Table of Contents

Linear Regression explainer

Denis Schlupppeck, 2018-11-09

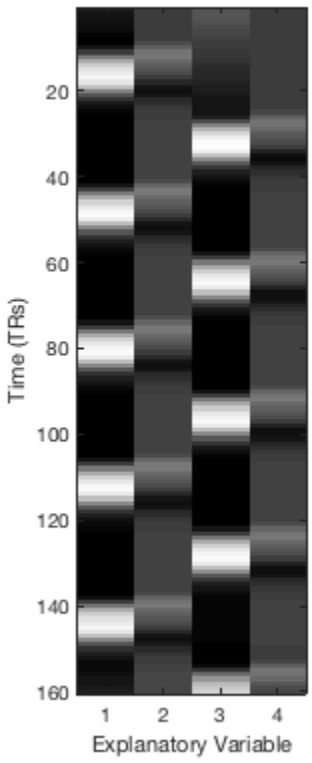
This is a script to explain the matlab mechanics of linear regression with the data set we saw in the Functional Imaging Methods module.

load the design matrix that FSL created (I chopped off a few lines at the top)

```
load('designMatrix.txt')
```

We can inspect what's in the design matrix. Using `imagesc()` works well for this.

```
figure
imagesc(designMatrix)
colormap(gray)
pbaspect([1,3,1]) % make the plot a bit taller 1:3 aspect ratio.
xlabel('Explanatory Variable')
ylabel('Time (TRs)')
```



Inspect and load some imaging data

Use the dafni_test image file (note to self: the NIFTI straight from the scanner can be *read* with `niftiread()` but `niftiinfo()` runs into problem because of the `qform/sform` matrices not being formatted correctly??

```
% img = niftiread('dafni_test');

hdr = niftiinfo('filtered_func_data')
img = niftiread('filtered_func_data');
```

```
hdr =

  struct with fields:

                 Filename: '/Users/lpzds1/matlab/learningMatlab/
imaging-03/faceVhouses/filtered_func_data.nii'
             Filemoddate: '09-Nov-2018 22:09:58'
                Filesize: 62914912
             Description: '5.0.10'
               ImageSize: [64 64 24 160]
          PixelDimensions: [3 3 3 1.5000]
                 Datatype: 'single'
            BitsPerPixel: 32
              SpaceUnits: 'Millimeter'
               TimeUnits: 'Second'
          AdditiveOffset: 0
    MultiplicativeScaling: 1
              TimeOffset: 0
               SliceCode: 'Unknown'
       FrequencyDimension: 0
           PhaseDimension: 0
         SpatialDimension: 0
    DisplayIntensityRange: [0 0]
           TransformName: 'Sform'
               Transform: [1×1 affine3d]
                 Qfactor: -1
                     raw: [1×1 struct]
```

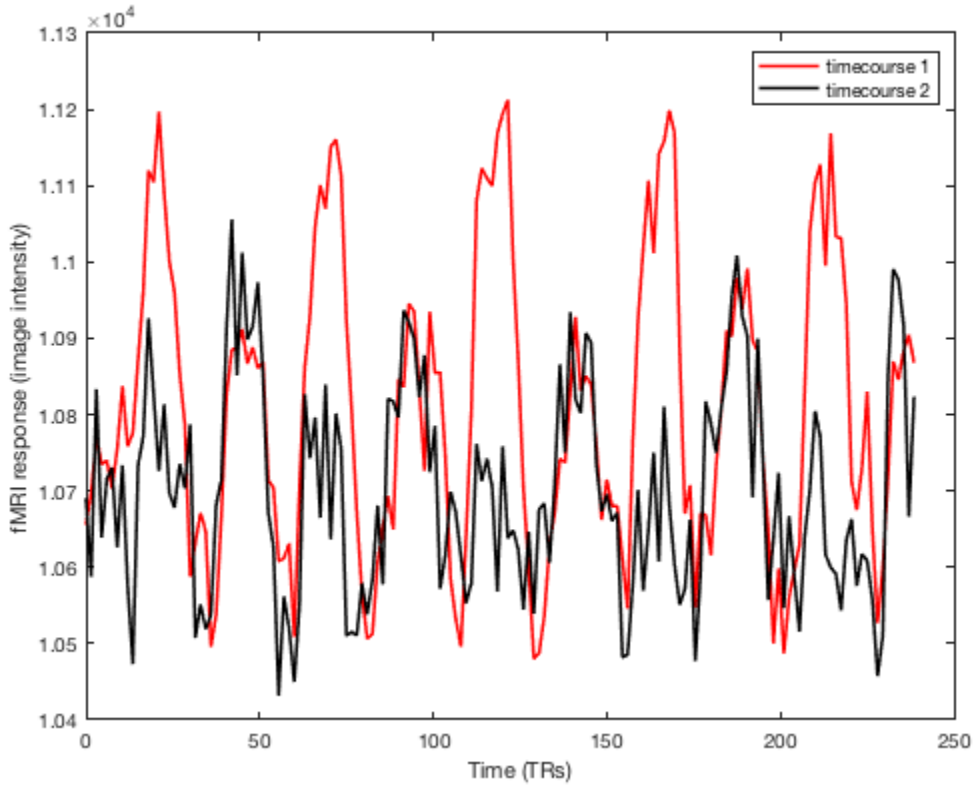
we can use the `returnTimecourse()` function we wrote last time to get a single timecourse back?? Or use indexing in plain matlab right here.

