HST.583

Lab 5: fMRI data analysis and SPM

Russ Poldrack, MGH-NMR Center

The purpose of this lab is to introduce some methods for analysis of fMRI data using the general linear model as implemented in the Statistical Parametric Mapping (SPM) software package (http://www.fil.ion.ucl.ac.uk/spm).

The goals of the lab are as follows:

- Understand the general linear model and how it is applied to an event-related fMRI timeseries.
- Evaluate the effects of constraining the model using a canonical hemodynamic response compared to an unconstrained (fixed impulse response) model.
- Understand the effects of colored temporal noise on statistical analysis, and evaluate the two basic approaches for dealing with this problem (pre-coloring and pre-whitening).
- Understand the effects of spatial autocorrelation on statistical analysis and how this is dealt with using the gaussian random field theory approach in SPM.
- Understand the importance of treating subjects as a random effect for group analyses, and explore the summary-statistic approach to group statistical analysis in SPM.

Getting started with Lab 5:

1) login your athena account
2) attach hst.583
3) source /mit/hst.583/bin/lab5-init.csh
   (this changes your directory to that of lab5)
4) matlab & (start matlab)
5) add_lab5
   (this adds the paths for matlab functions used in lab5 and also for the spm99 path)

Part 1. Estimating the signal

The general linear model is defined in its matrix formulation as follows:

$$\mathbf{Y} = \mathbf{XB} + \mathbf{e}$$

where:

Y: the observed timeseries
X: the design matrix, composed of a set of basis functions
B: parameter estimates (regression weights) for each basis function in the design matrix
e: residual error - assumed $N(0,\sigma^2\mathbf{I})$

We use ordinary least-squares estimation (which is also the ML estimate if the noise is $N(0,\sigma^2\mathbf{I})$) to estimate the weights on each of these basis functions:

$$\mathbf{B} = \mathbf{X}^{-1}\mathbf{Y}$$

SPM allows a choice of several basis functions for modeling of event-related responses. The most commonly used is a combination of two gamma functions, one that models the initial positive response and a second that models the later undershoot in the HRF. The basis function for an event-related design is thus a convolution of this *canonical HRF* with a stick function corresponding to the onsets of individual trials.

Exercise 1: Estimating a simple model

Create a 100-element stick function with 10 trials, and convolve it with a canonical hemodynamic response. A stick function contains ones at the onset of each trial and zeros otherwise. The canonical HRF can be obtained using the following function:

```
canonical_hrf = spm_hrf(2);  % where 2 represents the TR (in
seconds)
stick_function = zeros(1,100);
trial_times=round(rand(1,10)*99)+1;
stick_function(trial_times)=1;
```

Convolution is performed using the conv() function in MATLAB. Because convolution extends the size of the matrix, trim it back after convolution to a length of 100 elements.

```
convolved_stick_function=conv(stick_function,canonical_hrf);
convolved_stick_function= convolved_stick_function(1:100);
```

Plot the resulting basis function using plot(convolved_stick_function).

Now create a simulated fMRI timeseries (Ysim) by running the model forwards.

First, create a design matrix, including the basis function created above along with another column representing the mean signal (made up of a vector of ones). The design matrix should have dimensions 100 X 2.

```
X=zeros(100,2);
X(:,1)=convolved_stick_function';
X(:,2)=1;
```

View the design matrix using the MATLAB function imagesc(X). It may be useful to set the colormap to grayscale using the command colormap gray

Create a set of parameter weights for the canonical basis function as well as the mean:

```
B = [4; 100];
```

Create random noise to represent the residual error. For the moment we will use uncorrelated Gaussian noise with a mean of zero. The MATLAB function randn() will create a vector of values ~ N(0,1).

Thus, to create a synthetic dataset based on the model you use the following:

```
Y=X*B + randn(100,1);
```

Run the model forward to create the simulated dataset, and plot the resulting dataset. Run the model several times multiplying the noise vector by different values, in order to create a dataset that has some noise but still approximates the canonical basis function.

Now run the model backwards to estimate the parameter weights for the model. Note that because the design matrix is not square, the inverse is not defined; thus a psedoinverse technique must be used. You can use the pinv() function in MATLAB. They should approximate the weights that we specified in step b. Increasing the noise will increase the degree to which these estimates differ from the true values.

```
B_est = pinv(X)*y;
```

How well do these estimates match the values that you specified above?

