Assignment 2

Teacher Note:
3.e Visualize the mask, not the brain+mask

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

9 feb 2018

1. Linear regression

Participant 372 from the sleepstudy has the following data:

```
#Participant 372
Reaction372<-c(269.41, 273.47, 297.60, 310.63, 287.17, 329.61, 334.48, 343.22, 369
.14, 364)
Days372<-c(0,1,2,3,4,5,6,7,8,9)
```

1.a

```
#1.a Make a constant vector of the same length as the data, consisting of ones.
a1 <- rep(1,10)
```

1.b

```
#1.b Report the inner product (aka dot product) of the days vector and the constan
t vector.

#calculate inner product
inner_product <- sum(Days372*a1)

#print inner product
inner_product
```

```
## [1] 45
```

1.b Response:

The inner product of the days vector and the constant vector is 45

1.c

```
#1.c What does the dot product say about the possibility of finding an optimal lin
ear regression?
```

1.c Response:

If dot product is 0 we have optimal conditions for making a linear regression, because they are completely independent of each other. Our inner product is 45, which is not optimal for linear regression.

1.d

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

X <- matrix(c(Days372, a1), ncol=2)

#print matrix
X

##       [,1] [,2]
##  [1,]    0    1
##  [2,]    1    1
##  [3,]    2    1
##  [4,]    3    1
##  [5,]    4    1
##  [6,]    5    1
##  [7,]    6    1
##  [8,]    7    1
##  [9,]    8    1
## [10,]    9    1

#Calculating the optimal coefficients to reaction time with the least square metho
d
beta<-solve(t(X)%*%X)%*%t(X)%*%Reaction372

#print beta
beta

##           [,1]
## [1,]  11.29145
## [2,] 267.06145
```

1.d Response:

The optimal coefficients (slope and intercept) to reaction time is a slope of 11.29 and intercept of 267.06.

1.e

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

#Make the linear model
model <- lm(Reaction372~0+X)

#Summarise model
summary(model)

##
## Call:
## lm(formula = Reaction372 ~ 0 + X)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -25.057  -4.234   1.009   7.489  11.747
##
## Coefficients:
##    Estimate Std. Error t value Pr(>|t|)
## X1   11.291      1.243   9.083 1.73e-05 ***
## X2  267.061      6.636  40.241 1.60e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 11.29 on 8 degrees of freedom
## Multiple R-squared:  0.999,  Adjusted R-squared:  0.9988
## F-statistic:  4004 on 2 and 8 DF,  p-value: 9.921e-13
```

1.e Response:

We can see that the slope and intercept is the same as calculated earlier.

1.f

```
#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?

#calculate mean days
```

```r
mean_days <- mean(Days372)

#print the mean days
mean_days
```

```
## [1] 4.5
```

```r
#substracting the mean of days from the vector
day_vector <- Days372-mean_days

#print the results
day_vector
```

```
##  [1] -4.5 -3.5 -2.5 -1.5 -0.5  0.5  1.5  2.5  3.5  4.5
```

```r
#make a new vector with the day_vector and the constant vector
new_vector <- matrix(c(day_vector, a1), ncol=2)

#Replace days vector with new vector and do linear regression
model_2 <- lm(Reaction372~0+new_vector)

#summarise model
summary(model_2)
```

