The generality of self-control

Chen, Kent `kentschen` | Lee, Rachel `reychil` | LeRoy, Benjamin `benjaminleroy` | Liang, Jane `janewliang` | Udagawa, Hiroto `hiroto-udagawa`

December 12, 2015

Abstract

Self-control is an interesting field of behavioral research with broad implications for our day-to-day experiences. Being able to appropriately regulate and check our impulses and reactions to various everyday stimuli is necessary for maintaining health and high-functionality in society. Thus, relating a subjects ability to control risky behavior to an area of the brain or focusing on understanding what connects self control to neurological activity has been the focus of many studies. One approach to capturint these neurological facets is to use functional magnetic resonance imaging (fMRI). Our goal is to reproduce and extend the fMRI analysis for a Balloon Analogue Risk Task (BART) study described in *The Development and Generality of Self-Control* [2].

1 Introduction

The Development and Generality of Self-Control [2] and its associated fMRI studies are concerned with the relationship between impaired and normal self control, as well as similarities and differences across the brain relating to self-control. The paper in its entirety explores multiple studies (of multiple study types), but we will just focus on the third study. The full paper compares four different types of self control among healthy adults to see if they are related to each other. Very little relationship was found between these different behavioral tasks, in contrast to the vast majority of existing literature, which argues for a unified notion of self-control. So, we have decided to narrow our focus and data analysis approaches to just the Balloon Analogue Risk Task (BART) study, which purportedly measures control over risky behavior. fMRI scans from the study show blood flow to the brain, which may be relatable to control over risk-taking behavior during participation.

The rest of this report will detail the procedures from the original analysis of the data that we have tried to mimic. We experimented with different procedures to spatially smooth the voxels and the convolve and time-correct the time courses for each subject. After preprocessing the data, we turned to fitting simple and multiple linear regression models to each subject, with the option of including different study conditions. The resulting coefficients from the linear regression models can be used to perform t-tests to examine the significance of activity in different voxels. However, the validity of these tests is highly dependent on the validity of our model assumptions, such as the normality of our errors. Since we are performing a large quantity of tests, multiple comparison corrections are needed to control the false positive rate. We used k-means clustering to further identify regions of the brain across subjects with high activity and compared these results with the results of the original paper.

2 Data

The Balloon Analogue Risk Task (BART) measures risk-taking behavior by presenting participants with a computerized balloon. The participant can earn money incrementally by pumping up the balloon, but after an unknown threshold, the balloon will explode. At any time, the participant elect to cash out his or her earnings, but doing so eliminates the potential to gain additional money through pumps. If the balloon explodes, the participant loses all of the money for the trial. For this study, BART and fMRI data for 24 subjects was collected. The mean age of the subjects was 20.8, and ten of the subjects were female. Four behavioral variables were recorded for each subject: the average number of pumps for each balloon, the average amount of money earned across runs, the number of exploded balloons, and

the number of trials. There were also three model conditions: events for inflating the balloon (excluding the very last inflation of each trial), the last inflation before an explosion, and the event of cashing out (the balloon explosion was not included as an event). Of interest for our work is the blood-oxygen-level dependent (BOLD) imaging data recorded for each subject during the course of task. Each subject's BOLD data was recorded as 64 by 64 image matrices in 34 slices, with a variable number of time points.

Much of our analysis focuses on creating our own procedure for cleaning and preprocessing the raw BOLD data so that the true signal can be readily captured by our analyses. However, a cleaned version of the data was later made available by Ross Poldrack and the OpenfMRI project. The cleaned scans had receieved motion correction, high-pass filtering in time, and registration to the standard MNI anatomical template. We would eventually like to compare the results from using our cleaning procedure versus the provided procedures. One non-trivial issue that must be overcome is to design a reproducible pipeline to perform our analyses on either data format, since the preprocessed scans provided by OpenfMRI are in a space with different-sized dimensions.

It should also be noted that neither set of preprocessed data uses the exact same cleaning procedure as that used in *The Development and Generality of Self-Control* [2], which describes using various software and black-box methods that may not be available for our use. Additionally, while the paper is mostly concerned with comparing the areas of neural activity across different types of self control, we focused only on a single study on a single type of self control, for feasibility reasons.

3 Methods

3.1 Smoothing

Due to the inherently random nature of human subjects and their movements, a certain level of smoothing must be performed on the spatial dataset. That way, the "noisy" data can be cast off from the data that actually represents significant changes in blood flow in the brain. By doing so, researchers and anyone else investigating the data will be able to distinguish between non-brain scans versus actual brain scans. Each voxel of the brain is represented by a measure of blood flow intensity, and so a series of steps must be taken so that the data is correctly convolved to most closely and accurately depict what was happening at a certain point in the brain at a certain time. After researching quite extensively, we decided to use smoothing involving a Gaussian kernel in order to smooth the three dimensional data. Originally, we were going to try and write a smoothing function from scratch, by implementing a rudimentary average-over-neighbors method. However, discussion with mentors lead us to the scipy module `ndimage.filters` that has a function to performs a Gaussian filter on n-dimensional data. This was exactly what we needed so rather than reinventing the wheel, we will be smoothing the data with this module. An in-depth discussion of the Gaussian filter can be found in the appendix.

3.2 Convolution and Time Correction of Hemodyamic Response

3.2.1 Convolution

As our study is structured around event-related neurological stimulus, and not block stimulus as was discussed in class, a similar approach to representing the hemodynamic response could not be done. We note here that in general the hemodynamic response is only 5 % of the noise.

The general assumption is that there is a relationship between the hemodynamic response to neurological stimulation, where a single stimulation generates a delayed hemodyamic response that mirrors a double gamma function. And that there is a natural linear/additive nature when there are multiple stimulations (just adding the response functions started at different times), as defined by:

$$r(t) = \sum_{i=1}^{n} \psi_i \phi_i (t - t_i) \tag{1}$$

where ψ is the amplitude of the response stimulus (always 1 in our case), and ϕ_i is the hemodynamic response started at the ith stimulation (t_i).

