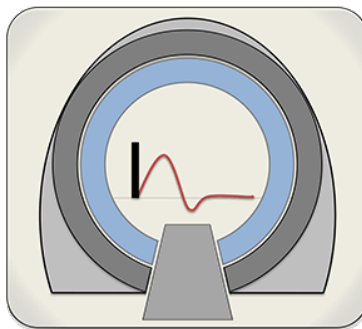


fMRI Design Explorer:

A program to design, optimize, and visualize task-based fMRI
experimental designs

Student Workbook



Contents

Preface	I
Introduction	II
Download and Installation	III
Overview of Program	IV
Unit 1: The Hemodynamic Response	1
1.1 Overview	1
1.2 Define and Configure Events	1
1.3 Visualize Event Signal	3
1.4 Unit Exercises	4
Unit 2: Event Hierarchy	5
2.1 Overview	5
2.2 Hierarchy Outlined	5
2.3 Hierarchy in the Design Explorer	6
2.4 Unit Exercises	9
Unit 3: Block Designs	10
3.1 Overview	10
3.2 The Design	10
3.3 Define and Configure Events	10
3.4 Visualize Event Signal	11
3.5 Unit Exercises	12
Unit 4: Slow Event Related Designs	13
4.1 Overview	13
4.2 The Design	13
4.3 Define and Configure Events	13
4.4 Visualize Event Signal	14
4.5 Unit Exercises	15
Unit 5: Assessing Design Quality	16
5.1 Overview	16
5.2 Optimization	16
5.3 Quality Metrics	16
5.4 Compare Designs	17
5.5 Unit Exercises	19
Unit 6: Fast Event-Related	20
6.1 Overview	20
6.2 Define and Configure Events	20
6.3 Visualize Events	21
6.4 Unit Exercises	22

List of Figures

1	Project Launcher	IV
2	Overview of Interface	VI
3	Single stimulus event hierarchy	1
4	Single stimulus event configuration	1
5	Single stimulus event plot	2
6	Single stimulus visualize configuration	3
7	Single stimulus hemodynamic response	3
8	Minimal Hierarchy	5
9	Run Level	5
10	Trial Level	6
11	Run Level Configuration	6
12	Run Level Events	7
13	Trial Level Configuration	7
14	Trial Level Events	8
15	Monetary Incentive Delay Task Hierarchy	9
16	Block design event hierarchy	10
17	Block design configuration	11
18	Block design event plot	11
19	Block design response plot	11
20	Slow event hierarchy	13
21	Initial Slow event related configuration	14
22	Initial slow event related event plot	14
23	Initial slow event related response plot	14
24	How to optimize	16
25	Optimization window	16
26	Visualize Quality Measures	17
27	Design toolbar with copied designs	18
28	Variance inflation factor	18
29	VIF exercises event configuration	19
30	Fast event hierarchy	20
31	Initial Fast event related configuration	20
32	Initial fast event related event plot	21
33	Fast event related signal	21

Preface

Say you, an enthusiastic, young cognitive neuroscientist, have an archaic research question such as, “Where in the brain responds more selectively to faces compared to other stimuli?” After all, humans are experts at quickly recognizing faces, so there must be specific cortical architecture and resources dedicated to face processing, right? Say you haven’t done a proper literature search and found Kanwisher et al., 1997 or you EVEN missed the compelling response by Haxby et al., 2001 that casts doubt onto the previous finding that there even IS such a brain module. All experiments in this workbook will be constructed to address this research question while slowly adding complexity to your experimental design.

This document is designed to be an instructive tool that illustrates functional magnetic resonance imaging (fMRI) experimental design using the fMRI Design Explorer. For Unit 1, It is assumed that you have the program in working order and that the program is open and ready to build an experiment. The installation section will provide you all system requirements, dependencies, and installation steps. You can save your progress in an experiment file and return at a later time. This workbook is meant to accompany PSYC 411 / PSYC 888P / NACS 728F - *Introduction to Functional Magnetic Resonance Imaging* at the University of Maryland - College Park, however, it can be useful to any who wish to begin using the fMRI Design Explorer program for all of their fMRI experimental design needs.

References

- Haxby, J. V., Gobbini, M. I., Furey, M. L., Ishai, A., Schouten, J. L., & Pietrini, P. (2001). Distributed and overlapping representations of faces and objects in ventral temporal cortex. *Science*, *293*(5539), 2425-30.
- Kanwisher, N., McDermott, J., & Chun, M. M. (1997). The fusiform face area: a module in human extrastriate cortex specialized for face perception. *The Journal of Neuroscience*, *17*(11), 4302-4311.



Introduction

About this workbook

This workbook provides an interactive and comprehensive description of the functionality of the fMRI Design Explorer. When this PDF is viewed electronically, it offers a number of helpful internal and external hyperlinks. In addition, hovering over the hyperlink with a mouse should result in a preview of the link's content. These hyperlinks have been tested on both Mac OS X and Linux operating systems. However, individual operating system and PDF viewer idiosyncrasies may prevent functional hyperlinks. Please report all hyperlink issues to Dustin Moraczewski at dmoracze@umd.edu.

Internal Hyperlinks

Hyperlinks colored [green](#) refer to internal links. Clicking an internal link will navigate the PDF to the corresponding section or figure. For example, if the text refers to [Figure 1](#), clicking on this link will navigate the PDF to Figure 1. In addition, the [Table of Contents](#) is fully interactive, however (as you probably noticed) the text color is not [green](#). This decision was made for readability.

External Hyperlinks

Hyperlinks colored [blue](#) refer to external links. Clicking an external link will initiate behavior external to the PDF. There are three types of external links:

1. URL link - this will open a browser and navigate to a specific URL.
2. Email link - this will open an email client, if one is configured (currently does not work on the University of Maryland's GLUE network)
3. Download link - this will download relevant files (e.g. the Design Explorer installer). This link may also open a browser that navigates to Dropbox on certain operating systems.

For example, if the text refers to the [AFNI](#) package for fMRI analysis, clicking on the corresponding link will open a browser that navigates to the AFNI homepage.



Download and Installation

System Requirements

64-bit machines running Mac OS X and Linux have been tested.
It is likely that the program will run on others and may fail with some.

Windows operating system is not supported.

Dependencies

Mac OS X – [XQuartz](#) is required.
Linux – no dependencies.

Download

Please download the current Design Explorer installer [here](#).

Install

1. Open Terminal console.
2. Navigate to the directory that contains the installer.

```
> cd <path_to_installer>
```
3. Change installer to executable, changing the version number appropriately.

```
> chmod 755 Install_DesignExplorer_XXXX-XX-XX.XXXX
```
4. Run the installer.

```
> ./InstallDesignExplorer
```
5. Pay attention to the output in the terminal. If there is an error, you should report this text to the developers.
6. Once the installer is finished, an executable file called `DesignExplorer` will be deposited into your home (`~`) directory.

Run

1. To run the Design Explorer you can either double click on the `DesignExplorer` file OR you can run the program in the terminal:

```
> ~/DesignExplorer
```

The executable file `DesignExplorer` is a bash script that points to a `~/DesignExplorer` hidden directory that contains a python distribution, all dependencies, [AFNI's 3dDeconvolve](#), and the [default project](#).

Uninstall

To uninstall the Design Explorer, simply open the terminal and type:

```
> rm -rf ~/DesignExplorer
```

Be aware, though, that this will also delete the [default project](#) as well.

Please email [Dustin Moraczewski](#) with any issues related to installation. Be sure to be as descriptive as possible and provide your system configuration, what you did, and what was the output.



Overview of Program

To see the forest rather than jumping right to the trees, this section gives users a quick overview of the multiple large-scale features of the Design Explorer. In addition, this section provides tips on [getting started](#), some [general information](#), and an [overview of the interface](#).

Getting Started

Upon [executing](#) the Design Explorer, the Project Launcher window will appear:

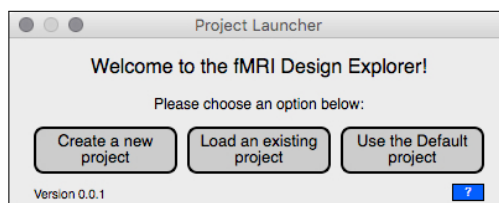


Figure 1: Project Launcher

Users have a three choices regarding how they want to initialize a project:

1. [Create a new project](#)
2. [Load an existing project](#)
3. [Use the Default project](#)

Create a new project button

To create a new project within a user-defined location, choose this option. A file directory dialog window will appear and the user will need to choose a location for their new project. Typically, prior to the creation of a new project, users will create a unique directory for their project. This unique directory will ensure that all files relevant to the project are contained within a single location.