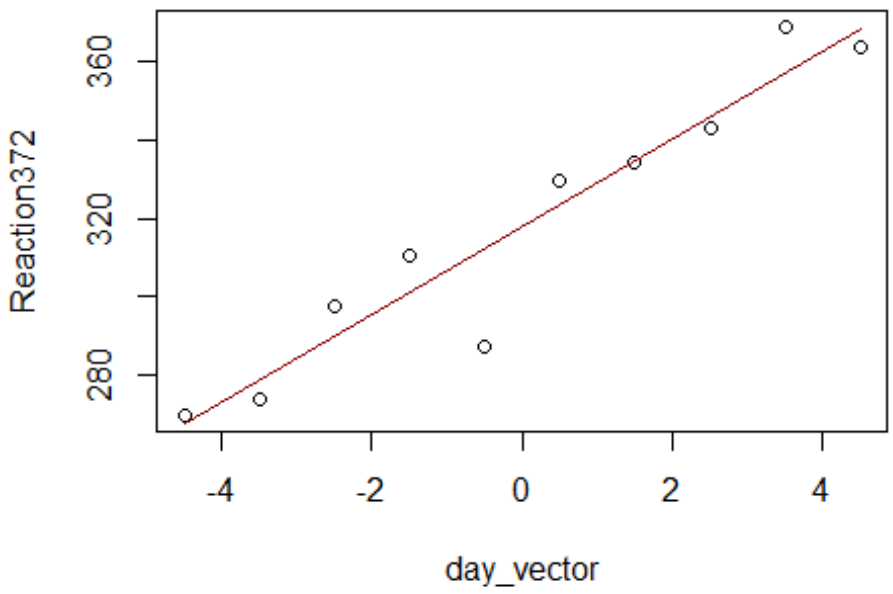
```
##
## Call:
## lm(formula = Reaction372 ~ 0 + new_vector)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -25.057  -4.234   1.009   7.489  11.747
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## new_vector1   11.291      1.243   9.083 1.73e-05 ***
## new_vector2  317.873      3.571  89.025 2.83e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 11.29 on 8 degrees of freedom
## Multiple R-squared:  0.999,  Adjusted R-squared:  0.9988
## F-statistic:  4004 on 2 and 8 DF,  p-value: 9.921e-13
```

1.f Response:

Yes, the intercept change. The intercept is now the middle day of the experiment, so on the middle day their reaction time is 317.87.

1.g

```
#1.g Make a scatter plot with the mean-centered days covariate against response ti
me and add the best fitted line.

#Make a plot with the mean days and reaction time
plot(day_vector,Reaction372,type='p')

#calculate the beta with the least squares fit - to get the best fit
beta2<-solve(t(new_vector)%*%new_vector)%*%t(new_vector)%*%Reaction372

#make a linear line to the plot
lines(c(day_vector[1],day_vector[10]),c(beta2[2]+beta2[1]*day_vector[1],beta2[2]+b
eta2[1]*day_vector[10]),col='darkred')
```



2. Images and matrices

```
#Load the data
man_matrix<-readJPEG('portfolio_assignment2_matrices_data.jpg', native = FALSE)
```

2.a

```
#2.a report how many rows and how many columns the matrix has. What are the maximu
n, minimum and mean pixel values?
```

```r
#find how many rows
nrow(man_matrix)
```

```
## [1] 900
```

```r
#find how many columns
ncol(man_matrix)
```

```
## [1] 606
```

```r
#maximum pixel
max(man_matrix)
```

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

```r
#minimum pixel
min(man_matrix)
```

```
## [1] 0.0627451
```

```r
#mean pixel
mean(man_matrix)
```

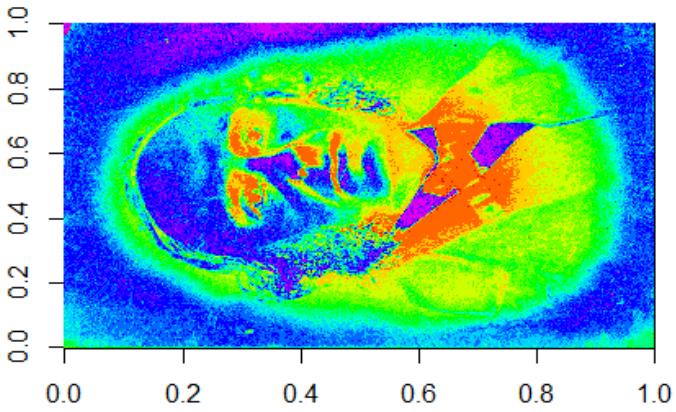
```
## [1] 0.5118474
```

2.a Response:

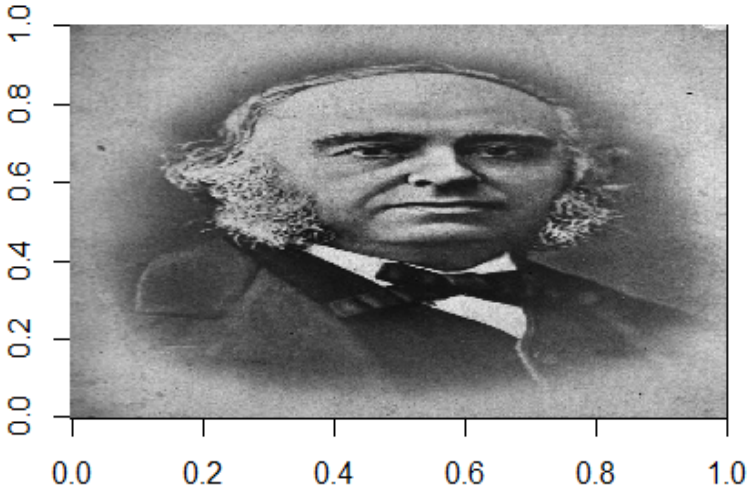
The matrix has 900 rows and 606 columns. The maximum pixel is 1 and the minimum is 0.06. The mean pixel is 0.51.

2.b

```r
#2.b Make an image of the loaded matrix. Be sure to rotate the image into the corr
ect orientation.

#Make an image of the man matrix
image(man_matrix,col=rainbow(15))
```

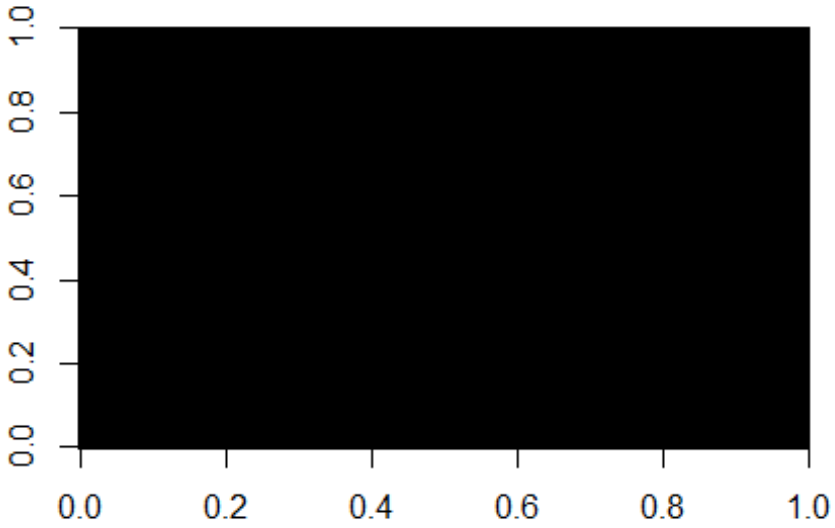



```
#Rotate the picture using t() and rev() to rotate the matrix for display
rotate <- function(x) t(apply(x, 2, rev))
man_matrix2 <-rotate(man_matrix)
#convert to an image
man_image <- image(man_matrix2,col=gray(1:100/100))
```



