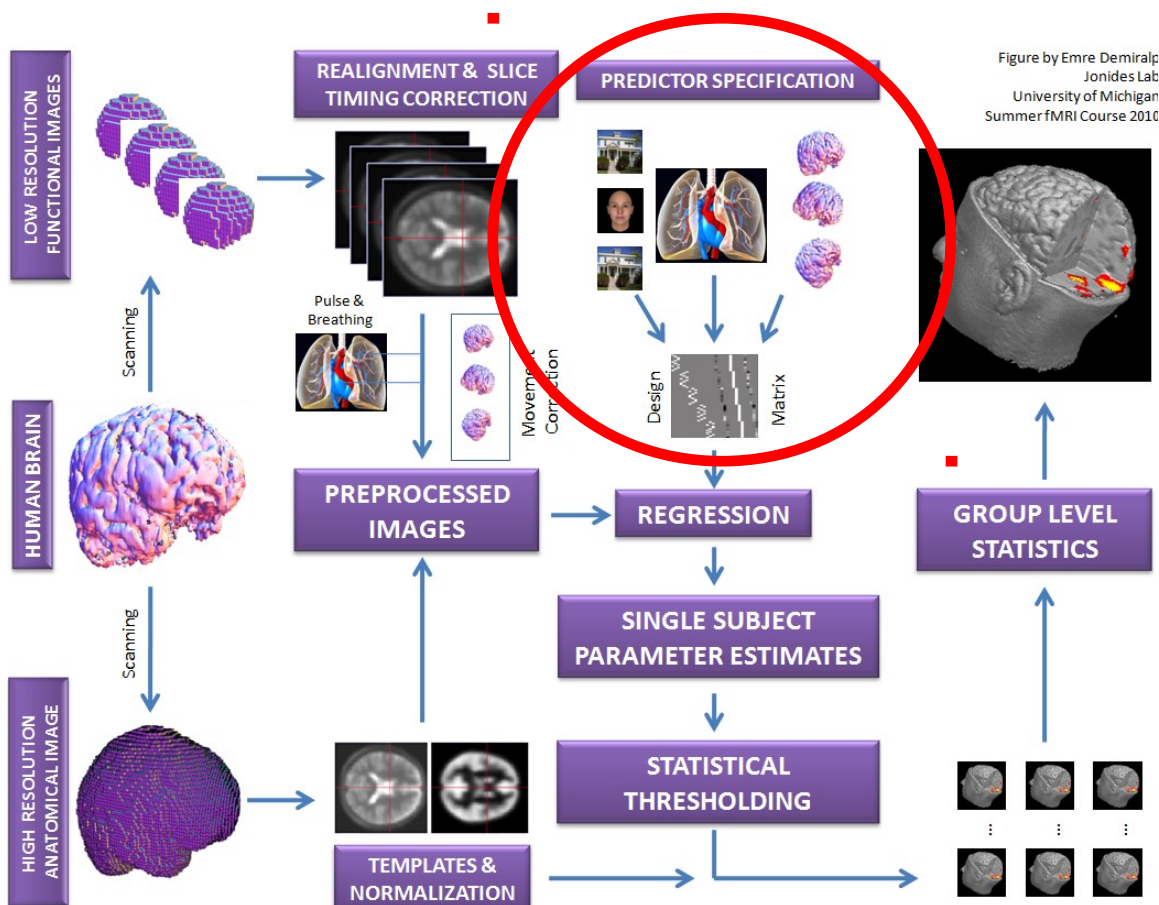




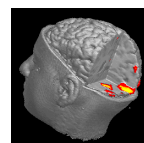
Optimal Experimental Design in fMRI Data Analysis

SPM Lab 2 BME 499/Biostat 642



Contents

Goals of this lab.....	2
Study Design Setup.....	3
Part 1: Design Setup Basics.....	4
Part 2: Constructing a Design Matrix.....	8
Part 3: Testing the efficiency of a Design Matrix.....	10
Part 4: Numerical Optimization Using Genetic Algorithms.....	13
What are the Alternative to the Genetic Algorithm Approach.....	14
When Should You Use the Genetic Algorithm.....	14

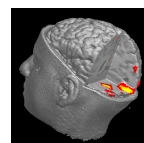




Goals of this Lab

After this lab you will be able to...

- 1) Understand the principles involved in translating a research question into an efficient, fMRI-compatible design (stimulus parameters, timing, and ordering)
- 2) Specify a linear model that reflects your design, and construct a “design matrix” that encapsulates reasonable choices about the design and the analysis
- 3) Examine the design matrix and test it for various problems that may be addressed prior to running the study



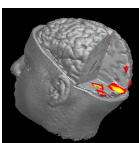


Study Design Setup

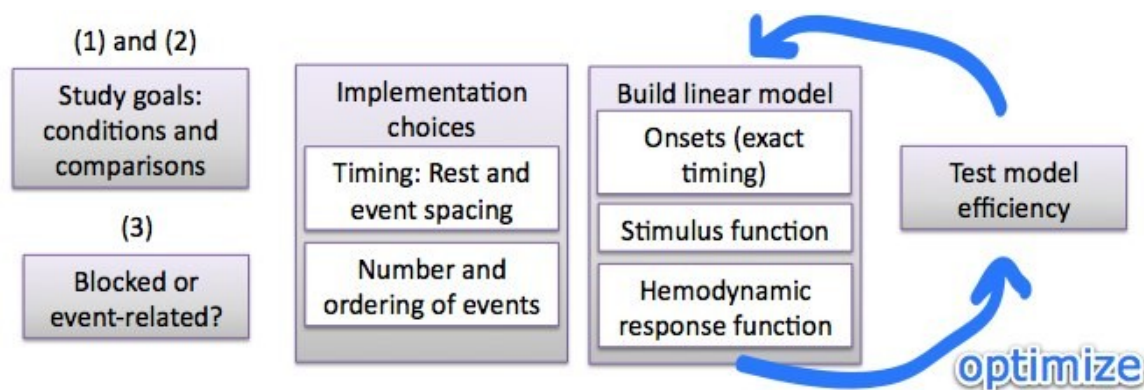
As an example, we will work on a **new design for a face/place paradigm**, similar to the one we've already been working with. Let's say the **goal of our new study** is to test whether successfully encoding items (faces or places) in memory involves boosting face- or place-selective activity in the inferior temporal/occipital cortex (the FFA or PPA). We will use a design that compares viewing of two types of items, faces and places, when each is to be remembered or ignored. We would like to identify FFA and PPA within individual participants, and then answer the following questions. We are interested in:

- 1) Whether intentionally remembering faces or places boosts activity in the FFA or PPA, depending on which type of item (face or place) is to be remembered
- 2) Whether ignoring a stimulus de-activates the posterior cortical area corresponding to its Item Type (FFA for face items, PPA for place items)
- 3) Whether the same frontal and medial temporal regions (MTL) are preferentially activated for to-be-remembered items, regardless of the item type (face or place)
- 4) Whether frontal/MTL areas are selectively connected to item type-specific posterior cortical regions (FFA or PPA) during intentional encoding
- 5) Whether activity in posterior cortical regions, frontal MTL regions, and/or connectivity between them predicts better subsequent memory

Given these goals, we can now specify the specific comparisons, stimuli, and design matrix, and test the statistical efficiency of the comparisons we will make before we ever start collecting data.

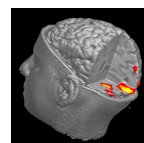


A primer on experimental design



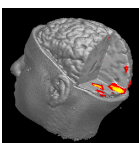
Part 1: Design setup basics

1. Consider the **types of “events” or task conditions** in your study. Are there a reasonable number? What alternatives are available? Does it make psychological sense to group events of the same type, or not?
 - a. In our hypothetical study, we have four basic conditions, organized into a 2 x 2 factorial design: Item Type, Face or Place, crossed with Memory Instruction, Remember or Ignore.
 - b. Is this a reasonable number
 - c. Should “rest” or some other low-level control condition be an event type?
 - i. If we are interested in comparing stimulus responses to rest/baseline, we need to include rest as a condition. If we are interested whether faces “activate” the FFA and/or show positive-going hemodynamic responses, for example, we must specify the baseline against which we will assess whether FFA is “activated” or “deactivated.” In a blocked design, this would entail blocks of resting fixation or whatever other low-level control condition you would like to serve as your baseline.
 - d. What other event types would it make sense to include?
 - e. What might the costs of including other event types be?
 - f. Does it make psychological sense to block faces and places? Does it make sense to block “Remember Face” and “Remember Place?”



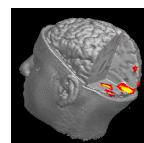


2. Second, consider **what specific comparisons you would like to test**. These will become “contrasts” in your design matrix, and statistic maps will be created for each contrast. What the conditions/contrasts are, and how you weight them in terms of importance, will determine what kind of design is optimal.
 - a. In our hypothetical study, we would like to compare activation to Face and Place stimuli under Remember or Ignore conditions
 - b. What contrast would test this?
 - c. What contrasts would correspond to the other comparisons?
3. Design basics: Consider whether **to block stimuli of the same type or use an event-related design**, approximately how long to present each stimulus for, how they should be spaced in time, and how many stimuli/runs to present
 - a. Is it important to identify whether activity is linked to specific events (i.e. face/place picture presentation) or merely to time periods when certain types of events are occurring? Is whether activity is a response to stimulus presentation specifically?
 - b. Given your answers above, which type of design (blocked/event related) is appropriate for Faces/Places? For “Remember Face (Ignore Place)” vs. “Remember Place (Ignore Face)” conditions?
 - c. How long should stimuli be presented on-screen so that they are perceptible? So that participants will not become disengaged? So that time is not wasted on imaging when participants are not doing the task (i.e., intentionally encoding/ignoring, depending on condition)
 - d. What should the approximate range of inter-stimulus intervals (ISIs) be?
 - i. Should we include “rest” ISIs at all?
 1. Is it necessary to make stimuli/trials psychologically distinct, so that subjects can do the task?
 2. Would you like to assess “activation” or “deactivation” relative to rest, or only relative to other active task conditions?
 3. Would you like to test for brain areas that respond to the sum/average across multiple event types (relative to rest), or only test for areas that respond to differences across event types?





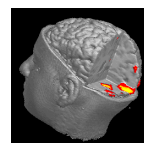
- ii. Are the ISIs between events long enough to avoid substantial nonlinearity in responses to repeated events?
- iii. Are they short enough that participants will stay engaged in the task?
- iv. Are they short enough/long enough so that a substantial proportion of the scanning time is spent performing the process of interest (face/place encoding)?
- e. For blocked designs, how long should blocks be?
 - i. You can make an initial guess based on the psychology of your design, your expected amount of low-frequency noise drift and planned high-pass filter
 - ii. Blocks, or the max time between repeated events in an event-related design, should be half the periodicity of the high-pass filter or shorter
- f. How long can you reasonably keep participants in the scanner? You should allow for 30 minutes of participant setup and structural (T1 and/or DTI) imaging time. Given that, how much time can you allocate for the functional task?
 - i. Will participants disengage or get fatigued?
 - ii. Is performance expected to be comparable across the whole functional task period?
- g. Given your answers above, how many stimuli will you be able to present in each condition?
 - i. The Central Limit Theorem describes a basic principle in statistics: The stability and reproducibility of an average of N measures increases in proportion to the square root of N . In this case, N is the number of trials in a particular condition. Typically, stability is quite low up to about $N = 30$ or 40 . Though more trials are always better, gains with larger N become progressively smaller after that, because \sqrt{N} is proportionately less of an increase as N increases. The number of trials in each condition is not the only (or, sometimes, even the primary) consideration in an fMRI design, because we may not be interested in the stability of activation estimates for individual trial types. Nevertheless, it's useful to





think about how stable the estimates for each event type are likely to be given the number of trials. Later, we will test the efficiency of *contrasts* across multiple event types, which is more precisely what we're testing when we construct statistic maps.

