AnalyzeFMRI: An R package for the exploration and analysis of MRI and fMRI datasets

by Jonathan Marchini

AnalyzeFMRI is a developing package for the exploration and analysis of large MRI and fMRI datasets. In this article we give a short introduction to MRI and fMRI and describe how we have used R when working with these large datasets. We describe the current version of the package using examples, describe our own approach for fMRI analysis and outline our plans for future functionality in the package.

MRI and fMRI

Magnetic Resonance Imaging (MRI) is a non-invasive medical imaging technique that provides images that represent slices of (brain) tissue. Essentially, measurements occur within each slice on a grid of cube-like *vol*ume *el*ements (*voxels*). These measurements reflect the chemical and magnetic properties of the tissue in each small voxel and result in *structural* MRI images that show detailed contrast between different tissue types (see `http://www.stats.ox.ac.uk/~marchini/pictures/high.res/gif`). Variations of the imaging parameters sensitize the images to different chemical and physical properties of interest.

The relationship between *functional* MRI (fMRI) and *structural* MRI is analogous to the relationship between still photography and movies. In simple terms fMRI is fast, repeated structural MRI and can be carried out in such a way so as to be sensitive to local changes in levels of brain activity. Increases in local brain activity increase the local levels of blood oxygen. This in turn causes the measured signal to increase. Thus the images produced are said to be *Blood Oxygenation Level Dependent* (BOLD). Through the BOLD mechanism we can use fMRI datasets to localize specific areas or networks of the brain that are responsible for cognitive functions of interest. In this fashion fMRI has revolutionized the area of neuroscience over the last 10 years. An excellent introduction to this area is given by Matthews et al. (2001)

In a typical fMRI experiment the subject is repeatedly imaged whilst performing a task(s) or receiving certain stimuli designed to elicit the cognitive function of interest. Traditionally, tasks/stimuli are applied in alternating blocks of 20–30 seconds. More recently, 'event-related' designs in which the stimulus is applied for short bursts in a stochastic manner have become popular.

In those areas of the brain that are activated by the tasks/stimuli we will observe that the voxel time series are correlated with the design of the experiment (see Figure 1). Focus usually centers on delineating those areas (clusters of voxels) of the brain that exhibit a significant response. Most often we will wish to make inferences using a group of subjects.

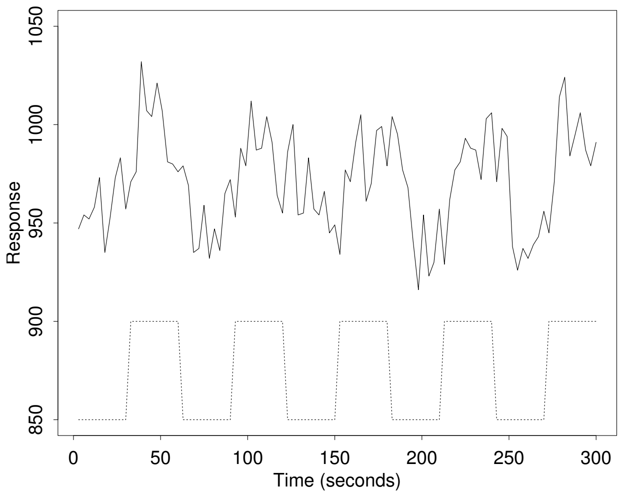

Figure 1: A real voxel time series (solid line) in an area of the brain activated by an alternating 'OFF/ON' visual stimulus (dotted line).

