# Lab 8 - linear algebra

**Using linear algebra to solve a system of linear equations**

I assume you understood the basics of what we covered in lecture. Now let's walk through some examples in the textbook and expand on them. We will discuss these points when walking through the lab demo, so be sure to take notes.

```
data(father.son, package="UsingR")
x <- father.son$fheight
y <- father.son$sheight
X <- cbind(1,x)
head(X) # see what this is, design matrix of 1 for beta not and x values with beta 1
```

```
##               x
## [1,] 1 65.04851
## [2,] 1 63.25094
## [3,] 1 64.95532
## [4,] 1 65.75250
## [5,] 1 61.13723
## [6,] 1 63.02254
```

```
colnames(X) <- c("B0","B1") # give the columns names that reference for the beta matrix
head(X)
```

```
##      B0       B1
## [1,]  1 65.04851
## [2,]  1 63.25094
## [3,]  1 64.95532
## [4,]  1 65.75250
## [5,]  1 61.13723
## [6,]  1 63.02254
```

```
# crossprod takes the t(x)*x when only one matrix provided. t(x)*y when provided.
crossprod(rep(1,3),1:3) # 1+2+3=6
```

```
##      [,1]
## [1,]    6
```

```
crossprod(matrix(1:6, ncol=3, byrow=TRUE),1:2) # 1+8+2+10+3+12=36
```

```
##      [,1]
## [1,]    9
## [2,]   12
## [3,]   15
```

```
            # 1+8=9
            # 2+10 = 12
            # 3+12 = 15


betahat <- solve(crossprod(X))%*%crossprod(X,y) # %*% is dot product. same as
betahat
```

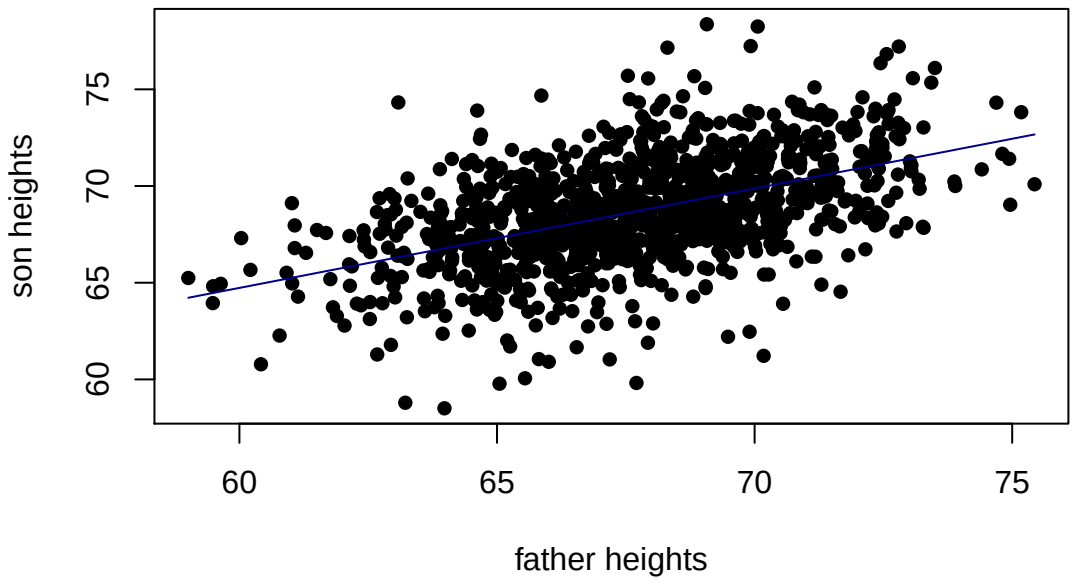
```
##         [,1]
## B0 33.886604
## B1  0.514093
```

```
lm(y~x)$coefficients
```

```
## (Intercept)           x
##   33.886604    0.514093
```

Make sure you understand what just happened before proceeding (refer to the lecture as well). Now we'll plot the data and the best fit model (i.e. the regression line).

```
plot(x,y,pch=16,main="",xlab="father heights",ylab="son heights")
newx <- seq(min(x),max(x),len=150) #put 150 points between, connecting so same as 3
X <- cbind(1,newx) # make sure you know why we do this....
fit.vals <- X%*%betahat # a design matrix of 1 for the B0 and some x values for B1 values
lines(newx,fit.vals,col="darkblue")
```




The "big result" that the book discusses: $\hat{\beta} = (X^T X)^{-1}(X^T Y)$ is really just another way to write the covariance between X and Y over the variance in X. Think about this - the Covariance in X and Y is the rise, and the variance in X is the run. Rise over run is slope...