The five approaches we attempted can be divided into 3 subcategories: **(1)** a strict replication of equation 1; **(2)** a similar function as **(1)** that utilizes matrix multiplication; and **(3)** a function that splits the intervals between each scan (2 seconds) into a certain number of even slices, then puts the

stimulus into the closed split, using `np.convolve` on this stimulus and a detailed hrf function, and then reducing the output back into the 2-second time intervals. Detailed exploration of this matter can be found in the appendix.

We compared these methods based on accuracy and speed. [Figure 1] displaces accuracy comparison, and the table in [Table 1] show the accuracy based off of `ipython`'s `timeit` function.

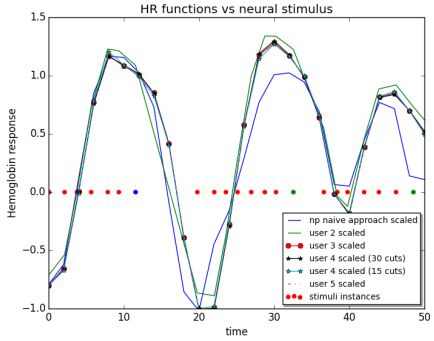

Figure 1: Different convolution functions vs. the Neural stimulus

name in graph	Speed per loop
np naive approach	14.4 μs
user 2	972 ms
user 3	1.15 s
user 4 (15 cuts)	98.3 ms
user 4 (30 cuts)	185 ms
user 5	110 ms

Table 1: Speed to create HRF predictions for Subject 001, all conditions

The first method in the table "np naive approach" which blindly plugs in our data into the `np.convolve` function. It is provided to showcase potential speed. The failure of the "np naive approach" was the motivating factor behind the rest of the hemodynamic response convolution analysis . The "user 2" and "user 3" runs fall under subcategory **(1)**. The "user 2" was the first approach to matches the theory, but matches the stimulation times and not the scan times. The "user 3" is the most theoretically sound model (and is our standard for accuracy). The "user 5" falls under subcategory **(2)**, "User 5" is our matrix version of the theory, and has the same accuracy as "user 3". The "user 4" model fall under subcategory **(3)**, the methods that use the grid cut usage of `np.convolve` with notations for the number of slices between each scan. We concluded that "user 4 (15 cuts)" was the best approach gives us speed and very close accuracy to the golden standard - "user 3".

3.2.2 Time Correction

The fMRI machine scans each voxel at a slightly different time. In our case, the lowest horizontal slice was scanned first, with the later scans moving progressively toward the top of the brain. The signs of this linear change in time of scan was observed when running simple regression on the data and the hemodyamic response beta values from all conditions grouped together. We corrected for this by shifting the times of stimulus "backwards" for voxels scanned later to directly correct for the delay of the scan (assuming that each layer of the scan took 2/34 of a second).

3.2.3 Multiple Conditions

Originally we used multiple regression to take into account the 3 different types of stimulus (pump, explode, cash- out) to see if the separation of these stimuli can better describe the response. We did this by creating separate predicted hemodyamic reposes for each to allow for different amplitudes for each type of condition. As will be noted in the *Model selection* portion below not observe a large difference in the $\hat{\beta}$ values we got, so we did not continue with this exploration. In figure 2, we can see the different conditions broken up.

A more detailed discussion about our approach and theory behind convolution of the hemodynamic response with the neurological response can be found in the Appendix on Convolution C.

3.3 Linear Regression

A simple and straightforward way to model the voxel time courses is to preform linear regression. Initially, we just used the convoluted predicted hemodyamic response (HR) using all the conditions together and each of conditions individually. After realizing that the HRs themselves didn't explain

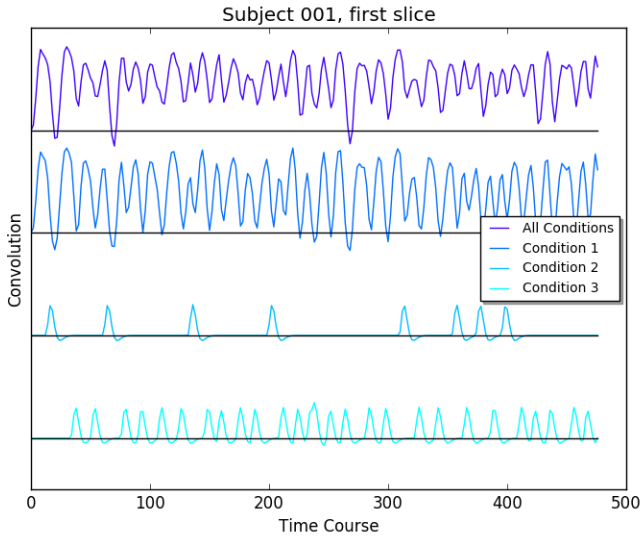

Figure 2: Plotting all predicted HR for conditions.

enough of the BOLD ratio, attended to add features in order to reduce/ explain the noise we observed. Additional features that we examined included a linear drift, a small set the fourier series, and a few principle components.

Linear regression suggests a linear relationship between an observation's response, y_i with it's features x_i (a vector who's first element is 1 and the rest are the values the observation i's features are) in the following manner:

$$y_i = x_i^T \beta + \epsilon_i \tag{2}$$

where $\epsilon_i \sim N(0, \sigma^2)$. Moveover we assume for each observation (i), the ϵ_i are independent and that the true β value is the same. As such we can rewrite the relationship we assume is correct in a matrix form:

$$Y = X\beta + \epsilon \tag{3}$$

This relationship requires a X feature matrix, which presents each distinct feature as a column and sees each observation get a single row to represent it's values. Assuming $\epsilon \sim N(0, \sigma^2 I)$ isn't automatic, so we will need to check the validity of this assumptions, as discussed in the *"Normality Assumptions"* section. As might be expected, we do not know the true value of β and as such need to estimate, using matrix algebra we get a closed from solution:

$$\hat{\beta} = (X^T X)^{-1} X^T \tag{4}$$

This equation does require $(X^T X)$ to be invertible, but even when it is not, we can use the psuedo inverse, represented $(X^T X)^-$ to get a non-unique value for $\hat{\beta}$.