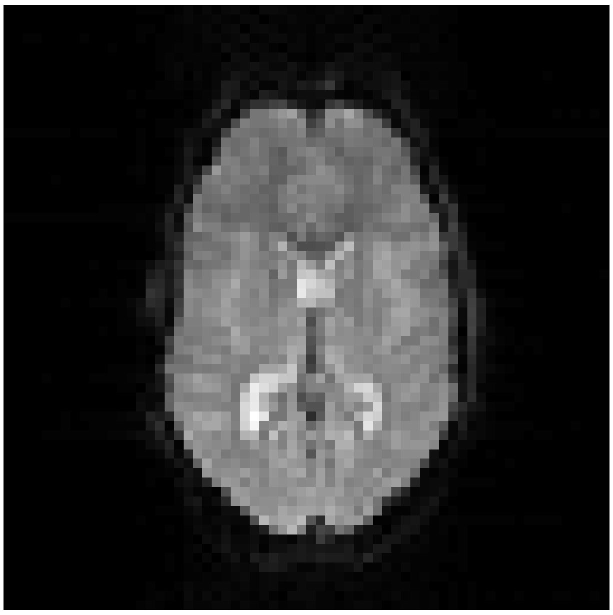

Figure 2: A functional MRI image

The resolution of fMRI datasets is very good. Typically datasets obtained using this technique consist of whole brain scans (≈ 20 images) repeated every 2-3 seconds. Usually each image consists of 64×64 voxels of size $(4\text{mm})^3$ (see Figure 2).

In summary, an fMRI dataset consists of a 3D grid of voxels, each containing a time series of measurements that reflect brain activity. Out of roughly 15,000 voxels that lie inside the brain we wish to identify those that were activated.

Using R for fMRI datasets

In order to analyze fMRI experiments we need to be able to manipulate and visualize the large amounts of data in a relatively easy fashion. This in turn allows us to implement and evaluate existing approaches and develop new methods of analysis. The analysis will often involve several computationally intensive steps and so it is also important to keep an eye on the efficiency of the code.

The high level environment that R provides has allowed us to implement and test our methods quickly and reliably. During this process we have used profiling facilities to identify areas of our code that can be speeded up (Venables, 2001) and where necessary we have written C and Fortran code to achieve this (Venables and Ripley, 2000). We have often found that the memory overheads of carrying out all calculation within R can be high. For this reason we have often used R simply to pass file names and analysis parameters to C code. All file I/O and computations are carried out in C before writing the results to a new data file. In addition we have found it useful to write simple Graphical User Interface's (GUI's) using the **tcltk** package. This allows analyses to be set up and run without recourse to command line functions.

The ANALYZE format and I/O

The ANALYZE image format is a general medical image format developed by the Mayo Clinic[1] that is commonly used within the brain imaging community. The format consists of an image file ('foo.img') together with a header file ('foo.hdr'). The header file details the binary storage type of the image file, dimensions of the dataset and certain imaging parameters used during acquisition. In addition the endianness of the image and header files can be detected by reading the first 4 bytes of the header file, as they contain the constant header file size (348 bytes). The size of a typical fMRI dataset stored in this format is 30Mb.

The package provides read and write capabilities for the ANALYZE format. For example, a simple summary of the dataset is available.

```
> f.analyze.file.summary("./ex.img")
      File name: ./ex.img
  Data dimension: 4-D
      X dimension: 64
      Y dimension: 64
      Z dimension: 21
  Time dimension: 1 time points
Voxel dimensions: 4 mm x 4 mm x 6 mm
       Data type: signed short
                  (16 bits per voxel)
```

[1]http://www.mayo.edu/bir/PDF/ANALYZE75.pdf

Functions of the form `f.read.analyze.*` allow whole or parts of the dataset to be read into R, i.e.,

```
> a <- f.read.analyze.volume("./ex.img")
> dim(a)
[1] 64 64 21 1
```

The function `f.write.analyze` allows an array to be written into a new ANALYZE format file, i.e.,

```
> a <- matrix(rnorm(1000),dim=rep(10,3))
> f.write.analyze(a, file="./ex2", size="float")
```

In addition `f.basic.hdr.list.create` and `f.write.list.to.hdr` allow the user to store custom information in the header files.

Exploratory Data Analysis

fMRI datasets are large and can contain significant noise structure and imaging artifacts. The quality of datasets can vary widely from scanner to scanner and even from day to day on the same scanner. For these reasons it is important to regularly 'eyeball' the datasets to examine their quality. The package provides two methods of examining the structure present in each fMRI dataset.