This little trick also works for curves, so long as the function for the curve is linear in the paramters (coefficients), such as is the case for the Pisa falling object example.

```
set.seed(1)
g <- 9.8 # graviational acceleration constant
n <- 25 # numebr of observations
tim <- seq(0,3.4,len=n) # time series, x values
y <- 56.67 - 0.5*g*tim^2 + rnorm(n) # y values
X <- cbind(1,tim,tim^2) # model matrix, allow for curve with the squared unit
colnames(X) <- c("B0","B1","B2") # need parameter estimate for change over time, and speeding up change
head(X,20) # check it out
```

```
##       B0        B1         B2
##  [1,]  1 0.0000000 0.00000000
```

```
##  [2,]  1 0.1416667 0.02006944
##  [3,]  1 0.2833333 0.08027778
##  [4,]  1 0.4250000 0.18062500
##  [5,]  1 0.5666667 0.32111111
##  [6,]  1 0.7083333 0.50173611
##  [7,]  1 0.8500000 0.72250000
##  [8,]  1 0.9916667 0.98340278
##  [9,]  1 1.1333333 1.28444444
## [10,]  1 1.2750000 1.62562500
## [11,]  1 1.4166667 2.00694444
## [12,]  1 1.5583333 2.42840278
## [13,]  1 1.7000000 2.89000000
## [14,]  1 1.8416667 3.39173611
## [15,]  1 1.9833333 3.93361111
## [16,]  1 2.1250000 4.51562500
## [17,]  1 2.2666667 5.13777778
## [18,]  1 2.4083333 5.80006944
## [19,]  1 2.5500000 6.50250000
## [20,]  1 2.6916667 7.24506944
```

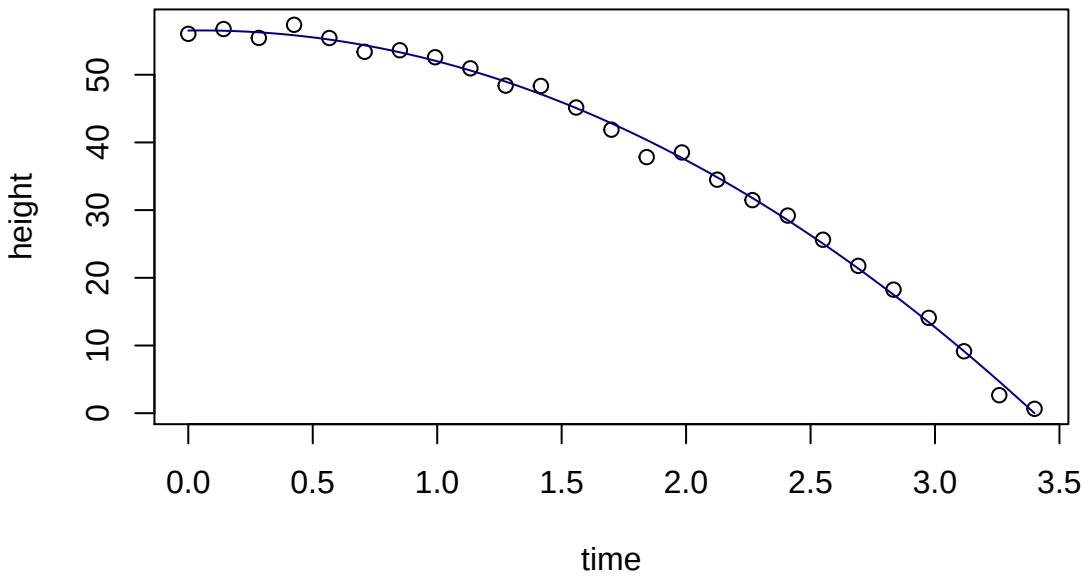
```
betahat <- solve(crossprod(X))%*%crossprod(X,y) #variance or sum of squares??
betahat # check the estimates
```

```
##          [,1]
## B0 56.5317368
## B1  0.5013565
## B2 -5.0386455
```

```
lm(y~tim+I(tim^2))$coefficients # compare "by hand" estimates (note use of Identity matrix)..?? I just
```

```
## (Intercept)         tim    I(tim^2)
##  56.5317368   0.5013565  -5.0386455
```

```
# now plot regression line
newtime <- seq(min(tim),max(tim),len=150) # make new x values for prediction
X <- cbind(1,newtime,newtime^2) # make new model matrix
fit.vals <- X%*%betahat # make new y values using best fit model
plot(tim,y,xlab="time",ylab="height",main="")
lines(newtime,fit.vals,col="darkblue")
```