1. Choose the **Create a new project** button
2. Navigate to desired directory
3. Click **choose**
4. Enter project name
5. Click **ok**

The new project will be created and the main window will appear ([Figure 2](#)).

Load an existing project button

If a previous project had been created within a user-defined location (using the [Create a new project](#) button), users can choose to load this already existing project.

1. Choose the **Load an existing project** button
2. Navigate to project's directory
3. Click **choose**

The project will load where the user left off, including all variants.



Use the Default project button

The default project exists so that users can rapidly create and interact with experimental designs without cluttering up the file system. Upon initialization of the default project, the Design Explorer will deposit a directory called `default_project` within the `~/.DesignExplorer` directory and the user will be asked to name the project. If a default project already exists, clicking the **Use the Default project** button will open the previously saved default project.


To remove the default project, open a terminal and type:

```
> rm -rf ~/.DesignExplorer/default_project
```

General Information

Autosave

All changes within a valid design are autosaved, thus, there is no need to save manually. However, there is a caveat to the autosave feature: in order for changes to be saved, the changes must occur within a valid design and be valid with respect to the parameters of interest. For example, the autosave feature will not save changes under the following scenarios:



1. If events are defined and/or configured without first creating a design using the  button, the changes will be lost because they did not occur within the context of a design.
2. Entering a character instead of a number into the **TR** field would not be saved since this field expects a numeric value.

These scenarios are meant to represent the types of behaviors that would fail to autosave and do **not** encompass all possibilities.

Overview of Interface

This section gives a brief overview of the main interface that is common to all Tabs. Please see the [Tabs](#) section for functionality unique to each Tab. [Figure 2](#) depicts the large-scale structure of the Design Explorer interface.

Consider the number annotations in [Figure 2](#):

1. The [Tab](#) menu:
This menu consists of the main functional divisions of the Design Explorer. The tabs consist of [Design](#), [Visualize](#), and [Simulate](#).
2. The Design menu:
This menu consists of the user-defined designs currently in the workspace. In [Figure 2](#), this menu is currently blank which means that no designs have been created yet. A design must be created before defining or configuring events.
3. The New Design button :
This button adds a new design to the Design Menu. When this button is clicked, users will be asked to enter a name for the new design. All designs must have unique names.
4. The [Help Button](#) :
This button is available in all three tabs.
5. The Plot window:
This window is available in all tabs, however, the information presented within this window is unique to each tab. For example, the plot window in the [design tab](#) shows the event plots whereas this window in the [visualize](#) and [simulate](#) tabs show the ideal and simulated BOLD signal, respectively.



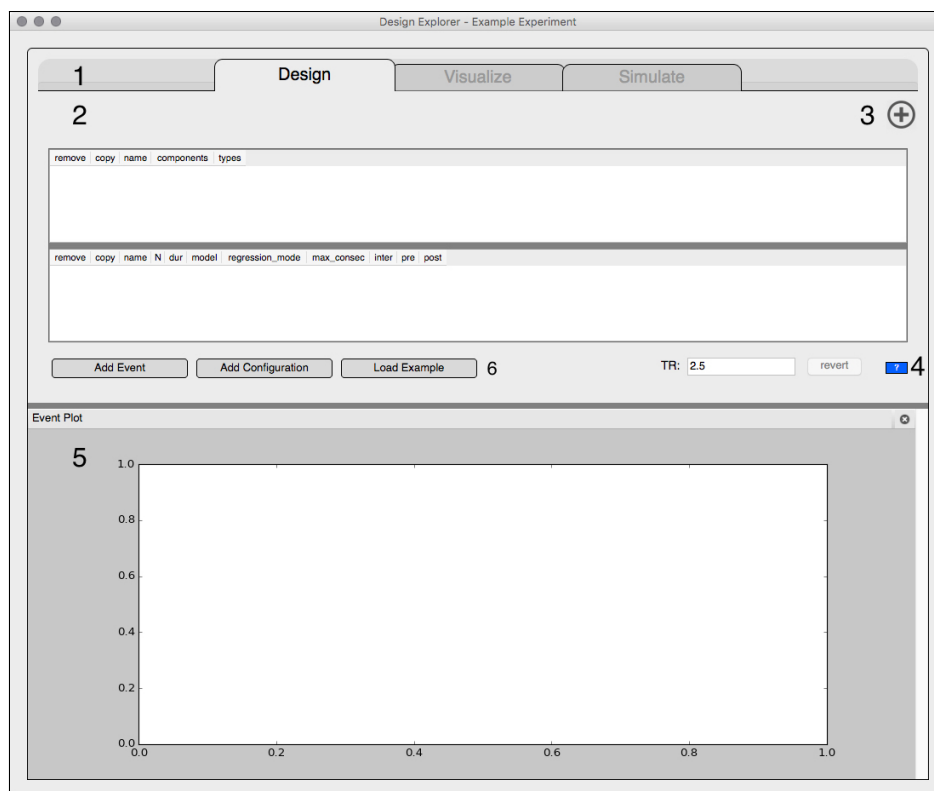


Figure 2: Overview of Interface

6. The [Load Example](#) button:

Click this button to load specific examples from this workbook.

Tabs

This section gives a brief, big-picture overview of the functionality of each of the tabs. Please see the corresponding chapters for greater detail regarding the implementation of the features.

Design

The purpose of the Design tab is three-fold:

1. Define events into an event hierarchy
2. Configure the various parameters of the events
3. View experimental design as an event plot

There are multiple ways in which users can define their event hierarchy, see the Event Hierarchy chapter for more information. With regards to event configuration, users have complete control of most parameters: number of events, event duration, hemodynamic response models, maximum consecutive events of the same type, pre, inter, and post baseline periods. Both events convolved with a hemodynamic response and not are configurable.

Visualize

The purpose of the visualize tab is to view the idealized (no noise added) BOLD signal on your experiment. Users can also view event-specific signal to examine the relative contributions of each event to the overall BOLD signal. This signal is derived from the events that are configured to include a hemodynamic response



model. In addition, the Event Configuration table is also supplied so that users can make adjustments to the event parameters and see the changes in the BOLD signal in real time.

Simulate


The purpose of the Simulate tab is to use an experimental design to simulate BOLD signal according to user-defined event-specific β weights, event-specific noise, as well as global noise. Then, once new BOLD signal is simulated, users can examine the error in the recovered β estimates. The number of simulations is a user-defined parameter.

Help

The Load Example Button

The load example button is a quick and easy way to load the designs discussed in the units of the workbook. Thus, users can load an example quickly and begin interacting with the designs without spending time defining and configuring each individual event.

The Help Button

The help button , as seen in the lower right corner of [Figure 1](#), will open this user manual to provide further assistance on the implications of these decisions. This button is available in the [Project Launcher](#), [Design tab](#), [Visualize tab](#), and [Simulate tab](#).

Debug Windows

In the current version, initializing the Design Explorer will also open two other windows: a terminal window and a debug window. Each of these windows provides different information about the background processes. Overall, these windows are not necessary to keep in view, however, if there are any errors or the program is not behaving as you expected it, the output of these windows can help.

Further Assistance

If you find a bug, please [Dustin Moraczewski](#) with a description of what happened and any text you see in these windows. In the interest of fixing all bugs, the more information provided to the developers the better. A detailed description of what you did, screen shots, and debug and terminal window text would provide a helpful overview in order to diagnose what happened.



Unit 1: The Hemodynamic Response

1.1 Overview

Before constructing an experiment, we must first understand how specific, transient events could be reflected in neurobiological terms. The hemodynamic response in the brain is slow relative to action potentials at the single-neuron and neural-population level. Therefore, this unit will give you the intuition of how events, which are computed by the brain at the neural level, translate to increased metabolic activity to replenish oxygen to the neural cell bodies.

1.2 Define and Configure Events

First we will explore how a single, transient event elicits a hemodynamic response. Here we are making an assumption that there is a one-to-one correspondence between the hemodynamic response and the event in question, say the presentation of a picture of a face. This is an ideal example for educational purposes, and in practice real data are far more variable. We will begin by setting up the stimulus presentation so that we can build a larger experiment from this template in later units.

When defining events, it is helpful to think about hierarchies. In the current example, our event hierarchy will look like [Figure 3](#). More on event hierarchies in Unit 2.