The first is a simple spectral summary of the dataset produced using `f.spectral.summary`. This function calculates the periodogram of each voxel time series normalized by the median periodogram ordinate. A plot is produced that shows the quantiles of the normalized periodogram ordinates at each frequency. This provides a fast look at a fMRI dataset to identify any artifacts that reside at single frequencies. Figure 3 shows the results of applying this function to a real fMRI dataset. The spike at frequency 18 is consistent with the frequency of the periodic stimulus of the experiment. The spike at frequency 60 highlights the existence of a Nyquist ghost image artifact that can occur in fMRI datasets.

```
> f.spectral.summary(file="ex3.img",
                     mask.file="ex3.m.img",
                     ret.flag=FALSE)
Processing slices... [1] [2] [3] [4] [5] [6] [7]
[8] [9] [10] [11] [12] [13] [14] [15] [16] [17]
[18] [19] [20] [21]
```

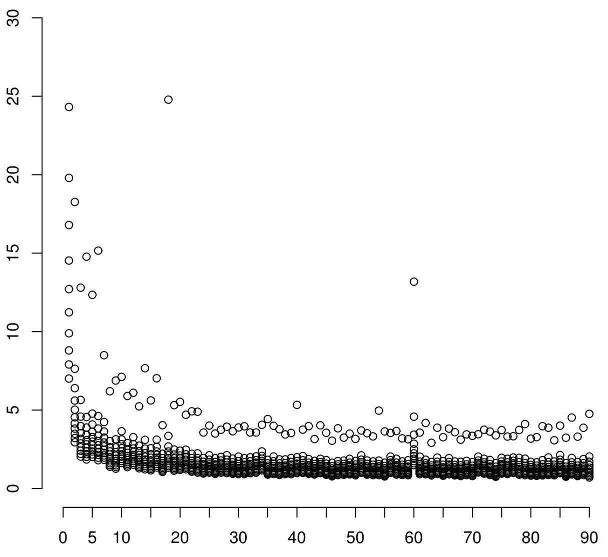

Figure 3: Spectral summary plot of an fMRI dataset.

We provide a GUI that allows visualization of individual voxel time-series, images of specific slices and functional volumes, movies through time of specific slices and spectral summary plots. This GUI is invoke through the function `f.analyzeFMRI.gui`.

Secondly, we have applied Independent Component Analysis (ICA) to fMRI datasets and found that the extracted components are extremely useful in uncovering otherwise hidden structure. In ICA the data matrix X is considered to be a linear combination of non-Gaussian (independent) components, i.e., $X = SA$ where columns of S contain the independent components and A is a linear mixing matrix. In short ICA attempts to 'un-mix' the data by estimating an un-mixing matrix W where $XW = S$.

Under this generative model the measured 'signals' in X will tend to be 'more Gaussian' than the source components (in S) due to the Central Limit Theorem. Thus, in order to extract the independent components/sources we search for an un-mixing matrix W that maximizes the non-Gaussianity of the sources. It is useful to note that in the absence of a generative model for the data the algorithm used is equivalent to Projection Pursuit. We use the Fast ICA algorithm (Hyvarinen, 1999) implemented in package **fastICA** to estimate the sources (see that package for more details).

For fMRI data we concatenate the dataset into a data matrix so that each row of X represents one voxel time series. Using this formulation allows us to estimate spatially independent sources that occur in the dataset. The function `f.ica.fmri` calls C code that reads the data directly from the ANALYZE image file, carries out the ICA decomposition and returns the results into R. We also provide a GUI written using **tcltk** that allows a fast way of visualizing the results of the ICA decomposition. This GUI is invoked through the function `f.ica.fmri.gui` (see Figure 4).

Figure 5 shows one of the estimated components (columns of S) plotted in its spatial configuration together with its associated weighting through time (row of A). This particular component shows the high frequency Nyquist ghosting artifact suggested by the spectral summary plot.

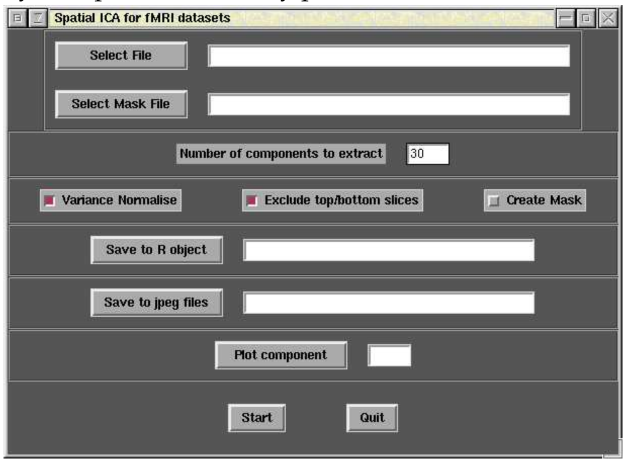

Figure 4: The GUI for spatial ICA for fMRI datasets.