To consider the strength of the effects of these predictors, we looked at t-tests of the corresponding estimated coefficients for each voxel, as discussed under *"Hypothesis Testing"*. The validity of these t-tests and their corresponding "p-values" is largely dependent on how good our assumptions of linearity and normally-distributed errors are.

3.3.1 More about potential features:

Other than the basic HR feature/features, and a column of 1s in our feature matrix (to account for a "intercept" term/ non-zero average value), we experimented with include a few features for the first principle components of a of time $\times$ time matrix and the first few functions of the Fourier series. As noted above these additional features helped account for the noise in the observed BOLD ratio fluxuation.

Principle Components

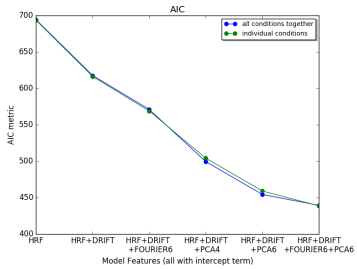

Figure 3: AIC

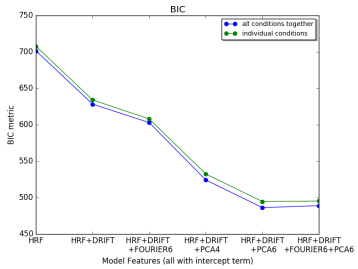

Figure 4: BIC

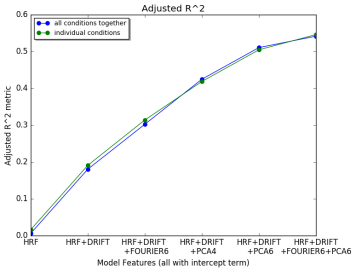

Figure 5: Adjusted R^2

This matrix use to obtain the principle components was the correlation between each individual voxel time course with other voxel's time course ($A^T A$ where A is a voxel *time* time matrix). It should be noted that this matrix $A^T A$ only used the voxels inside the brain. ***Add Jane's Stuff about why we'd need to use the first 6 pcs***

Fourier Series

We included 4 features related to the first few functions of the Fourier series. The full fourier series is represent as the following:

$$f(x) = \frac{1}{2} \cdot a_0 + \sum_{n=1}^{\infty} a_n \cdot cos(nx) + \sum_{n=1}^{\infty} b_n \cdot sin(nx) \tag{5}$$

but since strength of Fourier series strength comes from orthogonality of the range, and we are looking for a function of the range $(0, \text{num of TR})$, we change it to:

$$f(x) = \frac{1}{2} \cdot a_0 + \sum_{n=1}^{\infty} a_n \cdot cos(\frac{n}{\text{num of TR}} x) + \sum_{n=1}^{\infty} b_n \cdot sin(\frac{n}{\text{num of TR}} x) \tag{6}$$

We used $+\sum_{n=1}^{2} a_n \cdot cos(\frac{n}{\text{num of TR}} x) + \sum_{n=1}^{2} b_n \cdot sin(\frac{n}{\text{num of TR}} x)$ to be 4 features to try to get a low order sinusoidal fluxucations.

3.4 Model Selection

In order to select the best set of features for our X matrix, and also compare the use of a single condition feature vs each of the 3 different types of conditions as 3 seperate features we decided to utilize model comparison, specificially AIC,BIC, and Adjusted R^2 metrics. Using small expressive subset of the subjects (002,003, and 014) we abused the metric's by averaging across all voxels and people. We visualized values in the Figures 3,4, and 5.

From these plots, you can observe that we don't gain my from adding the conditions seperated, and from the BIC, we decided that the fourier features didn't add enough benefit to justify the increase of features, so we'll be using the model that includes a single HRF for all the conditions, a linear drift feature, and the first 6 principle components. It is possible that this is dangerous, and even with these features we could have over fit beyond just the noise and into the territory of the hemodyamic response.

3.5 Normality Assumptions

The validity of our hypothesis tests of the estimated $\hat{\beta}$ values from linear regression is largely dependent on whether we can assume that the errors in our model(s) are independent and identically distributed from some normal distribution mean zero and constant variance. We focus here on checking the normality assumption. It is generally wise to use visualizations, such as residual vs. fitted values plots and quantile-quantile plots to inspect residuals for patterns and abnormalities. However, with the sheer quantity of data we are working with— each of the 24 subjects has $64 \times 64 \times 34$ voxels that can each in turn be fitted to a model— visual inspection is not practical.

For this reason, we turned to using the Shapiro-Wilk test for normality, which tests the null hypothesis that the data in question is normally distributed. A Shapiro-Wilk test was performed for each set of residuals corresponding to a single voxel's time course. That is, each test used around 200 observations, or the number of time points for that particular subject. 200 observations is not an especially large sample size, and for this reason, we express some concern because normality tests have low power for small sample sizes. Shapiro-Wilk may incorrectly fail to reject the null hypothesis due to this bias [3].

3.6 Hypothesis Testing

Our simple linear regression model was created to better understand the relationship between the voxels in a given subject's brain and the convolved time course. In order to measure the strength of the association between these two measurements, we ran a hypothesis test on the coefficients of the simple linear regression model for each subject.

There is an individual linear model associated with each voxel in a subjects image (and a total of $64 \times 64 \times 34$ voxels per subject). Thus we ran a t-test on each voxel's β coefficient that is associated with the HRF response. The null hypothesis for each test was that $\beta = 0$, with the alternative hypothesis that $\beta \neq 0$. Once we had obtained each t-statistic, we compared this value across voxels in two ways. First, we simply compared this t-values with voxels within a subject. In this case, we took into account the sign of the t-statistic in our analysis. Second, we converted this t-statistic to a "p-value", in which case the sign of the t-value will become irrelevant and we compared across voxels without taking into account this sign. Later, we also run a multiple comparison test using a BenjaminiHochberg in order to find the voxels that are most significant.