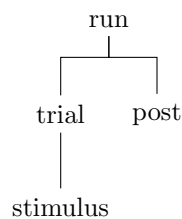


Figure 3: Single stimulus event hierarchy

Now let's build it. For the purposes of this workbook, column names (see below) will be in **bold** and text that needs to be entered into text boxes will be in "quotes". Begin in the [Design](#) tab to construct your experiment:

Design | Visualize | Simulate

Single Event ■

1

	remove	copy	name	components	types
1	X	*	trial	stim	face
2	X	*	run	trial	

2


	remove	copy	name	N	dur	model	regression_mode	max_consec	inter	pre	post
1	X	*	run	1	1			0	0	0	0
2	X	*	trial	1	1			0	0	0	20.0
3	X	*	stim	1	1	dmBLOCK(1.0)	AM1	0	0	0	0

3

Add Event
Add Configuration
Load Example
 TR: 2.5 revert

Figure 4: Single stimulus event configuration



1. Click the **New Design** button 
2. Enter “Single Event” as the name of the design, click **ok**
3. Add events to your experiment by clicking **Add Event** (Figure 4, blue circle), which will insert a column in the Event Definition table (Figure 4, ①). You will add two different parts to your experiment: “run” and “trial”. Add “run” and “trial” in to the **name** column of the first and second rows of the table, respectively. While these two events are not immediately applicable to a presentation of one stimulus, it will become clear once we expand the complexity of the experiment.
4. Next continue to define the events in the Event Definition table. The run event will consist of a **component** named “trial”. Intuitively, this illustrates that within the run event you will have smaller events called trials. For the trial Event (row 2 of the Event Definition Table) the **component** will be named “stimulus” and we will define one **type**: “face”. This definition is saying that within each run we will have events called trials. With each trial we will have events called “stimulus”, and, for this setup, we will only have one type of stimulus: “face”. “stimulus” will correspond to the onset of each face presentation.
5. Now we need to configure our events. Keep in mind that we are constructing a special case of an experiment that only has one stimulus presentation, we will build on this design in later units. Click the **Add Configuration** button (Figure 4, green circle) to insert a line into the Configure Events table (Figure 4, ②). We will need to insert 3 rows into this table, corresponding to “run”, “trial”, and “stimulus”. Enter each of these into the **name** column, respectively. It does not matter which order these rows are in, however, it is essential that the text in the **name** column be identical to the events in the Event Definition table.
 - (a) For the run configuration, we can use all of the defaults, except for **post** which denotes a post-run event where the participant views a fixation cross for a defined period of time. For this example, enter a post-run fixation event of 20 seconds.
 - (b) For the trial configuration, we can use the default settings.
 - (c) For the stimulus configuration, we will use the defaults except that we need to define a hemodynamic response model with which to convolve the event. In the **model** column, we will select **dmBLOCK** with an amplitude of 1. We also need to select a **regression_mode** of AM1. AM stands for amplitude modulation, which means that the height of the hemodynamic response will be dictated by the value entered into the dmBLOCK model specification. More on different types of models will be discussed later.
6. For example purposes, we will also change the **TR** (repetition time) to an unrealistic 0.01 seconds (Figure 4, ③). You will explore the effects of a changing TR on the response behavior in the unit exercises.

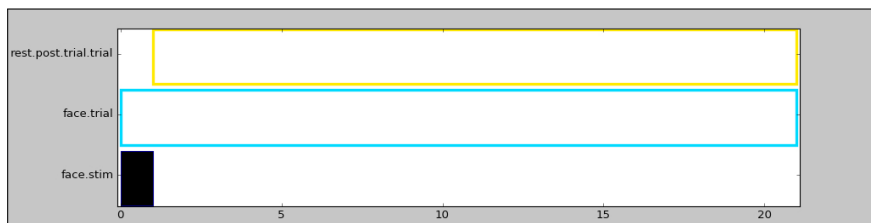


Figure 5: Single stimulus event plot

Once you have defined and configured your events, your single-stimulus experiment should look like Figure 5. In this plot, we can see a graphical illustration of the events in our experiment with events on the Y-axis and time in seconds on the X-axis. The top line (empty yellow box) represents the 20 second post-run fixation we entered under the Run event. The second line (empty cyan box) denotes the trial event we defined. Finally, the solid black box shows the actual presentation of the stimulus. Events are depicted



as either empty or solid boxes. Empty boxes correspond to events where we are not going to estimate the hemodynamic response whereas solid events are events where we will estimate this effect (the manner in which we estimate the response is set using the **model** and **regression_mode** columns of the Event Configuration table).

1.3 Visualize Event Signal

Now that we defined and configured the events in our simple design, we can now visualize the translation of these events into a hemodynamic response. Keep in mind that this is an idealized example. *In a perfect world*, these response plots depict that a presentation of a face elicits an *ideal* hemodynamic response - real experimental data are never this clean.

To view the response plots, click the **Visualize** tab (Figure 6, blue circle):

	name	N	dur	model	regression_mode	max_consec	inter	pre	post
1	run	1	1			0	0	0	0
2	trial	1	1			0	0	0	20.0
3	stim	1	1	dmBLOCK(1.0)	AM1	0	0	0	0

Collinearity Metric Quality
Variance Inflation Factor Normalized Standard Deviation

View Signal TR: 0.01 revert

Figure 6: Single stimulus visualize configuration

Just as in the **Design** tab (Figure 3), Figure 6, ①, shows the Event Configuration table that you created earlier. Here you will be able to edit the event parameters and visualize how the hemodynamic response changes based on your design changes. You can also change the TR length in Figure 6, ②. To visualize the ideal hemodynamic response of the defined events, click **signal** button (Figure 6, green circle).

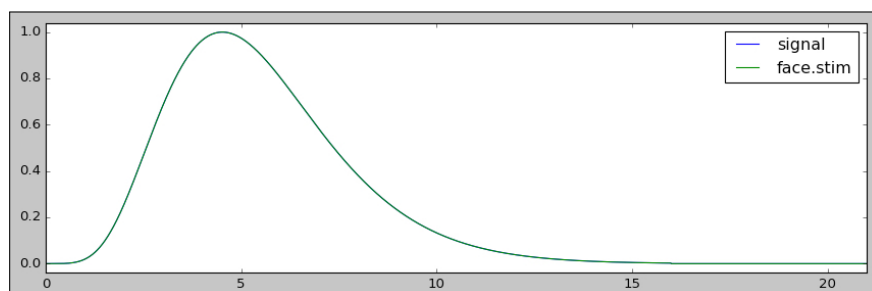


Figure 7: Single stimulus hemodynamic response

Figure 7 shows the hemodynamic response corresponding to the events in the experiment. Time is on the X-axis and hemodynamic amplitude (the strength of the response) is on the Y-axis. Time 0 would correspond to the onset of the 1-second long stimulus. Note that the peak of the hemodynamic response is at roughly the 5 second mark, which should be consistent with what you have learned about the time course of the response. Now that you have configured the events and can visualize the corresponding response, you are now equipped to explore the end of the unit exercises.



1.4 Unit Exercises

1. Change the TR to a more realistic 2.5 seconds. How does the response change and why?
2. Change the TR to .1, 1, 2, 3, 5, 8, and 13. What happens as the TR length increases? Pay attention to the shape of the curve.
3. What happens to the amplitude as you go from 5, 8, and 13 second TRs? Why?
4. Change the TR back to 0.01 seconds. Now change the duration of the stimulus to 3, 5, 8, and 13 seconds. How does the response change and why?
5. Reset the stimulus duration to 1 second and remove the post Run fixation period (set to 0). How does the response change and why?
6. Reset the post Run fixation period to 20 seconds. Now change the number of Stimulus presentations from 1 to 2. How does the response and amplitude change? Is this what you expected? why?
7. Now change the **inter-stimulus interval** (ISI, the amount of time between stimulus presentations) to 3, 5, 8, and 13. How does the shape and amplitude of the response change?
8. Say you want a proper control for your study. You realize that comparing the presentation of a face to baseline (a fixation cross) does not isolate brain regions that solely respond to faces. In a faces vs. baseline comparison there are also low-level visual differences in luminance, contrast, edges, and complexity of the visual stimuli. To further control for these effects you decide to show a scrambled image as well. These scrambled images will have the same luminance, contrast, edges, and complexity as the face stimuli, but they will not be in the form of a face. Thus, you want to add a new condition: Scrambled.
 - (a) Return to the **Design** tab (Figure 4).
 - (b) Add another **Type** (separated from “face” with a comma i.e. “face,scrambled”) to the trial event called: “scrambled”, Figure 4, ①.
 - (c) In the Event Configuration table (Figure 4, ②), set the number of stimulus events to 2.
 - (d) Return to the **Visualize** tab (Figure 6, blue circle) and view the new signal (Figure 6, green circle).

What does the overall signal (blue line) look like? Can we differentiate the two conditions from this signal? Change the **Inter-Trial Interval** (ITI, space between trials) to 1, 3, 5, 8, and 13 seconds. How does this change affect the overall shape and amplitude of the signal?