Analysis

The most widely used strategy for the analysis of fMRI experiments is carried out using a two-stage approach. In the first stage a linear model with correlated errors is used for each individual voxel time-series Y. That is,

$$Y = X\beta + \epsilon \quad \epsilon \sim N(0, \sigma^2 V) \tag{1}$$

where the design matrix X contains terms that model the BOLD response to the stimuli and non-linear trends that are often observed in fMRI voxel time-series.

Once this model has been fitted at each voxel, summary statistics (typically t and F statistics) are calculated that reflect the components of the response under study. For example, we might calculate an F statistic that reflects the variance component attributable to the BOLD response. In the context of a group study the statistic at each voxel is calculated from the model fits of all the subjects. These statistic values can then be plotted spatially as a statistic image (see figure 8). The second stage then focuses on the analysis of the statistic image in order to identify those areas of the brain that were activated by the stimuli.

Time-series modelling

In general, the linear model (1) used at each voxel will contain three main components. These being (i) non-linear trends, (ii) the BOLD response to the stimulus, and (iii) auto-correlated noise.

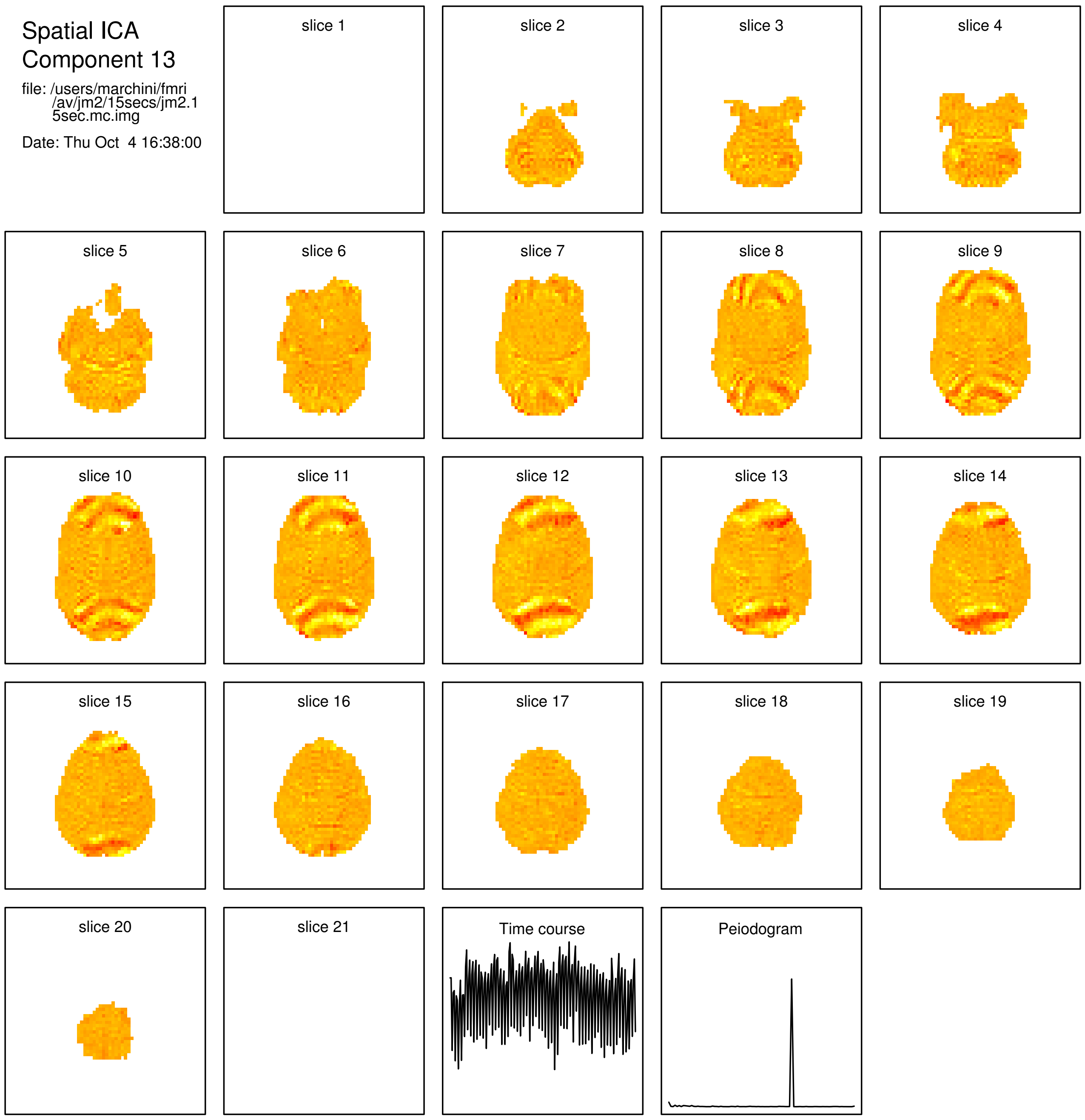

Figure 5: An extracted spatial ICA component showing a high frequency Nyquist ghosting artifact.

Trends are commonly modelled using polynomial, cosine or spline basis terms. Alternatively the trends can be removed from the data before analysis by applying an appropriate filter/smoother. We have found a Gaussian weighted running lines smoother reliably removes the non-linear trends (Friedman, 1984). For efficiency we avoid multiple calls to smoothing functions (such as `supsmu`) by initially calculating a smoothing matrix S and applying it at each voxel.

The nature of the BOLD response implies that in areas of activation we will observe a delayed and blurred version of the stimulus design. This is commonly modelled through the convolution of the stimulus design $x(t)$[2] with a *Haemodyanmic Response Function* (HRF), $h(t)$. That is,

