Portfolio 3, Study Group 10

Jesper Fischer, Lasse Hansen & Sarah Hvid Andersen

2/19/2020

```r
pacman::p_load(astsa, ggplot2, reshape, pracma, tidyverse)
```

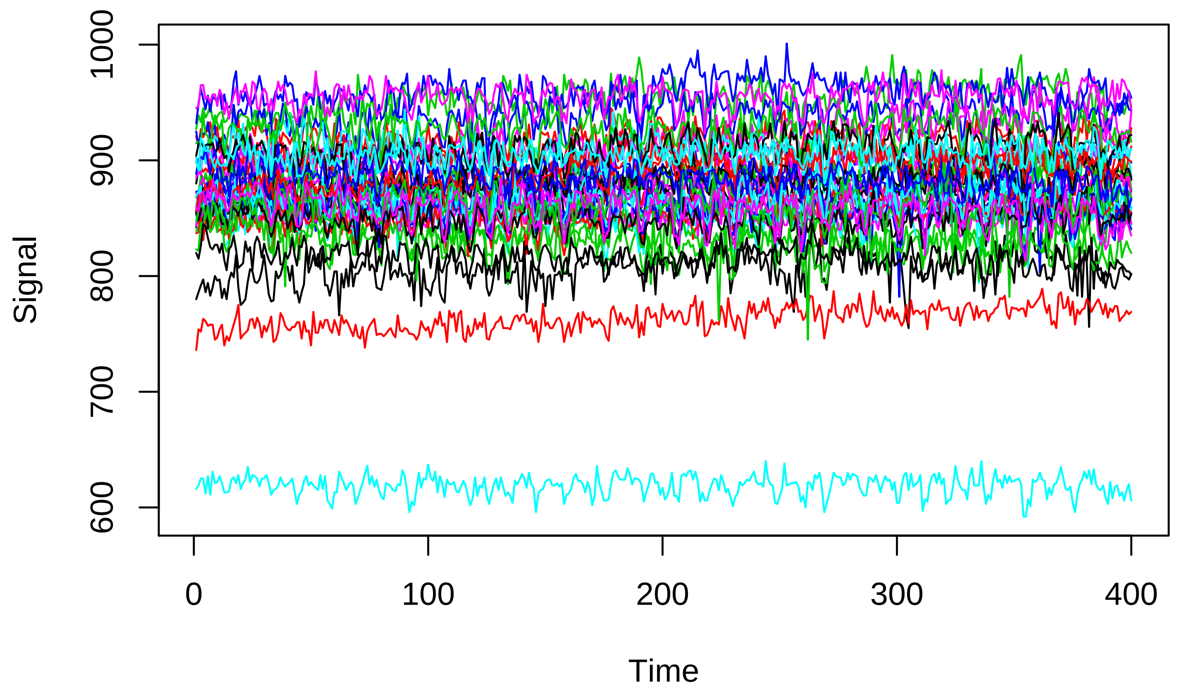
Loading data

```r
#Loading the data as matrices and making it timeseries.
fmri <- as.matrix(read.csv("fmri_data37.csv", header=FALSE))
fmri2 <- ts(fmri)
fmrides <- as.matrix(read.csv("fmri.design.csv", header=FALSE))
fmrides2 <- ts(fmrides)
```

1.a. A figure with lineplots of the data from all participants as a function of time in one figure

We are using the function matplot to plot the time-series data.