Having implemented a method to compare the voxels within a single subject, we next examined our results for the same voxels across subjects. Our initial approach was to aggregate the t-statistic data between all subjects for each voxel. This allowed us to decrease the variability of the fit on each voxel and detect a more clear signal.

In order to do this, we ran the hypothesis test as stated above on all 24 subjects of the study. Then for each voxel, we took the average of the t- statistics across the subjects. An issue with our data was the presence of empty space detected by the scanner that is not directly part of the brain. To account for this, we took the masked data of the brain and "cut out" the parts of the images that were not relevant to our analysis. Ultimately, we were left with a single 3-d image with each voxel representing the average t-statistics across all subjects in the study. This image will later be used in our clustering step in order to pinpoint the regions of the brain that have the strongest relationship with the convolved time course.

3.7 Clustering

3.8 Benjamini-Hochberg Correction

When conducting multiple comparisons, it is important to have an idea of the quantity of Type I errors that may be prevalent in the analysis. In our analysis of voxel data, we decided that limiting/controlling the number of Type I errors is important to the process. The processes of limiting the number of Type I errors are called FDR-controlling procedures. In the grand scheme of things, FDR-controlling procedures give greater statistical power with the cost of more Type I errors that can fall through.

Once we implemented the hypothesis function that would return t-test values and "p-values", we implemented the Benjamini-Hochberg procedure to control the proportion of rejected null hypotheses in the data. The Benjamini- Hochberg procedure works by multiplying each of the "p-values" to a ratio of the number of tests and the chosen false discovery rate – from these adjusted "p-values", only the values that are less than the chosen false discovery rate will be chosen to be returned. This way, we are

able to adjust the proportion of null hypotheses that will be rejected and the proportion that will return the desired proportion of significant tests. This will reduce the number of false positives returned in the data and extend greater statistical power in later analysis performed on the voxel dataset.

3.9 Hierarchical Clustering

Now that we have the across-subject average t-statistics for every voxel in the brain, we are left with a 3-d array of t-statistics that contain both negative and positive values. Instead of manually observing patterns in these images, we instead implemented a clustering algorithm to split the entire 3-d images into clusters based on the voxels' relative location to each other as well as the values of their t-statistics.

In order to find a proper clustering algorithm, we decided to treat this problem like a grayscale image segmentation problem and implemented a agglomerative hierarchical cluster using Ward's method. Agglomerative means that the clusters are built bottom up which each observation starting as its own cluster and pairs being moved up the hierarchy. Ward's method creates clusters based on a minimum variance criterion that miniizes the total within-cluster variances. An example of this implimentation for a 2d image is seen here: `http://scikit-learn.org/stable/auto_examples/cluster/plot_lena_ward_segmentation.html`.

In our implimentation, we defined a structure to our data using a connectivity graph in order to ensure that each cluster is spatially constrained. Also, since our scenario uses a 3d image, the connectivity graph will also have to take into account this extra dimension.

Ultimately, the goal of clustering is to have a better understanding of which parts of the brain are related to the signal based on the t-statistics from performing hypothesis tests on the coefficients from linear regression. Once we obtain our clusters, we will both measure the within-cluster mean of t-statistics as well measure the centroids of the clusters. By doing this, we hope to see the parts of the brain that have the strongest relationship with the signal and compare them to the results of the origin research paper.

4 Results

To develop linear models, we looked at the HR from the neural response as a single feature. Originally we used multiple regression to take into account the 3 different types of stimulus (pump, explode, cash-out) to see if the separation of these stimuli can better describe the response, but it wasn't that good. In figure 2, we can see the different conditions broken up. Using smoothed data, fourier and drift features in linear regression we obtained fitted values for fMRI BOLD contrast of a random voxel and also calculated the residuals for a random voxel for subject 001, voxel $(30, 40, 15)$ [Figure 6, 7].

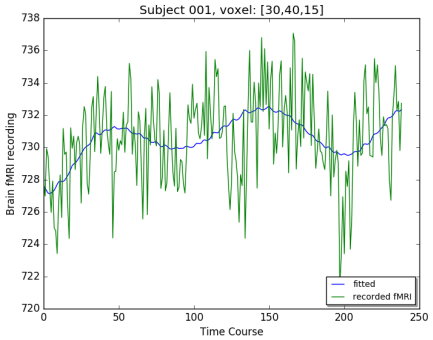

Figure 6: Fitted/Predicted vs Actual fMRI BOLD contrast

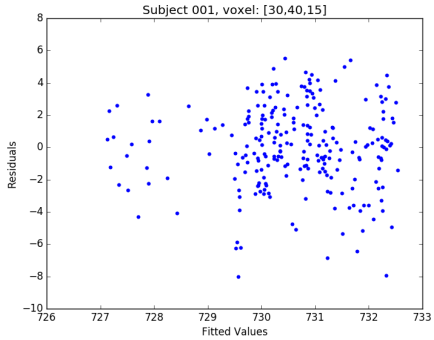

Figure 7: Fitted fMRI BOLD contrast vs Residual from Linear Regression

*****As we also obtained $\hat{\beta}$ values (coefficients) from the linear regression models, we looked at the 3-dimensional reports of the $\hat{\beta}$ values, a less rigorous analysis than hypothesis testing with t-statistics [Figure reference]. ∼Remove or update****

The numerous other multiple regression models discussed in *Linear Regression* should be analyzed similarly in the future.

The results of our simple linear regression t-statistic comparisons across subjects are shown in [Figure 8]. We can see each slice of the brain from top to bottom in each section of the image. The blue areas shows parts of the brain that had a negative t-statistic while the red parts of the image shows parts of the brain that had a positive t- statistic.