- ii. Is the number you came up with above a reasonable number?
- h. Consider how to avoid some obvious confounds
 - i. Can events of different types be evenly spread over scanning runs? (Randomization, blocked by time)
 - ii. Is it important for the design to be psychologically unpredictable? Consider transitional probabilities / counterbalancing of trial history

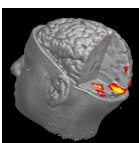




Part 2: Constructing a design matrix

Now we have what we need to put these choices into an experimental design matrix, so that we can estimate activation parameters for each event type and contrasts across event types using a linear model. Here are some additional things to consider?

1. Specify the analysis goal: Contrast detection (e.g., a powerful A – B subtraction) or hemodynamic response function (HRF) estimation, or a combination of both?
 - a. If contrast detection only, consider a block design
 - b. If HRF estimation only, you need an event-related design (see above for additional considerations)
2. Consider the type of stimulus function to assume. You must assume that “neural” responses (more accurately, the brain metabolic signals that give rise to the BOLD signal we’re measuring) have a particular form. Basic choices are “events” or “epochs.” Block designs are often, but not always, modeled with long epochs lasting the duration of a block.
 - a. Do you expect neural responses to the process of interest to occur only for a brief moment, or over a more prolonged period? If the former, an event-related design is appropriate, but if the latter, an epoch-related design is better.
 - b. If “events,” should they be placed at stimulus onset?
 - c. If “epochs,” how long and when should they occur?
3. Consider the type of hemodynamic model to use
 - a. For powerful detection of brief events, a constrained basis set (e.g., HRF + time derivative) is recommended
 - b. For HRF estimation, a smooth FIR (best) or FIR model is recommended



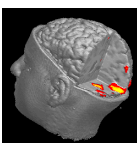


4. Specify the general order/placement of event
 - a. For block designs, should blocks repeat regularly or be randomized?
 - i. If regular, consider the psychological impact of knowledge about the block structure, and consider order effects (does Condition A always occur first?)
 - ii. Consider the spatial frequency of the design. Randomized blocks have a frequency profile that is more dispersed (more low-frequency power), and thus can preclude use of an otherwise desirable high-pass filter during analysis
 - b. For block designs, approximately how long should blocks be?
 - i. All other (psychological) factors being equal, blocks of 16-20 sec are usually optimally efficient in a two-condition design. For three conditions (or two conditions and rest), 12-sec blocks are efficient, and for four conditions, 8 sec blocks are efficient. Do the psychological constraints permit this timing?
 - c. For event-related designs, can stimuli be grouped into “mini blocks” without changing the psychological nature of the task?
 - i. If so, detection power will be improved, and design optimization to maximize contrast efficiency is recommended
5. Now you are ready to actually construct a design matrix!
We will use SPM to construct a design matrix using the graphical interface.
The instructions are encapsulated in the files:

SPM_design_spec1.swf and
SPM_design_spec2.swf

Locate these files in the folder for this laboratory on your hard drive. Drag this file into Firefox web browser to view the video. The main things to keep in mind are:

1. Use the SPM5 GUI to “specify 1st level” in “design only” mode. You can access this by pressing any button (e.g., “Smooth”) to bring up the SPM Job Manager, and then using the TASKS->Stats menu at the top.
2. Create a new directory to save your design in. The SPM.mat file created automatically in this directory when you “Run” the job will contain all of your design details.
3. Before you “Run,” remember to “Save” your job file so that you can load and edit it again later and re-run if necessary





Part 3: Testing the efficiency of a design matrix

1. First, explore the design matrix using SPM's graphical interface. Instructions for this are in the lab you've already done on first-level analysis.
2. Second, let's look at a particular session "manually," using the Matlab command interface. This is instructive because it will give you freedom: It will give you familiarity with how to use the commands to do whatever you want to do. As you'll see, we can do things that we can't do in the SPM graphical interface.

(a) First, load and get the regressors of interest

load SPM

```
X = SPM.xX.X(:, SPM.Sess(1).col); % pick out Session 1
```

This creates a variable called "X," which stores the columns in the design matrix corresponding to Session 1 (minus the intercept). If the Sessions correspond to different runs, as they typically are in SPM, then we can test each Session of a multi-run design separately. So the Session (run) will be our unit of analysis.

(b) Second, get variance inflation factors

Now let's get Variance Inflation Factors for these columns (regressors). This is an overall measure of how much each column can be explained by a combination of the other columns. It's a much better summary than the bivariate correlations among regressors, because a column may be relatively uncorrelated with each other regressor when taken separately, but highly related to a **combination** of the other regressors. You will need the OptimizeDesign11 toolbox. It is at:

<http://psych.colorado.edu/~tor/>

See if you have the function on your path by typing:

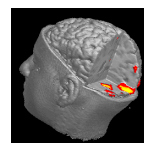
```
which getvif
```

```
vif = getvif(X); % variance inflation factors for each column  
vif
```

Are the VIFs near 1? There is no hard and fast rule for how high is high, but higher values mean more unstable estimates of a parameter. 2 is twice as much variance (error) as 1.

Check out this movie by pasting it into FireFox for help:

SPM_design3_testvifs.swf





(c) Next, specify and apply contrasts

However, we are not interested in the efficiency of the columns, really, but of the contrasts among them that define our factorial design!

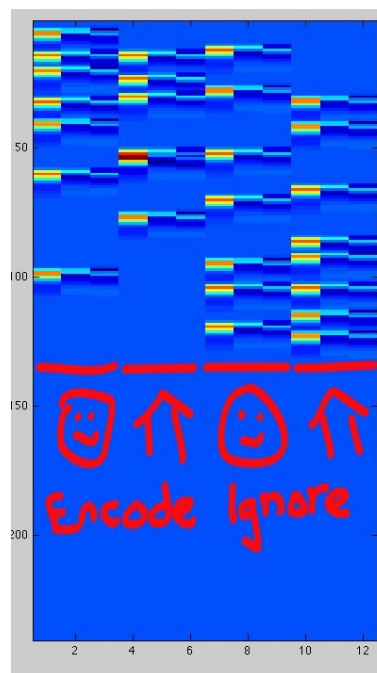
Every linear model can be expressed in terms of a design matrix. Consider the factorial design that we are using.
How many columns does it have (including the intercept?)

What contrasts will specify the main effects of [Face vs. Place] and [Encode vs. Ignore], and their interaction?

We now need to create a matrix of contrast weights for the factorial design.