2.c

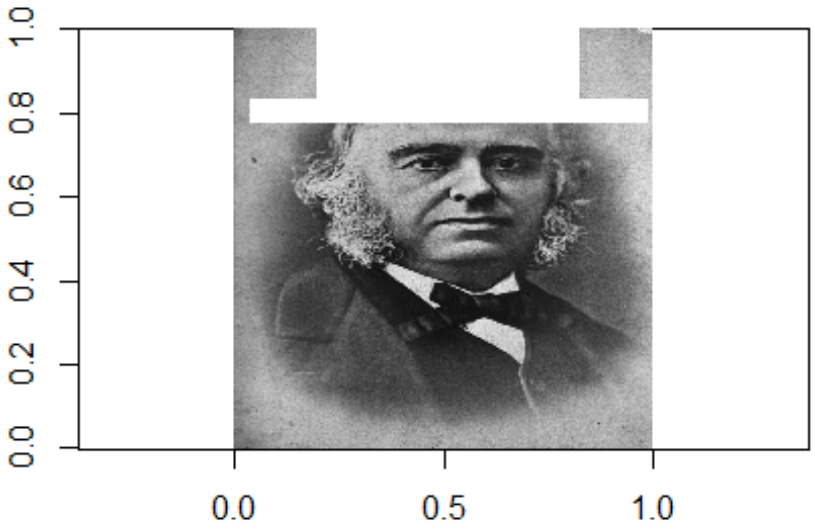
```
#2.c Draw an image with the same dimensions as that from 2.b. But this image shoul
d be completely black (hint: use zeros).
#Make a new matrix with all zeros
black_image <- matrix(0, nrow = 900, ncol = 606)
black_image <- rotate(black_image)

#Make it into an image that is all black
image(black_image, col= gray(0:1))
```



2.d

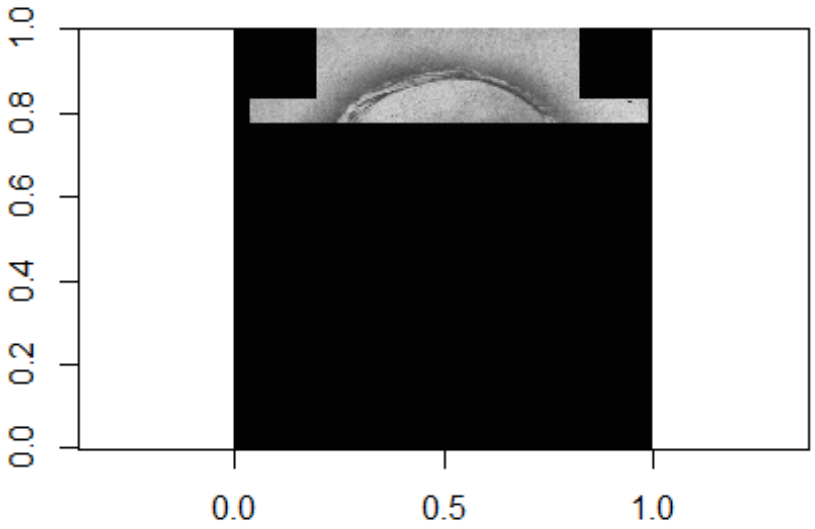
```
#2.d Draw a white hat on the image from 2.b (hint: use ones).

#Put the picture into a new data frame
make_hat <- man_matrix2

#Make the long part of the hat by making these intervals ones
make_hat[25:600, 700:750] <- 1

#Make the top part of the hat by making these intervals ones
make_hat[120:500, 750:900] <- 1
```

```r
#Draw new image with the changed intervals
image(make_hat, col=gray(1:100/100), asp = 1)
```



2.e

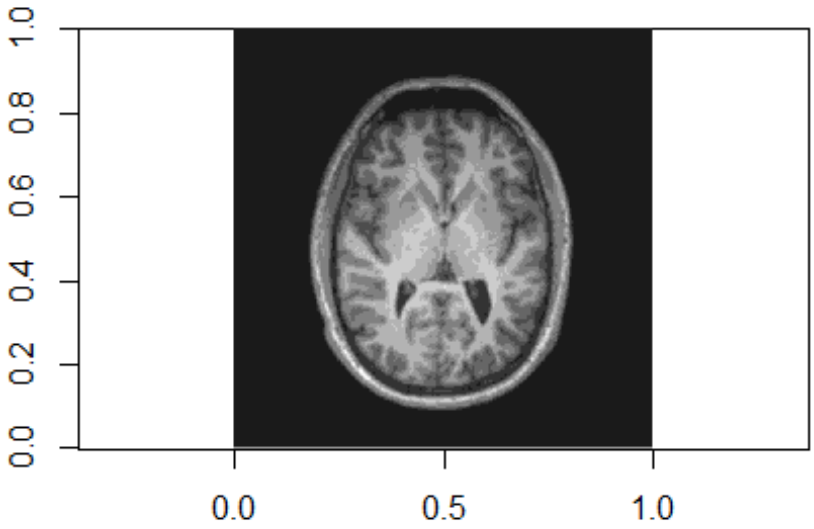
```r
#2.e Make an image which has the same dimensions as in 2.b., and which only contai
ns the parts which was hidden behind the hat in 2.d. The rest should be black.

#We take the black image from before and make it a new matrix

only_hat <- black_image

#Make the hat again
only_hat[25:600, 700:750] <- 1
only_hat[120:500, 750:900] <- 1

#Multiplying the two matrices only hat and the original photo
pic_hat <- only_hat*man_matrix2

#Make image
image(pic_hat, col=gray(1:100/100), asp = 1)
```