As a specific example, I have chosen two 3d coordinates (from areas with different kinds of responses). They are `[18, 14, 4]` and `[40, 19, 3]`

Two plot them against the proper time units, let's make a time vector and the pick out the two different 1s timecourses.

```
t = (0:159)*1.5; % 0 ... 1.5 ... 3 ... 4.5..

timecourse1 = squeeze( img(18, 14, 4,:) );
timecourse2 = squeeze( img(40, 19, 3,:) );
```

```
figure
plot(t, timecourse1, 'r', 'linewidth', 1.5);
hold on
plot(t, timecourse2, 'k', 'linewidth', 1.5);
xlabel('Time (TRs)')
ylabel('fMRI response (image intensity)')
legend('timecourse 1', 'timecourse 2')
```



Linear regression in action

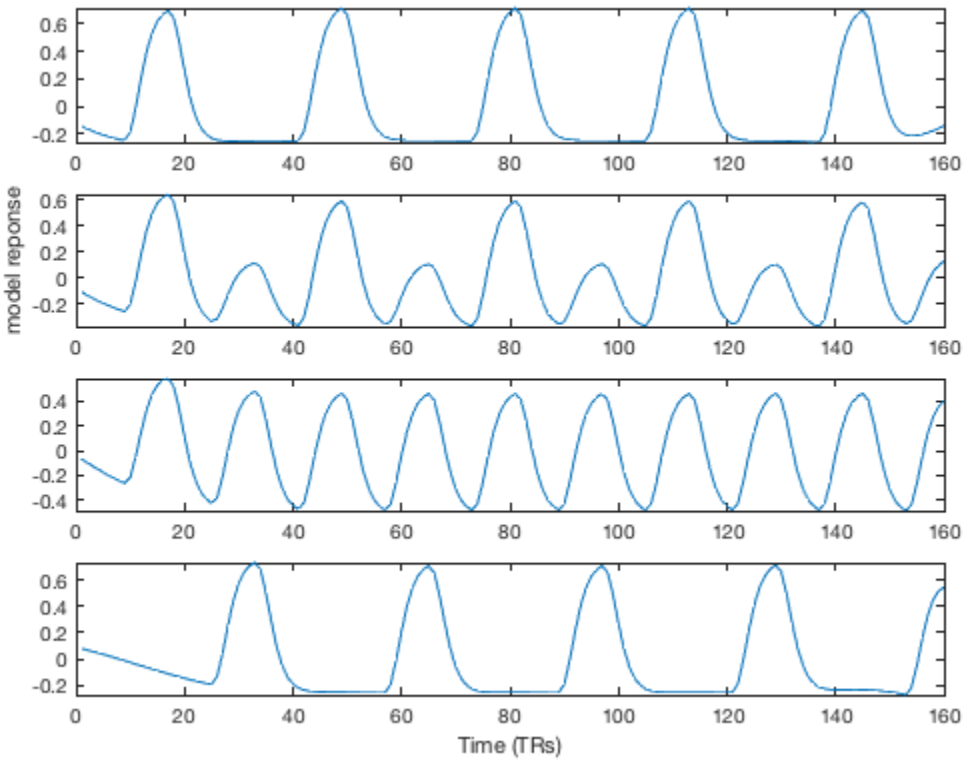
$$\mathbf{y} = \mathbf{X}\beta$$

The $\beta$$ weights in the column vector control how much of each column in the design matrix is in the "mix". The following plot shows you what happens when you mix the designMatrix columns in different proportions:

- `[1; 0; 0; 0]` one unit of column 1
- `[1; 0; 0.5; 0]` one unit of column 1 and half a unit of column 3
- `[1; 0; 1.0; 0]` one of col1 and one of col3
- you get the idea...

```
figure
```

```
subplot(4,1,1)
plot( designMatrix * [1; 0; 0; 0] )

subplot(4,1,2)
plot( designMatrix * [1; 0; 0.5; 0] )
ylabel('model reponse');

subplot(4,1,3)
plot( designMatrix * [1; 0; 1.0; 0] )

subplot(4,1,4)
plot( designMatrix * [0; 0; 1.0; 0] )
xlabel('Time (TRs)');
```



Now we can look at how we can find out the numbers that provide the best fit.

One subtlety is that we also need deal with a possible constant "offset" between model and data. So we add a column of ones... to the designMatrix

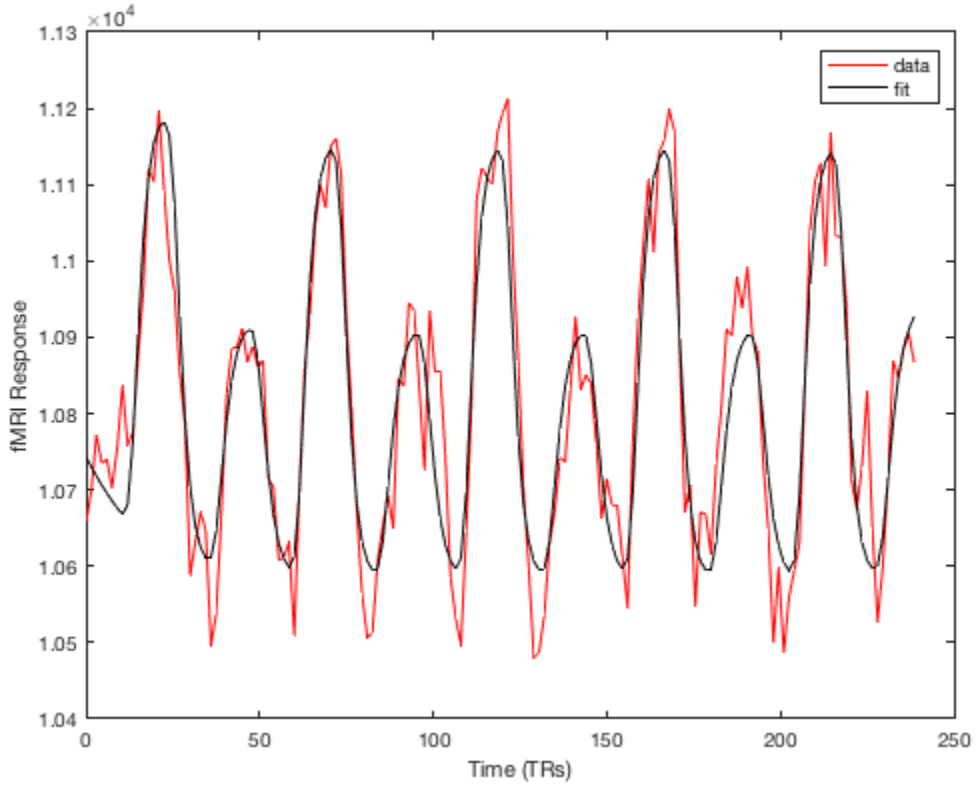
The `X\y` step looks simple but it actually is a complicated command. Have a look at `doc \` or `doc mldivide` (which does the same).

```
X = [designMatrix, ones(length(designMatrix), 1)];

b =  X \ timecourse1;
```

how big are those numbers? `round()` to get a rough idea when we display them in the command window:

```
round(b)

% once we have those numbers (not perfect! but the best ones...) we
 can use
% them to calculate what that FIT is...

modelfit = X*b;

figure
plot(t, timecourse1, 'r', t, modelfit , 'k')
xlabel('Time (TRs)');
ylabel('fMRI Response')
legend('data', 'fit')
```

```
ans =

  5×1 single column vector

        572
        579
        331
        194
      10814
```



Published with MATLAB® R2018a