At the right is a sample design matrix I created using SPM. I used the SPM HRF 3-parameter basis set, so there are 3 columns for each condition. I'd like to create a main effects and interaction contrast for each of the 3 basis functions. Basis function (HRF1,2,3) always moves "fastest" in the column order.

Remember: You do not want to directly contrast parameter estimates for different basis functions, because they don't mean the same thing. If you were only going to use the canonical HRF, you'd only have 3 contrasts, but since I've used a basis set of 3 parameters, I have 9 contrasts, specifying the main effects and interaction for each basis function. Thus, for a group analysis I'd have to take 3 images per contrast (one per basis function) to the second level and do an F-test.



If you have the same setup as I did, you can paste this code. Otherwise, you have to roll your own.

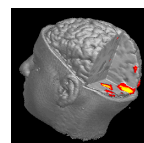
% Main effect of Enc – Ignore, for each basis function

```
C = [1 0 0 1 0 0 -1 0 0 -1 0 0; 0 1 0 0 1 0 0 -1 0 0 -1 0; 0 0 1 0 0 1 0 0 -1 0 0 -1];
```

% Main effect of Face – Place, for each basis function

```
C(:, 4:6) = [1 0 0 -1 0 0 1 0 0 -1 0 0; 0 1 0 0 -1 0 0 1 0 0 -1 0; 0 0 1 0 0 -1 0 0 1 0 0 -1];
```

% Interaction, for each basis function





$C(:, 7:9) = [1\ 0\ 0\ -1\ 0\ 0\ -1\ 0\ 0\ 1\ 0\ 0; 0\ 1\ 0\ 0\ -1\ 0\ 0\ -1\ 0\ 0\ 1\ 0; 0\ 0\ 1\ 0\ 0\ -1\ 0\ 0\ -1\ 0\ 0\ 1];$

The expression $(X * C)$ multiplies the design matrix by each contrast and returns the contrast vectors. The efficiency/variance inflation of these contrast vectors is what is of interest.

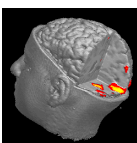
$c =$ ~~$E-I$~~ ~~$\Phi-\pi$~~ ~~X~~

1	0	0	1	0	0	1	0	0
0	1	0	0	1	0	0	1	0
0	0	1	0	0	1	0	0	1
1	0	0	-1	0	0	-1	0	0
0	1	0	0	-1	0	0	-1	0
0	0	1	0	0	-1	0	0	-1
-1	0	0	1	0	0	-1	0	0
0	-1	0	0	1	0	0	-1	0
0	0	-1	0	0	1	0	0	-1
-1	0	0	-1	0	0	1	0	0
0	-1	0	0	-1	0	0	1	0
0	0	-1	0	0	-1	0	0	1

(d) Now, test variance inflation factors for the contrasts

$vifs = \text{getvif}(X * C)$

If we were measuring reaction times, the VIFs for the contrasts would be 1, because the factorial design is defined in terms of orthogonal contrasts. However, the VIFs for the contrasts in the fMRI Design are not 1, indicating that they are less than optimally efficient. Why not?



Part 4: Numerical optimization using the Genetic Algorithms

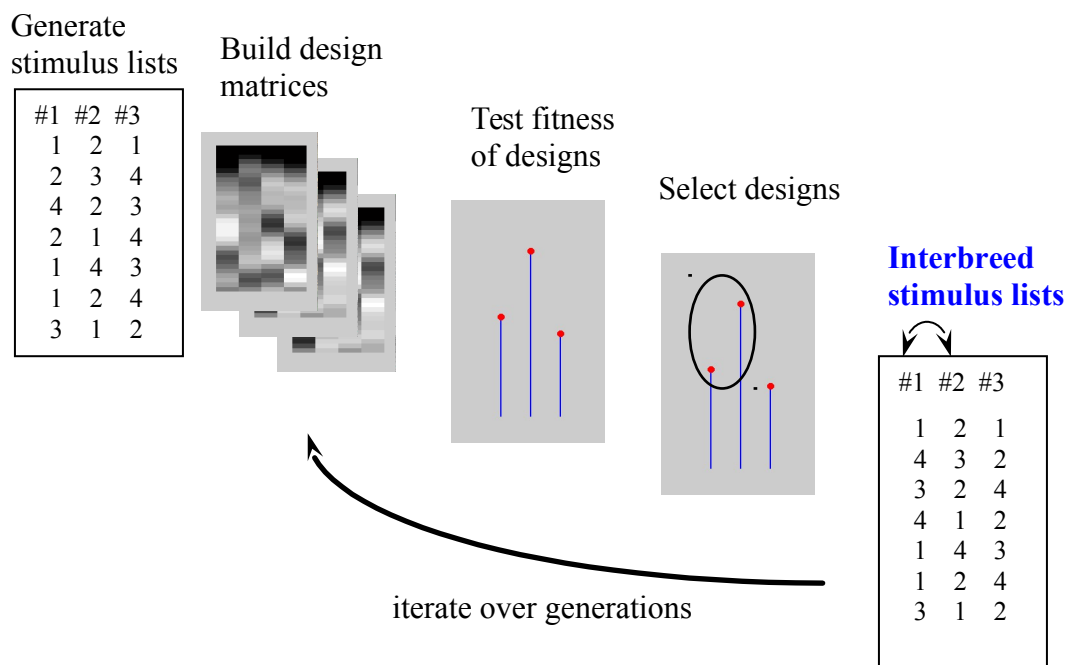
Once you've made some informed choices about what the comparisons are you're most interested in and what the general design parameters (number of conditions, timing) and analysis parameters (HRF model, high-pass filter) should look like, computer-aided design algorithms can help you both choose a design that will maximize statistical power and test how efficient that design is relative to other candidates.

How does the GA work?

First, you specify the design parameters in a script. The GA will construct random designs that conform to your specifications, build design matrices, test them, and save and "inter-breed" the best of them. It iterates through a number of "generations" until you tell it to stop, and returns the best overall design it's found. What it returns is a sequence of which event types (including rest intervals) should be presented at which times during your experiment.