3. Brains and matrices Load the brain data using something like

3.a: Make an image of the brain.

```
##Load data
brain<-readJPEG('portfolio_assignment2_matrices_data2.jpg', native = FALSE)
brain<- rotate(brain)

#Make an image of the brain
image(brain, col = grey(1:10/10), asp = 1)
```



3.b:

3.b: We will attempt to ???nd the interesting areas of this brain image, e.g. only areas with gray matter.

To do this we will create two masks, one that ???lters all darker areas away, and one that ???lters the white matter away.

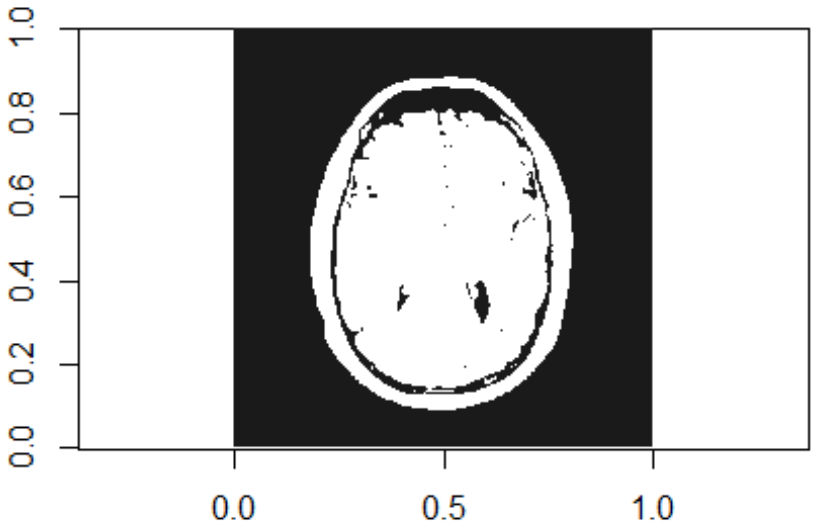
The masks will work by having zeros at the areas we want to ???iter away, and ones at the interesting areas.

Thus, the mask will have the intended e???ect if we do element-wise mutiplication 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.

```
#Creating a matrix with the values from the matrix "brain"
mask1 <- brain

#Making a matrix where all values larger than the mean is set to a value of 1
mask1[mask1 > mean(mask1)] <- 1
```

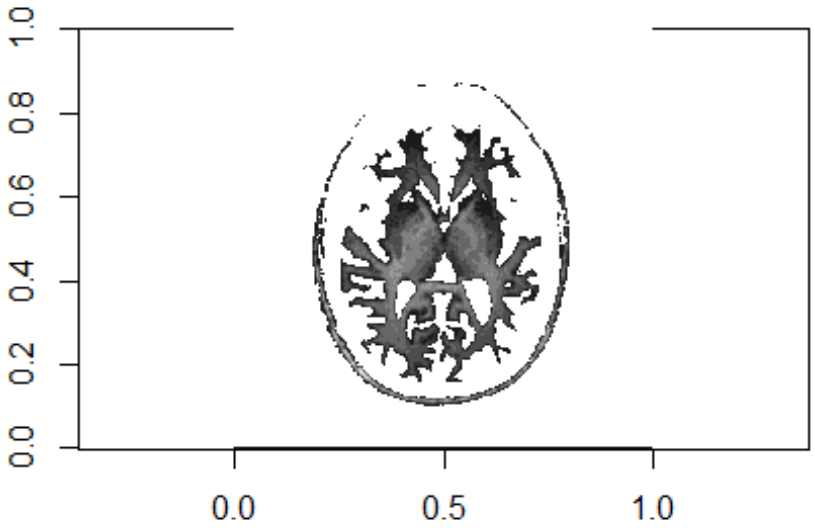
```r
#Changing all values less than 1 to 0
mask1[mask1 < 1] <- 0

#Make image
image(mask1, col = grey(1:10/10), asp = 1)
```