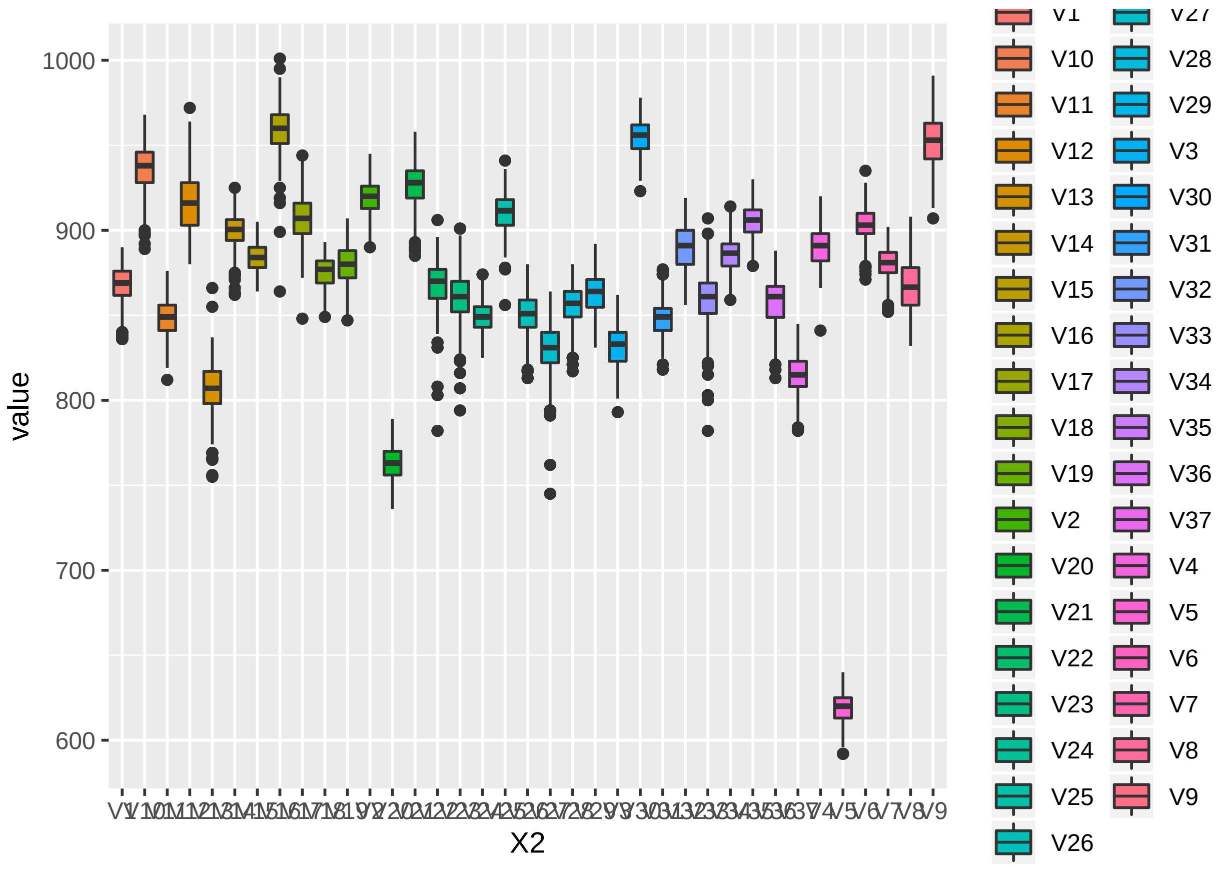
```r
matplot(fmri2, type = 'l', lty = 1, xlab = 'Time', ylab = 'Signal')
```



1.b. A boxplot with the signal intensity for each participant. Note how much the baseline signal can vary between participants.

To be able to display the data as a boxplot for the signal intensity for each participant, we will change the data from wide data into long data. This is done with the melt() function.

```
melt <- melt(fmri2)
ggplot(melt, aes(X2, value, fill = X2)) + geom_boxplot()
```