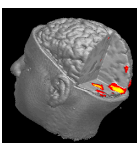
The Genetic Algorithm is described in:

Wager, T. D. & Nichols, T. E. (2003) Optimization of Experimental Design in fMRI: A General Framework Using a Genetic Algorithm. *Neuroimage*, 18, 293-309.



What are the Alternatives to the Genetic Algorithm Approach?

You could create your own scripts to generate designs and test them, and find a good design by generating a whole bunch of them and then picking the best one at the end. This is a Monte Carlo simulation. Doug Greve's **OptSeq** program does this. The trouble is





that there are so many possible designs that you might not get a very good one, even if you generate, say, a million or more of them.

Buracas' **m-sequence** program is good for HRF shape estimation, but not for contrast detection.

Kao's new GA (2009, Neuroimage) may be pretty good as well, but I haven't tried it. They compared it to our GA and said it performed better.

When Should You Use the GA?

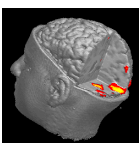
Using the GA is good if you have to optimize a randomized event-related design, and you want to detect one or more contrasts across different event types. (A contrast is a linear combination of signal estimates for each event type; for example, 2 different "visual" events versus two different "auditory" events.) Or, it could be good if you want to recover hemodynamic response estimates (HRF shapes) for each event type. Or, it could be good if you want to counterbalance the order of event types up to any number of time steps back (i.e., each trial follows each other one equally often).

If you really want to maximize detection power for a single contrast, consider using a blocked design. That will give you the best detection power. If you **ONLY** care about HRF estimation and don't have any contrasts, then try using an m-sequence (see Buracas' toolbox).

If you can't use a blocked design because you want to isolate particular events (and avoid confounds), the task can't be done in a blocked fashion, or for any other reason, the GA may be able to help. Some block designs are better than others, especially with more than two conditions, but the problem can be solved analytically—the strong suit of the GA is in cases where the solution is too hard to predict a priori. ER designs with multiple contrasts generally fall under this category, as do designs that have mixed goals (e.g., detection power for a contrast and **ALSO** hemodynamic response function estimation for linking responses to particular events in time and/or showing plots).

If you optimize for contrast detection, what the GA will try to do with your event-related design is to create mini-blocks, or clumps of similar event types. This may make the design somewhat predictable to subjects—after all, it's not truly random once you start selecting specific trial orders. You can counteract this by specifying maximum numbers of the same trial types that can occur in a row (hard constraint) or optimize for HRF shape estimation and/or counterbalancing in addition to contrast detection.

The original GA was designed to work with single events as models, not sustained epochs. However, its results will still be reasonably accurate for short epochs (less than a few seconds). If you have a longer epoch-related design, there is an alternative GA function that has worked for me in the past, but hasn't been extensively tested. The normal event-related GA function (that does most of the work) is called `optimizeGA.m`.





The epoch-design GA function is called `optimizeGA_epochs.m`. Using this second function might be frustrating at this point.

1. Set up the GA for this study

First, make sure you have the OptimizeDesign11 Toolbox downloaded and added to your Matlab path.

You'll start by editing an example script, and customizing the script for your design. Check whether the script is there and open it in the Matlab Editor by typing:

`which ga_example_script.m`
`edit ga_example_script`

I've created a version of the script for our study, called:

`ga_example_script_faceencoding.m`

In the script, here are some of the options I changed and what they mean. Additional explanation of the options can be found in the comments in the script itself!

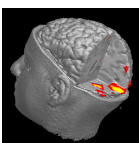
`GA.conditions = [1 2];`

- This means there are two conditions, 1 and 2. We will assign Face=1, Place=2. This is arbitrary.
- I've decided to create a mixed block/event-related design, with Face vs. Place event related, and Encode vs. Ignore blocked. The idea is that every 20 sec, an instruction will appear on the screen that says "Remember Face (Ignore Place)" or "Remember Place (Ignore Face)". By default, the two block types will alternate in an ABAB fashion. Because Encode/Ignore is BLOCKED, I'm not including them in the "event types" here. There's a special option for the Mixed Block/ER design, which will replicate these events (1, 2, etc.) within alternating blocks. We will come to that below.
- If I wanted to use a pure event-related design, I'd specify [1 2 3 4] for the 4 event types.

`GA.freqConditions = [.34 .34];`

- These are the frequencies of Face and Place events (34% of the events). They don't sum to 1 (100%) because I've decided to leave blank (unmodeled) intervals, or in other words, "jitter" the presentation times. The unmodeled blank intervals are marked in the program as Condition 0, and the algorithm will start off with $1 - (.34 + .34) = .22 = 22\%$ blank intervals.
- You could also model rest by adding an event type for rest specifically (i.e., condition 3 here), but it's not necessary to do it that way.

`GA.scanLength = 480;`





- We'll optimize one run's worth, or 8 min. We can run the whole GA multiple times, one for each run. I like to run it even more times than the number of runs, and create a pool of optimized runs that can be used, so that not all subjects have the same trial ordering.
- You might also consider counterbalancing across subjects which optimized sequence is used for which run, and/or using the same runs but reversing the assignments of numbers to conditions for half the subjects (i.e., for some subjects, 1 would be Place, and 2 would be Face). You don't need to re-run the GA to do this, just change what stimulus you assign to each condition number when you program the experiment.

GA.ISI = 2;

- This is the inter-stimulus interval, the basic time unit of the GA. Basically, a condition is assigned to each ISI, and time bins are created to fill the whole duration of the run you specified above. 2 means that we'll present either a stimulus, rest (blank screen), or instruction stimulus every 2 secs.
- It might be advantageous to use a fractional TR or ISI to "oversample" the HRF, but you may get warnings if you do that with the GA for silly technical reasons that could be fixed...