Unit 2: Event Hierarchy

2.1 Overview

When building an experiment, it can be useful to think of each event as nested within an event hierarchy rather than a series of independent events. It is not only conceptually useful, it is how experiments are defined in the fMRI Design Explorer. The program then uses this hierarchy to optimize stimulus timing, which makes the world a better place. This unit will explain how to think of an experiment as a hierarchy of events and how to define that hierarchy in the Design Explorer.

2.2 Hierarchy Outlined

In our face viewing experiment we can think of two types of events: when a stimulus is on the screen and when the screen is blank (the inter-trial interval). Now, imagine that every single face presentation is independent of each ITI, that is, no stimulus is bound to an ITI. The event structure of this experiment would look something like [Figure 8](#). However, there is a practical constraint to a “minimal hierarchy” design - typical

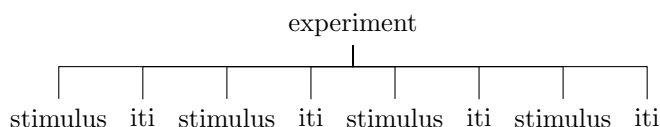


Figure 8: Minimal Hierarchy

fMRI experiments last more than one functional run. Researchers like to exert control over their experiments and so, to ensure that there is no unintended effect of run on the data, runs are typically counterbalanced across participants. This means that the order in which the runs are presented to participants is shuffled. In a two run experiment (with runs A and B), some participants will receive run A first whereas some participants will receive run B first. It is important to ensure that each run is not significantly different from other runs with regard to the number of trials and timing. Thus, we can think about the stimulus and ITI events as occurring *within* each run. The event structure of an experiment that considers stimuli and ITI as nested within run is depicted in [Figure 9](#). While this ensures that each run is independent, there

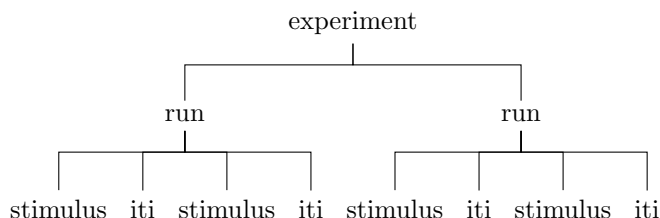


Figure 9: Run Level

are still many problems with constructing your experiment in this manner. If we randomized the order of the run event in [Figure 9](#), we could still counterbalance runs and test to ensure that runs did not differ in timing. However, we also want to make sure that the order of stimulus presentation does not affect our results. While the above examples only have 4 stimulus events (2 per run), keep in mind that the typical experiment has many more. We want to ensure that there are no systematic effects with respect to stimulus timing. For example, we want to make sure that participants cannot predict what trial will be next or else expectancy effects could influence the results in unintended ways. If we were to randomize the stimulus and ITI events in [Figure 9](#) we would have a problem. We have the stimulus and ITI at the same level which means that we could potentially get two (or even three!) ITI events back to back, and the same goes for the stimulus events. We want to make sure that each stimulus is followed by an ITI, but we want to be able to randomize the stimulus. Thus, we can create a hierarchy like [Figure 10](#): Here we are able to randomize run



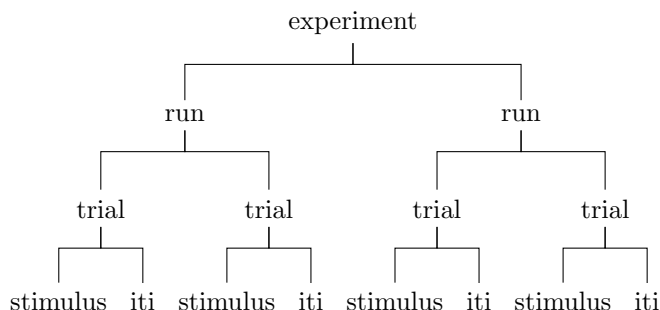


Figure 10: Trial Level

and trial order while ensuring that every stimulus is followed by an ITI event.

NOTE: In the Design Explorer, the Experiment level of the above hierarchies is automatically defined and does not need to be included

2.3 Hierarchy in the Design Explorer

Since the Design Explorer assumes a minimal run hierarchy, this is the level at which we can begin implementing hierarchical designs in the program. In the program, we will begin by building the experiment outlined in [Figure 9](#).

To do this:

1. Begin in the [Design](#) tab
2. Choose **New Design** button 
3. Enter “Hierarchy” as the name of this variant
4. Click **ok**

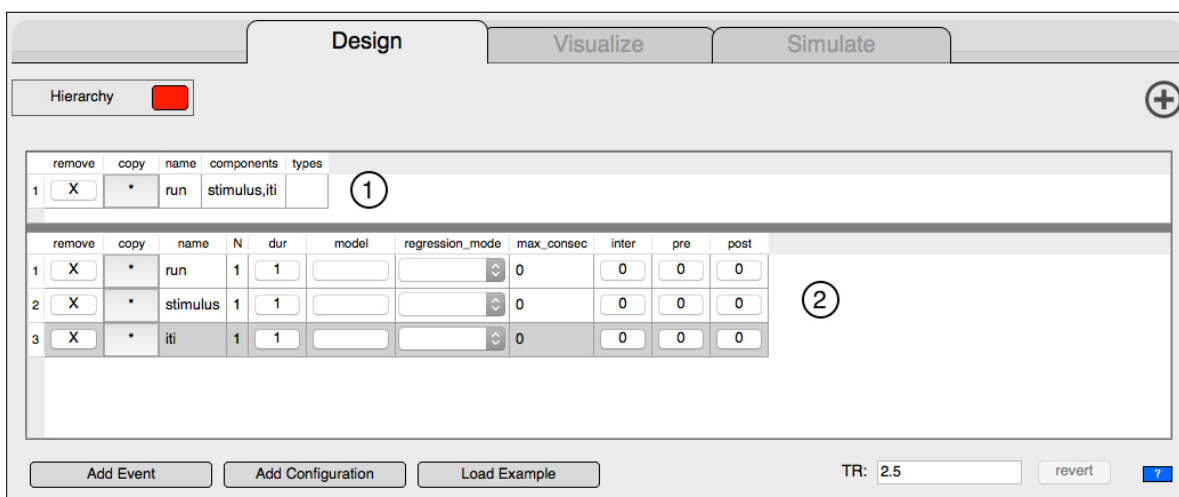


Figure 11: Run Level Configuration

To build the design:

1. Click the **Add Event** button to add a row into the Event Definition Table ([Figure 9](#), ①)



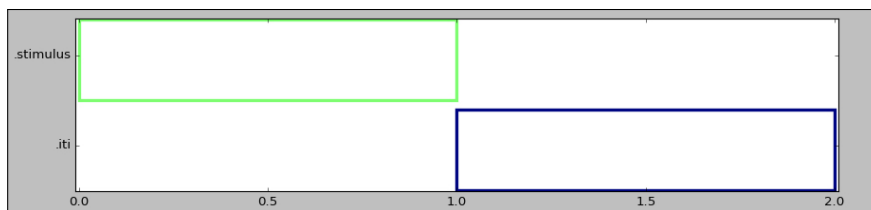


Figure 12: Run Level Events

2. While the order of the rows does not matter, there must always be an event named run. Enter “run” into the **name** column of the first row in the Event Definition table
3. In the **components** column of the run event, enter “stimulus,iti”
4. Click the **Add Configuration** button three times to add three rows into the Event Configuration table (Figure 9, ②)
5. Add “run”, “stimulus”, and “iti” into the **name** column for each of these events

After you have defined and configured all events, your event plot should look like Figure 12. While this does not look like a very interesting experiment, it outlines the events you defined. The empty green box shows the event that corresponds to the stimulus, whereas the empty blue box corresponds to the ITI event. Now that you have built this experiment, you can explore why this event organization is sub-optimal:

Run Level Exercises

1. Change the number of stimulus and ITI events to 2, and then 3. How did the event plot change? Is this what you would want to happen?
2. To see the hierarchy in practice, change the number of runs to 2, and then 3. how did the event plot change?

The experiment, as we have defined it, groups all like-events together. This is probably not what we would want out of an experiment because it makes no sense to have a group of stimuli followed by a group of ITI events.

In order to define a much more flexible experiment that knows to put an ITI event *always* after a stimulus event, we need to implement the hierarchy as defined in Figure 13.

remove	copy	name	components	types
1	X	*	run	trial
2	X	*	trial	stimulus,iti

remove	copy	name	N	dur	model	regression_mode	max_consec	inter	pre	post
1	X	*	run	1	1		0	0	0	0
2	X	*	trial	1	1		0	0	0	0
3	X	*	stimulus	1	1		0	0	0	0
4	X	*	iti	1	1		0	0	0	0

Add Event Add Configuration Load Example TR: 2.5 revert ?

