functional Magnetic Resonance Imaging – Methods
Denis Schluppeck
Visual Neuroscience Group
University of Nottingham, UK
2/4

Next 4 3 lectures
1. Spatial and temporal properties of fMRI (+ linearity, convolution)
2. Signal and Noise (+ Fourier domain, convolution)
3. Preprocessing of fMRI data (+ common software tools)
4. Statistics + experimental design (+ linear regression, GLM, multiple comparisons)

Matlab

Quick recap: data
1. numbers (=pixel/voxel)
2. a bunch of numbers on a grid (=slice)
3. a collection of slices (=volume)
4. many volumes over time, acquired every TR (=timeseries)

Data: indexing
• if we have a timeseries of volumes (in 3D + 1D = 4D), we need 4 “indices” or coordinates to uniquely identify a voxel (x,y,z,t)
• multi-dimensional arrays
• we can slice this data in different ways:
  >> data(:,:,12,1) % get slice z=12 at t=1
  >> data(32,:,:,1) ??
  >> data(1,1,12,:) % get timeseries at [1,1,12]

Data: indexing
• if we have a timeseries of volumes (in 3D + 1D = 4D), we need 4 “indices” or coordinates to uniquely identify a voxel (x,y,z,t)
• multi-dimensional arrays
• we can slice this data in different ways:
  >> data(:,:,12,1) % get slice z=12 at t=1
  >> data(32,:,:,1) % y/z slice at x=32, t=1
  >> data(1,1,12,:) % get timeseries at [1,1,12]
imaging (fMRI) responses. Left: hypothetical haemodynamic impulse response function (HIRF) average neuronal activity over time. Bottom row: corresponding functional magnetic resonance states, along with the amplitude of the transients.

Average resonance imaging (fMRI) responses from the visual cortex for 6- and 12-s stimulus presentations.

Figure 2 | fMRI response

Temporal summation of fMRI responses.

The linear transform model of fMRI responses.

Heeger & Ress, NRN (2002)

fMRI response as a linear system

Heeger & Ress, NRN (2002)

Temporal Summation

Heeger & Ress, NRN (2002)
Neural activity: input to fMRI transform

1. fMRI response is approximately a linear system
2. Neural activity is not a linear transform of e.g. visual stimulus
   - neuronal firing rates are > 0 (so at least half-rectifying)
   - response to visual contrast saturates (contrast response function)


Linearity does not always hold

1. very brief events (threshold)
Linearity does not always hold

1. very brief events (threshold)
2. "refractory" effects for very closely spaced events

cf. fMRI adaptation

Simulation
fMRI Response in a block design experiment

Block alternation
Stimulus alternation frequency = 1/12 Hz; (12s cycle)

Block alternation
Stimulus alternation frequency = 1/12 Hz; (12s cycle)

Block alternation
Stimulus alternation frequency = 1/12 Hz; (12s cycle)

Block alternation
Stimulus alternation frequency = 1/12 Hz; (12s cycle)
Noise

- measured data is never perfect...
- sources of (unwanted) variability: heart beat, breathing, movements, ...
- in fMRI data we usually (high-frequency) ‘noise’ and drift

Block alternation / drift
Stimulus alternation frequency = 1/12 Hz; (12s cycle)

Time domain versus Fourier domain

- compare to what you know about image domain → k-space
- two different ways of looking at a signal: one in terms of time: s, ms, the other in terms of frequencies: Hz, s^{-1}, cycles/scan
- MATLAB tools (Eero Simoncelli, NYU)
  http://www.cns.nyu.edu/~eero/math-tools/
  contains additional links to www / books
Frequency domain

$$T$$

Time / signal domain

$$F$$

Frequency domain

{x} \rightarrow {y}

Fourier transform

{x} \leftarrow {y}

Inverse Fourier transform

Where do the specific numbers come from?

$$1/12 \text{s} = 0.083 \text{Hz}$$

Retinotopy / topographic Mapping

Lecture on “Vision” by Dr Peirce

Clever choice of stimulus:
map “spatial location” into temporal delay (travelling wave of activity)

Lots of Fourier transforms...