GA.cbalColinPowerWeights = [0 1 1 0];

- **This is a vector of weights to specify how much I care, in relative terms, about the objectives of the study. The first number is the weight for counterbalancing, the second for contrast detection efficiency, the third for HRF shape estimation efficiency, and the fourth for maintaining the same frequencies of trial types as those I put in.**
- **The GA will start with the input frequencies, but it will choose designs with either more or fewer events of the different types depending on the designs' efficiency. This last weight is a way of constraining that, usually at the cost of efficiency.**
- **In our design, I specified that I care equally about HRF estimation for each of the 4 event types, and detecting the 5 contrasts I specified below.**

GA.dflag = 0;

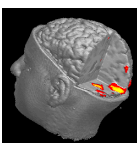
- **Use a-optimality. This will tend to prioritize the contrasts a little bit.**

GA.numGenerations = 1000;

GA.sizeGenerations = 300;

GA.maxTime = 120;

- **How many iterations of the GA to run. Here, 1000 iterations of 300 designs each. It will complete all these iterations, or stop at maxTime, which I set to be 1 hour (3600 sec) in the example I ran. However, if you set it to 120, it will run for just 2**





minutes, and you can proceed ahead without waiting a long time. For real designs, I like to let each run cook for at least 12 hours. The longer the better, though you won't get much benefit after a while.

- At least 300 designs/generation is a good minimum

GA.HPlength = [100];

- I specified a 100 sec high-pass filter, which will remove a good deal of low-frequency noise. I don't want my design to be such that the filter removes lots of design-related variance, so including the filter choice will help ensure that it doesn't.
- A 100 sec filter means that any frequencies slower than 50 sec periodicity (i.e., 25 sec alternating blocks) will be penalized. Thus, this will push the design towards 25 sec or shorter blocks. This is convenient, because we know that blocks about this long (18-20 sec) are optimal, and I've planned a new instruction for a new Encode Face or Ignore Face Block every 20 sec (see below).

GA.NumStimthresh = [5];

- Don't permit designs with more than 5 events of the same type (Face or Place) in a row.

**GA.contrasts = [1 1 -1 -1; ... % Encode - Ignore Main Effect
1 -1 1 -1; ... % Face - Place Main Effect
1 -1 -1 1; ... % Interaction
1 0 1 0; ... % Face - Rest
0 1 0 1]; % Place - Rest**

- I've specified that I care about 5 contrasts. Here, each contrast is entered as a row vector.

GA.contrastweights = [.5 1 1 .5 .5];

- This is a weighting function for the contrasts (rows of GA.contrasts) that tells me which contrasts, relatively speaking, I care the most about. Here, I've specified that I care the most about Face-Place and the Interaction (with equal weights), and also care, but a little bit less, about Encode-Ignore and Face-Rest and Place-Rest.

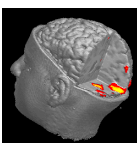
GA.restlength = [2];

% if inserting probe or instruction periods, modeled with periodic rests in the simulation,

% this is the length of the rest/probe/instr periods to insert, in units of the ISI

GA.restevery = [10];

% This is how many regular events you want to have (in ISIs) between rests.





% Leave restlength and restevery blank to avoid using these options.

GA.trans2switch = 0;

% 1 or 0. This option works if the conditions field is 2 elements long.

% This option creates new conditions 3 and 4 that occur whenever 1 is followed by 2 or 2 by 1

% So this option creates trial history-dependent predictors. 1 and 2 are repeats, 3 and 4 are switches

% Useful when studying habituation, switching between items, etc.

GA.trans2block = 1;

% 1 or 0. This option doubles the input length of conditions - so [1 2] is transformed to [1 2 3 4]

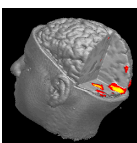
2. Run the GA script

Just type its name.

The GA shows you lots of output that reflects your input options and some calculations it's made. Once you see the results of each "generation" appear, you know that it's working. The numbers for efficiency (contrast and HRF) should generally go up, though if you weight both, sometimes one can go up at the expense of the other going down.

These are relative metrics rather than absolute ones, as:

- (1) they depend on the scaling of the regressors and number of volumes you collect;
- (2) they are only proportional to power, because they specify relative efficiency for a given level of noise variance (which you don't know until you collect the data); and
- (3) they don't say anything about the psychological/neural strength of activation, so they are proportional to power for a given true effect size





```
Starting Model 1
OptimizeGA
=====
Saturation threshold = 2
Setting up variables
...contrasts entered; last number is added to account for intercept

contrasts =
  1  1 -1 -1  0
  1 -1  1 -1  0
  1 -1 -1  1  0
  1  0  1  0  0
  0  1  0  1  0

...fyi: Num contrasts does not equal num conditions
...Using rests of length 2 and resting every 10

===== Parameters =====
Total stimuli in scan : 240
Fixed rest length in ITIs : 2
Fixed Rest stimuli in scan : 38
Num blank intervals : 64
Num of task conds : 2
Num stimuli each cond : 69 69

=====
...generating lists of organisms
...USING TRANS2BLOCK >
freqConditions = 0.17 0.17 0.17 0.17
conditions = 1 2 3 4
REST INTERVALS: not used (to use, freqCond should be one longer than conditions)
...randomizing organism start state
...restMatrix is 29 X 300

ans =
  10  10  10  10  10  10  10  10  10  10
  10  10  10  10  10  10  10  10  10  10
  10  10  10  10  10  10  10  10  10  10
  10  10  10  10  10  10  10  10  10  10
  10  10  10  10  10  10  10  10  10  10

===== Num stimuli, averaged across lists (before instruction/rest stimuli) =====
Condition 0: 64 stimuli, 26.6667% of total stimuli
Condition 1: 69 stimuli, 28.75% of total stimuli
Condition 2: 69 stimuli, 28.75% of total stimuli
Condition 3: 0 stimuli, 0% of total stimuli
Condition 4: 0 stimuli, 0% of total stimuli

best of generation 1 / Wtd. con eff = 26.894 / Wtd. HRF eff = 16.0521 / ok lists = 253 / Pop var. = 0.94014 / time = 11.1
best of generation 2 / Wtd. con eff = 26.5759 / Wtd. HRF eff = 16.8173 / ok lists = 294 / Pop var. = 0.9375 / time = 12.57
best of generation 3 / Wtd. con eff = 28.9189 / Wtd. HRF eff = 14.8879 / ok lists = 294 / Pop var. = 0.93514 / time = 12.48
best of generation 4 / Wtd. con eff = 28.9189 / Wtd. HRF eff = 14.8879 / ok lists = 293 / Pop var. = 0.9329 / time = 12.41
best of generation 5 / Wtd. con eff = 28.7789 / Wtd. HRF eff = 15.1432 / ok lists = 297 / Pop var. = 0.9323 / time = 12.55
best of generation 6 / Wtd. con eff = 28.7789 / Wtd. HRF eff = 15.1432 / ok lists = 298 / Pop var. = 0.93051 / time = 14.77
best of generation 7 / Wtd. con eff = 28.7789 / Wtd. HRF eff = 15.1432 / ok lists = 297 / Pop var. = 0.92967 / time = 15.17
best of generation 8 / Wtd. con eff = 31.424 / Wtd. HRF eff = 13.9639 / ok lists = 294 / Pop var. = 0.9288 / time = 12.74
>>
```