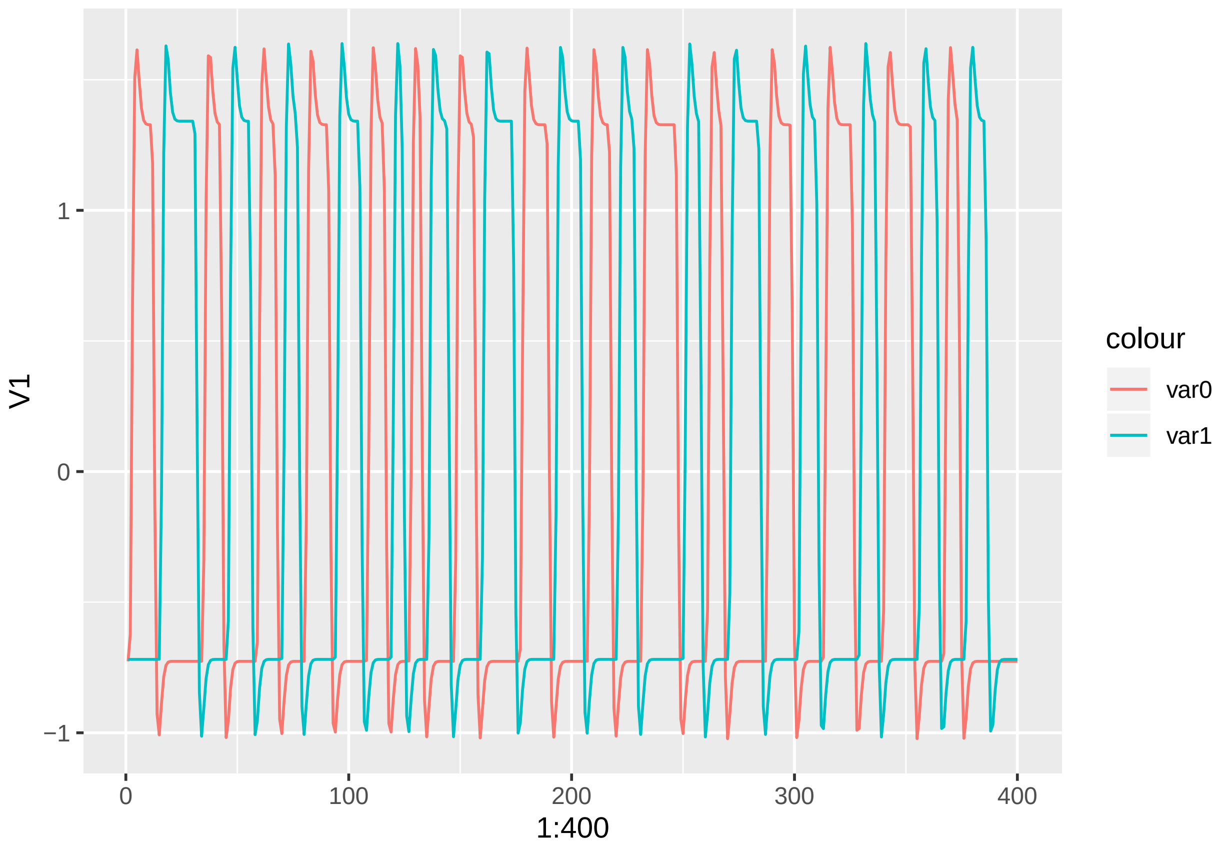


The plot shows that the baseline signal indeed varies a lot between participants.

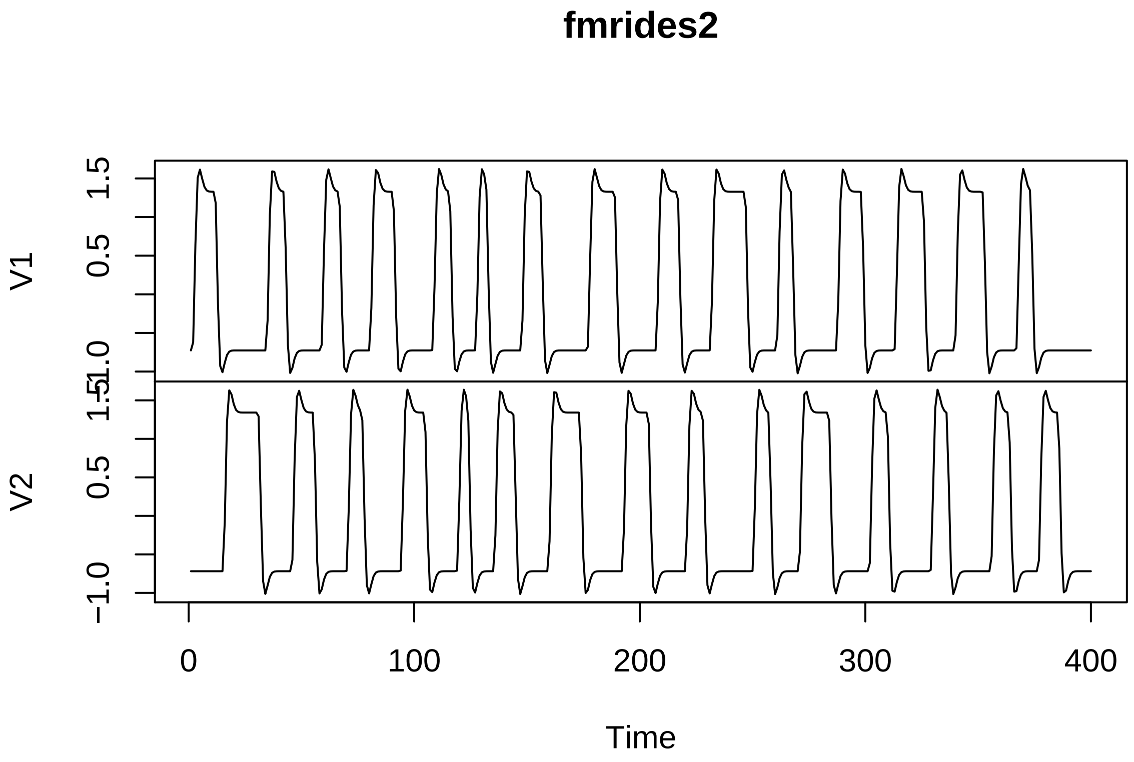
1.c. A lineplots figure with the model covariates.

