Portfolio 4: fMRI preprocessing exercise

Studygroup 2: Sebastian Engen, Fredrik Sejr, Signe M.R Holdgaard, Roxana Petrache, Nanna Bernth

In this exercise we are going to prepare fMRI data for analysis and look at some of the output. The data is the same dataset as last week, although this week we will be looking at all the fMRI data from one participant (participant 1).

Deadline

March 1, 2018.

Data

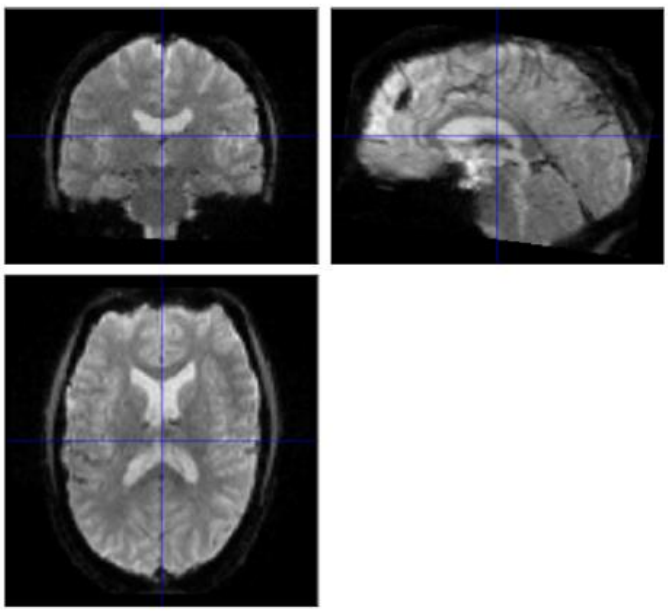
The data can be found in a zip-file at blackboard entitled "fMRI_data_raw.zip". Note that this file contains
400 functional images (called f. . . nii) and 1 structural anatomical image (called s. . . nii).
Save the structural data to a separate file.

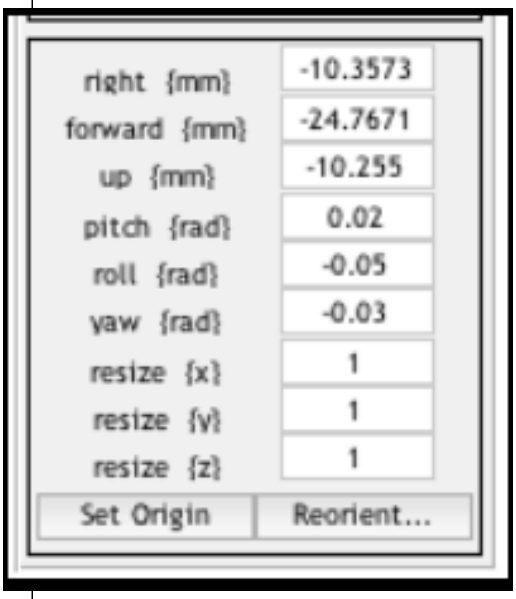
Tasks

1. Initial alignment of data to standard stereotachtic space (MNI-space).
Attempt to position the anterior commissure in [0,0,0] of the first functional image using the Display functionin SPM.

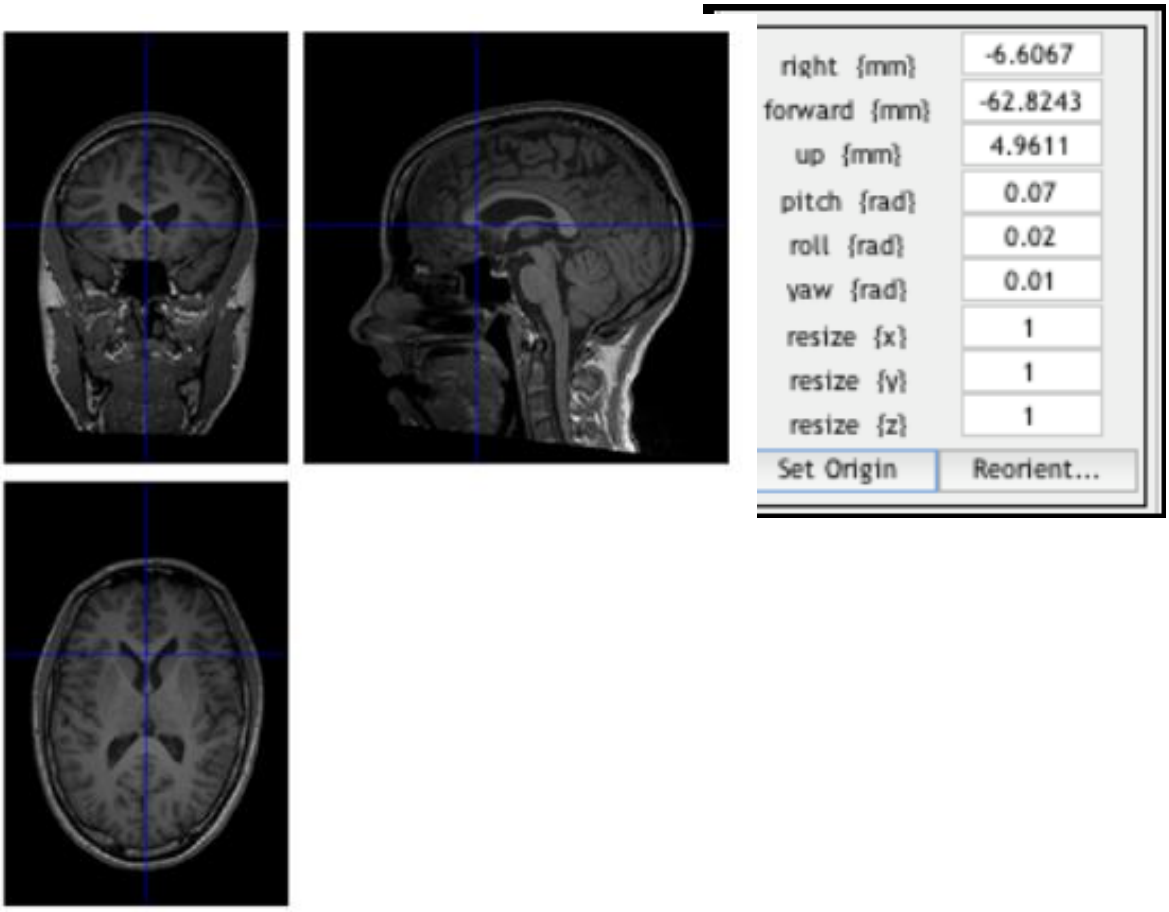
1.a. How much does it have to be moved (indicate 3 translations and 3 rotations)?
Apply transformation to all functional images. Align the anterior commisure of the structural image to [0,0,0].

We set origin and then reorient all the photos in the map.