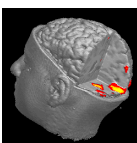
Handwritten notes:

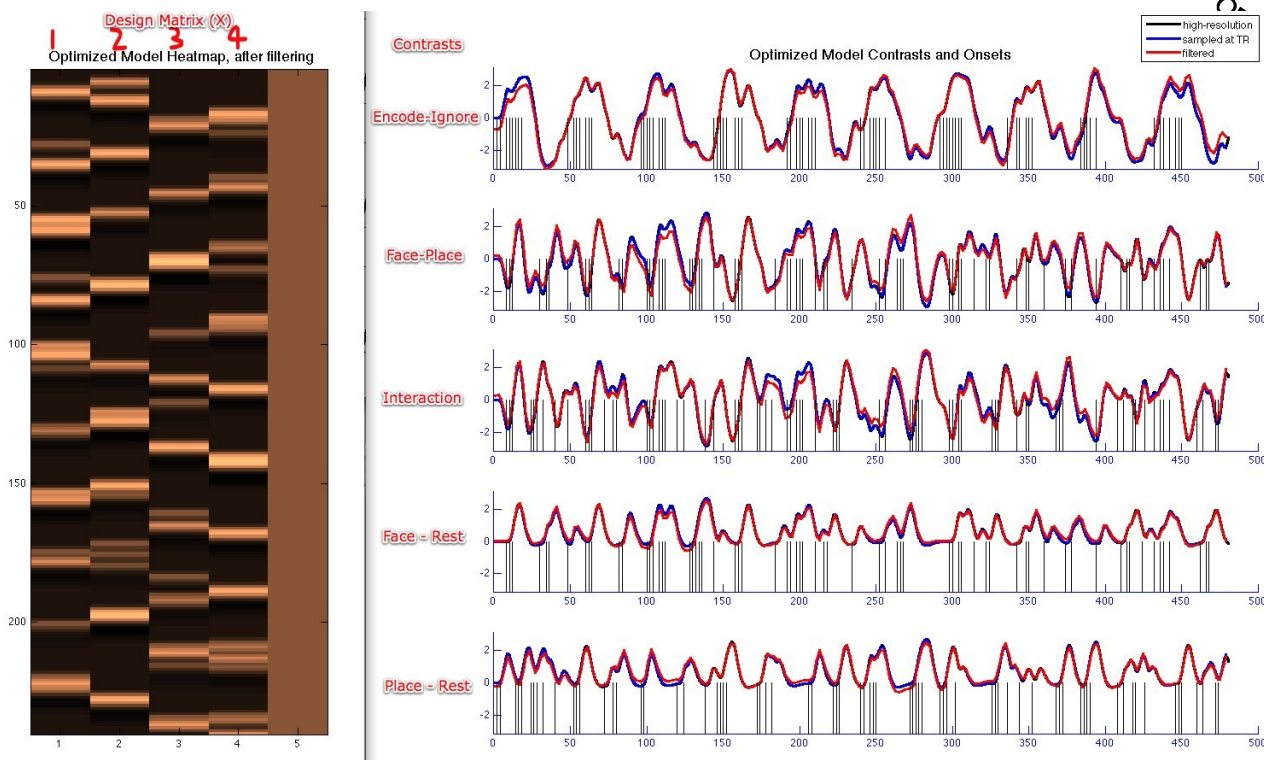
- Factorial + simple effects** (pointing to contrasts)
- ← one run, 480 sec.** (pointing to Total stimuli in scan)
- } Block / ER mix** (pointing to TRANS2BLOCK)
- ignore this** (pointing to REST INTERVALS)
- } Breaks btwn. blocks** (pointing to ans matrix)
- Detection Power** (pointing to Wtd. con eff)
- ← rest** (pointing to Condition 0)
- HRF shape eff.** (pointing to Wtd. HRF eff)

3. Look at the results of the GA

If you've turned the "plot" option on (set it to a value of 1) in the script, the GA will make some plots that show you the design. Some of them look like the ones below.

The GA also returns a structure with lots of information, `M`, to the workspace. This is saved to disk in the file: **modell1_8-11-2009.mat**, so you can load it later. The date in the filename will change, depending on what day it is.





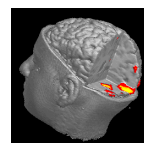
I've labeled the columns and contrasts here for our example study, but in general you will have to figure this out on your own, depending on what you had in mind for mapping condition numbers and contrasts to psychological/task conditions. Vertical lines indicate onsets that belong to a particular contrast.

Question 1: Does the model matrix (left in the plot; look at your own on-screen) make sense? Do the predictors look like what you intended when you set up the GA?

Question 2: Do they show a reasonable level of rise and fall? Do they seem relatively uncorrelated to the naked eye?

Question 3: Is there rise and fall in your contrast values (right in the figure above)? Are there patterns that seem psychologically unacceptable (to the naked eye)?

Question 4: Do the different event types seem reasonably well balanced, based on what you intended? Do they seem spread relatively evenly across time?



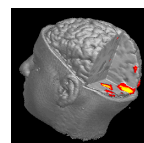
Question 5: Does filtering or downsampling to the TR introduce big variations from the original regressors? (These things can only reduce design variance and increase inaccuracy, so less is good!)

Extras for Experts

You can make your own plots using all kinds of tools that are available, but that we don't have time to document fully!