$$BOLD(t) = \sum_s h(t-s)x(s) = h \otimes x(t) \qquad (2)$$

Suggested forms for the HRF include discretized Poisson, Gamma and Gaussian density functions. To allow for spatial variation in the HRF at each voxel a small set of basis HRF's can be used that vary in the amount they blur and delay the stimulus design

[2]normally a vector of 1's and 0's that specify the times when the stimulus is 'ON' and 'OFF' respectively.

(Friston et al., 1995). Each HRF is convolved with the design to make one column of the design matrix X.

The final component in the linear model is the auto-correlated noise term. The auto-correlation structure varies considerably throughout the brain and the chosen model should be chosen to reflect this fact. Suggested models include AR(p) (Bullmore et al., 1996), ARMA models (Locascio et al., 1997) and non-parametric spectral density estimates (Lange and Zeger, 1997). Alternatively, Worsley and Friston (1995) suggest imposing a known correlation structure on the model (using a low-pass filter) before using OLS to fit the model at each voxel. The authors point out that this scheme works for periodic designs[3] but is inefficient for more interesting event-related designs.

In general the chosen noise model should be flexible enough to capture the spatially varying levels of correlation throughout the brain. The model should provide well-calibrated levels of significance for estimated responses and be resistant to commonly occurring image artifacts.

A spectral domain approach

For periodic designs the specification of the HRF and noise model at each voxel time-series is greatly simplified by working in the spectral domain (Marchini and Ripley, 2000). The simplest example of a periodic design involves c repeats of a block which consists of b volumes scanned during a baseline stimulation, followed by b volumes scanned during a stimulus/task of interest. For such designs the majority of the power of the BOLD response will lie at just one frequency $\omega_c = 2\pi c/n$ and will result in a spike at that frequency in the periodogram of the voxel time-series, i.e. the spike at the 18th frequency in Figure 6.