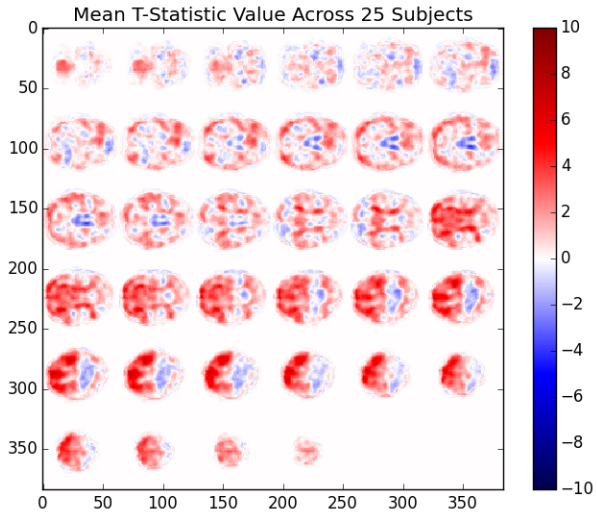

Figure 8: Across-subject mean of t-Statistic per voxel.

The parts of the image that were cut out by the mask are white so we can more clearly see the contrast in our results. Based on a cursory look at this image, we can see a pattern of dark red (high positive t-statistics) in the lower left parts of the brain, and area of dark blue (high negative t-statistics) in the center and lower middle parts parts of the brain.

5 Discussion

5.1 Discussion of Results

While very much still a work in progress, our analysis thus far includes methods for both data processing and modeling voxel time courses. Prior to doing any serious analysis, we had to smooth the data spatially for each subject. We also generated a reasonable convolved time course with time shift corrections, based on event-related neurological stimuli with non-constant intervals.

A basic but nevertheless important model to consider is linear regression. We implemented both simple and multiple regression models at the individual subject level. In addition to the convolved time course, our multiple linear regression models attempt to account for more of the noise in our data by including terms for event conditions, linear drift, and Fourier transforms. We then checked the assumptions for the fitted models and performed hypothesis testing on the resulting coefficients for each voxel. Though the linear regression models were designed to handle each voxel for each subject's data individually, we aggregated the data across the 24 subjects by taking the means of the t-statistics corresponding to each voxel. However, one major concern for hypothesis testing is the issue of multiple comparisons, which we attempted to address using the Benjamini-Hochberg procedure. Finally, we used k-means clustering to further identify areas of the brain with high neurological activity during the events of the BART study.

5.2 Discussion of Future Work

We have specified several models for linear regression and considered various methods for hypothesis testing across multiple voxels and subjects. Now our main objective is to distill all of our work into interpretable results that can be used to identify brain regions with high neurological activity during the course of the BART events.

Additional future work will concern reproducing our analysis on the clean version of the data provided by OpenfMRI and comparing those results with the results from using our own preprocessing techniques. One issue that arises with the addition of the new cleaned data is that the dimension sizes of the scans are different, which will create challenges for comparing the two data sets.

If we have the time, permutation tests, which have few assumptions and are easy to interpret (but computationally intensive), would be a useful tool for identifying brain regions with significant activity. Additional tests and checks for model assumptions would also be valuable for assessing the appropriateness of our existing hypothesis testing.

Appendix

A Appendix: Outlier Removal

We considered following the procedure implemented as part of Homework 2 to detect and remove outlier 3-d volumes from the 4-d image scans of each subject. The process involves finding the root mean squares (RMS) of each 3-d volume across time and then getting the difference values. When a given volume is very different from the preceding volume, this may indicate a potential outlier or the sign of an artifact. We used thresholds based on $1.5 \times$ IQR added to the 75th percentile and subtracted from the 25th percentile to create the cutoffs for the RMS difference outliers. We then extended the RMS difference outliers by labeling the volumes either side of the outlier RMS difference as being outliers.

Each subject was considered independently, as considerable variation in measurement is expected between different subjects. However, we found that reductions in mean residual sum of squares from running simple linear regression before and after removing the extended RMS difference outliers were minimal. Visually speaking, we also did not observe the presence of egregiously different points. So, we opted to refrain from removing outliers, at least through the extended RMS difference method.

B Appendix: Smoothing

We used a Gaussian filter to smooth away noise in the brain images. At first, we played around with implementing a function similar to a mean filter, where each voxel would be the average of a certain radius of its neighbors. Ultimately, we decided to use a kernel with a Gaussian (bell-like) shape.

The property that makes the Gaussian filter a stronger and more reliable candidate for smoothing the voxel data is that it outputs a weighted average of the voxel and its neighbors. The more heavily weighted values are at the center of each of the neighborhood of voxels we examine. On the other hand, a regular mean filter would use a uniformly weighted average, which means that there is the risk of oversmoothing, along with additional complications of handling the voxels along the edges of the brain image data. Furthermore, a Gaussian filter is ideal for use in noisy voxel data because of its steady frequency response. By choosing an appropriate filter, we have can gain more control of the range of spatial frequencies left after smoothing the data. Additionally, Gaussian filters are non-negative for all voxel data. Thus, the output of smoothing the voxel data with a Gaussian filter will still be a valid image.

In conclusion, we decided to go with the module for a Gaussian filter for several reasons, the main one being that Gaussian filters can remove noise and yet preserve the high frequency edges in the brain image data. Rather than use a self-implemented mean filter function that would cause issues with high-frequency edge cases as well as conglomerating data into thoughtless averages, a Gaussian filter handles these situations better because the smooth, bell- shaped curve of the convolution does not have a sharp cutoff at edges. The Gaussian filter also distributes weighted averages across the voxels such that the smoothing keeps high-frequency data points into consideration for the end product.

C Appendix: Convolution Analysis

C.1 Introduction

fMRI data presents a distinct challenge for relating neural stimuli to BOLD (blood-oxygen-level dependent) response. fMRI scans record changes in oxygenation levels of hemoglobin in the brain. However, there is a delay between the neural stimulus and the change in blood oxygen levels in a given area. In our case, the neural stimulation comes from a event-related style of experiment. A commonly assumed hemodynamic response to a neurological stimulation is the double gamma function that can be seen in Figure 12. The complete hemodynamic response function needs to be modeled in order to better relate stimulation and the BOLD response from the fMRI scan. It should be noted that the BOLD response is highly noises and we're really trying to capture the blood oxygenation level change to the stimulation.

