   : Temporal resolution of the estimate (TR)
   -Tres  <float>   : Temporal resolution of the stimulus presentation (TER)
   -tprescan <float>   : time before first scan to present first stim (0)
   -timewindow <float> : Time Window of estimate (20 sec)
   -prestim   <float>  : part of timewindow to use for prestim baseline (0)
   -nsearch <int>   : number of sequences over which to search (100)
   -tsearch <float> : number of hours over which to search
   -seed    <int> : seed for random number generator (auto)
   -mail  user    : mail user when finished
   -monly mfile   : generate matlab file only
```

3 Command-line Arguments

-npercond n1 ¡n2 ...¿ : this is the number of stimuli per condition per session where $n1$ is the number of times that condition type 1 will be presented; $n2$ is the number of times type 2 is presented; etc. List as many as there are conditions. There must be at least one condition. Do not list the null condition. The identification number of each condition (as found in the paradigm file) corresponds to the sequence in which it is listed after the *npercond* flag. Identification number 0 always corresponds to the null condition.

-tpercond t1 ¡t2 ...¿ : this is the amount of time (in seconds) that will be allocated for each event presentation. $t1$ is the time for condition 1, $t2$ is the time for condition 2, etc. Note that this allows for conditions to be of different durations. *Durations that are not integer multiples of the TER will be rounded up to the next TER.*
-o outprefix: this is the prefix used to construct the name of the paradigm files. The paradigm file for a run will be *outprefix-sXXX-rYYY.dat* where YYY is the three-digit, zero-padded session number, and XXX is the three-digit, zero-padded run number.

-Ntp ntp: number of scans, aquisitions, or time-points to be collected for each run.

-label l1 ¡l2 ...¿: list of labels for each non-null condition. This label will be appended to each line of the paradigm file where the condition is presented. It is for ease of reference only.

-nsessions : number of sessions. Each session has a number of runs specified by the $-nruns$ flag. A paradigm file will be generated for each run, but the optimization procedure will be applied to all the runs in a session. Increasing the number of sessions does not significantly increase the amount of time it takes the program to run.

-nruns : number of runs per session.

-TR TR: temporal resolution of the fMRI scans (ie, the time between volume aquisitions) in

seconds.

-TER TER: temporal resolution of the estimate of the hemodynamic response in seconds. Default is to set the TER to the TR. If sub-TR estimation is desired, the TER must be an integer divisor of the TR.

-Tres Tres: temporal resolution (in seconds) of the stimulus presentation time. This sets the resolution of the "clock" used to present the stimuli. The presentation time is the first column in the paradigm file, and each time entry will be an integer multiple of $Tres$. By default, $Tres$ is set to the TER, and it would be unusual to do otherwise.

-tprescan: time (in seconds) before the first acquistion of a run to begin presenting stimuli.

-timewindow ¡float¿: time window in seconds over which the hemodynamic response will be estimated. This should be equal to the maximum stimulus duration plus the expected hemodynamic recovery time plus any prestimulus estimation time (see $-prestim$). Default is 20 sec.

-prestim : specify a portion of *timewindow* to estimate the hemodynamic response *before* stimulus onset.

-nsearch, -tsearch : The number of possible legal sequences is enormous and cannot be exhaustively searched. Therefore, optseq performs a random search of legal sequences and keeps the ones that are best. These two options determine how long optseq will search before returning an answer. $-nsearch$ allows the user to specify the number of sequences to sample. Alternatively, $-tsearch$ specifies the number of hours that optseq should run before returning an answer. It would not be unusual to let optseq run for 12 hours, and several days would not be out of the question.

-seed : specifies the seed for the random number generator. By default, optseq will automatically choose a seed based on the computer's system clock. Setting the seed to -1 explicitly forces optseq to choose a seed.

-mail user: send mail to user when the program finishes. Handy for those long search times.

-monly mfile: only generate the matlab file which would accomplish the analysis but do not actually execute it. This is mainly good for debugging purposes.

4 Selecting Parameters

Optseq requires that the user supply the following parameters:

1. Number of presentations per condition/event-type (N_{pc})
2. TR/TER
3. Duration of each condition/event-type presentation (T_{pc})
4. Total number of time points (N_{tp}), Average Stimulus Onset Asynchrony (ASOA).

The selection of these parameters depends upon what the experimenter is trying to accomplish as well as the realities of analyzing the data.

The number of times each event type is presented will affect how reliably the response to that event type can be distinguished from fixation and other event types. The minimum number of presentations needed to achieve a given level of significance is dependent upon the effect size, which is usually not known in advance (and is often the object of the experiment). However, to see a main effect (eg, responses in visual cortex), one generally needs about 20 presentations. To see a subtle effect in a cognitive area, one may need as many as 40 presentations. In general, each event type should be presented the same number of times. Note that -npercond expects the total number of presentations over the entire session.

The TR needs only be fast/short enough to capture changes in the hemodynamic response. A TR of 2 seconds seems to work well for this. There is an inverse relationship between the TR and the number of slices that can be acquired, so, while a very fast/short TR is generally better for signal processing, it will reduce the number of slices and so the amount of brain coverage. If the TR must be extended beyond 2 seconds, then one can still reconstruct the hemodynamic response at a resolution finer than the TR by specifying the TER (temporal estimation resolution). The TER must be an integer divisor of the TR. If the TER equals the TR, then events will always start at the beginning of the TR. TR. If the TER equals half the TR, then events will always start at either the beginning of the TR or midway through the TR. A TER less than the TR will allow the hemodynamic response to be estimated at a finer resolution; however, it can significantly reduce the power of subsequent statistical tests (if one does not assume a shape to the hemodynamic response).

The duration of each event type is usually up to the experimenter. However, optseq will work best if the duration is an integer multiple of the TER. The duration of each event type will, of course, affect the total amount of scanning time.

Choosing the number of time points. Once the duration of each event type and the number of times each event type will be presented are determined, one can compute the total amount of time that will be needed to present all the task-related stimuli:

$$T_{stim,tot} = \sum_{i=1}^{N_c} N_{pc,i} T_{pc,i}, \tag{2}$$

where N_c is the number of non-null event types. As a minimum, then, the number of time points must be $\frac{T_{stim,tot}}{TR}$. However, this does not account for presentations of the null stimulus needed to optimize the schedule. As a rule of thumb, the total time allocated to presentation of the null stimulus should equal to the average time given to the task conditions. This is equal to:

$$T_{null,tot} = \frac{T_{stim,tot}}{\sum_i N_c N_{pc,i}} \tag{3}$$

This makes the total time needed to present all stimuli (task and null):

$$T_{tot} = T_{null,tot} + T_{stim,tot} \tag{4}$$

The number of time points needed then will be $\frac{T_{tot}}{TR}$. Note: Optseq forces the null stimulus duration to be an integer multiple of the TER.

5 Example

Consider an experiment with 2 sessions and 3 runs per session. There will be 120 acqusitions per run with a TR of 2 seconds. There are three non-null event types (or conditions): "normal", "anomolous", and "nonsense". Over all 3 runs, there will be 60 presentations of the normal stimulus type, and each presentation will last 2 seconds. There will be 65 presentations of the

anomalous stimulus type, and each presentation will last 1 second. There will be 44 presentations of the nonsense stimulus type, and each presentation will last 3 seconds. The data will be analyzed with a temporal estimation resolution 1 second over a window of 20 seconds, 4 seconds of which will be used to acquire a prestimulus baseline. Paradigm files for this design can be generated by running optseq with the following options:

```
optseq  \
  -nsessions 2 -nruns 3 -TR 2 -ntp 120   \
  -npercond 60 65 44   \
  -label normal anomalous nonsense   \
  -tpercond 2 1 3   \
  -timewindow 20  \
  -tprestim 4   \
  -TER 1   \
  -o par   \
  -tsearch .05
```

This instructs optseq to run for .05 hours (only 3 minutes). It will produce six paradigm files: par-s001-r001.dat, par-s001-r002.dat, par-s001-r003.dat, par-s002-r001.dat, par-s002-r002.dat, and par-s002-r003.dat. It will also produce a log file as shown below.

```
par.log
Wed Jan  5 20:15:31 EST 2000
Nsessions:  2
Nruns:      3
Ntp:        120
nPerCond:   60 65 44
tPerCond:   2 1 3
TR:         2 seconds
TER:        1 seconds
Tres:       1 seconds
TimeWindow  20 seconds
TPreStim    4 seconds
TPreScan    0 seconds
nsearch:    0
tsearch:    .05
seed = 2.056159e+05
Event Types:
 1 60 2.000 normal
 2 65 1.000 anomalous
 3 44 3.000 nonsense
tsearched = 180.019
nsearched = 1796
Session   1  Efficiency = 0.642871
Session   2  Efficiency = 0.640874
Wed Jan  5 20:18:33 EST 2000
```

The log file shows the arguments with which optseq was invoked as well as a summary of results of the search. The search started on $WedJan5 20:15:31 EST 2000$ and ended on $WedJan5 20:18:33 EST 2000$. Optseq automatically chose a seed of $2.056159e+05$. The time of the search was 180 seconds (ie, .05 hours) during which it sampled 1769 sequences of which it kept 2. The best sequence (Session 1) had an efficiency of 0.642871. The next best sequence (Session 2) had an efficiency of 0.640874.

The first 10 lines of par-s001-r001.dat are:

```
 0.000    3     3.000     nonsense
 3.000    1     2.000     normal
 5.000    1     2.000     normal
 7.000    0     1.000     null
 8.000    2     1.000     anomalous
 9.000    0     1.000     null
10.000    3     3.000     nonsense
13.000    0     1.000     null
14.000    1     2.000     normal
```

The first column is the time at which the event begins with respect to the first image to be acquired. The second column is the ID of the event. The third column is the duration of the event, and the fourth column is the label of the event.