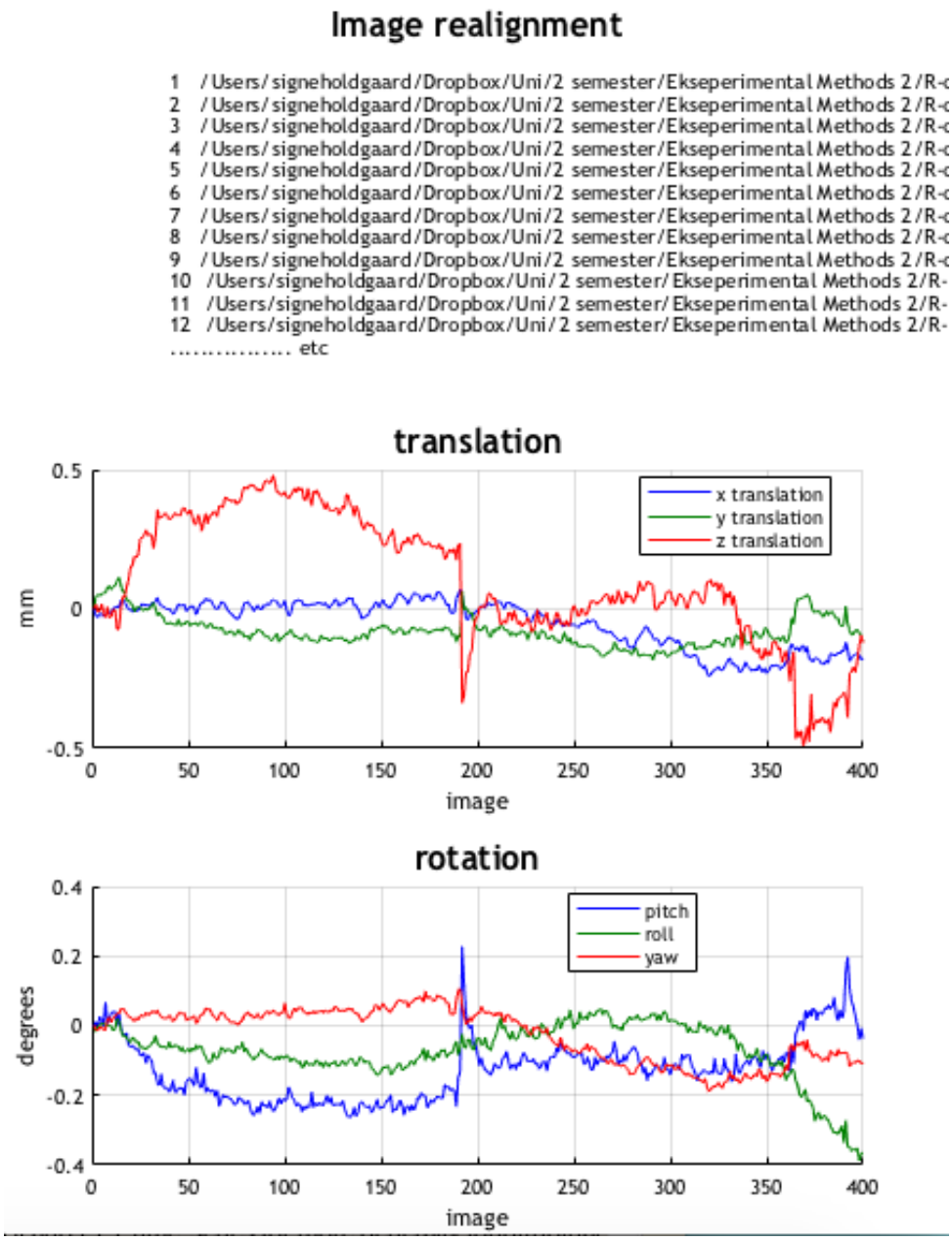


1.b. How much does that have to be moved?

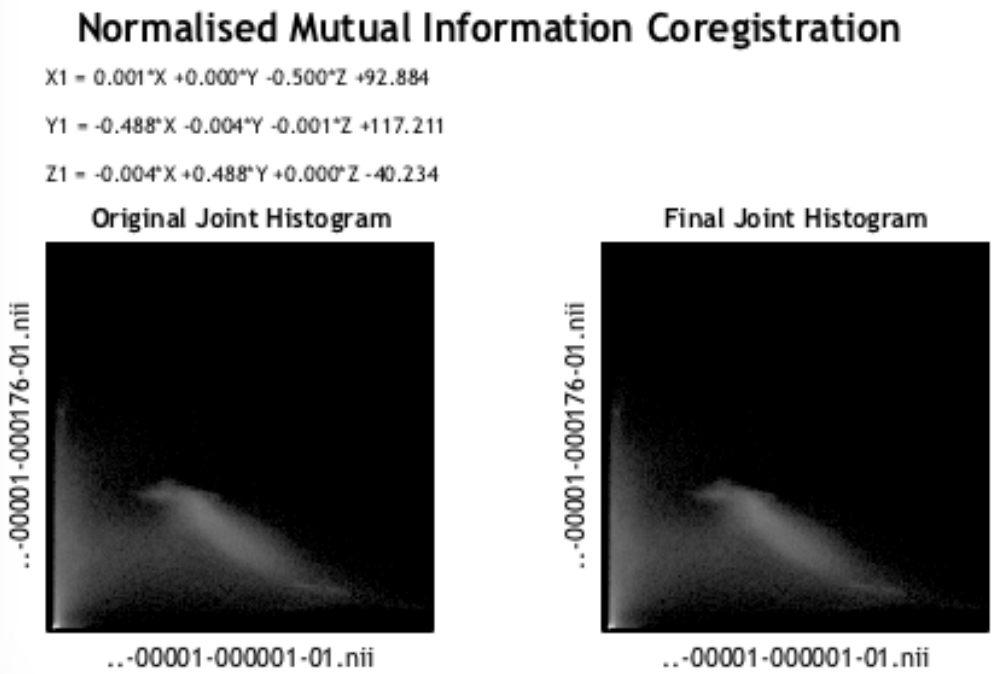


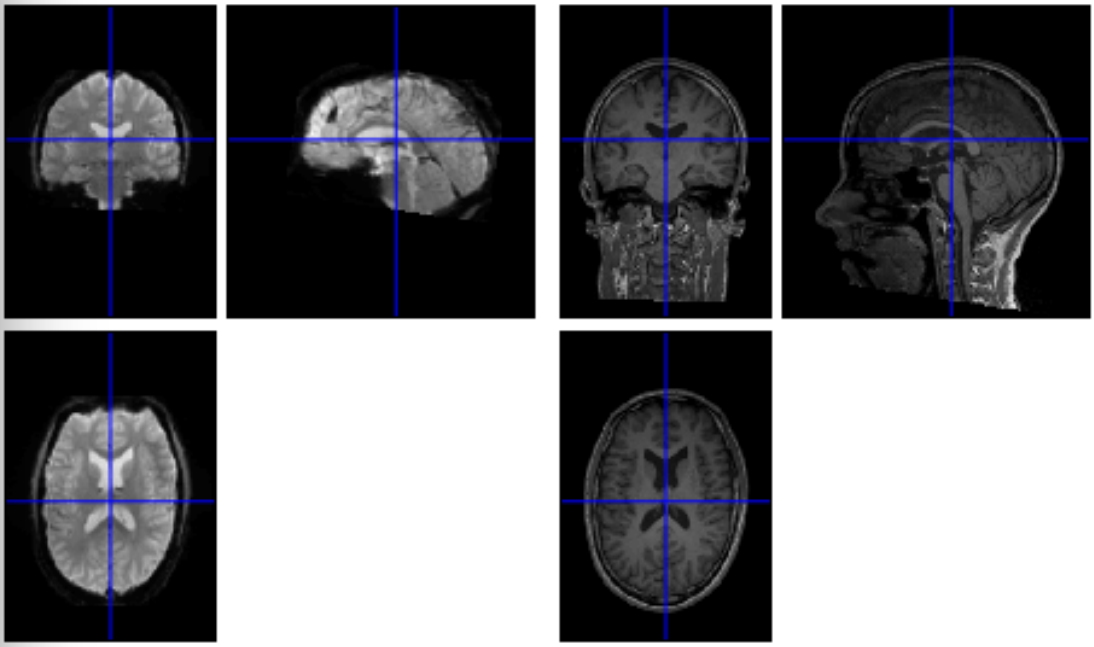
2. Preprocessing of fMRI data
Follow the example in the SPM12 manual chapter 30. Apply the same preprocessing procedure to the current data. This means:

2.a. realignment,

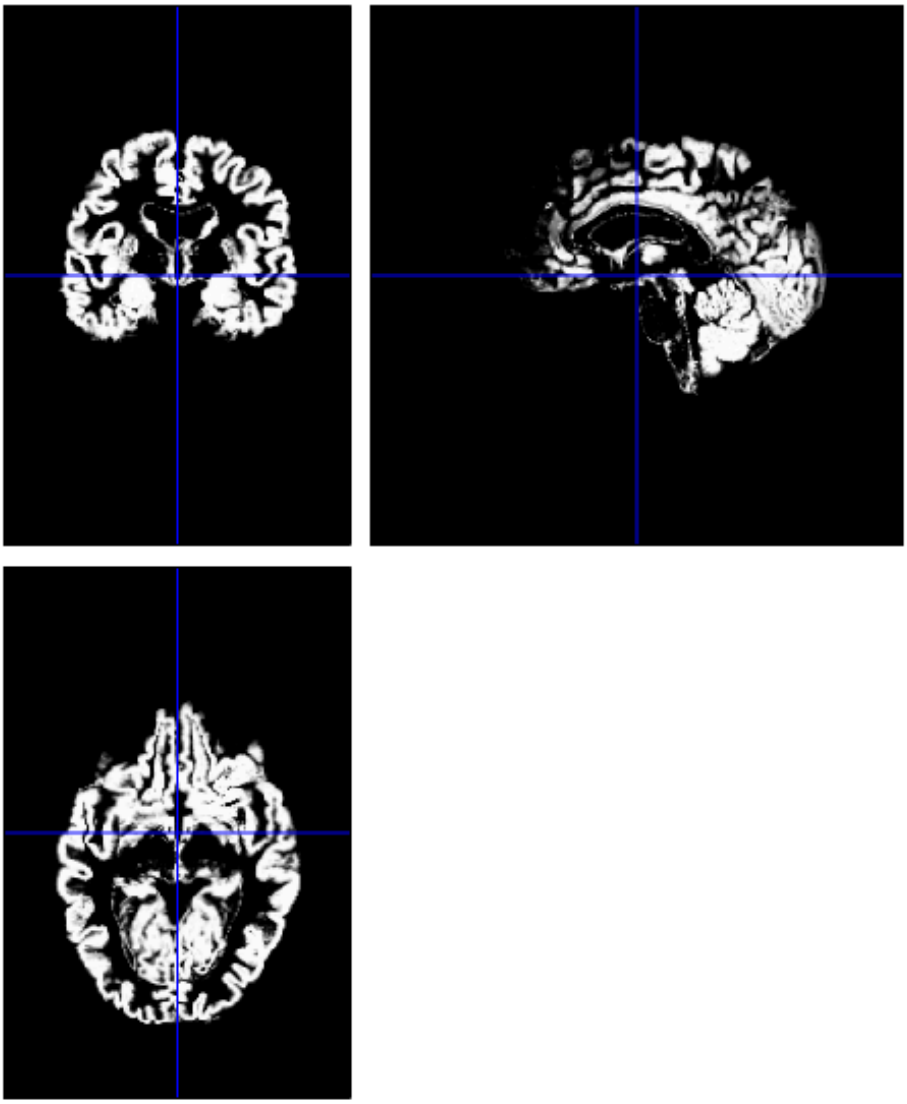


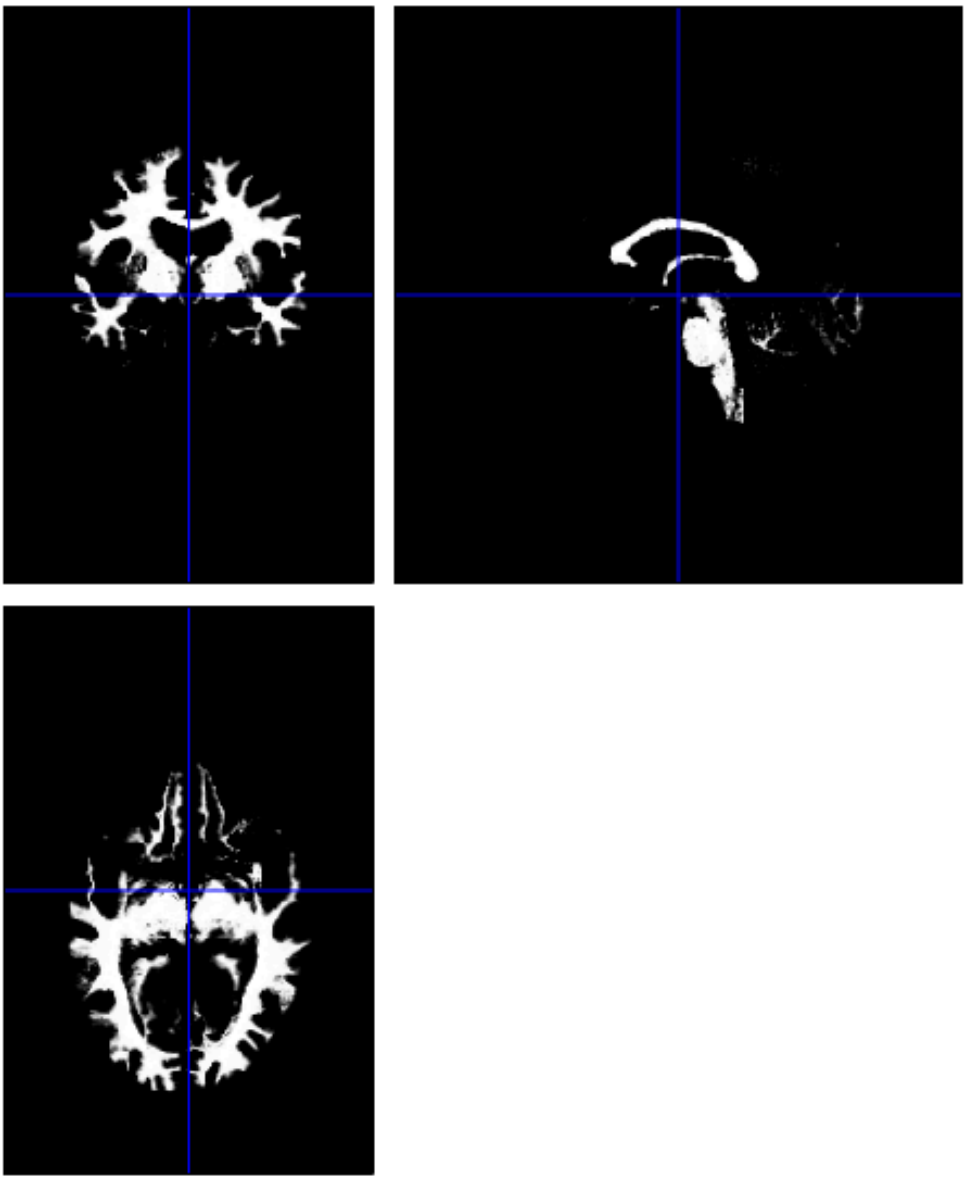
2.b. co-registration of function and structural data (hint: use "dependency" to point to the mean functional image),