Figure 13: Trial Level Configuration



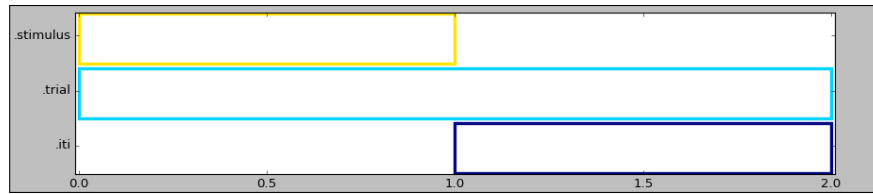


Figure 14: Trial Level Events

In the **Design** tab:

1. Click the **Add Event** button to add a row into the Event Definition Table (Figure 13, ①).
2. Enter “run” into the **name** column of the first row in the Event Definition table.
3. In the **components** column of run, enter “trial”.
4. Add another row into this table. Under **name**, enter “trial”.
5. In the **components** of the trial event enter “stimulus,iti”. Thus, for each “run” we will have events called “trial” and each “trial” event will consist of “stimulus” and “iti”.
6. Click the **Add Configuration** button four times to add four rows into the Event Configuration table (Figure 13, ②).
7. You will need to name the rows the same as in the Event Definition table. Enter “run”, “trial”, “stimulus”, and “iti” in the **name** column of the rows, respectively.

Now that you have defined and configured the events in this hierarchy, your event plot should look like Figure 14. You still have empty boxes for the stimulus (yellow box) and ITI (blue box) events. In addition, there is another empty box (cyan box) that encompasses the stimulus and ITI events. This is the box that corresponds to the trial event.

Trial Level Exercises

1. Change the number of stimulus and ITI events to 2, and then 3. How did the event plot change? Is this different than in the previous example?
2. Change the number of stimulus and ITI back to 1. Now change the number of trials to 2, and then 3. How did the event plot change?
3. Now add a post-run fixation period of 10 seconds to allow the hemodynamic response to return to baseline. To do this, there is no need to define another event, but rather you change the **post** column of the run row in the Event Configuration table. This will add a set amount of time after every run event.



2.4 Unit Exercises

While some experiments can be as simple as a trial that consists of a stimulus and an ITI, many are more complicated. Take, for example, a classic monetary incentive delay task ([Figure 15](#)). In this experiment, participants can win money if their reaction time is fast enough on a button pushing task. Within each trial there is a cue where the participants are told what condition they are in (e.g. how much money they could win). This is followed by a delay period where they are waiting for the target. Then the target appears and participants must push a button as fast as they can (while holding still!). After the target, they are notified whether they pushed the button fast enough. Finally, this is followed by an ITI where participants are waiting for the next trial. Keep in mind that there are usually different trial types as well, this is where the **types** column of the Event Definition table will come in to play.

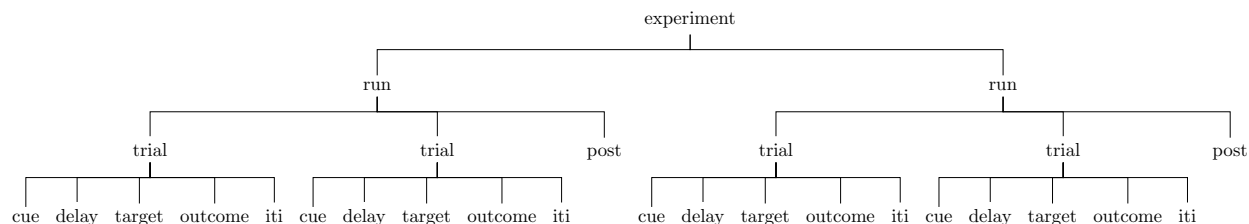


Figure 15: Monetary Incentive Delay Task Hierarchy

1. Create an experiment in the Design Explorer that reflects the hierarchy in [Figure 15](#). For this exercise, create 2 runs that consist of 2 trials each. Each trial will need a cue, delay, target, outcome, and ITI event. Finally, enter a post run fixation of 10 seconds after each run. While this exercise only includes 2 trials per run, keep in mind that there are many many more trials in a real experiment.



Unit 3: Block Designs

3.1 Overview

In unit 1, we discussed how single, transient events can be translated into a hemodynamic response and how event duration, ITI, and other aspects can change the subsequent response. Then, in unit 2, we discussed how experiments can be thought of as hierarchical events and how to define these events in the Design Explorer. Now it is time to put these together and design your first experiment. Consider two properties of the hemodynamic response: 1) the hemodynamic response is slow compared to electrical activity at the neural level and 2) two (or more) events in quick succession will elicit a greater response (in terms of amplitude) compared to a single event. It would be beneficial to design an experiment that exploits these properties to increase our statistical power to detect an effect of one condition showing greater response compared to another condition. These properties are what led early fMRI researchers to design experiments in block designs. In this unit you will construct a block design for your “faces” experiment and see how this experimental design translates into a hemodynamic response.

3.2 The Design

Block designs are built around a block of repeating stimuli (of the same class) in quick succession followed by a block of rest (baseline; usually a fixation cross). [Figure 16](#) shows the event hierarchy of the design we will construct. While this figure shows two task blocks and two rest blocks, keep in mind that an actual experiment will consist of many more blocks. Also, note that the rest blocks have smaller events also called rest. This is to show that the duration of the rest blocks will be equivalent to the duration of the task blocks.

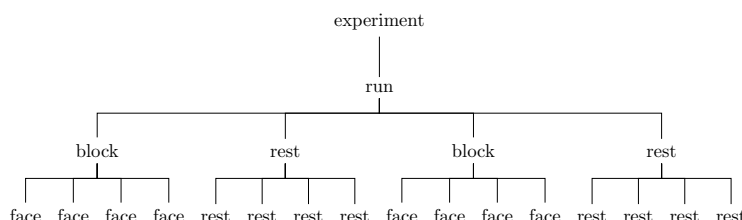


Figure 16: Block design event hierarchy

3.3 Define and Configure Events

Now, let's build the experiment in the Design Explorer. Refer to [Figure 17](#).


1. Click the **New Design** button .
2. Enter “Block Design” as the name of the design, click **ok**
3. Add two rows into the Event Definition table, [Figure 17](#), ①, using the **Add Event** button.
4. **Name** the first row “run”, and add a **component** called “block”.
5. **Name** the second row “block”, add a **component** called “stimulus”, with one **type** called “face”. Thus, our event “run” will consist of “block” events which, in turn, will consist of “stimulus” events that have one type: “face”.
6. Now, configure the events in the Event Configuration table, [Figure 17](#), ②: Add three rows using the **Add Configuration** button.
7. **Name** the first row “run”. You can keep the defaults except for the **post** column where you will enter a 20 second post-run fixation period.



Figure 17: Block design configuration

8. **Name** the second row “block” and set the number of blocks (**N**) to 4. This is the event with which we will want to model the hemodynamic response. In the **model** column, select “dmBLOCK” with an amplitude of 1. In the **regression_mode** column, select “AM1”. Finally, we want to make sure that there is a consistent rest period in between blocks. Set the **inter** column to equal 20 seconds.
9. **Name** the third row “stimulus”. Set the number of stimuli (**N**) to 20. We can keep the duration equal to 1 second, this makes our block duration equal to 20 seconds.

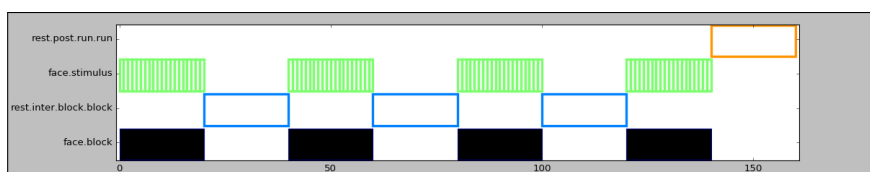


Figure 18: Block design event plot

After defining and configuring your events, your event plot should look like [Figure 18](#). The **green** boxes show an individual presentation of a face (twenty per block). Compare this to the **black** box that corresponds to the block as a whole. This block is solid colored because this is the event we will use to model the hemodynamic response. Instead of estimating a response for each stimulus, we are estimating a response for each block. The **blue** boxes are the 20 second inter-block fixation period and the **orange** box corresponds to the 20 second post-run fixation block. While there are a few different events in this design, essentially we are telling the program that we want to alternate 20 seconds of stimulus with 20 seconds of rest.