Exercise 2: Compare the canonical model to the fixed impulse response model.

The canonical HRF model used in Exercise 1 used only a single basis function. However, analyses in SPM generally use a set of canonical basis functions to model the event-related response. This basis set is comprised of the canonical (2-gamma) basis function that you worked with in Exercise 1, as well as the derivatives of this basis function with respect to two of its parameters, namely its delay and dispersion. This basis set is able to adequately model many kinds of responses that occur in fMRI studies, but it does give biased estimates for responses that fall outside the space modeled by these basis functions.

It is possible to analyze fMRI data using a basis set that can model any possible response. This is generally referred to as a fixed impulse response (FIR) model, in that it models an impulse response at each point in peri-stimulus time. This model is the same as "selective averaging" in the sense that the parameter estimates reflect the average response at each point in peristimulus time.

In this exercise you will explore the differences between the canonical model and the FIR model. The demonstration function called basis_function_example()will let you examine how the canonical models fits various types of responses.

First, examine the canonical basis set and the effects of varying the temporal and dispersion derivaties by calling the function with no arguments:

```
basis_function_example;
```

This will create a figure showing you what the canonical basis set looks like.

Next, examine the fit to the canonical:

```
hrf=spm_hrf(2);
basis_function_example(hrf);
```

This will create three figures. Figure 1 shows the design matrices for the FIR and canonical models. The program will then run the model 100 times using random normal errors. Figure 2 shows an example of one run for both models. Figure 3 shows the average model fit for each model along with the standard error across simulation runs. What differences do you notice between the two models?

Now examine some different types of responses. First, examine responses that are delayed in time. You can create a delayed response as follows:

```
hrf=spm_hrf(2);   % canonical with 2 second timesteps
delayed_hrf=[0;hrf]; % delay response by 1 timestep (2 seconds)
delayed_hrf=delayed_hrf(1:size(hrf,1));  %trim the vector
```

The response can be delayed further by adding additional zeros at the beginning of delayed_hrf. How well does the canonical model capture this delayed response?

Now examine a sustained response, which can be created as follows:

```
hrf=spm_hrf(2);
sustained_hrf=[hrf(1:3);hrf(4)*ones(2,1);hrf(5:17)];
sustained_hrf=sustained_hrf(1:size(hrf,1));
```

The length of the sustained response can be changed by changing the first argument to the ones() function; when set to 1 it is the standard hrf, and each unit increment increases the length of the sustained portion of the response by two seconds.

How well does the canonical model deal with sustained responses?

Try other types of responses (such as double-peaked responses). What kinds of responses cause problems for the canonical model?

Part 2. Temporal autocorrelation in fMRI data

The temporal noise in an fMRI timeseries is colored due to both physical and physiological processes. In this exercise we will examine the effects of these noise characteristics on the resulting statistical analyses, and examine the techniques used to address these effects.

The general linear model assumes that the temporal noise is white, and any temporal autocorrelation will inflate the rate of false positive test results. The GLM can be extended to account for temporally autocorrelated noise ~N(0,Λ). This model, known as the generalized linear model, is stated as:

$Y=X*K*B + K*e$

where $K*K^T=\Lambda$. In the case where $\Lambda \neq \sigma^2\mathbf{I}$, the OLS estimates of the standard errors for the parameters will be underestimates, and Type I error will thus be inflated, as you will see below. In the special case of temporally white noise ($\Lambda = \sigma^2\mathbf{I}$), $\mathbf{K}$ becomes an identity matrix and the

model reduces to the general linear model.

There are two approaches to dealing with correlated noise. One is to remove the autocorrelation by pre-multiplying the data and design matrix with an estimate of the autocorrelation:

$$\mathbf{K^{-1}Y} = \mathbf{K^{-1}*X*b} + \mathbf{e}$$

This approach is known as "pre-whitening" because it attempts to make the residuals white. If the autocorrelation can be properly estimated, then this approach will provide a best (i.e., most efficient) estimate of the model parameters. However, it has proven difficult in practice to estimate the autocorrelation, and the use of this approach with incorrectly specified autocorrelation will lead to biased estimates.