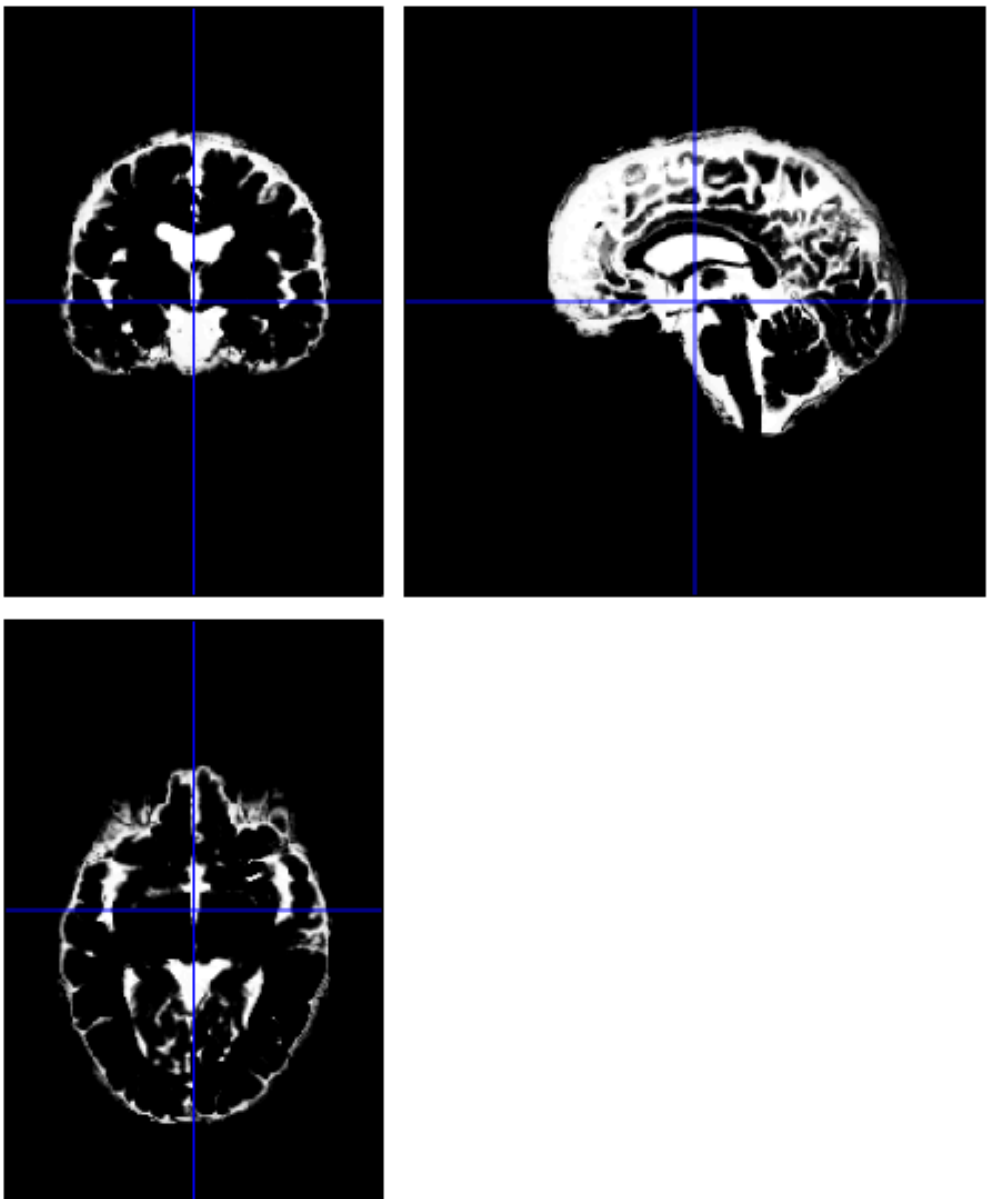


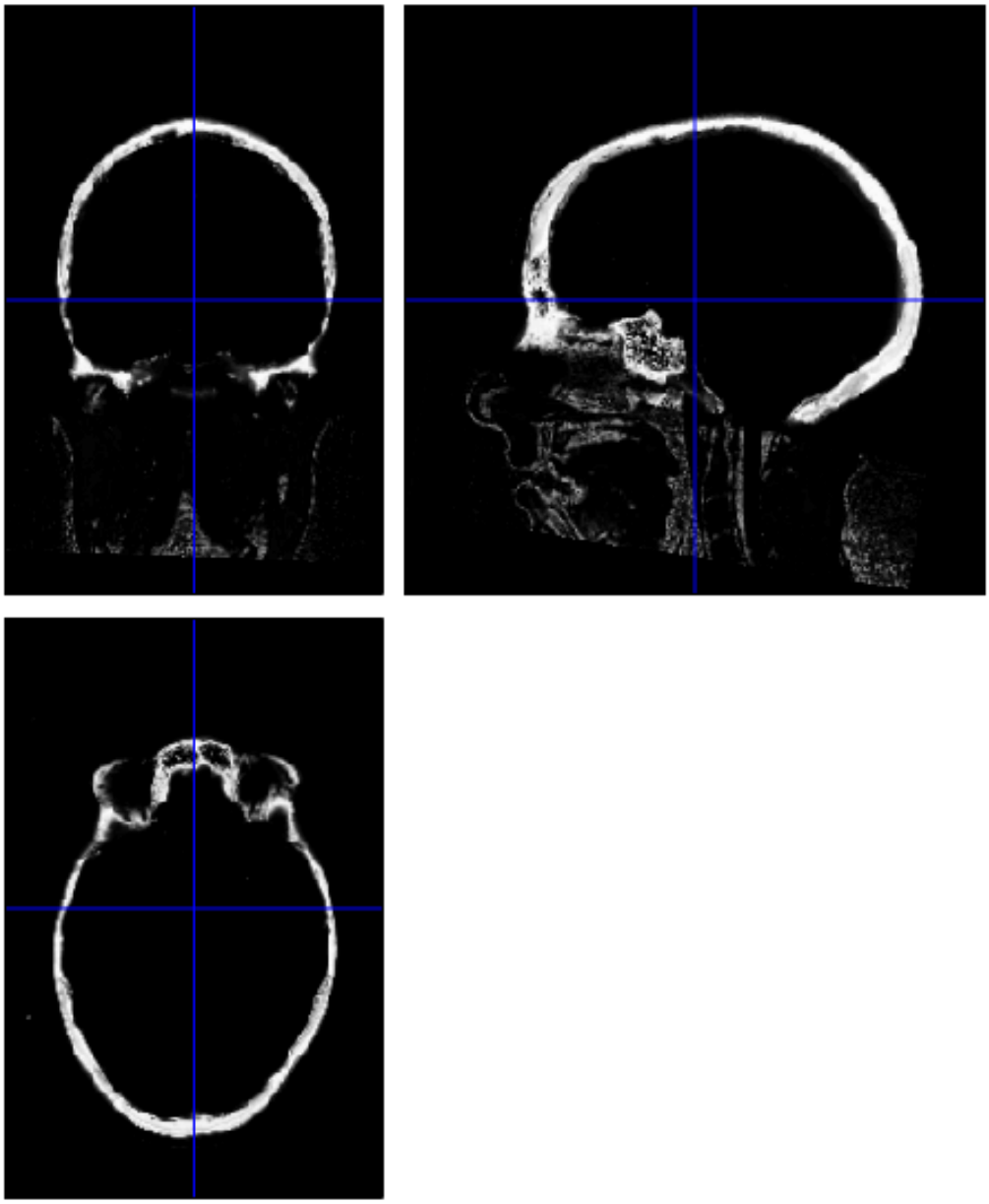


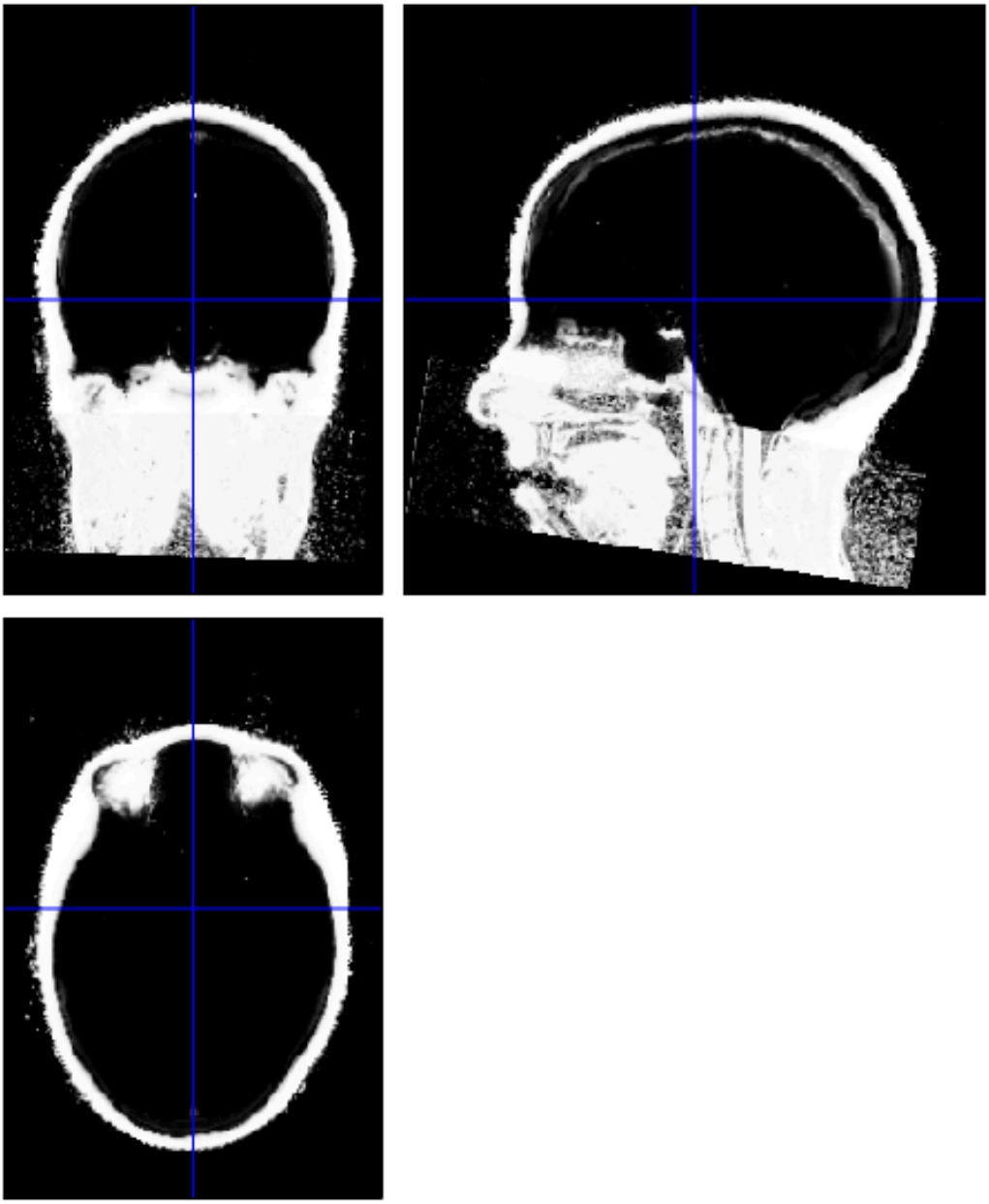
2.c. segmentation of structural data (again, use dependency to point the co-registered structural image)

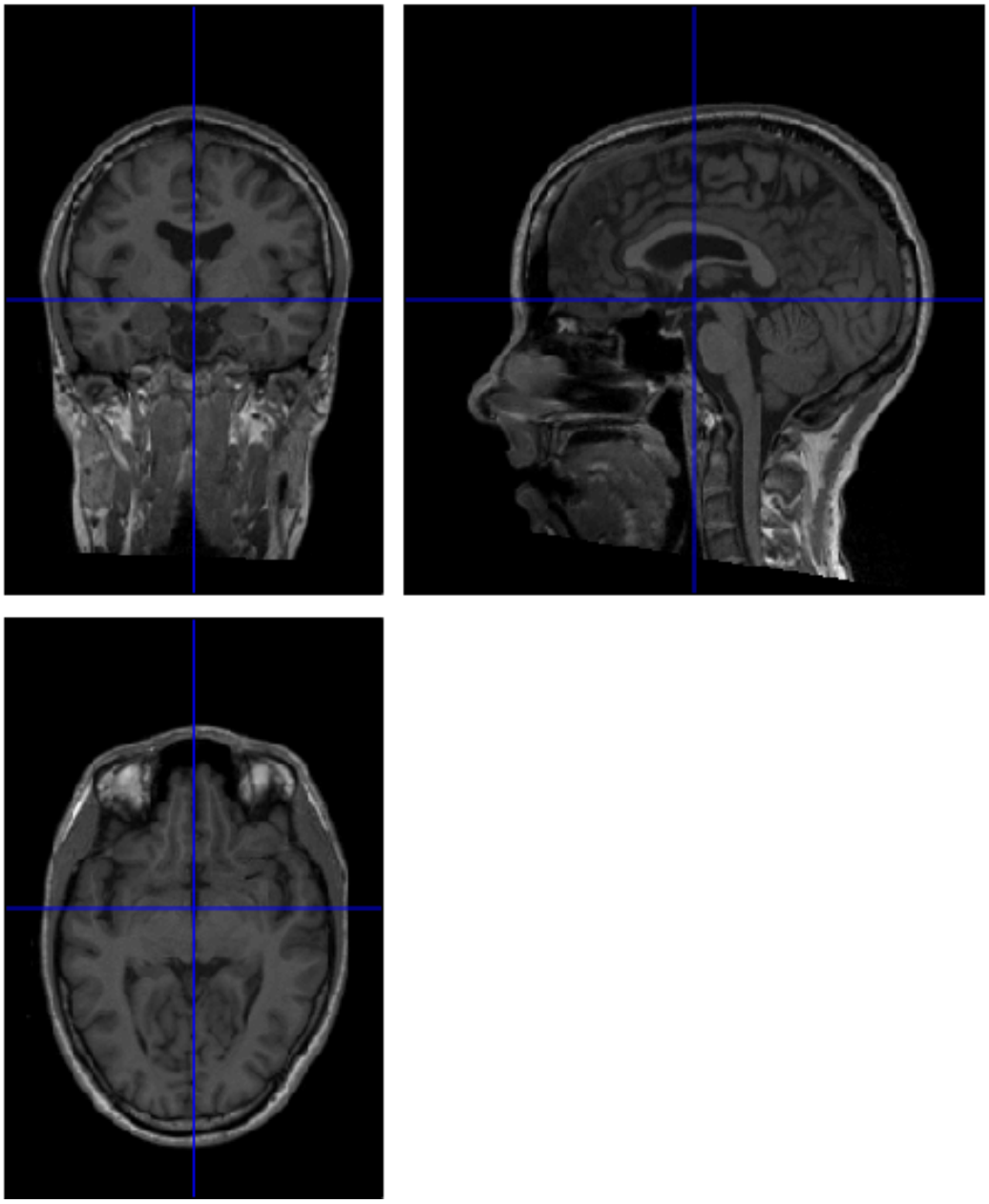












2.d. normalization using the forward deformation field from segmentation (hint: dependency and NB: No
need to change voxel size), and 1

✓

This creates new files in the folder that has been normalized

2.e. smoothing (choose dependency and output from normalization) using a [8,8,8] mm FWHM gaussian
kernel.