We have made two different lineplots with two different functions. GGplot is used to plot the two covariates next to eachother. The other plot is made using plot.ts, where the two covariates are plotted under each other:

```
df2 <- as.data.frame(fmrides2)
ggplot(df2, aes(1:400)) +
  geom_line(aes(y = V1, colour = "var0")) +
  geom_line(aes(y = V2, colour = "var1"))
```



```r
plot.ts(fmrides2)
```



2. Based on the shape of the model: How many stories did the participants listen to in each condition?

We assume that each peak in the model symbolises a story being told to the participants, as the design matrix is made on the assumption that a story makes brain activity spike.

```
#We use nrow() and findpeaks() as the findpeaks function gives us rows of the different peaks
nrow(findpeaks(df2$V1))
```

```
## [1] 15
```

```
nrow(findpeaks(df2$V2))
```

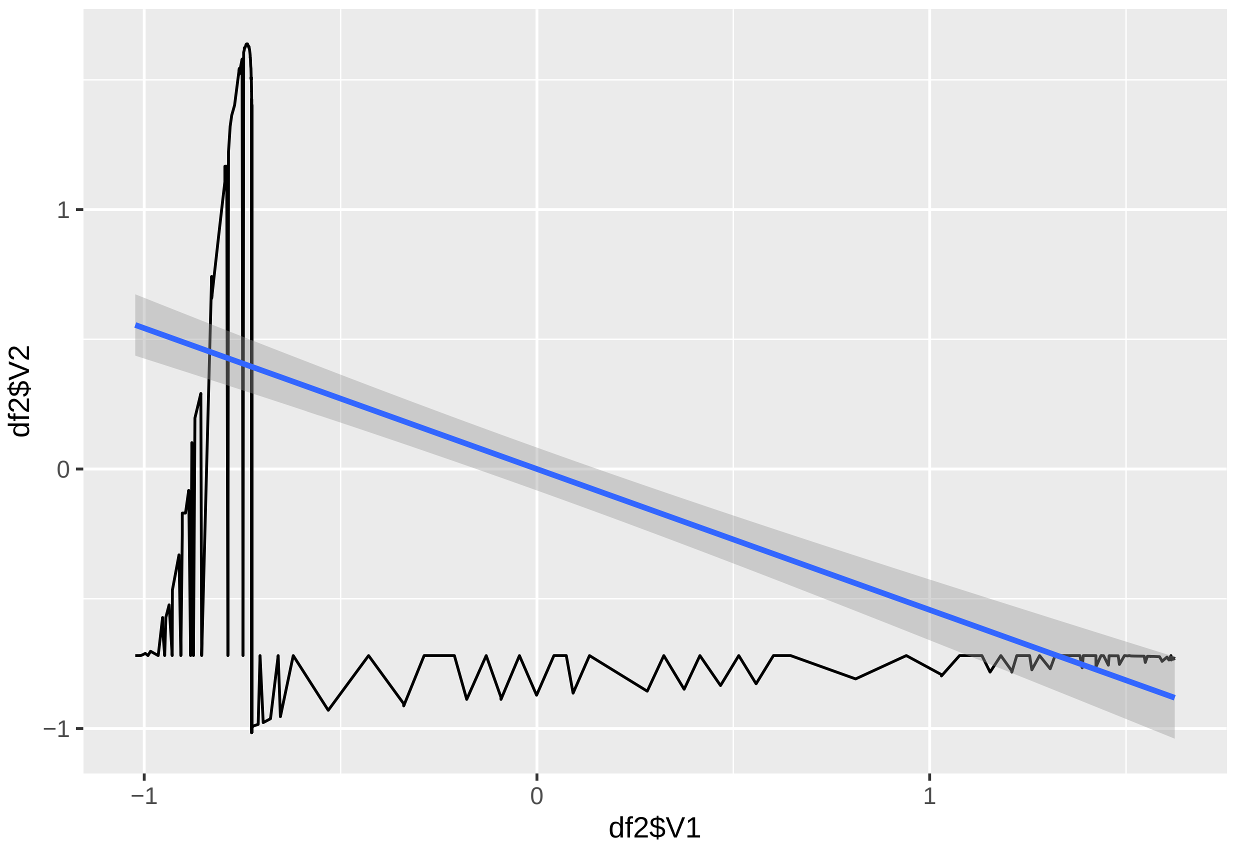
```
## [1] 15
```

rom this we can conclude that the participants listened to 15 stories in each condition

###€ 3.a. Are the two model covariates correlated?

If the two covariates are correlated we will see a linear tendency between them. Therefore we start by plotting the two in a line plot. Afterwards we add a linear regression line to the data.

```
ggplot(df2, aes(df2$V1, df2$V2)) + geom_line() + geom_smooth(method = "lm")
```



Viusally, it seems that there is a negative linear tendency between the two covariates.

To find out if there is a correlation between the covariates we use a correlation test to find the correlation coefficient.

```
cor.test(df2$V1, df2$V2, method = 'pearson')
```

```
##
##  Pearson's product-moment correlation
##
## data:  df2$V1 and df2$V2
## t = -12.894, df = 398, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.6084775 -0.4697617
## sample estimates:
##        cor
## -0.5428111
```

As expected the test shows a negative correlation between the two which is quite high at r = -0.54. Therefore the two model covariates are correlated to some extent.

3.b. Have the covariates been mean-centered?