A second way to approach the problem has been called "pre-coloring", which is the approach used in SPM. In this method, one specifies a particular matrix K describing the autocorrelation, and the pre-multipies both the data and design matrix by this matrix beore fitting this model:

$$\mathbf{K*Y} = \mathbf{K*X*b} + \mathbf{K*e}$$

giving residuals distributed as $N(0,\sigma^2KK^T)$. The model parameters can then be estimated using generalized least squares. The pre-coloring approach is less efficient than the pre-whitening approach unless the specified convolution matrix exactly matches the actual covariance structure, but it is less likely to be biased, because the imposed "coloring" should largely dictate the nature of the autocorrelation structure thus leading to unbiased estimation by generalized least squares.

In addition to pre-coloring, SPM also calculates the effective degrees of freedom for the dataset. This is necessary because the t-distribution with N degrees of freedom is in appropriate when the errors are serially correlated (as they are in the pre-coloring approach).

Exercise 3: Examine the statistical effects of colored noise

In this exercise we will examine the effects of temporal autocorrelation on statistical analysis, and the effects of temporal smoothing and whitening.

The function simulate_ar1() will perform a simulation to demonstrate these effects. This program takes the following arguments:

```
simulate_ar1(n_runs,ar1_param)
```

n_runs: number of simulated runs of noise data over which to simulate
ar1_param: the weighting parameter for the ar1 model (varies from 0 for independence to 1 for maximal autocorrelation)

The program performs the following steps, repeating them n_runs times:

create a simulated noise timeseries using uncorrelated noise
Perform GLM using a boxcar basis function convolved with SPM canonical HRF
Perform statistical test in 3 ways: standard OLS assuming independence, pre-coloring w/ effective degrees of freedom, and pre-whitening

Over all runs, the number of statistical results is calculated. Because the statistical tests are thresholded at alpha = .05, we should observe spuriously significant results on roughly 5% of trials.

Try the simulation using various values of the ar1 weighting parameter (100 runs is suggested for each trial). How does smoothing (i.e., ar1 weighting) affect the number of spuriously significant results, and how does it differ for the test corrected using pre-coloring and effective degrees of freedom and pre-whitening?

Part 3. Spatial autocorrelation in fMRI data

Any particular fMRI acquisition contains thousands of voxels. If an independent statistical test is performed at each voxel, a large number of these will be significantly active simply by chance. If each test is truly independent, then the overall error rate can be controlled using the Bonferroni correction:

alpha = (overall alpha)/(number of tests)

However, because of the spatial autocorrelation (smoothing) inherent in fMRI data, all voxels are not independent, and the Bonferroni approach is thus too conservative (i.e., results in a false positive rate less than the nominal rate).

SPM uses an approach known as random field theory to deal with the effects of spatial autocorrelation. This approach assumes that the model residuals approximate a Gaussian random field; in order to fulfill the assumption of a good lattice representation of a Gaussian field, the smoothness should be at least twice the voxel size. Gaussian random field theory then

provides corrected p-values for significantly activated clusters of a given size at a given t-threshold value.

Exercise 4. Examine the effects of spatial autocorrelation

In this exercise we will examine the effects of spatial autocorrelation on statistical results, and the effects of random field theory corrections.

The function simulate_spatial_noise() will perform a simulation to demonstrate these effects. The usage of this program is:

```
simulate_spatial_noise(n_runs, smoothing)
```

n_runs: number of simulated runs of noise data over which to simulate
smoothing: the amount of spatial smoothing (full width at half-maximum of a Gaussian kernel, in voxels, where the program assumes that the voxel size is 3 mm cubic)

The program creates images with random normal variates and counts the number of voxels exceeding the .05 threshold given the specified amount of spatial smoothing, both with and without the random field theory correction. Try this with several smoothing values, from 0 for no smoothing up to some large value such as 6 or 8 voxels.

Questions:
How does spatial smoothing affect the number of spuriously ignificant results?
Does RFT adequately control the number of significant results?