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.

```r
#Creating a duplicate of brain called mask2
mask2 <- brain

#Changing all values less than 2.5 times the mean value to 1 (making them white)
mask2[mask2 < mean(brain)*2.5] <- 1

#Make image
image(mask2, col = grey(1:10/10), asp = 1)
```



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?

```
#Combining the two matrices into one logical matrix in which overlap is equal to t
rue and different values equals to false
combined_mask<-mask1==mask2

#Chaning the logical matrix into an numeric. When encounting artimetic "true" and
"false" cease to exist, therefore multipying with one (or adding 0) will do the jo
b
combined_mask <- combined_mask*1
```

#Mathematical procedure

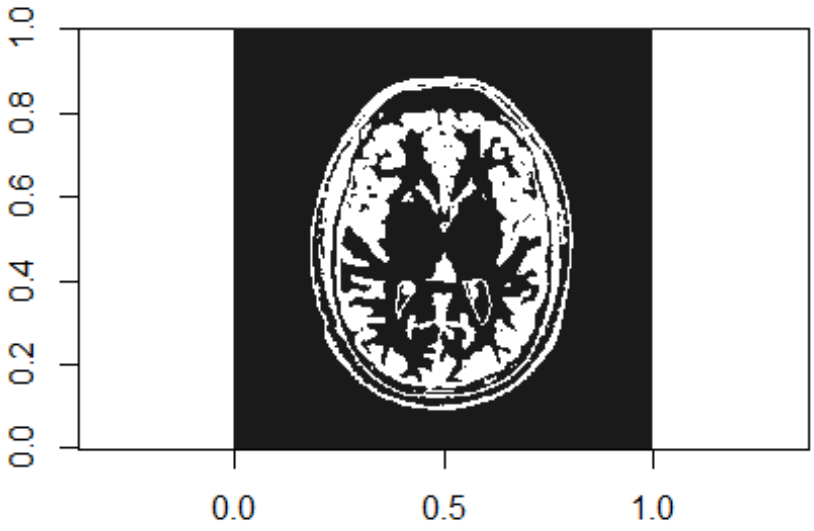
Basically, we are multiplying the two masks. 1*1 equals 1 and 0*0 equals 0.

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?

```
image(combined_mask, col = grey(1:10/10), asp = 1)
```



Response 3.d

Yes we did manage to filter all the white matter away and maintain only the grey matter. However, we also still see the bone structures.

3.e:

Count the number of pixels in the combined mask.

```
#Calculating the number of pixels in the combined matrix using sum
sum(combined_mask)

## [1] 50004
```

3.e Response

The picture contains 50004 pixels.

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 2

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?

```
#We create a matrix with the values. We set men to x and women to y. The informati
on can be written as:
#3x+y = 116
#x+2y= 92

#Make matrix with values
friday_bar <- matrix(c(3,1,1,2), nrow=2)
friday_bar
```

```
##      [,1] [,2]
## [1,]    3    1
## [2,]    1    2
```

```
b <- matrix(c(116,92), nrow=2)
b
```

```
##      [,1]
## [1,]  116
## [2,]   92
```

```
#Inversing the matrix and multiplying with b to solve the equation
solve(friday_bar,b)
```

```
##      [,1]
## [1,]   28
## [2,]   32
```

Response 4.a

The friday bar was attended by 28 males and 32 woman.