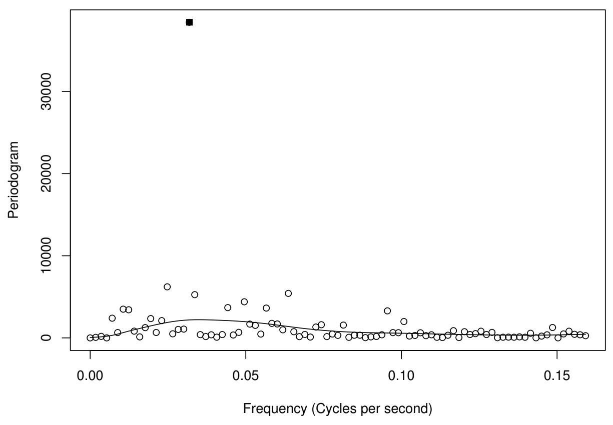

Figure 6: Periodogram and spectral density estimate.

The asymptotic distribution of the periodogram ordinate $I(\omega_c)$ is given by

$$I(\omega_c) \sim \frac{1}{2} f(\omega_c).\chi^2_2, \tag{3}$$

where $f_(\omega)$ represents the non-normalized spectral density of the (second order stationary) noise process. This suggests that a test for an unusually large component at frequency ω_c can be constructed through the use of the statistic R_c where

$$R_c = \frac{I(\omega_c)}{\widehat{f(\omega_c)}}, \tag{4}$$

where $\widehat{f(\omega_c)}$ is an estimate of the non-normalized spectral density. If we can obtain a sufficiently accurate estimate of $f(\omega_c)$ then R_c will have a standard exponential distribution.

Spectral density estimation

Lange and Zeger (1997) used a rectangular smoothing kernel in both space and frequency to estimate $f(\omega)$. This approach tends to produce biased estimates at boundaries between obviously different spectral densities.

We use a smoothing spline to estimate the spectral density from the log-periodogram of the voxel time-series (Wahba, 1980). One advantage of working on log-scale is that large high frequency artifacts are down weighted and thus avoids the biased estimates exhibited by other methods.

To improve the estimation we use a spatially anisotropic filter that only smooths spectral density estimates that are similar. Specifically we use a multivariate normal approximation to the difference between two independent[4] spectral density estimates. This allows us to decide when two estimates are close enough that they can be smoothed. We have also extended this approach to the case of event-related designs by working with the Fourier transform of the time-series as opposed to just the periodogram (Marchini, 2001).

Calibration

One nice property of this method is that we can calculate R_j at all the other frequencies at which we know there to be no response. Under the null model of no activation all these statistics should have the same distribution and can be used to self-calibrate the method.

Figure 7 illustrates the results of our calibration experiments. We applied the method to 6 null datasets[5] with (blue lines) and without (red lines) spatial smoothing. We compared the distribution of

[3]when the columns of the design matrix are close to being eigenvectors of the low-pass filter.
[4]the approximation works well even with the spatial correlation that exists between estimates
[5]datasets in which there is no activation because no stimulus was applied to the subject.

the statistic values at both the design frequency (dotted lines) and at a range of other frequencies (solid lines) to the theoretical exponential distribution. The statistic values are shown on $-\log_{10}(p\text{–value})$ scale to focus on the tails of the distribution. The line at 45^o represents exact agreement with the theoretical distribution. This plot shows how spatial smoothing provides a more accurate estimate of $f(\omega)$. This method of calibration also allows us to choose the smoothing spline parameter in an objective fashion.

