ExpMeth2\_PF2

Martine Lind Jensen

10/2/2020

# Tasks

## 1. Linear regression

Participant 372 from the sleepstudy has the following data:

Reaction372<-c(269.41, 273.47, 297.60, 310.63, 287.17, 329.61, 334.48, 343.22, 369.14, 364.12)  
Days372<-c(0,1,2,3,4,5,6,7,8,9)

### 1.a:

Make a constant vector of the same length as the data, consisting of ones.

const <- rep(1,10)

### 1.b:

Report the inner product (aka dot product) of the days vector and the constant vector.

sum(Days372\*const)

## [1] 45

### 1.c:

What does the dot product say about the possibility of finding an optimal linear regression?

*When the dot product is 0 the two vectors are orthogonal and then there is no linear correlation. 45 tells us that there can be fitted a linear regression. It is not negative and that means that they are not anti-correlated.*

### 1.d:

Create a 10x2 matrix called X with the days vector and constant vector as columns and use the least squares method manually to find the optimal coefficients (i.e. slope and intercept) to reaction time.

#Getting the intercept from the matrix of the days and the constant   
X <- matrix(c(Days372,const),ncol=2)  
  
#Getting the fit, the slope and intercept. This is the formula for the least squares   
model <- solve(t(X)%\*%X)%\*%t(X)%\*%Reaction372  
  
model

## [,1]  
## [1,] 11.298  
## [2,] 267.044

### 1.e:

Check result using lm(). Use the formula lm(Reaction372~0+X) - the zero removes the default constant.

#Checking linear model  
lm(Reaction372 ~ 0 + X)

##   
## Call:  
## lm(formula = Reaction372 ~ 0 + X)  
##   
## Coefficients:  
## X1 X2   
## 11.3 267.0

### 1.f:

Subtract the mean of Days372 from the Days372 vector. Replace the days vector with the new vector in X and redo the linnear regression. Did the coefficients change? (we will return to why this happened in a later class, but if you are curious, you can check this website out: <https://www.theanalysisfactor.com/center-on-the-mean/>)

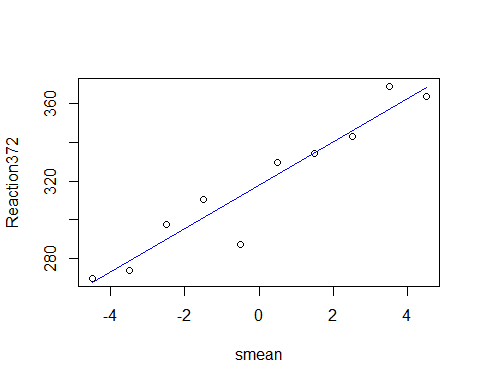
meand372 <- mean(Days372)  
  
smean <- Days372 - meand372  
  
X2 <- matrix(c(smean,const),ncol=2)  
  
model2 <- solve(t(X2)%\*%X2)%\*%t(X2)%\*%Reaction372  
  
model2

## [,1]  
## [1,] 11.298  
## [2,] 317.885

### 1.g:

Make a scatter plot with the mean-centered days covariate against response time and add the best fitted line.

#Make a scatter plot  
plot(smean,Reaction372,type='p')  
  
#Adding a line to the plot Days on X-axis, intercept plus slope\*days on y-axis  
lines(c(smean[1],smean[10]),c(model2[2]+model2[1]\*smean[1],model2[2]+model2[1]\*smean[10]),col='blue')



## 2. Images and matrices

Load the data using something like:

pacman::p\_load(jpeg)  
  
#Load the picture into a matrix  
matrix <- readJPEG("portfolio\_assignment2\_matrices\_data.jpg", native = FALSE)

### 2.a:

report how many rows and how many columns the matrix has. What are the maximun, minimum and mean pixel values?

#Number of rows  
nrow(matrix)

## [1] 900

#Number of columns  
ncol(matrix)

## [1] 606

#Maximum  
max(matrix)

## [1] 1

#Minimum  
min(matrix)

## [1] 0.0627451

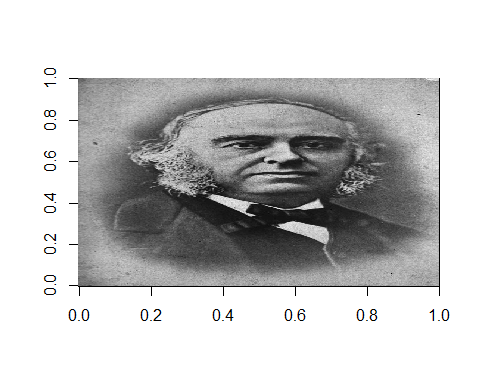
#Mean  
mean(matrix)

## [1] 0.5118474

### 2.b:

Make an image of the loaded matrix. Be sure to rotate the image into the correct orientation. The functions needed are found the in lecture slides. Furthermore, grey scale the picture with gray(1:100/100) - this will color values near 0 black/dark and values near 1 white/light.