<table>
<thead>
<tr>
<th>Transform</th>
<th>time domain</th>
<th>Fourier domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fourier Transform</td>
<td>continuous, infinite</td>
<td>continuous, infinite</td>
</tr>
<tr>
<td>Fourier Series</td>
<td>continuous, periodic</td>
<td>discrete, infinite</td>
</tr>
<tr>
<td>DFT</td>
<td>discrete, infinite</td>
<td>continuous, infinite</td>
</tr>
<tr>
<td>DFT</td>
<td>discrete, periodic</td>
<td>discrete, periodic</td>
</tr>
<tr>
<td>DFT</td>
<td>discrete, finite</td>
<td>discrete, finite</td>
</tr>
</tbody>
</table>

FFT Algorithm

• Computes DFT (discrete Fourier Transform) of finite length input
• Very efficient for inputs of lengths $$N = 2^n$$
• Produces 2 outputs, each of size/length equal to that of the input:
  - real part ( cosine coefficients)
  - imaginary part ( sine coefficients)

>> fftdemo % matlab
Convolution

Discrete-time signal: \( x[n] = [x_1, x_2, x_3, \ldots] \)

A system or transform maps an input signal into an output signal:
\[
y[n] = T(x[n])
\]

A shift-invariant, linear system can always be expressed as a convolution:
\[
y[n] = \sum x[m] \cdot h[n-m]
\]
where \( h[n] \) is the impulse response.

Convolution as a sum of impulses
**Convolution as sequence of weighted sums**

<table>
<thead>
<tr>
<th>Past</th>
<th>Present</th>
<th>Future</th>
<th>Input (impulse)</th>
<th>Weights</th>
<th>Output (impulse response)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1/8</td>
<td>1/8</td>
<td>1/2</td>
<td>1/4</td>
<td>7/8</td>
<td>7/8</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1/2</td>
<td>1/4</td>
<td>7/8</td>
<td>7/8</td>
</tr>
</tbody>
</table>

**Matrix multiplication ??**

A is a 2 by 2 matrix

\[
\begin{bmatrix}
1 & 0 \\
0 & 2 \\
\end{bmatrix}
\begin{bmatrix}
2.5 \\
3.2 \\
\end{bmatrix}
\]

\(x\) is a vector (2 by 1 matrix)

\[
\begin{bmatrix}
1 \\
0 \\
\end{bmatrix}
\begin{bmatrix}
2.5 \\
3.2 \\
\end{bmatrix}
\]

\(x\) is a vector (2 by 1 matrix)

\[
1 \times 2.5 + 0 \times 3.2 \\
0 \times 2.5 + 2 \times 3.2
\]

**Convolution as matrix multiplication**

\[
\begin{bmatrix}
1 \\
2 \\
-3 \\
4 \\
\end{bmatrix}
= 
\begin{bmatrix}
1 & 2 & 3 & 0 & 0 \\
0 & 1 & 2 & 3 & 0 \\
0 & 0 & 1 & 2 & 3 \\
0 & 0 & 0 & 1 & 2 \\
\end{bmatrix}
\begin{bmatrix}
1 \\
1 \\
-1 \\
0 \\
\end{bmatrix}
\]

Linear system ↔ matrix multiplication
Shift-invariant linear system ↔ Toeplitz matrix
**Matrix multiplication ??**

A is a 2 by 2 matrix

\[
\begin{bmatrix}
1 & 0 \\
0 & 2 \\
\end{bmatrix}
\begin{bmatrix}
2.5 \\
3.2 \\
\end{bmatrix}
\]

\[
\begin{bmatrix}
1 \times 2.5 + 0 \times 3.2 \\
0 \times 2.5 + 2 \times 3.2 \\
\end{bmatrix}
\]

\[
\begin{bmatrix}
\text{weighted sum of columns} \ldots ax_1 + bx_2 \ldots \text{should ring a bell!} \\
\end{bmatrix}
\]