C.2 Mathematics

C.2.1 Convolution Theory and Mathematics

To relate stimuli to BOLD response, we convolved the time courses of discrete stimulation with the assumed response to a single stimulation.. At a basic level, convolution is a distinct combination of two functions (say f and g). This combination is just the "integral that expresses the amount of overlap of f as it is shifted over another function g" [4]. There are many examples of this, but the following is basic idea that we will expand off of later.

Let us define the function f as a sum of two gamma functions and g as a "continuous" specialized step function (we will examine why these functions are valuable later). Graphically, we can see their plots in Figure 9 and 10, and mathematically we will define them as in the following equations 7 and 8, respectively.

$$f(t) = \frac{.6}{.17} \cdot \left[G_1(6,t) - .35 \cdot G_1(12,t)\right] \tag{7}$$

where $G_1(k,t) = \frac{1}{\Gamma(k)} t^{k-1} e^{-t}$ (the gamma pdf with $\theta = 1$)

$$g(t) = \begin{cases} 0 & \text{if } 5.85 \leq \text{t} \leq 6.15 \\ .6 & \text{otherwise} \end{cases} \tag{8}$$

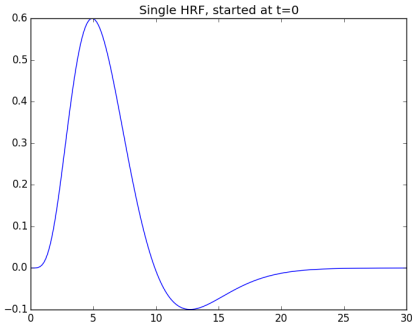

Figure 9: f ("Stabilized Function").

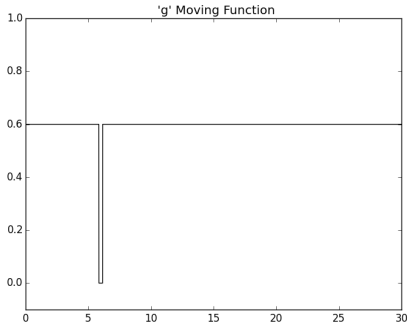

Figure 10: g ("Moving function").

As mentioned in the earlier definition, if we move g across f from left to right, we will see something similar to Figure 11 for discrete time intervals. If we plot these values (the integration of the differences), we will get a plot very similar to that of f when f is starting at a certain point. (The plot actually "cheats" when f is negative, and we would have to alter definitions a little bit). If we had multiple peaks in our g function (i.e. multiple distinct"zero" places), we would expect to get multiple non-zero differences between the functions at each time capture.

C.2.2 Convolution Applied to Stimulus

The "continuous" nature of the step function "g" does not extend well into the discrete time series that we have. However, one approach for fMRI analysis is to approach the convolution as something slightly different: mathematical sums. For example, in the previous section, we can treat f as the same, and g as g' defined in equation 9.

$$g'(t) = \begin{cases} 1 & \text{if t=6} \\ 0 & \text{otherwise} \end{cases} \tag{9}$$

We could then find the value of the convolution of g' and f for discrete integers as in equation 10.

$$r(t) = f(t-6) \tag{10}$$

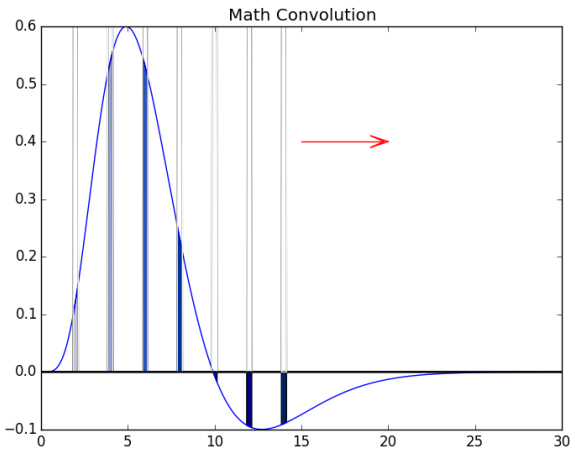

Figure 11: Convolution of f and g.

If we allow for multiple non-zero periods in g', we can get a more general model in equation 11, where each t_i is a value when $g'(t_i) \neq 0$:

$$r(t) = \sum_{i=1}^{n} f(t - t_i) \tag{11}$$

This equation gives a good glimpse into what the hemodynamic response would be after stimulus at time t_i for $i \in 1, ..., n$. Moreover, you could extend the idea to include a "strength" value of the stimulus by changing the $g'(t_i)$ to values other than 1. If that was the case, we would change the response equation to equation 12. Equation 12 also allows us to include all discrete t into the equation where $g'(t_i)$ is now expected to be zero (so n becomes much larger).

$$r(t) = \sum_{i=1}^{n} g'(t_i) f(t - t_i) \tag{12}$$

With this new equation, we can consider function f and g' displayed graphically [Figure 12, 13, respectively] and their "convolved" output [Figure 14].

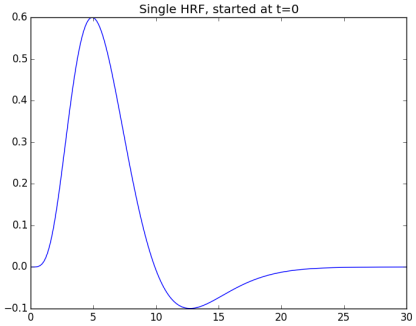

Figure 12: f ("Stabilized Function").

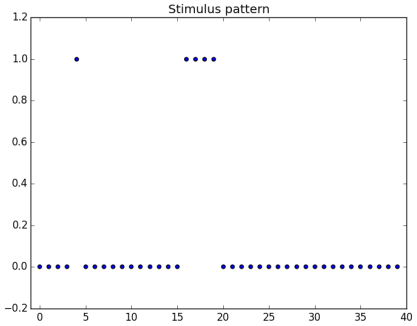

Figure 13: g' ("Moving function").