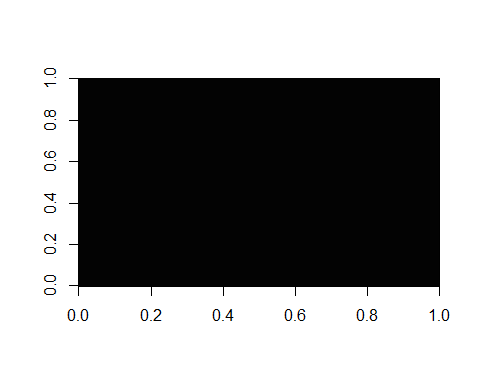
rotate <- function(x) t(apply(x,2,rev))  
  
pic <- matrix  
  
pic2 <- rotate(pic)  
  
image(pic2, col = gray(1:100/100))



### 2.c:

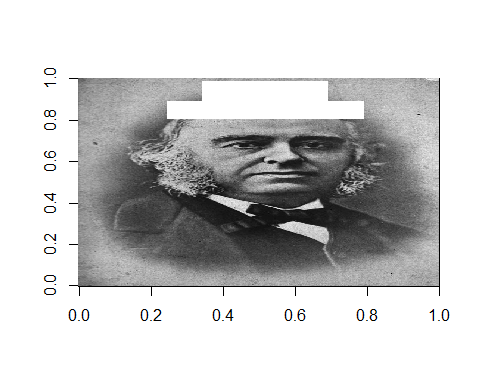
Draw an image with the same dimensions as that from 2.b. But this image should be completely black (hint: use zeros).

matrix\_pic\_black <- matrix(0, nrow = 900, ncol = 606)  
  
image(matrix\_pic\_black, col = gray(1:0/100))



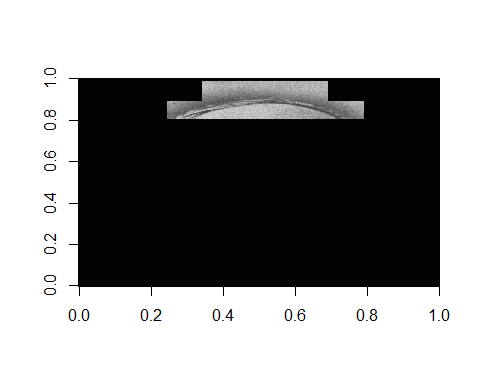
###2.d: Draw a white hat on the image from 2.b (hint: use ones).

hat <- pic2  
hat[150:480, 725:800] <- 1  
hat[210:420, 750:890] <- 1  
  
image(hat, col = gray(1:100/100))



###2.e: Make an image which has the same dimensions as 2.b., and which only contains the parts which was hidden behind the hat in 2.d. The rest should be black.

#Making a new matrix, where i rotate the matrix first so it fits with the rotated picture from hat   
behind\_the\_hat <- rotate(matrix(0,0, nrow = 900, ncol = 606))  
  
#Taking the specific numbers from the matrix and creating them from the Broca picture   
behind\_the\_hat[150:480, 725:800] <- pic2[150:480, 725:800]  
behind\_the\_hat[210:420, 750:890] <- pic2[210:420, 750:890]  
  
image(behind\_the\_hat, col = gray(1:100/100))



## 3. Brains and matrices

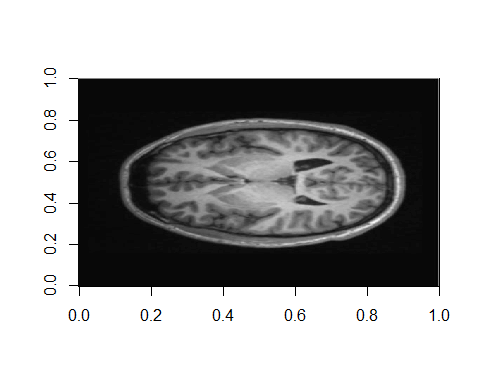
Load the brain data using something like:

library(jpeg)  
  
brain <- readJPEG("portfolio\_assignment2\_matrices\_data2.jpg", native = FALSE)

### 3.a:

Make an image of the brain.