Part 4. Single-subject versus group analyses

The final goal of most fMRI studies is to produce knowledge about functional anatomy that generalizes across individuals. In order to determine whether a particular effect generalizes across individuals, it is necessary to perform group analyses. In addition, we often want to know whether two groups (e.g., patients and controls) differ in activation, for which group analysis is also necessary. In this section we will discuss two approaches to group analysis, and you will get hands-on experience analyzing a group dataset in SPM.

An essential concept to understand for this section is the distinction between fixed and random effects. A random effect is one where the particular values in the study represent samples from a larger population, whereas a fixed effect is one where all of the levels of interest of the effect are included in the study (such as task conditions, or trials on a particular task). The important

implication of this distinction comes about at the level of inference: Inference for a random effect generalizes across the entire population from which the samples were acquired, whereas inference for a fixed effect generalizes only to the particular levels of that effect included in the study.

Historically, fMRI data were analyzed treating subjects as a fixed effect. This was accomplished by including all images for all subjects in the analysis, and calculating the variance and degrees of freedom over all of these data points. This can lead to a very large number of degrees of freedom and thus to highly significant effects, but because variance was calculated over scans, subjects are treated a fixed effect and inference is thus limited to the specific set of subjects included in the study. More recently, it has been suggested that subjects should be treated as a random effect, so that inference can extend to the entire population from which our subjects are sampled (i.e., college students willing to lie in a loud, dark, confined space for $50).

The terminology "random-effects analysis" has come to be applied to this practice in the literature, though it is more properly a mixed-effects model with subjects as a random effect and other factors (such as experimental conditions) as fixed effects. In SPM analysis of the random effect is accomplished using what is referred to as a "summary-statistic" approach. In essence, this involves computing an image reflecting the size of a particular effect for each subject at each voxel, and then submitting those images to a second-level test where the variance is computed across subjects.

A practical reason to desire random-effects group analysis is that group analyses that treat subjects as a fixed effect can show results that are driven by a small subset of the subjects. However, it is generally accepted that random effects analysis requires a greater number of subjects to find a particular effect compared to fixed effects analysis. In general one needs at least 8, and preferably at least 12, subjects for a random effects fMRI analysis to have sufficient power.

Exercise 5: Subjects as a fixed vs. random effect

In this exercise you will explore the differences between treating subjects as a fixed versus a random effect.

The function simulate_group_analysis() will perform a simulation to show you the differences in fixed versus random effects approaches, particularly with respect to their sensitivity to activity in a small subset of subjects. The usage of this function is:

```
simulate_group_study(subs,noise_within,noise_between,nruns)
```

subs: a vector with effect sizes corresponding to different subjects
e.g., for eight subjects, half of whom show no effect and half of whom show a 1% signal change on the task, you would enter [0 0 0 0 1 1 1 1]
noise_within: scaling factor for within-subject variability (default=1)
noise_between: scaling factor for between-subject variability (default=1)
n_runs: number of simulation runs (default=100)

You should try this with various parameter values. In particular, you should explore the following questions:
What happens to eacb model when there is a small group of subjects (or even a single subject) that shows a large effect?
What happens when all subjects show a very small effect?
How does the number of subjects affect each type of model?

Further readings

Web sites:

SPM web site: http://www.fil.ion.ucl.ac.uk/spm/
Matt Brett's SPM tutorial site: http://www.mrc-cbu.cam.ac.uk/Imaging/spm.html
SPM Toolbox (extensions to SPM): http://spm-toolbox.sourceforge.net/
Will Penny's notes on random effects in SPM:
http://www.fil.ion.ucl.ac.uk/~wpenny/publications/notes.ps

Papers:

Analysis of fMRI Time Series Revisited. Friston KJ, Holmes AP, Poline JB, Grasby PJ, Williams SCR, Frackowiak RSJ, Turner R (1995) *Neuroimage* 2:45-53

Analysis of fMRI Time-Series Revisited – Again. Worsley KJ, Friston KJ (1995) *Neuroimage* 2:173-181

To smooth or not to smooth? Bias and efficiency in fMRI time-series analysis. Friston KJ, Josephs O, Zarahn E, Holmes AP, Rouquette S, Poline J. (2000) *Neuroimage* 12:196-208.

Stochastic designs in event-related fMRI. Friston KJ, Zarahn E, Josephs O, Henson RN, Dale AM. (1999) *Neuroimage*. 10:607-19.