C.3 Approach to our Specific Problem

C.3.1 Returning to Our fMRI Data

We can now apply this discrete approach to convolution between f and g' to our data. The f is actually a common representation of the hemodynamic response, and the g' is a good representation of

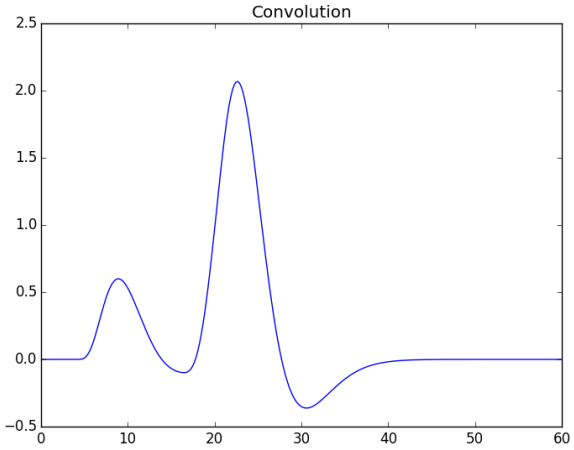

Figure 14: Convolution of f and g'.

the stimuli from an event-related trial [1].

C.3.2 Naive Approach (using `np.convolve`)

A `numpy` function `np.convolve`, which takes advantage of fast Fourier transforms for efficiency (it boils down to fewer computations using roots of unity) is commonly used to do discrete convolution. A naive approach for convolving two functions would use this function directly, but `np.convolve` assumes that the intervals between stimuli mirror the desired intervals between prediction intervals. As such, we can not naively apply the `np.convolve` for our data (though we do for a base model). Even still, exploring this naive approach to convolve the hemodynamic response gives an idea about what not to do, and a high bar for efficiency.

C.3.3 Needed Improvements: Moving beyond naive `np.convolve`

The reason motivating this appendix is the fact that our data fails to meet the assumption that the intervals are in equidistant which was required to naively apply `np.convolve`. Especially in our case, there was not an simple fix, such as performing some basic rounding in order to then correctly utilize `np.convolve`. All the following approaches improve on the basic `np.convolve` approach's accuracy, but ultimately circle back to incorporating `np.convolve` to improve the speed of the convolution.

Our condition file (`cond1`) lists stimulus times for when the individual pumped the balloon but did not pop it. For subject 001, the first 10 data points are as follows [Figure 15]:

0.0671	2.1251	3.7681	5.6601	7.8673	9.3443	19.7831	22.0402	23.5837	25.1434

Figure 15: First 10 values for Sub 001, condition 1.

Clearly, this short time series does not align with idealized scans that start at $t = 0$ and occur every 2 seconds apart. As such, we had to go back to the drawing board to try to reproduce our expected hemodynamic response for the entire time course.

C.4 Summary of Approaches

Our first approach attempts to correctly match the theory underlying our data. Our second approach tries to utilize `np.convolve` by expanding the grid of desired results (thanks to advice from Matthew Brett, Jean-Baptiste Poline, and Jarrod Millman).

C.4.1 Initial Correction to Represent Theoretical Idea

To account for our data's lack of any easily identifiable grid structure between when a stimulus was recorded and when our scans occurred (on the order of every 2 seconds), we went back to the theory of convolution and implemented code to recreate equation 12 directly. To do so, we also had to create a function that works with all discrete points of f, the stimulus response as potential starts of the hemodyamic response, multiplied by the actual value of f, as seen in equation 13:

$$r(t) = \sum_{i=1}^{n} g_i' f(t - t_i) \tag{13}$$

where g_i' is the value of g' at t_i (allowing for zeros and varying non-zero values of g').

C.4.2 Matrix Multiplication

Equation 13, reproduced below

$$r(t) = \sum_{i=1}^{n} g_i' f(t - t_i)$$

can be rewritten as a matrix multiplication problem (thanks to Jane Liang), and can be seen below:

$$r(t) = g^*(t)^T f^*(t) \tag{14}$$

where g^* is a vectorized function of g' of t as a scalar output and f^* is the vector of f values (irrespective of location, as the t^* takes that into account). This is a useful representation, since matrix multiplication is faster for Python's numpy arrays.

C.4.3 Using FFT with `np.convolve`

The "theoretical" solution lacked computation efficiency (despite considerable speed improvements from matrix multiplication), so we also approached the problem by creating a denser grid between each scan (2 seconds apart). Then we rounded the actual times of the stimulus to meet this more finely scaled grid. This allowed us to utilize `np.convolve` with its faster algorithms (thanks to FFT), before reducing back down to our 2 second grid.

C.5 Example

Now that we have discussed that theoretics of and possible implementations of the convolving event-related stimulation, let's look at a basic example from our data. (specifically using subject 001 condition files), and in doing so examine the trade-offs between theoretical accuracy and computational efficiency.

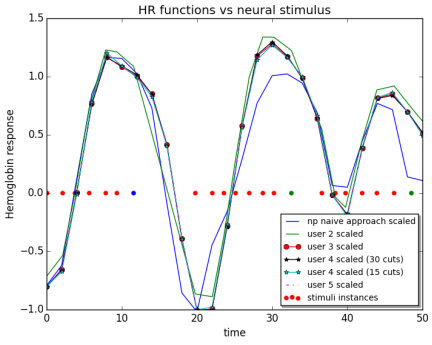

Figure 16: Different convolution functions vs. the Neural stimulus

name in graph	Speed per loop
np naive approach	14.4 μs
user 2	972 ms
user 3	1.15 s
user 4 (15 cuts)	98.3 ms
user 4 (30 cuts)	185 ms
user 5	110 ms

Figure 17: Speed to create HRF predictions for Subject 001, all conditions

The first method in the table "np naive approach" method which blindly plugs in our data into the `np.convolve` function, provided to showcase potential speed. The "user 2" method was the first

approach to matches the theory, but matches the stimulation times and not the scan times. The "user 3" method is the most theoretically sound model (and is our standard for accuracy). "User 5" model is our matrix version of the theory, and has the same accuracy as "user 3", but is observably faster. The "user 4" models fall under use the grid cut usage of `np.convolve`with notations for the number of slices between each scan. We concluded that "user 4 (15 cuts)" was the best approach gives us speed and very close accuracy to the golden standard - "user 3".