If data that has been mean centered is modelled, the intercept of that model should be equal to the mean. Therefore we investigate this by making a model. Afterwards we see if the mean is significantly different from the intercept:

```
v1 <- lm(df2$V1 ~ df2$V2)
v2 <- lm(df2$V2 ~ df2$V1)
v1coef <- v1$coefficients
v2coef <- v2$coefficients

t.test(mu = v1coef[1:1], df2$V1)
```

```
##
##  One Sample t-test
##
## data:  df2$V1
## t = -2.2201e-05, df = 399, p-value = 1
## alternative hypothesis: true mean is not equal to -2.387072e-07
## 95 percent confidence interval:
##  -0.09829771  0.09829502
## sample estimates:
##    mean of x
## -1.34875e-06
```

```
t.test(mu = v2coef[1:1], df2$V2)
```

```
##
##  One Sample t-test
##
## data:  df2$V2
## t = 1.4642e-05, df = 399, p-value = 1
## alternative hypothesis: true mean is not equal to 1.31288e-06
## 95 percent confidence interval:
```

```
##  -0.09829484  0.09829893
## sample estimates:
## mean of x
## 2.045e-06
```

Yes, it seems that the covariates have been mean centered, because the t.test showed that there was no signifcant difference between the intercepts of the two models and the mean of the models.

4. Please report the percentage of shared variance in the two covariates.

The shared percentage of variance (R^2) is calculated by taking our r value to the power of 2:

```
corrtest <- cor.test(df2$V1, df2$V2)
corrtest$estimate^2
```

```
##       cor
## 0.2946439
```

Afterwards we can test if the r^2 value is the same when we put the two variables in a linear model:

```
summary(lm(df2$V1 ~ df2$V2))
```

```
##
## Call:
## lm(formula = df2$V1 ~ df2$V2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.41315 -0.75018  0.00973  0.93735  1.22740
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -2.387e-07  4.205e-02    0.00        1
## df2$V2      -5.428e-01  4.210e-02  -12.89   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.8409 on 398 degrees of freedom
## Multiple R-squared:  0.2946, Adjusted R-squared:  0.2929
## F-statistic: 166.3 on 1 and 398 DF,  p-value: < 2.2e-16
```

They are the same. So we conclude that the shared variance of the two covariates is r^2 = 0.2946.

5. Pick one participant's data set.

We filter one participant out of the data set by specifying the first column using hard brackets.

```
p1 <- fmri2[,1]
```

Conduct 6 analyses using lm():

5.a. Fit the model as it is, including intercept.

```r
model <- lm(p1 ~ fmrides2)
summary(model)
```

```
##
## Call:
## lm(formula = p1 ~ fmrides2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -18.8673  -5.2246   0.4506   5.0971  20.0510
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 867.3275     0.3557 2438.64   <2e-16 ***
## fmrides2V1    9.5823     0.4240   22.60   <2e-16 ***
## fmrides2V2    8.9265     0.4240   21.05   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.113 on 397 degrees of freedom
## Multiple R-squared:  0.6097, Adjusted R-squared:  0.6077
## F-statistic: 310.1 on 2 and 397 DF,  p-value: < 2.2e-16
```

5.b. Fit the model as it is, excluding intercept.

Fitting a model that excludes the intercept, is done by adding 0 as an independent variable. This sets the intercept to 0.

```r
model2 <- lm(p1 ~ fmrides2 + 0)
summary(model2)
```

```
##
## Call:
## lm(formula = p1 ~ fmrides2 + 0)
##
## Residuals:
##    Min     1Q Median     3Q    Max
##  848.5  862.1  867.8  872.4  887.4
##
## Coefficients:
##            Estimate Std. Error t value Pr(>|t|)
## fmrides2V1    9.582     51.832   0.185    0.853
## fmrides2V2    8.928     51.831   0.172    0.863
##
## Residual standard error: 869.5 on 398 degrees of freedom
## Multiple R-squared:  0.0001043,  Adjusted R-squared:  -0.00492
## F-statistic: 0.02075 on 2 and 398 DF,  p-value: 0.9795
```

5.c. Fit only the 1st covariate as a model

The first model is created by calling the 1.st covariate using hard brackets.

```r
model2 <- lm(p1 ~ fmrides2[,1])
summary(model2)
```

```
##
## Call:
## lm(formula = p1 ~ fmrides2[, 1])
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -27.8865  -6.8788   0.8642   7.1143  25.2019
##
## Coefficients:
##               Estimate Std. Error  t value Pr(>|t|)
## (Intercept)   867.3275     0.5168 1678.388   <2e-16 ***
## fmrides2[, 1]   4.7369     0.5174    9.155   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 10.34 on 398 degrees of freedom
## Multiple R-squared:  0.174,  Adjusted R-squared:  0.1719
## F-statistic: 83.81 on 1 and 398 DF,  p-value: < 2.2e-16
```

5.d. Fit only the 2nd covariate as a model.