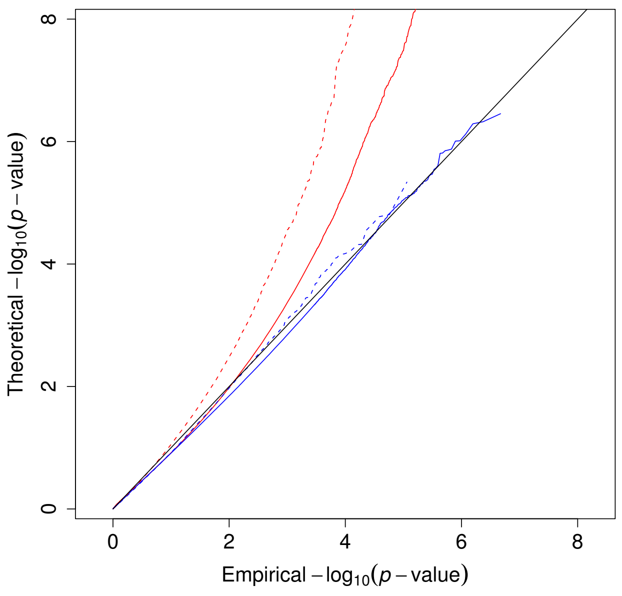

Figure 7: Calibration results

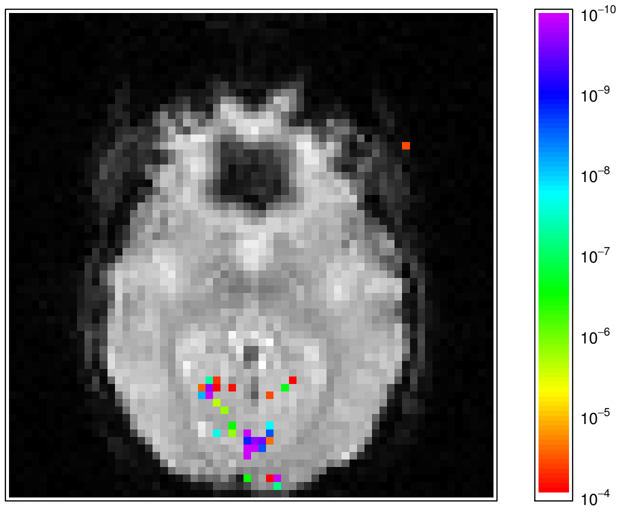

Figure 8: p-value image using spline smoothing spectral density estimation.

Figures 8 and 9 show statistic images produced by using a smoothing spline and an AR(1) model (Bullmore et al., 1996) for the spectral density at each voxel. The images are shown on p-value scale, thresholded to show only values below 10^{-4} and overlaid onto an image of the slice. The spline smoothing approach shows activation just in the visual cortex. The AR(1) approach identifies the same areas but with additional amounts of false positive activation indicating that this approach is poorly calibrated.

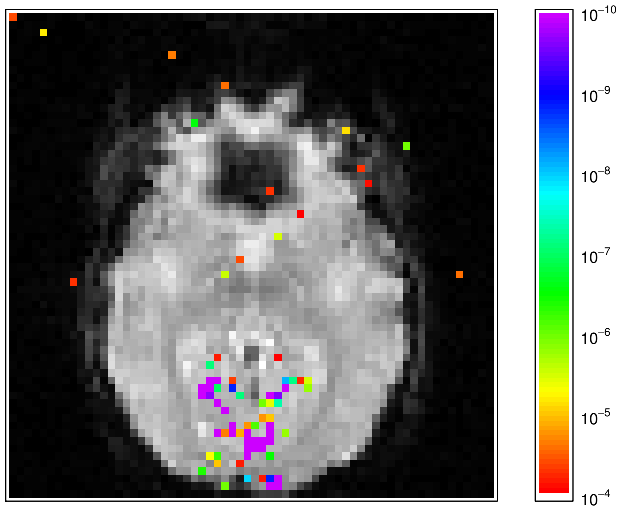

Figure 9: p-value image using AR(1) spectral density estimation.

We have implemented our approach using a mixture of R, C and FORTRAN code. We are currently working on incorporating this method into the **AnalyzeFMRI** package.

Identifying areas of activation statistic

Many different methods are available for the analysis of fMRI statistic images. The most widely used approaches are based on thresholding the statistic map (Worsley and Friston, 1995). The level of the threshold is chosen to protect against false positive activation under the null hypothesis of no activation anywhere in the image. Results from Random Field theory are used to set the appropriate threshold.

This approach tends to attract alot of criticism. In some experiments we know the effect will occur somewhere and we want to estimate the location of the response rather than use a hypothesis test to see if it's there. Also, to validate assumptions required by the Random Field theory results the datasets are significantly spatially smoothed. This tends to blurs the good resolution fMRI affords.

Despite these criticisms this approach has been very successful in answering the questions that neuroscientists want to ask. In addition, for group studies, the resolution is limited by structural variability between individuals. Thus using a small amount of smoothing doesn't matter and can in fact help detection of a response.