✓

This creates new files in the folder that has smoothed the normalized files

Portfolio 4

Nanna Bernth, Sebastian Engen, Fredrik Sejr, Signe Rahbek, Roxana Petrache

23 feb 2018

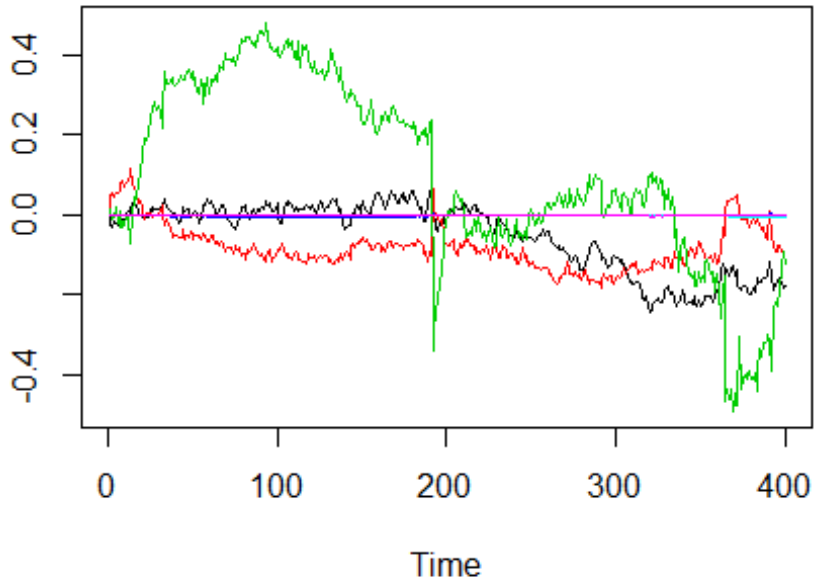
Load the data and libraries

```
library(pacman)

p_load(Hmisc, corrgram)

#load in design
fmri <-as.matrix(read.delim("rp_fSubjectNo0001-0005-00001-000001-01.txt",
header=FALSE, sep = ""))

#making it into a time-series
fmri2 <-ts(fmri)

#Make it into a data frame
fmri_df <- data.frame(fmri2)
```

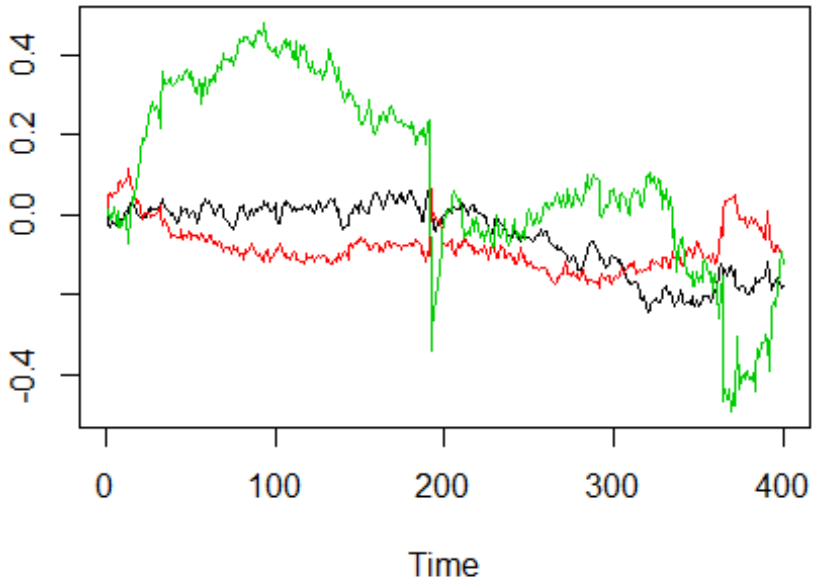
3.a.

Make a lineplot of the realignment parameters in R.

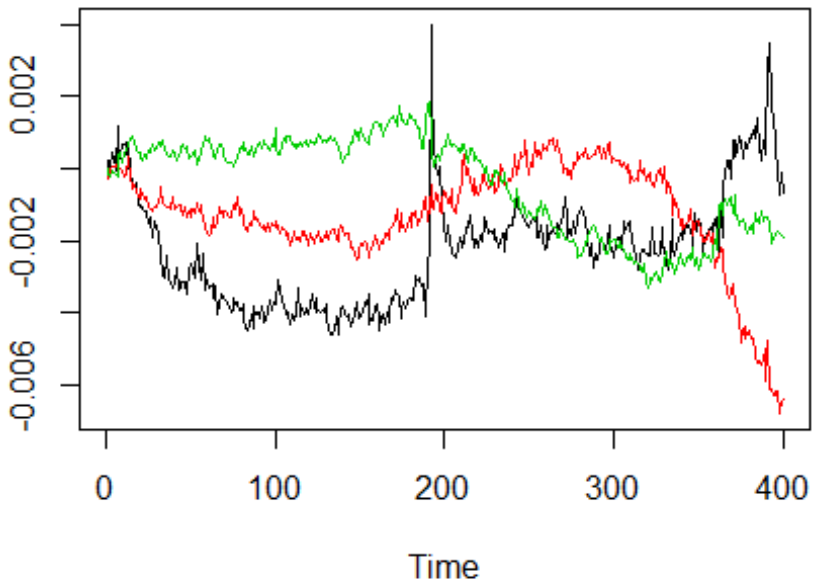
```
#Plotting with ts.plot
ts.plot(fmri_df, col = 1:6)
```




```
#We observe that some of the varibales appears to be on a straight line. This we
investigate further by adjusting for the differences with two plots


#make two plots to be true to the diffenrent scales
ts.plot(fmri_df[,1:3], col = 1:3)
```




```r
ts.plot(fmri_df[,4:6], col = 1:3)
```



3.b.

How far has the participant moved for each dimension during the experiment (Hint: use "apply()" to run a function across columns)?

```
#Use apply to find the maximum and minimum values
range <- data.frame(apply(fmri_df, 2, range))

#Subtract the two rows, to see how much the participant moved
movement <- range[2,] - range[1,]
movement

##          V1      V2        V3          V4          V5          V6
## 2 0.3133471 0.29759 0.9715261 0.008585008 0.007625465 0.005127003

#We now have the values for how much the participant moved within each parameter.
```

3.c.

Are any of the realignment parameters significantly correlated with the fMRI model (same model as used in exercise 3)?