The second model is created with the same method.

```r
model3 <- lm(p1 ~ fmrides2[,2])
summary(model3)
```

```
##
## Call:
## lm(formula = p1 ~ fmrides2[, 2])
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -28.6493  -6.3764   0.8989   7.5855  25.3560
##
## Coefficients:
##               Estimate Std. Error  t value Pr(>|t|)
## (Intercept)   867.3275     0.5371 1614.768  < 2e-16 ***
## fmrides2[, 2]   3.7251     0.5378    6.927 1.74e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 10.74 on 398 degrees of freedom
## Multiple R-squared:  0.1076, Adjusted R-squared:  0.1053
## F-statistic: 47.98 on 1 and 398 DF,  p-value: 1.739e-11
```

5.e. Fit the 2nd covariate to the residuals from analysis 5.c., the 1st covariate only analysis

The residuals from model 5.c is saved in a vector and thereafter used in the lm model as the dependent variable.

```r
res <- model2$residuals
model4 <- lm(res ~ fmrides2[,2])
summary(model4)
```

```
##
## Call:
## lm(formula = res ~ fmrides2[, 2])
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -23.0729  -5.3106   0.9192   5.8245  20.3670
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)   -1.288e-05  4.095e-01    0.00        1
## fmrides2[, 2]  6.296e+00  4.100e-01   15.36   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.19 on 398 degrees of freedom
## Multiple R-squared:  0.3721, Adjusted R-squared:  0.3705
## F-statistic: 235.8 on 1 and 398 DF,  p-value: < 2.2e-16
```

5.f. Fit the 1st covariate to the residuals from 5.d., the 2nd covariate only analysis

The residuals from model 5.d is saved in a vector and thereafter used in the model as the dependent variable.

```r
res2 <- model3$residuals
model5 <- lm(res2 ~ fmrides[,1])
summary(model5)
```

```
##
## Call:
## lm(formula = res2 ~ fmrides[, 1])
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -23.3301  -5.1474   0.5896   5.9121  20.6890
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)   9.116e-06  4.171e-01    0.00        1
## fmrides[, 1] 6.759e+00  4.177e-01   16.18   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.343 on 398 degrees of freedom
## Multiple R-squared:  0.3969, Adjusted R-squared:  0.3953
## F-statistic: 261.9 on 1 and 398 DF,  p-value: < 2.2e-16
```

5.g. Does the order in which the predictor variables are fitted to the data matter for the estimates? If it does, what can explain this?

The estimates of the different models are different. This is because the two covariates are correlated as we found in assignment 3a.

6. Fit the full model to each of the 37 participants' data and extract the coefficients for each participant.

The full model is constructed from the participant matrix and the design matrix. Afterwards we extract the coefficients from the model for display.

```r
fullmodel <- lm(fmri2 ~ fmrides2)
coefficients <- fullmodel$coefficients
```

6.a. Test the two individual hypotheses that the set of coefficient from each covariate is different from zero across the whole group.

To test this we are checking whether the coefficients from the full model are different from zero using a t-test. This is done with a one-sided t-test as we assume that the coefficients are larger than zero, because otherwise our design matrix would not be very effective.

```r
t.test(coefficients[2,], mu = 0, alternative = "two.sided")
```

```
##
##  One Sample t-test
##
## data:  coefficients[2, ]
## t = 16.607, df = 36, p-value < 2.2e-16
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  4.512224 5.767586
## sample estimates:
## mean of x
##  5.139905
```

```r
t.test(coefficients[3,], mu = 0, alternative = "two.sided")
```

```
##
##  One Sample t-test
##
## data:  coefficients[3, ]
## t = 15.603, df = 36, p-value < 2.2e-16
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  4.413274 5.731982
## sample estimates:
## mean of x
##  5.072628
```

From the t-test it is made clear that the coefficients are different from zero.

Make a contrast that investigates the difference between the two covariates, i.e. the two types of stories.

6.b. Test the hypothesis that the contrast is different from zero across participants.

The contrast is made by substracting the one covariate from the other. Afterwards we use a one sided t-test to test if the contrast is significantly different from 0:

```r
contr <- fmrides2[,1]-fmrides2[,2]
t.test(contr, mu = 0, alternative = "two.sided")
```

```
##
##  One Sample t-test
##
## data:  contr
## t = -3.864e-05, df = 399, p-value = 1
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  -0.1726706  0.1726639
## sample estimates:
##    mean of x
## -3.39375e-06
```