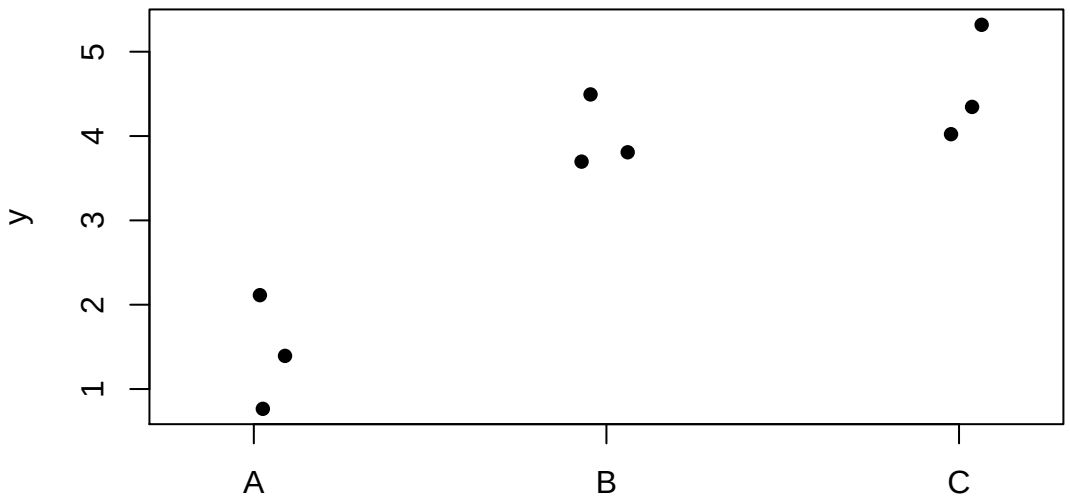

---

Fine. Now the hard part - transferring what we learned just now to the third example in the textbook. This specific problem is left for you to solve as part of the lab, but it's a tricky one so I'll walk through a mock example. To deal with nominal variables (cofactors) we need to understand "dummy" variables.

```
set.seed(23)
dat <- data.frame(cell.line = c(rep("A",3),rep("B",3),rep("C",3)),
                  y=c(rnorm(3,mean=1.2),rnorm(3,mean=2.7),rnorm(3,mean=4.3)))
dat # look at the data
```

```
##   cell.line         y
## 1         A 1.3932123
## 2         A 0.7653179
## 3         A 2.1132671
## 4         B 4.4933881
## 5         B 3.6966051
## 6         B 3.8074905
## 7         C 4.0219137
## 8         C 5.3192055
## 9         C 4.3454372
```

```
stripchart(y~cell.line,data=dat,pch=16,vertical=T,method="jitter") # plot the data
```




OK, so you get the general structure. Now let's get the y and X matrices. X will be odd, so pay attention.

```
y <- dat$y
y # look at the vector of responses, 9 long
```

```
## [1] 1.3932123 0.7653179 2.1132671 4.4933881 3.6966051 3.8074905 4.0219137
## [8] 5.3192055 4.3454372
```

```
X <- matrix(c(rep(1,9),rep(0,3),rep(1,3),rep(0,9),rep(1,3)),9,3)
colnames(X) <- c("A","B","C")
X # look at the X matrix # B0 the first column, mean to start, then B is yes or no in group 2, and C is
```

```
##       A B C
##  [1,] 1 0 0
##  [2,] 1 0 0
##  [3,] 1 0 0
##  [4,] 1 1 0
##  [5,] 1 1 0
##  [6,] 1 1 0
##  [7,] 1 0 1
##  [8,] 1 0 1
##  [9,] 1 0 1
```

Now, what we have here is a set of three "dummy" variables, and it becomes clear why we need to do this if we revisit our linear model for nominal scaled cofactors. Here, we have $y_i = \beta_0 + \beta_1 * x_{1i} + \beta_2 * x_{2i}$. Notice that the X matrix tells us which x values are 0 or 1. Write this out to convince yourself you understand this notation. If you do, you should be able to explain the $\beta's$ in terms of averages. For example, here, $\beta_0$ is the average for cell line A, $\beta_1$ is the difference between the average for cell line A and cell line B, and $\beta_2$ is the difference between cell line A and cell line C. Rename the columns if that helps:

```
colnames(X) <- c("B0","B1","B2")
```