Remove linear effects of time from the realignment parameters (hint: 1:400, fit a line and use residuals).

```
#Load in data from portfolio 3
fmrides<-as.matrix(read.csv("portfolio_assignment3_aud_fmri_design.csv",
header=FALSE))

#making it into a time-series
fmrides2<-ts(fmrides)

#Make it into a data frame
fmri_des_df <- data.frame(fmrides2)


#Running cortest on the two dataframes manually
cortest1 <- cor.test(fmri_df[,1], fmri_des_df[,1])
cortest2 <- cor.test(fmri_df[,2], fmri_des_df[,1])
cortest3 <- cor.test(fmri_df[,3], fmri_des_df[,1])
cortest4 <- cor.test(fmri_df[,4], fmri_des_df[,1])
cortest5 <- cor.test(fmri_df[,5], fmri_des_df[,1])
cortest6 <- cor.test(fmri_df[,6], fmri_des_df[,1])

cortest7 <- cor.test(fmri_df[,1], fmri_des_df[,2])
cortest8 <- cor.test(fmri_df[,2], fmri_des_df[,2])
cortest9 <- cor.test(fmri_df[,3], fmri_des_df[,2])
```

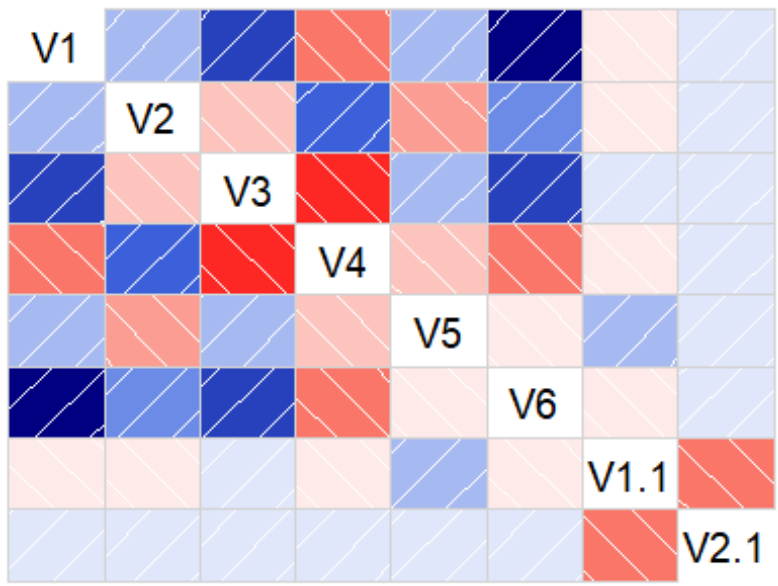
```r
cortest10 <- cor.test(fmri_df[,4], fmri_des_df[,2])
cortest11 <- cor.test(fmri_df[,5], fmri_des_df[,2])
cortest12 <- cor.test(fmri_df[,6], fmri_des_df[,2])

#making all cortests into one dataframe
cortest <- data.frame(rbind(cortest1, cortest2, cortest3, cortest4, cortest5,
cortest6, cortest7, cortest8, cortest9, cortest10, cortest11, cortest12))

#finding significantly correlateded variables
cortest_significant <- which(cortest[,3] < 0.05)
cortest_significant
```

```
## cortest5
##        5
```

```r
#We get that only cortest 5 is significantly correlated with V1.




#Using only one line to find out the same (the smart student-way)
table1 <- rcorr(as.matrix(cbind(fmri_df,fmri_des_df)),type = "pearson")
table1
```

```
##       V1    V2    V3    V4    V5    V6    V1    V2
## V1  1.00  0.23  0.67 -0.49  0.17  0.95 -0.02  0.01
## V2  0.23  1.00 -0.21  0.43 -0.29  0.39 -0.02  0.02
## V3  0.67 -0.21  1.00 -0.85  0.22  0.59  0.03  0.03
## V4 -0.49  0.43 -0.85  1.00 -0.18 -0.46 -0.08  0.01
## V5  0.17 -0.29  0.22 -0.18  1.00 -0.08  0.17  0.02
## V6  0.95  0.39  0.59 -0.46 -0.08  1.00 -0.03  0.03
## V1 -0.02 -0.02  0.03 -0.08  0.17 -0.03  1.00 -0.54
## V2  0.01  0.02  0.03  0.01  0.02  0.03 -0.54  1.00
##
## n= 400
##
##
## P
##    V1     V2     V3     V4     V5     V6     V1     V2
## V1        0.0000 0.0000 0.0000 0.0008 0.0000 0.6807 0.8672
## V2 0.0000        0.0000 0.0000 0.0000 0.0000 0.6838 0.6796
## V3 0.0000 0.0000        0.0000 0.0000 0.0000 0.6045 0.5279
## V4 0.0000 0.0000 0.0000        0.0003 0.0000 0.0949 0.8493
## V5 0.0008 0.0000 0.0000 0.0003        0.0971 0.0005 0.6466
## V6 0.0000 0.0000 0.0000 0.0000 0.0971        0.5295 0.5677
## V1 0.6807 0.6838 0.6045 0.0949 0.0005 0.5295        0.0000
## V2 0.8672 0.6796 0.5279 0.8493 0.6466 0.5677 0.0000
```

```r
#Luckily, we get the same results when we read table1.


#Visually showing the cortest results.
#This is solely for visual use.

Red indicates a negative relation, the darker; the stonger, and the darker blue
the more positive.
corrgram(cbind(fmri_df, fmri_des_df))
```

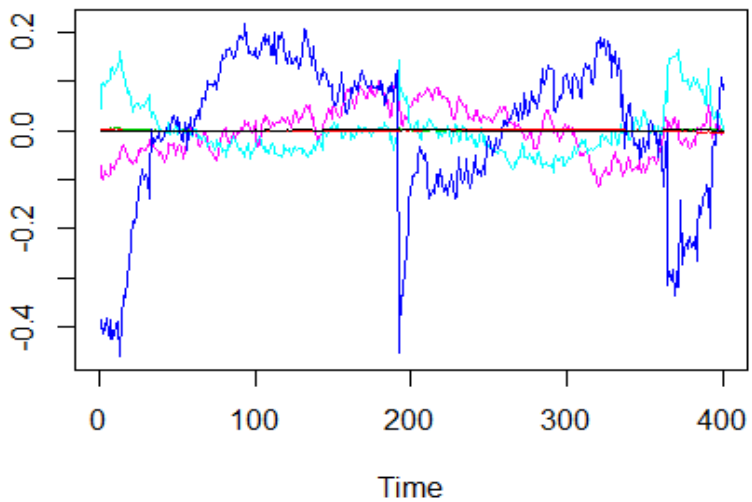



```r
#Removing the effect of time

#creating a dataframe with a colunm that has values from 1-400
time <- data.frame("time" = 1:400)

#make linear model
linear_model <- lm(fmri ~ time$time)

#Call residuals and make a dataframe
residuals <- data.frame(linear_model$residuals)


#make a time series with residuals (to use later for ts.plot, because we love
ts.plot more than matplot)
ts_residuals <- ts(linear_model$residuals)
```

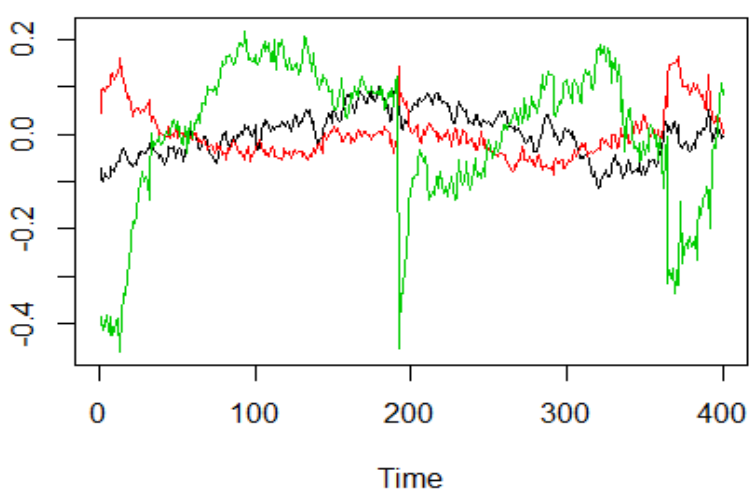
3.d.