3.4 Visualize Event Signal

Now that the experiment is constructed, we can see what the hemodynamic response would look like under ideal conditions. Change to the **Visualize** tab and click the **signal** button to see this response. Your response should look like [Figure 19](#).

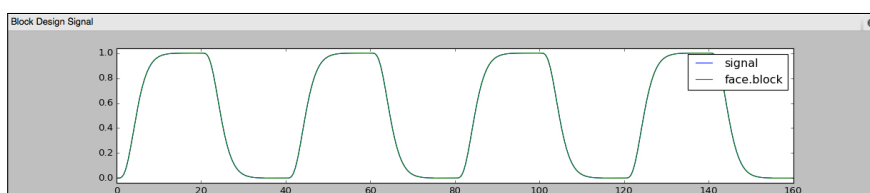
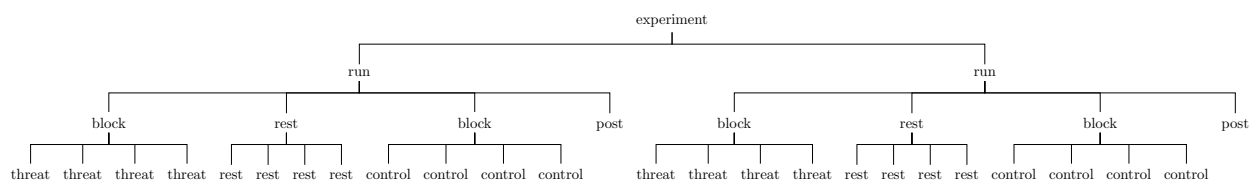


Figure 19: Block design response plot



3.5 Unit Exercises

1. Change the TR to a more realistic 2 seconds, then an unrealistic 5, 10, and 20 seconds. How does the response change?
2. Change the TR back to 2 seconds. Now let's make the rest blocks shorter: change the **inter** column of the block row from 20 to 10, 5, 1, and 0 seconds. How does the response change?
3. Change the **inter**-block interval back to 20 seconds. Now return to the **design** tab and enter our control condition, scrambled, as another **type** of stimulus in the Event Definition table. How does the event plot change? Is this an ideal event ordering, why or why not?
4. In the **Design** tab, change the **max_consec** column to 1. This means that we want a maximum of one consecutive block per trial. Now, the experiment should alternate blocks by condition. Return to the **Visualize** tab, and view the **signal**. As in question 2, change the **inter**-block interval from 20 to 10, 5, 1, and 0. What happens as the inter-block interval gets smaller and why might this spell trouble for our statistical analyses?
5. Now, create an all new design (\oplus). Call this experiment whatever you want. In this experiment you will show your participants threatening pictures (i.e. someone pointing a gun at the camera) and a control condition. Your experiment should have the following hierarchy:



With respect to event timing, each task and rest block should be 15 seconds long. Each run should consist of 10 task blocks and there should be no more than 1 consecutive block per trial. In addition, add a post-run fixation period of 15 seconds. Tell the program to model the hemodynamic response of the entire block length same as in [section 3.3](#).



Unit 4: Slow Event Related Designs

4.1 Overview

The previous unit explored the powerful fMRI design known as block designs. In this type of design, you take advantage of the fact that the BOLD signal scales linearly with the amount of stimuli, that is, more stimuli in a quick period of time will give stronger BOLD response (if the brain region in question is actually responsive to the stimulus) compared to baseline. While this design is effective at generating large differences in the amplitude of the BOLD response, there are some drawbacks. As we saw, for a block design to elicit the strongest difference in response, there needs to be adequate time for the response to return to baseline. Block designs are less about the response to an individual stimulus and more in line with the summation of many stimuli. Another type of design, event related, allows for multiple stimulus presentations while estimating the hemodynamic response of individual stimuli. This unit will focus on slow event related designs, which allows the hemodynamic response to return to baseline.

4.2 The Design

Initially, we will construct a flexible, but simple, slow event-related design to which we will later add complexity. We will continue to address our research question: where in the brain are responses stronger to faces compared to other stimuli? The event hierarchy of the experiment will look like Figure 20. Once difference between this design and the previous unit will be that the duration of the stimulus presentation (face events) and the duration of the ITI (rest) will not be equal. The stimulus presentation (i.e. face events) will consist of transient, 1 second, presentations whereas the ITI events will allow the hemodynamic response to return to baseline after estimating the response to a single stimulus event.

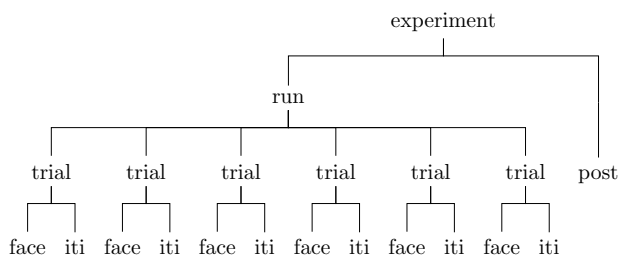


Figure 20: Slow event hierarchy

4.3 Define and Configure Events

We will define our events similar to the previous unit, with some important differences:

1. Using the **Add Event** button, add a row into the Event Definition table (Figure 21, ①). Add an event called “run” with a **component** called “trial”.
2. Add another row called trial with a **component** called “stimulus” with only one **type** called “face”.
3. Using the **Add Configuration** button, add an event into the Event Configuration table (Figure 21, ②) called “run”. We can use all of the defaults but, as before, we will enter a post run fixation of 20 seconds.
4. Add a row called “trial”, keep the defaults except for the number of events (N) in which we will add 20 events and change the inter-trial interval (ITI) to a constant 20 seconds.
5. Add another row called “stimulus”, keep the defaults except that this is the event where we will want to estimate a hemodynamic response. Under the **model** column select the **dmBLOCK** model with an amplitude of 1. Also, in the **regression_mode** column, select **AM1**.



Design | Visualize | Simulate

Block Design ☒ Slow Event Related ☒

	remove	copy	name	components	types
1	<input checked="" type="checkbox"/>	*	trial	stimulus	face
2	<input checked="" type="checkbox"/>	*	run	trial	

	remove	copy	name	N	dur	model	regression_mode	max_consec	inter	pre	post
1	<input checked="" type="checkbox"/>	*	stimulus	1	1	dmBLOCK(1.0)	AM1	0	0	0	0
2	<input checked="" type="checkbox"/>	*	trial	20	1			0	20.0	0	0
3	<input checked="" type="checkbox"/>	*	run	1	1			0	0	0	20.0

TR: 2.5

Figure 21: Initial Slow event related configuration

6. We can keep the TR at 2.5 seconds.

Once all events are defined and configured your event plot should look like [Figure 22](#). Each run (we have only prescribed one, for now) will consist of smaller events called trials. Within each trial there will always be two smaller events: a stimulus and an associated ITI. The stimulus event will be the presentation of the stimulus (e.g. a face) and the ITI event is an accompanying period of rest that allows the hemodynamic response to return to baseline. Of note in [Figure 22](#) is the solid event denoted **face.stimulus**. These are the events that will serve as the basis for the estimation of the hemodynamic response. The **green** box denotes the ITI, whereas the **blue** box denotes the face trial. This block box is colored as an empty box, however, the event is so short that the box does not look empty.

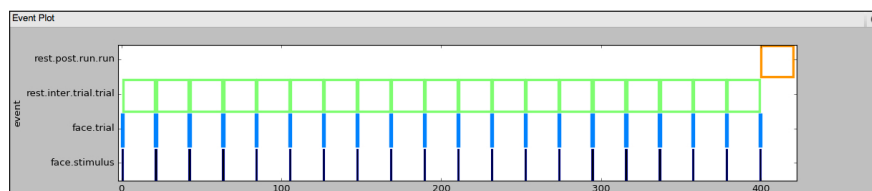


Figure 22: Initial slow event related event plot

4.4 Visualize Event Signal

To view the response plots, click the [Visualize](#) tab. You will see the [Visualize](#) tab's Event Configuration table which corresponds to table you configured in the previous step. As in the previous unit, you will edit the event parameters and explore the changes in the hemodynamic response plots. Click the **signal** button to see the response plot. Your initial response plot should look like [Figure 23](#).