**Matrix multiplication ??**

A is a 2 by 2 matrix

\[
\begin{bmatrix}
1 & 0 \\
0 & 2 \\
\end{bmatrix}
\begin{bmatrix}
2.5 \\
3.2 \\
\end{bmatrix}
\]

\[
\begin{bmatrix}
1 \times 2.5 + 0 \times 3.2 \\
0 \times 2.5 + 2 \times 3.2 \\
\end{bmatrix}
\]

**Matrix multiplication ??**

A is a 2 by 2 matrix

\[
\begin{bmatrix}
1 & 0 \\
0 & 2 \\
\end{bmatrix}
\begin{bmatrix}
2.5 \\
3.2 \\
\end{bmatrix}
\]

\[
\begin{bmatrix}
1 \times 2.5 + 0 \times 3.2 \\
0 \times 2.5 + 2 \times 3.2 \\
\end{bmatrix}
\]

**Convolution as matrix multiplication**

\[
\begin{bmatrix}
5 \\
2 \\
3 \\
4 \\
\end{bmatrix}
\begin{bmatrix}
1 & 2 & 3 & 0 & 0 & 0 \\
0 & 1 & 2 & 3 & 0 & 0 \\
0 & 0 & 1 & 2 & 3 & 0 \\
0 & 0 & 0 & 1 & 2 & 3 \\
\end{bmatrix}
\begin{bmatrix}
1 \\
2 \\
1 \\
0 \\
\end{bmatrix}
\]

\[
\begin{bmatrix}
5 \\
2 \\
3 \\
4 \\
\end{bmatrix}
\begin{bmatrix}
1 & 2 & 3 & 0 & 0 & 0 \\
0 & 1 & 2 & 3 & 0 & 0 \\
0 & 0 & 1 & 2 & 3 & 0 \\
0 & 0 & 0 & 1 & 2 & 3 \\
\end{bmatrix}
\begin{bmatrix}
1 \\
2 \\
1 \\
0 \\
\end{bmatrix}
\]

**Convention as matrix multiplication**

\[
\begin{bmatrix}
5 \\
2 \\
3 \\
4 \\
\end{bmatrix}
\begin{bmatrix}
1 & 2 & 3 & 0 & 0 & 0 \\
0 & 1 & 2 & 3 & 0 & 0 \\
0 & 0 & 1 & 2 & 3 & 0 \\
0 & 0 & 0 & 1 & 2 & 3 \\
\end{bmatrix}
\begin{bmatrix}
1 \\
2 \\
1 \\
0 \\
\end{bmatrix}
\]

**Convolution Theorem 1**

\[
\begin{bmatrix}
\text{Multiplication in the time domain} \\
\text{"Gabor"}
\end{bmatrix}
\begin{bmatrix}
\text{Convolution in the frequency domain} \\
\text{easier to calculate here} \\
\text{easier to understand here}
\end{bmatrix}
\]

Eero Simoncelli, NYU
Convolution Theorem 2

Multiplication in the time domain ↔ Multiplication in the frequency domain

But why bother with this seemingly complicated business of transforming?

Convolution in the time domain ↔ Convolution in the frequency domain

Convolution in the time domain ↔ Multiplication in the frequency domain

Multiplication in the frequency domain ↔ Multiplication in the time domain

But why bother with this seemingly complicated business of transforming?

For large data sets

- Convolution is a computationally expensive operation
- FFT / IFFT is very efficient
- Point-by-point multiplication is cheap

In some cases...

- ... it's easier to see periodic events, e.g. artefacts due to cardiac cycle / breathing in the frequency domain

Lance Armstrong?