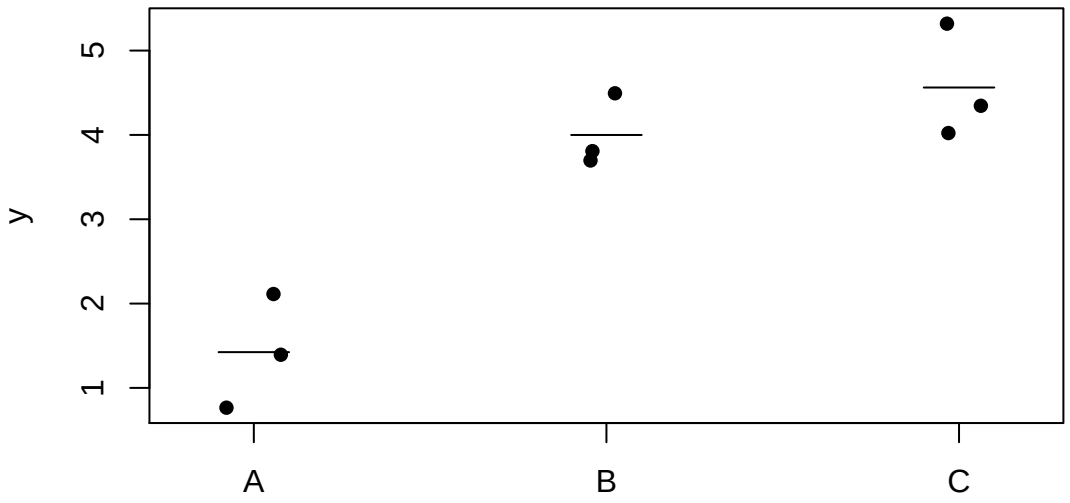
Good? Now let's get the $\hat{\beta}'s$, the estimates and check against lm() results...

```
betahat <- solve(crossprod(X))%*%crossprod(X,y)
betahat
```

```
##        [,1]
## B0 1.423932
## B1 2.575229
## B2 3.138253
```

OK. Stop. Now, write some code to solve for the means for cell line A, cell line B, and cell line C. We'll compute them and then add them to the plot as a "gut check".

```
A.mean <- betahat[1]
B.mean <- sum(betahat[1:2])
C.mean <- betahat[1]+betahat[3]
stripchart(y~cell.line,data=dat,pch=16,vertical=T,method="jitter") # plot the data
segments(0.9,A.mean,1.1,A.mean)
segments(1.9,B.mean,2.1,B.mean)
segments(2.9,C.mean,3.1,C.mean)
```




Looks good. Now compare to lm() results. Notice that this is an ANOVA....

```
betahat # remind ourselves of hand computed result
```

```
##        [,1]
## B0 1.423932
## B1 2.575229
## B2 3.138253
```

```
lm(y~cell.line,data=dat)$coefficients
```

```
## (Intercept)  cell.lineB  cell.lineC
##    1.423932    2.575229    3.138253
```

MAGIC!!

Here's a little trick you can use with lm() if you would rather just see the means, and not the contrasts. You can check these results against the hand-computed A.mean, B.mean and C.mean above...

```
lm(y~cell.line -1, data=dat)$coefficients #force intercept to 0
```

```
## cell.lineA cell.lineB cell.lineC
##   1.423932   3.999161   4.562185
```

```
print(c(A.mean,B.mean,C.mean))
```

```
## [1] 1.423932 3.999161 4.562185
```

---

OK. Now on to the assignment. To start, let's get some easy points.

---

Consider the matrix A: matrix(c(2,1,3,2),2,2)
byrow = false is the default:
2 3 1 2

Q1. Write the transpose of A.
2 1 3 2

Q2. Which of the following is the inverse of A? Use row operations and/or R to prove your answer is correct.

2 -3
-1 2 / 4-3

transpose would be -3 -2 –> Inverse would be: –>

a. matrix(c(-2,-1,-3,-2),2,2)

b. matrix(c(2,-3,-1,-2),2,2)
c. matrix(c(2,-1,-3,2),2,2) - This is the inverse
d. matrix(c(-2,3,1,-2),2,2)

---

Consider the dataset d: data.frame(x=c(-0.2,-1.5,-0.5,0.4,1.4,0.4,-1.4),y=c(3.0,6.3,5.2,2.8,-0.3,2.8,5.5))

Q3. Write the system of linear equations to describe the linear regression of y on x in d. Hint: you will be using the numbers given above.

Q4. Write the same system using matrix notation. Hint: you will have a y-matrix, and x-matrix, and a beta-matrix.

---

Q5. Fetch the mice weight data. e.g. mice <- read.csv("femaleMiceWeights.csv"). Make the y vector from the bodyweights and build the X matrix for the two diets. Do this in R and show each matrix (y and X).

Q6. Use solve() to find and interpret the $\hat{\beta}'s$ and compare with results from lm()