Try the following code:

```
[XX] = getPredictors(M.stimlist, spm_hrf(2));
figure; plot_matrix_cols(XX, 'horiz');
for i = 1:size(XX, 2)
    hh{i} = plot_onsets(find(M.stimlist == i) - 1, 'r', i + .3, -.5);
end
```





Now, let's look quantitatively:
Type "M"

```
>> M
M =
    origlist: [202x1 double]
    restlist: [29x1 double]
    stimlist: [240x1 double]
    ga: [1x1 struct]
    bestLists: [240x287 uint8]
    date: '8-11-2009'
    modelatTR: [240x5 double]
    model: [4800x5 double]
    delta: [4800x4 logical]
    smoothed: [240x5 double]
    contrasts: [5x5 double]
    conmodel: [240x5 double]
    consmooth: [240x5 double]
    energy: [107.05 115.52 87.131 108.95 5.0834e-29]
    conEnergy: [715.62 406.47 398.2 132.94 147.45]
    eff_fitness: 39.124
    eff: [39.427 25.513 24.338 26.845 30.853]
    se_contrasts: [0.025363 0.039196 0.041089 0.037251 0.032411]
    resample_eff_loss: 19.93
    smoothing_eff_loss: 1.0227
    DX: [240x61 double]
    hrf_eff_fitness: 15.304
    hrf_eff: [61x1 double]
    hrf_eff_avg: 15.267
    vif: [1.3668 1.3543 1.2988 1.3728 -0.33551]
    conVif: [1.0116 -2.2518e+15 1.0225 -4.5036e+15 3.0024e+15]
    cond: 3.847
    conCond: 1.1833e+16
    colin: [5x5 double]
    conColin: [5x5 double]
    cBal: 92.469
    cbalfreqs: [4x4 double]
    freq: [0.15417 0.14583 0.125 0.15833]
    timeBtwn: [12.056 13.176 15.103 12.162]
```

the final sequence

a structure with all your input options

design matrices

contrast matrix

overall "goodness"

efficiency for each contrast

HRF estimation design matrix

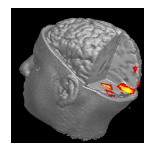
average HRF est. efficiency

pairwise correlations

transitional probabilities

frequencies of events

avg. time between repeats





We won't go over everything, but we'll look at some key things. Answer the following questions for your own design!

Look at M.eff. Are some contrasts much more efficient than others?

Type M.colin. Are any predictors strongly correlated with any others?

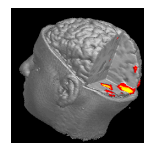
Look at M.vif and M.conVif. Are there high variance inflation factors for some regressors (vif) or contrasts (conVif)?

Is this OK, or not? Why/why not?

If you entered some non-independent contrasts (i.e., some contrasts can be predicted from a combination of others), then their VIFs may be very high. That's OK, potentially, because you're testing the contrasts one at a time. But if contrasts are very highly correlated, then it will be difficult to tell for sure which one is the better descriptor of the true effect! Statistics can tell you whether each contrast explains a significant amount of the variance, but the standard stats can't tell you whether one contrast is a better predictor than another – that's a “model comparison” problem.

High regressor VIFs are more likely to signal a problem, but remember, this isn't the bottom line either. If a regressor has a high VIF, you won't be able to estimate its activation parameter reliably. If you care about whether individual regressor betas are zero, positive, or negative, this definitely matters. But sometimes, you don't care about whether a regressor is individually significant; you just care about whether a set of them is significant, or a contrast across them, etc. So if your VIFs are low, you're good to go, but if they're high you're not necessarily sunk—you just need to think carefully.

Look at M.vif and M.conVif in the output above. Does this seem OK, or not?





Here are some other files that are output from the GA, and saved to disk:

-rw-r--r-- 1 tor tor 580669 Aug 11 10:56 GAworkspace.mat

The whole workspace when you ran the GA script. Probably redundant...

-rw-r--r-- 1 tor tor 4817689 Aug 11 10:56 modell.log

The log of everything printed to screen when you ran the GA script

-rw-r--r-- 1 tor tor 278393 Aug 11 10:56 modell_8-11-2009.mat

The final model output M structure. This is what you really need.

-rw-r--r-- 1 tor tor 928664 Aug 11 10:56 GAreults.mat

All the variables within optimizeGA.m when it finished; has intermediate output, but not the final M structure...no pressing need to save this unless you know what you're doing with it

-rw-r--r-- 1 tor tor 928550 Aug 11 10:55 listMat_working.mat

No idea why I ever saved this...

-rw-r--r-- 1 tor tor 451198 Aug 11 09:56 initial_designs.mat

The starting designs...no pressing need to save this unless you know what you're doing with it

4. Get the GA results in a form you can use with SPM

Now you will run a little function that makes the output SPM-compatible.

Type: `onsets = ga_model2onsets(M)`

This function takes a GA output M structure and returns onsets for each condition/event type in seconds. Zero is the start of the first event (i.e., the start of the run, after discarded acquisitions).

Onsets are stored in a cell array, one cell per condition, in order of the integer condition numbers stored in M.stimlist. The onsets cell array is compatible with the format SPM5 (and 8?) uses for its "multiple conditions" input in the first-level experimental design specification

You will have to add "names" and "durations" variables, but then you should be set to import the results of the GA into SPM.

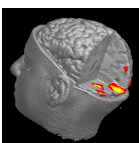
Example:

(You will have to replace the name of the .mat file with your own!)

`load modell_8-11-2009.mat`

`onsets = ga_model2onsets(M)`

You will need to add names and durations to be compatible with SPM5/8:





```
names{1} = 'Face_EncodeFace'; names{2} = 'Place_Encode'; names{3} =  
'Face_EncodePlace'; names{4} = 'Place_encodePlace';  
durations = {2 2 2 2}
```

Now let's save the results in the same .mat file, so we have the onsets.
You will have to replace the name of the .mat file with your own!

save model1_8-11-2009 -append onsets names durations

5. Re-build your SPM model with the optimized design!

Now it's time to return to SPM and insert your onsets, and see how it looks.

Make a new directory and go to it:

`pwd`

`mkdir Example_SPM_design_optimized`

`cd Example_SPM_design_optimized/`

See this movie file for a walk-through:

[Movie_put_GA_design_in_SPM.mp4](#)