Cardiac pulsation

Respiration

F

T
How would we make a high-pass filter?

As an example, consider the Gaussian function, which allows for a high-pass filter. This simplification, often conceptualized as a Gaussian impulse, is used to compute the Fourier transform of the signal. The behavior of the impulse response, often revealing quite significant results, is a good way to visualize the Fourier transform of the signal.

Eero Simoncelli, NYU
**Linear Algebra / FFT**

- Eero Simoncelli, NYU
  - [http://www.cns.nyu.edu/~eero/math-tools/](http://www.cns.nyu.edu/~eero/math-tools/) contains additional links to www/books
- MIT OpenCourseWare (video lectures)
  - Mathematics, Gilbert Strang, 18.06 course
- Linear Algebra and Its Applications, Gilbert Strang, book

---

**HRF**

- Time (s)
- fMRI response (arbitrary units)

---

**Glover, 1999**

\[ H(t) = \left( \frac{t}{d_1} \right)^{a_1} \exp\left( -\frac{(t - d_1)}{b_1} \right) - \left( \frac{t}{d_2} \right)^{a_2} \exp\left( -\frac{(t - d_2)}{b_2} \right) \]

Default params \([a_1, a_2, b_1, b_2, c] = [6 12 0.9 0.9 0.35]\)


---

**Plot a simple version in Matlab?**

\[ H(t) = \left( \frac{1}{\tau} \right)^2 \frac{\exp(-t/\tau)}{2\tau} \]

\( \tau = 2; \) \% time constant
\( \delta = 2; \) \% time shift
\( t = [0:1:30]; \) \% vector of time points
\( t\text{shift} = \max(t - \delta,0); \) \% shifted, but not < 0

HRF = (tshift/tau).^2 .* exp(-tshift/tau) ... / (2*tau); \% function

figure(1), plot(HRF, 'r'); \% plot it

---

**Signal-to-noise ratio (SNR)**

**Quantifying Signal / Noise**

Raw SNR: used by physicists + engineers to quantify image quality

- Intensity in sample (e.g. brain)
  - 2000
- Intensity outside sample (e.g. outside head)
  - 100
**Contrast-to-noise ratio (CNR)**

CNR: e.g. how good is T1 contrast between white matter (WM) and gray matter (GM) – take two small regions of interest.

<table>
<thead>
<tr>
<th></th>
<th>mean GM</th>
<th>mean WM</th>
<th>noise (o)</th>
<th>cnr</th>
</tr>
</thead>
<tbody>
<tr>
<td>image 1</td>
<td>150</td>
<td>250</td>
<td>100</td>
<td>x</td>
</tr>
<tr>
<td>image 2</td>
<td>60</td>
<td>70</td>
<td>5</td>
<td>?</td>
</tr>
</tbody>
</table>

**Contrast-to-noise ratio (CNR)**

CNR: e.g. how good is T1 contrast between white matter (WM) and gray matter (GM) – take two small regions of interest.

<table>
<thead>
<tr>
<th></th>
<th>mean GM</th>
<th>mean WM</th>
<th>noise (o)</th>
<th>cnr</th>
</tr>
</thead>
<tbody>
<tr>
<td>image 1</td>
<td>150</td>
<td>100</td>
<td>250</td>
<td>x</td>
</tr>
<tr>
<td>image 2</td>
<td>60</td>
<td>10</td>
<td>70</td>
<td>?</td>
</tr>
</tbody>
</table>

**functional signal-to-noise ratio**

functional SNR: (sometimes called functional CNR)

- **signal**: difference between two states of the brain caused by experiment.
- **noise**: variability in those states over time...

**low functional SNR**
how to increase functional SNR?

Signal averaging

Summary

- recap: linear systems
- Matlab
- simulated block design data
- drift + (high-frequency) noise
- Fourier domain, convolution
- raw SNR, CNR, functional SNR

higher field: 1.5T vs 4T (visual cortex)