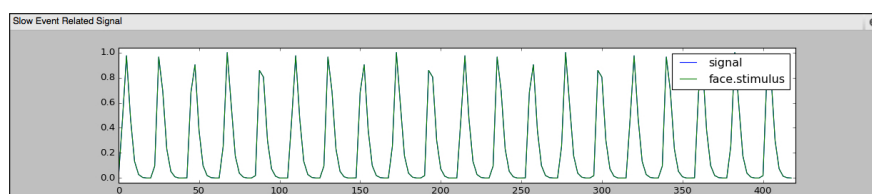


Figure 23: Initial slow event related response plot



4.5 Unit Exercises

1. Change the duration of the **ITI** from 20 to 15, 10, 5, 3, 2, 1, and 0. How does the signal change in amplitude and shape? Why?
2. Set the duration of the **ITI** back to 20 seconds. Now lets add our control condition from the previous unit: scrambled. Return to the **Design** tab and enter another **type** of trial called scrambled into the Event Definition table (**Figure 6**, ①). How did the event plot at the bottom of the **Design** tab change from [Figure 22](#) ?
3. How did the response plot in the **Visualize** tab change with the addition of the second condition? Change the duration of the **ITI** from 20 to 15, 10, 5, 3, 2, 1, and 0. When do the event-specific peaks in the response begin to disappear?
4. What is one reason that a long ITI of 20 seconds would be good to disentangle the relative contribution of each condition to the overall signal? **HINT:** think about the time course of the hemodynamic response



Unit 5: Assessing Design Quality

5.1 Overview

In addition to visualizing the event and signal structure of your design, the Design Explorer also gives you the ability to quantify design quality and compare to this quality to other designs. It is essential to quantitatively evaluate your design because you will not be able to choose an ideal timing based on visual inspection alone. This unit will focus on how to define and assess design quality.

5.2 Optimization

One feature of the Design Explorer is that it gives you the ability to optimize your experimental design based on design quality. This feature allows the computer to generate a random stimulus order and timing (with the constraints specified by the user) and compare quality metrics to choose the best timing. In order to compare the quality of multiple designs, each design should be optimized before examining the quality metrics. The program uses a **red** box to denote designs that need to be optimized and a **green** box to denote designs that have been optimized.

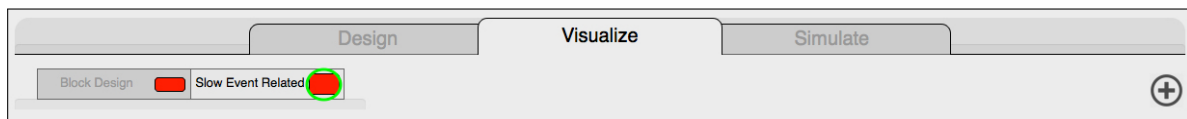


Figure 24: How to optimize

To optimize:

1. Click on the **red** box of the design you wish to optimize (Figure 24, **green** circle) - this will open the optimize window (Figure 25).
2. You can keep the defaults for now, but this window gives you the ability to customize your optimization. However, make sure you set the number of **iterations** to a sufficiently high number. For the purposes of this workbook, 1000 is good, but if you are fully optimizing a real study, you should choose a very large number like 10,000 or 100,000. 1000 iterations on a 2-core machine Macbook Pro takes about 40 seconds.
3. Click **run** to run the optimization. In the terminal window, you should see the iteration number. Once the optimization is finished, the optimization window will disappear and the optimization box will change to **green**.

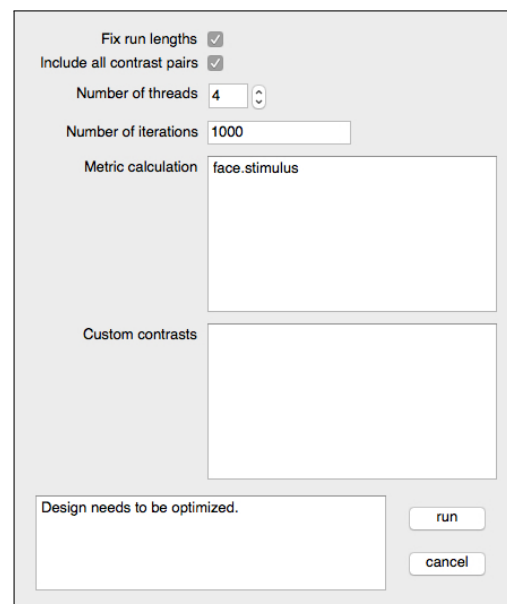


Figure 25: Optimization window

5.3 Quality Metrics

The following are quantitative measures that can tell you different things about the quality of your design, remember that to accurately compare designs, you must optimize the timing first.



Normalized Standard Deviation: Normalized standard deviation refers to the standard deviation in your beta estimates from the linear regression. You want the normalized standard deviation to be as small as possible, the smaller the uncertainty in your beta estimates, the better. This metric gives you a normalized standard deviation for each modeled event.

Metric Quality: This quantitative measure of design quality summarizes the normalized standard deviation for all events provided in the **metric** list during optimization. Thus, there is only one value per design. Optimization is required to examine this measure. Because this number summarizes the normalized standard deviation of the beta estimates, the lower this is the better.

Collinearity: This is the correlation between the time series of different events. For all events, you want the collinearity to be as small as possible.

Variance Inflation Factor (VIF): When there is a strong correlation between two events (i.e. say one event always follows another), also called collinearity, it becomes difficult to disentangle the relative influence of each individual event. The VIF is a measure of how much the variance of an estimated beta coefficient increases because of collinearity. You want the VIF to be as small as possible, as well.

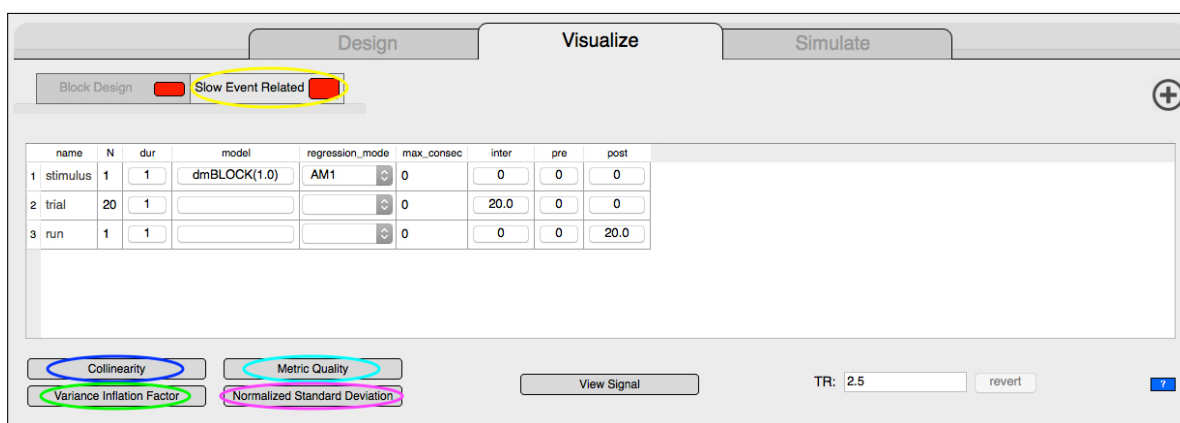


Figure 26: Visualize Quality Measures

To visualize the metric quality, click on the **metric quality** button (cyan circle). To visualize the collinearity metric, click on the **collinearity** button (blue circle). To visualize a histogram of the VIF, click on the **variance inflation factor** button (green circle). To visualize the normalized standard deviation metric, click on the **normalized standard deviation** button (magenta circle). In the next section we will create two different designs and compare the quality measures.

5.4 Compare Designs

Now we are in position to examine how small changes can quantitatively affect the quality of an experimental design. To do this, we first need to have 2 similar variants to which we can compare:

1. Right click on your Slow Event Related design tab, Figure 26, yellow circle.
2. Select **copy**
3. You can keep the default design name (Slow Event Related (2)). This will duplicate this variant so that you can change one parameter and compare.
4. We no longer need the Single Event design, so you can delete it:
 - (a) Right click on the Single Event tab
 - (b) Select **delete**
5. Make sure you optimize each of these designs using 1000 iterations.