D Appendix: Clustering

E Appendix: Time Series Analysis

Cohen's paper [2] discusses analyzing the data with time series using FILM (FMRIBs Improved Linear Model). While we are not familiar with the FILM method, we did try modeling individual voxels in the framework of an autoregressive integrated moving average (ARIMA) process. We focused only on a single voxel from the first subject, but the method could easily be extended to additional or aggregate voxels. Let $\{Y_t\}$ be a single volume's value at time t and assume that the dth difference $W_t = \nabla^d Y_t$ is weakly stationary, defined to be when W_t has a constant mean function and autocovariance dependent only on lag k and not time t. Then we can try to model W_t as a linear combination of p autoregressive terms (or the number of most recent values to include) and q moving average terms (the number of lags to include for the white noise error terms):

$$W_t = \phi_1 W_{t-1} + \phi_2 W_{t-2} + ... + \phi_p W_{t-p} + e_t - \theta_1 e_{t-1} - \theta_2 e_{t-2} - ... - \theta_q e_{t-q}.$$

White noise is defined as a sequence of independent, identically distributed random variables. In order to fit an ARIMA process, the three orders p, d, and q must be first be specified, and the the associated coefficients estimated. We used a combination of visual inspection and quantitative methods to specify the ARIMA orders, and then used the maximum likelihood method to estimate parameters.

Having specified the order for d, we turned to the problem of specifying p and q. We used a combination of visually inspecting the autocorrelation and partial autocorrelation plots of the first difference, and looking at the Akaike information criteria (AIC) and Bayesian information criteria (BIC) computed from a grid of possible models. The latter method suggested specifying $p = 1$ and $q = 1$ (based on either the AIC or the BIC), which was also supported by the visual inspections.

We estimated the parameters for an ARIMA(1,1,1) model using the exact maximum likelihood estimator via Kalman filter. The residuals appear to be normally distributed, and its autocorrelation and partial autocorrelation plots also do not raise any red flags. Furthermore, when visually comparing the fitted time series to the true observed data, the ARIMA process seems to approximate the observed data much better than any of the linear regression models. However, since specifying the correct ARIMA process orders and estimating the associated parameters must be done separately by hand for each individual voxel of interest, we decided to eliminate this direction of analysis from our main pipeline.

Had we decided to continue pursuing time series analysis, we may have tried to forecast future observations based on previous ones. As an example, we modeled an ARIMA(1,1,1) process based on the first half of the observations for a single voxel. This process was then used to forecast the second half of the observations. A comparison between the true observations and the forecasted predictions is shown in [Figure 18]. While the forecasted observations look reasonable for approximating the true values, more quantitative and robust metrics for assessing performance need to be implemented.

One such procedure for assessing performance would be to design a permutation test. A null voxel time course could be simulated from by performing a Fourier transform on the observed time course, permuting the phases, and then transforming back to the original space. That way, the simulated time course has the same autocovariance as the observed time course, but random signal (as under the null case). We can then fit the same ARIMA model to the permuted process and examine how much, if at all, the ARIMA process fitted to the observed data makes improvements over the null case. Generating confidence intervals for the parameter estimates and forecasting future observations may also be of interest. Other considerations for modeling voxels as time series include exploring efficient and reasonable techniques for comparing multiple voxels both within and across subjects.

Other approaches for modeling time series could also be considered, such as spectral density or modeling the conditional variance instead of the conditional mean with an ARCH (autoregressive conditional heteroskedasticity) model.

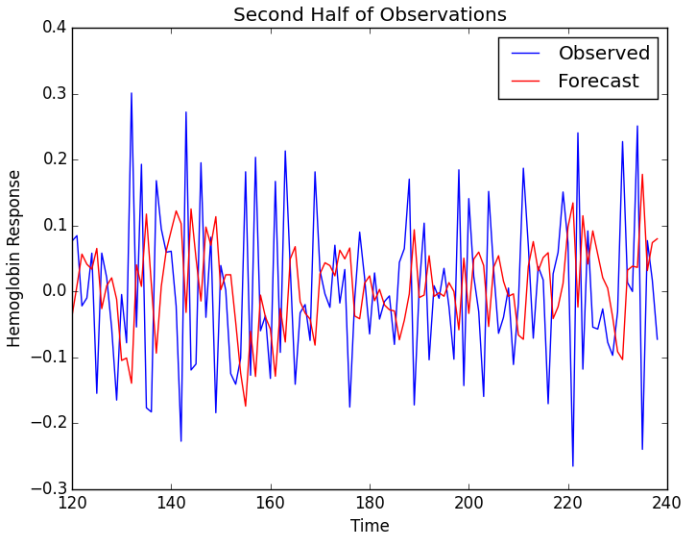

Figure 18: Forecasting the second half of observations based on the first half.

References

[1] M. Brett and J.-B. Poline, *Course on practical neuroimaging in python — practical neuroimaging analysis*. "`http://practical-neuroimaging.github.io/on_convolution.html`".

[2] J. R. Cohen, *The development and generality of self-control*, ProQuest, (2009), p. 164. "`http://gradworks.umi.com/34/01/3401764.html`".

[3] A. Ghasemi and S. Zahediasl, *Normality tests for statistical analysis: a guide for non-statisticians*, International journal of endocrinology and metabolism, 10 (2012), p. 486. "`http://www.ncbi.nlm.nih.gov/pmc/articles/PMC3693611/`".

[4] E. W. Weissten, *Convolution, mathworld - a wolfram web resource*. "`http://mathworld.wolfram.com/Convolution.html`".