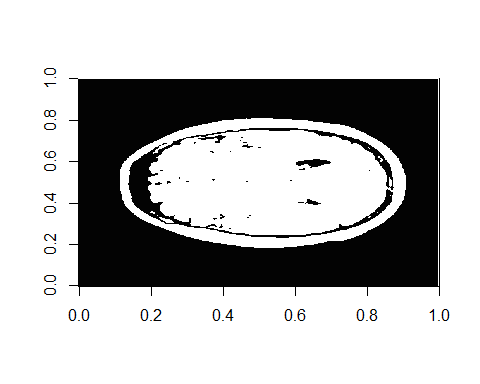
brainpic <- image(brain, col = gray(1:100/100))



### 3.b:

We will attempt to find the interesting areas of this brain image, e.g. only areas with gray matter. To do this we will create two masks, one that filters all darker areas away, and one that filters the white matter away. The masks will work by having zeros at the areas we want to fliter away, and ones at the interesting areas. Thus, the mask will have the intended effect if we do element-wise multiplication of it with the brain matrix. Start by making an image which is white (have ones) where the pixel values of the brain image are larger than the mean value of the whole image. Let the image be black (have zeros) everywhere else. Call this matrix mask1.

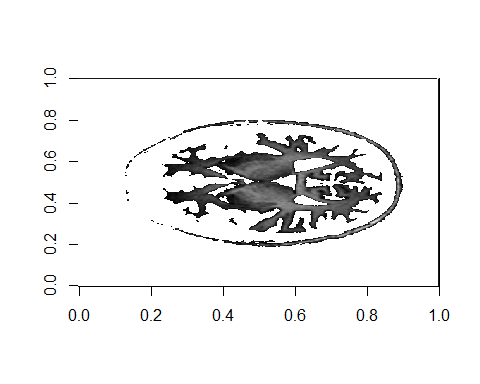
brain\_mean <- mean(brain)  
  
mask1 <- brain  
  
mask1[mask1 > brain\_mean] <- 1  
mask1[mask1 < brain\_mean] <- 0  
  
image(mask1, col = gray(1:100/100))



### 3.c:

Make an image which is white where the pixel values of the brain image are smaller than 2.5 times the mean value of the whole image. Call this matrix mask2

brain\_mean2 <- 2.5\*brain\_mean  
   
mask2 <- brain  
  
mask2[mask2 < brain\_mean2] <- 1  
  
image(mask2, col = gray(1:100/100))

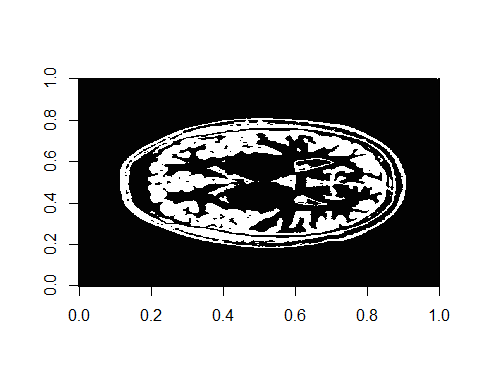


### 3.d:

Convert mask1 and mask2 into one mask with ones where the two masks overlap and zeros everywhere else. What type mathematical procedure can be used to produce this?

*We just use logical statements*

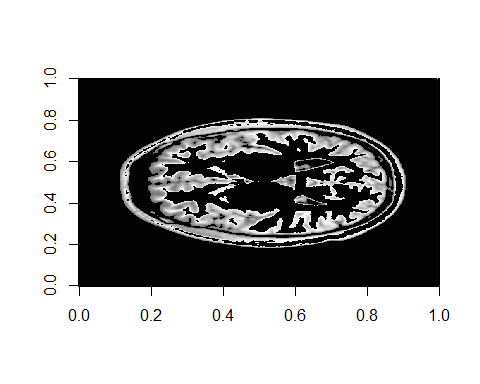
mask3 <- brain  
  
mask3[mask1 == mask2] <- 1  
mask3[mask1 != mask2] <- 0   
  
image(mask3, col = gray(1:100/100))



### 3.e:

Use the combined mask on the brain image to give you an image with only the image values where the mask has ones, and zeros everywhere else. Did we successfully limit our image to only contain gray matter?

f\_mask <- brain\*mask3  
  
image(f\_mask, col = gray(1:100/100))

 *Yes, i believe we made a pretty picture*

### 3.f:

Count the number of pixels in the combined mask.

sum(mask3)

## [1] 50004

## 4. Two equations with two unknowns

Two linear equations with two unknowns can be solved using matrix inversion. For example, see here: <https://www.mathsisfun.com/algebra/matrix-inverse.html>

### 4.a:

In the friday bar, men were three times as likely as women to buy beer. A total of 116 beers were sold. Women were twice as likely as men to buy wine. 92 glasses of wine were sold. How many men and women attended the Friday bar?

sold <- matrix(c(116,92), ncol=2)  
  
beer\_wine <- matrix(c(1,3,2,1), nrow = 2, ncol = 2) #top is beer, wine, buttom is women, men  
#x=BA^-1   
  
inv\_bw <- solve(beer\_wine)  
  
wom\_men <- sold%\*%inv\_bw  
  
wom\_men

[,1] [,2]

[1,] 32 28

*There were 32 women and 28 men*