More attractive approaches are based on mixture models for levels of activation (Everitt and Bullmore, 1999; Hartvig and Jensen, 2000). It has been suggested that such an approach provides a less arbitrary way of delineating areas of activation although (so far) they have only been used with a 0-1 loss function. It will be interesting and important to investigate the effect of varying the loss function.

More recently it has been suggested that thresholds be defined by controlling the *False Discovery Rate* (FDR) (Genovese et al., 2001). This approach is more focussed on modelling the areas of activation as it is based on controlling the amount of false positive tests compared to all positive tests. In this way the method shares a property of mixture model approaches in that the threshold adapts to the properties of the statistic image.

We are currently in the process of implementing these approaches into the **AnalyzeFMRI** package.

Future plans

The long term goal of the **AnalyzeFMRI** package is to provide a comprehensive software package for the analysis of fMRI and MRI images. We aim to take full advantage of the existing (and future) graphics facilities of R by providing fast interfaces to widely used medical image formats and several graphical exploratory tools for fMRI and MRI datasets. In this way we hope to provide a valuable resource for statisticians working in this field. Hopefully this will negate the need for replication of coding effort and allow researchers new to the field to quickly familiarize themselves with many of the current methods of analysis.

Bibliography

E. Bullmore, M. Brammer, S. C. R. Williams, S. Rabe-Hesketh, N. Janot, A. David, J. Mellers, R. Howard, and P. Sham. Statistical methods of estimation and inference for functional MR image analysis. *Magnetic Resonance in Medicine*, 35:261–277, 1996. 21, 22

B. S. Everitt and E. D. Bullmore. Mixture model mapping of brain activation in functional magnetic resonance images. *Human Brain Mapping*, 7:1–14, 1999. 22

J. H. Friedman. A variable span scatterplot smoother. Technical Report 5, Laboratory for Computational Statistics, Stanford University, 1984. 20

K. Friston, C. Frith, R. Frackowiak, and R. Turner. Characterising evoked hemodynamics with fMRI. *NeuroImage*, 2:157–165, 1995. 21

C. R. Genovese, N. A. Lazar, and T. Nichols. Thresholding of statistical maps in functional neroimgaging using the false discovery rate. Technical report, Department of Statistics, Carnegie Mellon University, 2001. 23

N. V. Hartvig and J. L. Jensen. Spatial mixture modelling of fMRI data. *Human Brain Mapping*, 11:233–248, 2000. 22

A. Hyvarinen. Fast and robust fixed-point algorithms for independent component analysis. *IEEE Transactions on Neural Networks*, 10:626–634, 1999. 19

N. Lange and S. L. Zeger. Non-linear time series analysis for human brain mapping by functional magnetic resonance imaging. *Applied Statistics*, 46:1–29, 1997. 21

J. L. Locascio, P. L. Jennings, C. I. Moore, and S. Corkin. Time series analysis in the time domain and resampling methods for studies of functional magnetic resonance brain imaging. *Human Brain Mapping*, 5:168–193, 1997. 21

J. L. Marchini. *Statistical Issues in the Analysis of MRI Brain Images*. PhD thesis, Department of Statistics, Oxford University, UK, 2001. 21

J. L. Marchini and B. D. Ripley. A new statistical approach to detecting activation in functional MRI. *NeuroImage*, 12:366–380, 2000. 21

P. M. Matthews, P. Jezzard, and S. M. Smith, editors. *Functional Magnetic Resonance Imaging of the Brain: Methods for Neuroscience*. Oxford University Press, 2001. 17

W. N. Venables. Programmers's niche. *R News*, 1(1): 27–30, 2001. 18

W. N. Venables and B. D. Ripley. *S Programming*. Springer, New York, 2000. 18

G. Wahba. Automatic smoothing of the log periodogram. *Journal of the American Statistical Association*, 75:122–132, 1980. 21

K. Worsley and K. Friston. Analysis of fMRI time series revisited – again. *NeuroImage*, 2:173–181, 1995. 21, 22

Jonathan L. Marchini
University of Oxford, UK
marchini@stats.ox.ac.uk