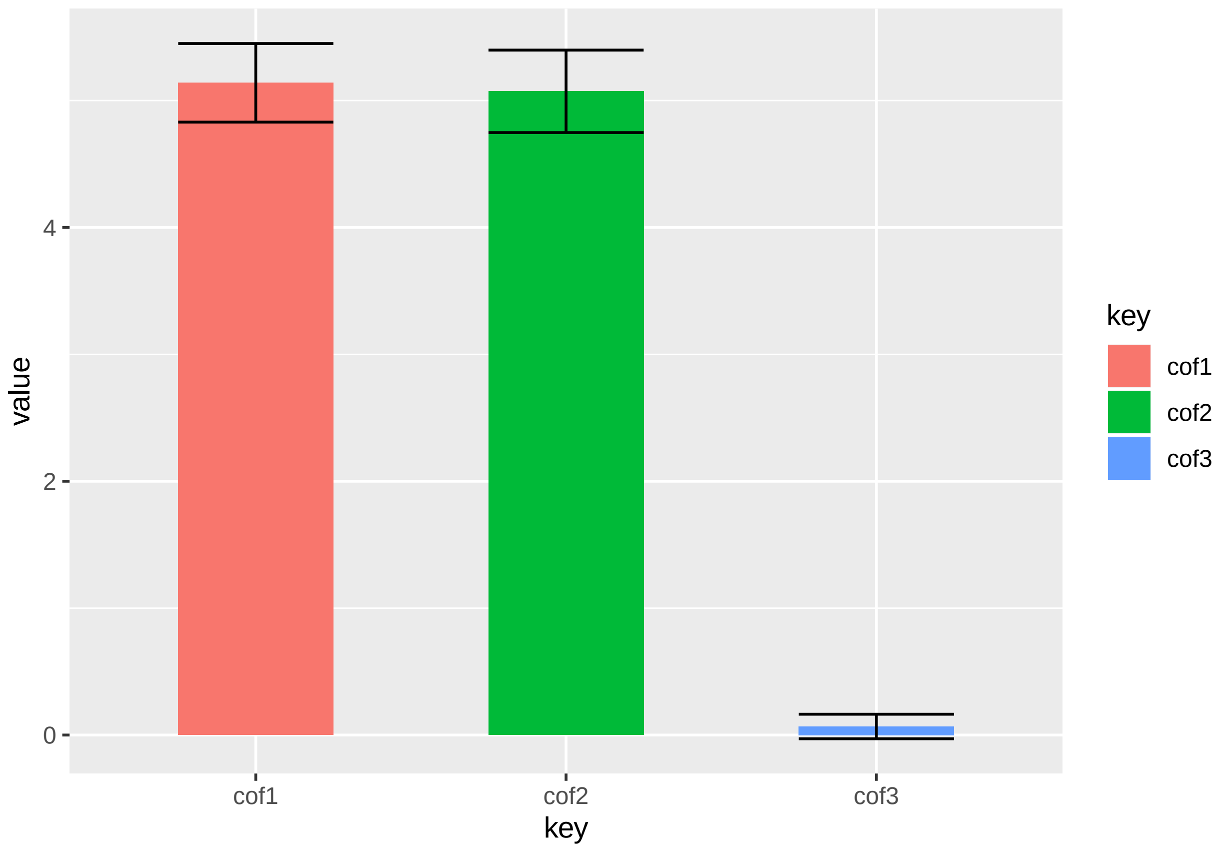
With a p-value = 1 we can see that the contrast is not significantly different from 0.

6.c. Make a bar diagram including the mean effect of the two coefficents and the contrast, including error bars (indicating standard error of mean)

The different coefficients from the model are called using hard brackets. Afterwards the contrast is drawn by substracting the covariates coefficients from each other. Then we use ggplot to visualise their effect including an errorbar.

```r
cof1 <- coefficients[2,]
cof2 <- coefficients[3,]
cof3 <- coefficients[2,] - coefficients[3,]
con1 <- data.frame(cof1,cof2,cof3)
con <- gather(con1)

ggplot(con, aes(key, value, fill = key)) + geom_bar( stat = 'summary', fun.y = mean, width = 0.5) +
geom_errorbar(stat = 'summary', fun.data = mean_se, width = 0.5)
```



7.a. For each partipant, add a covariate that models the effect of time (hint: 1:400)

A covariate called time is added:

```
time <- 1:400
```

7.a. Does that improve the group results in term of higher t-values?

First a model is made with the new covariate as an independent predictor. Afterwards we transpose the vectors containing coefficients from the original model and the model where time is added. This is done so that we can compare the covariates modelled with and without time with a t-test.

```
#The new model with the time vector added as an independent variable
modd <- lm(fmri2 ~ fmrides2 + time)
#The coefficients are saved in a matrix
newcoef <- modd$coefficients
#Transposing the coefficients from the new model
newcoef <- t(newcoef)
#Transposing the coefficients from the old model
coefs <- t(coefficients)
#Two t-test are made
t.test(newcoef[,2], coefs[,2], alternative = "two.sided")
```

```
##
##  Welch Two Sample t-test
##
```

```
## data:  newcoef[, 2] and coefs[, 2]
## t = 0.34288, df = 71.658, p-value = 0.7327
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.6987869  0.9890780
## sample estimates:
## mean of x mean of y
##  5.285051  5.139905
```

```
t.test(newcoef[,3], coefs[,3], alternative = "two.sided")
```

```
##
##  Welch Two Sample t-test
##
## data:  newcoef[, 3] and coefs[, 3]
## t = 0.42593, df = 71.491, p-value = 0.6714
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.692215  1.068331
## sample estimates:
## mean of x mean of y
##  5.260686  5.072628
```

From the t-tests it is shown that there is no significant difference between the two coefficients.