Make a lineplot of the realignment parameters with time removed.

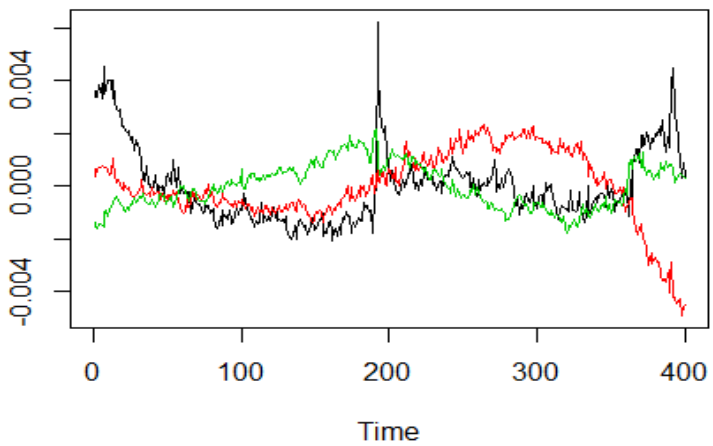
```
#Making two lineplots with the residuals
ts.plot(residuals, col = 22:11)
```




```
#Agian making two plots with the data
ts.plot(residuals[,1:3], col = 1:3)
```



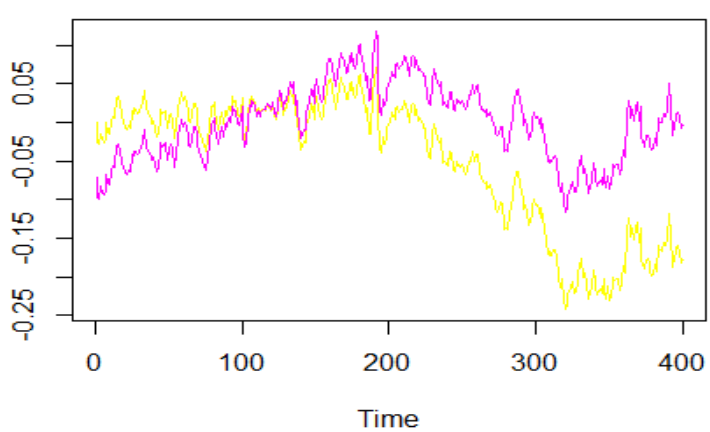
```r
ts.plot(residuals[,4:6], col = 1:3)
```



3.e.

Make a lineplot including only the first realignment parameter before and after removal.

```r
#Making a dataframe with the first parameter from the data and the first column
with residuals
colunm1 <- cbind(fmri_df[,1], residuals[,1])

#Making a line plot
ts.plot(colunm1, col = 7:6)
```



3.f.

Are the realignment parameters (corrected for effects of time) now correlated with the fMRI model?

```
#Same procedure as in 3.c
#First the studendt with too much time method:
corrected_cortest1 <- cor.test(residuals[,1], fmri_des_df[,1])
corrected_cortest2 <- cor.test(residuals[,2], fmri_des_df[,1])
corrected_cortest3 <- cor.test(residuals[,3], fmri_des_df[,1])
corrected_cortest4 <- cor.test(residuals[,4], fmri_des_df[,1])
corrected_cortest5 <- cor.test(residuals[,5], fmri_des_df[,1])
corrected_cortest6 <- cor.test(residuals[,6], fmri_des_df[,1])

corrected_cortest7 <- cor.test(residuals[,1], fmri_des_df[,2])
corrected_cortest8 <- cor.test(residuals[,2], fmri_des_df[,2])
corrected_cortest9 <- cor.test(residuals[,3], fmri_des_df[,2])
corrected_cortest10 <- cor.test(residuals[,4], fmri_des_df[,2])
corrected_cortest11 <- cor.test(residuals[,5], fmri_des_df[,2])
corrected_cortest12 <- cor.test(residuals[,6], fmri_des_df[,2])


#making all cortests into one dataframe
corrected_cortest <- data.frame(rbind(corrected_cortest1, corrected_cortest2,
corrected_cortest3, corrected_cortest4, corrected_cortest5, corrected_cortest6,
corrected_cortest7, corrected_cortest8, corrected_cortest9, corrected_cortest10,
corrected_cortest11, corrected_cortest12))


#finding significantly correlateded variables
corrected_cortest_significant <- which(corrected_cortest[,3] < 0.05)
corrected_cortest_significant

## corrected_cortest5
##                  5

#We get the result that still only cortest 5 is significant. However, the p-values
has changes a lot which can also be seen in the visual depiciton of the data in
the corrgram.


#The smart table for the smart student
table2 <- rcorr(as.matrix(cbind(residuals, fmri_des_df)),type = "pearson")
table2

##       V1    V2    V3    V4    V5    V6    V1    V2
## V1  1.00 -0.17  0.11 -0.24 -0.06  0.86 -0.07 -0.07
## V2 -0.17  1.00 -0.83  0.73 -0.43  0.16 -0.03  0.00
```

```
## V3  0.11 -0.83  1.00 -0.88  0.06  0.00  0.01 -0.02
## V4 -0.24  0.73 -0.88  1.00 -0.08 -0.19 -0.08  0.04
## V5 -0.06 -0.43  0.06 -0.08  1.00 -0.45  0.17  0.01
## V6  0.86  0.16  0.00 -0.19 -0.45  1.00 -0.08 -0.03
## V1 -0.07 -0.03  0.01 -0.08  0.17 -0.08  1.00 -0.54
## V2 -0.07  0.00 -0.02  0.04  0.01 -0.03 -0.54  1.00
##
## n= 400
##
##
## P
##    V1     V2     V3     V4     V5     V6     V1     V2
## V1        0.0009 0.0230 0.0000 0.2114 0.0000 0.1655 0.1675
## V2 0.0009        0.0000 0.0000 0.0000 0.0017 0.5311 0.9794
## V3 0.0230 0.0000        0.0000 0.2082 0.9646 0.7900 0.7102
## V4 0.0000 0.0000 0.0000        0.1053 0.0000 0.1008 0.4281
## V5 0.2114 0.0000 0.2082 0.1053        0.0000 0.0005 0.8547
## V6 0.0000 0.0017 0.9646 0.0000 0.0000        0.1132 0.5954
## V1 0.1655 0.5311 0.7900 0.1008 0.0005 0.1132        0.0000
## V2 0.1675 0.9794 0.7102 0.4281 0.8547 0.5954 0.0000
```

```r
#Make a visual depiction of the data
corrgram(cbind(residuals, fmri_des_df))
```