Your **Design toolbar** should now look like [Figure 27](#):

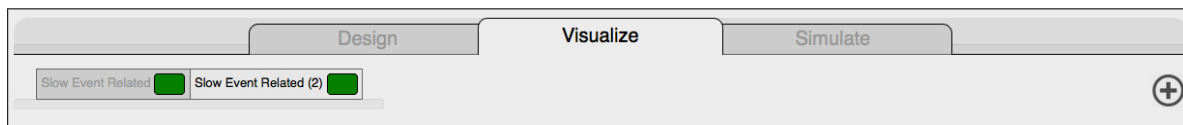


Figure 27: Design toolbar with copied designs

Now, in the **Visualize** tab, let's compare the VIF of both designs ([Figure 28](#) green circle):

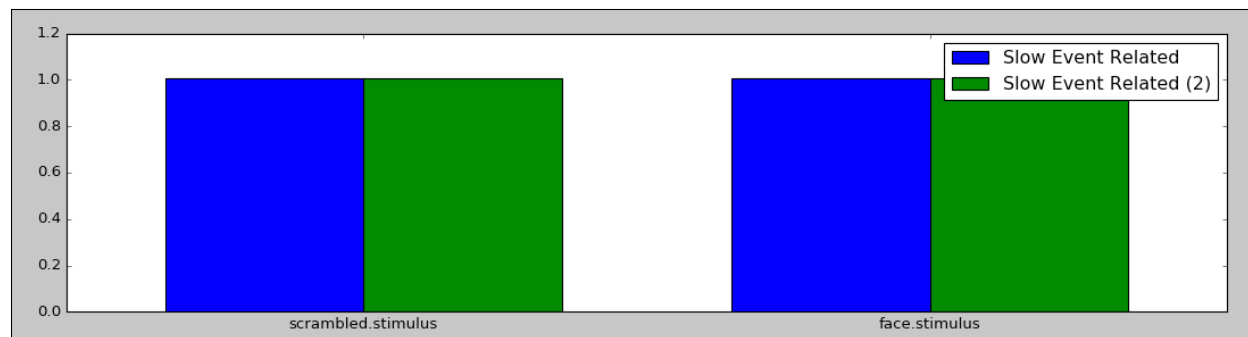


Figure 28: Variance inflation factor

It appears as though the VIF is the same for both designs. This makes sense because they are exactly the same, we haven't changed anything yet.



5.5 Unit Exercises

	name	N	dur	model	regression_mode	max_consec	inter	pre	post
1	stimulus	1	1	dmBLOCK(1)	AM1	0	0	0	0
2	trial	20	1			0	0	0	0
3	run	1	1			0	0	0	20.0
4	iti	1	20.0			0	0	0	0

Figure 29: VIF exercises event configuration

For the following exercises, ensure that both of your Slow Event Related variants look like [Figure 29](#). We will keep the original variant the same (Slow Event Related) and alter the copy (Slow Event Related (2)). Make sure that after every alteration in parameters that you re-optimize the design for a proper comparison. Also, make sure your TR is set to 2.5 seconds and then proceed to the following exercises:

1. Change the **ITI** on Slow Event Related (2) to 15, 10, 5, and 1 second. How does making the ITI shorter affect the VIF and the metric quality?
2. How does the response plot change with the shorter ITI and what does this mean in the context of how the VIF changes?
3. Change the **ITI** back to 20 seconds. How does making the **stimulus** duration longer affect the VIF and metric quality? Why? Test with a stimulus duration of 5, 10, 20, and 30 seconds.



Unit 6: Fast Event-Related

6.1 Overview

In the previous section, we learned how to construct and test the quality of a slow event related design. Slow event related designs are nice because they give the hemodynamic response a chance to return to baseline (or almost, depending on the duration of the ITI, as you saw). However, an MRI scan is very expensive and we want to make sure that we are efficient with the time. In addition, we want to make sure that we have enough stimulus presentations per condition to have enough power for statistical analysis. This section will explore how we can speed up stimulus timing while still trying to maximize experimental quality. To begin, we will construct an experiment with similar hierarchy as in Unit 4 (Slow Event Designs), see [Figure 30](#). In this hierarchy, “scram” is short for scrambled. The major difference between this design and the design in Unit 4 is that the timing between events will be much faster (which is not apparent in the [Figure 30](#), but will be more clear as we configure our events).

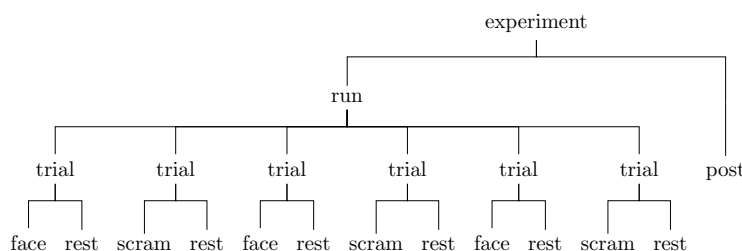


Figure 30: Fast event hierarchy

6.2 Define and Configure Events

Now let's define this in the Design Explorer. Your Event Definition and Event Configuration tables should look like the tables in [Figure 31](#).

remove	copy	name	components	types
1	X	stimulus	1	face, scrambled
2	X	run	trial	

remove	copy	name	N	dur	model	regression_mode	max_consec	inter	pre	post
1	X	stimulus	1	1	dmBLOCK(1.0)	AM1	0	0	0	0
2	X	trial	40	1			1	0	0	0
3	X	run	1	1			0	0	0	20.0
4	X	iti	1	5.0			0	0	0	0

Figure 31: Initial Fast event related configuration

1. Create a new design, call this “Fast Event Related”
2. Using the **Add Event** button, add a row into the Event Definition table. The **name** of this event will be “run” with a component called “trial”.
3. Add another row into the Event Definition table called “trial” with components called “stimulus,iti” and two **types** called “face,scrambled”.



4. Now we'll configure the events. Use the **Add Configuration** button to add a row into the Events Configuration table.
5. **Name** this event "run" and keep the default parameters.
6. Add another row and **name** it "trial". Set the number of trials (**N**) to "40". Also, for the beginning of this example, let's set the **max_consec** column to "1" - this means that there will be no more than 1 consecutive trial type (face and scrambled will always alternate).
7. Add another row and **name** it "stimulus". Set the **model** to "dmBlock" with an amplitude of 1. Set the **regression_mode** to "AM1".
8. Finally, add a row **named** "iti" and change the duration (**dur**) to 10 seconds.

When you are finished, your event plot should look like [Figure 32](#).

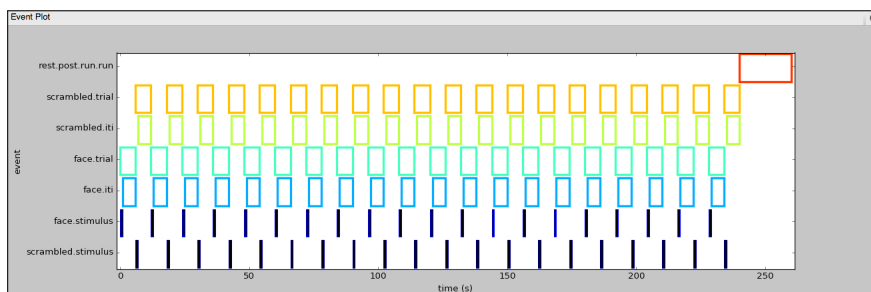


Figure 32: Initial fast event related event plot

6.3 Visualize Events

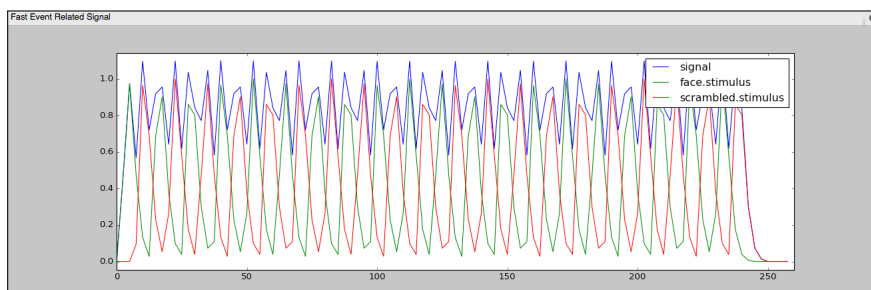


Figure 33: Fast event related signal

Once your experiment is constructed, the **signal** in the **visualize** tab should look like [Figure 33](#). The **green** line corresponds to the signal specific to the face condition whereas the **red** line corresponds to the signal specific to the scrambled condition. The **blue** line corresponds to the sum of the two conditions, which would be the ideal hemodynamic response.



6.4 Unit Exercises

Since we initially constrained the trials to alternated, there is no need to optimize the design because there are only two, equal, solutions (face,scrambled; scrambled,face). However, set **max_consec** to 2 so that the trial order can vary. Now you can properly optimize trial order based on the prescribed stimulus timing and do the following exercises.

1. Create a copy of the fast event design variant, and change the ITI duration (**dur**) to a jittered duration (**uni** option) with a lower bound of 3 seconds and an upper bound of 7 seconds. Make sure each design is optimized using 1000 iterations. How does the VIF and metric quality change between these two designs?
2. Make the designs equal again by setting the jitter back to a constant 5 second ITI. How does changing the **max_consec** option to 2 in one of the experiments change the metric quality and VIF?