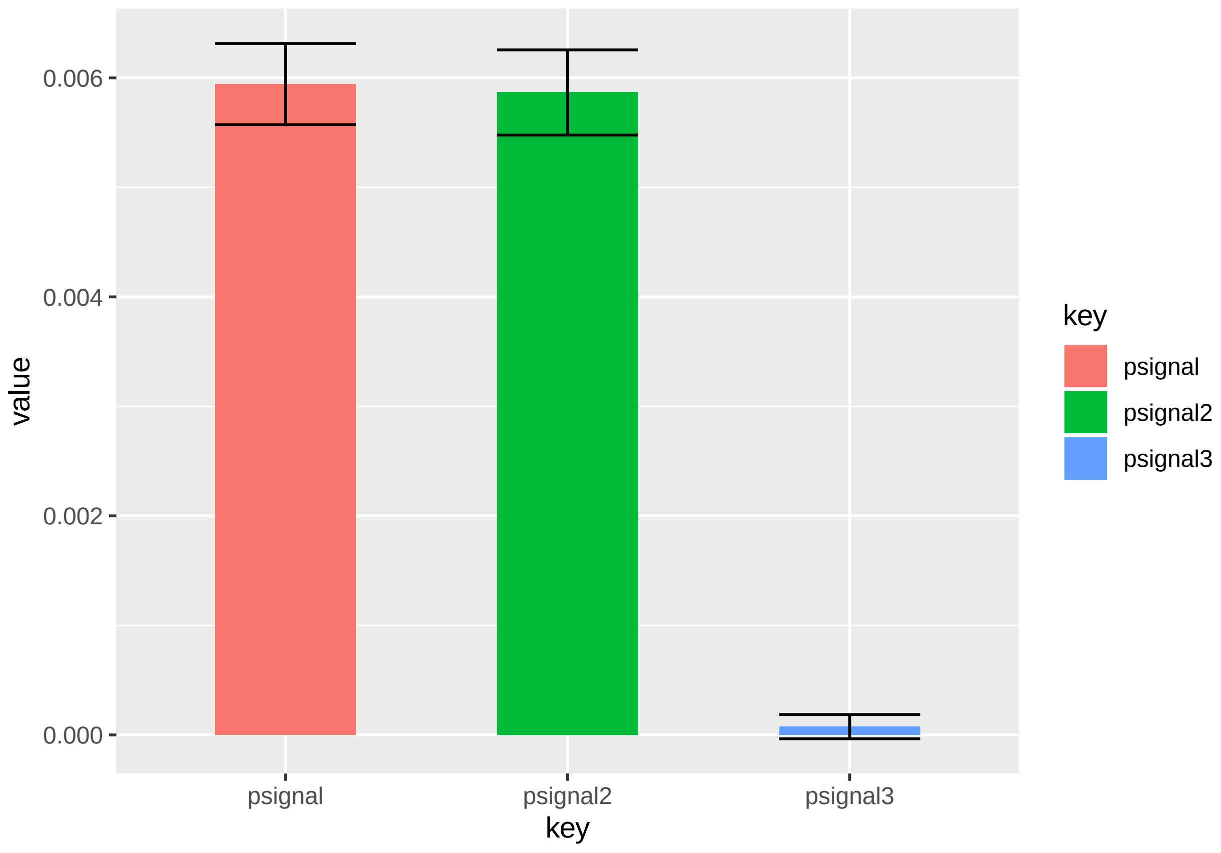
8. Make a bar diagram like in 6.c, but display effects as percent signal change (hint: percent signal change is slope divided by intercept

First three new vectors are made with the three different percent signal changes. Afterwards a dataframe is made with the 3 vectors. This dataframe is turned from a wide format to a long format. Then we plot this data in a bar plot using ggplot and adding errorbars:

```
psignal <- coefficients[2,]/coefficients[1,]
psignal2 <- coefficients[3,]/coefficients[1,]
psignal3 <- (coefficients[2,] - coefficients[3,])/coefficients[1,]
pchange <- data.frame(psignal,psignal2,psignal3)

pchange <- gather(pchange)

ggplot(pchange, aes(key, value, fill = key)) + geom_bar( stat = 'summary', fun.y = mean, width = 0.5) +
geom_errorbar(stat = 'summary', fun.data = mean_se, width = 0.5)
```


0.006
0.004
0.002
0.000
value
psignal
psignal2
psignal3
key
key
psignal
psignal2
psignal3