Computational Statistics and Data Analysis
Sheet No. 10

David Bubeck, Patrick Nisblè

July 5, 2017

Problem 1: Higher-Order Fitting

For the first estimation of the linear fit coefficients by finding the slope a and intercept b to minimize the given formular we found:

$$a = 7.0$$
$$b = 820$$

Here is the printed summary of the lm function:

```
lm(formula = metabolic.rate ~ body.weight, data = data)
```

Output

```
Residuals:
    Min      1Q  Median      3Q     Max
-245.74 -113.99  -32.05  104.96  484.81

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 811.2267    76.9755  10.539 2.29e-13 ***
body.weight   7.0595     0.9776   7.221 7.03e-09 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 157.9 on 42 degrees of freedom
Multiple R-squared:  0.5539,^^IAdjusted R-squared:  0.5433
F-statistic: 52.15 on 1 and 42 DF,  p-value: 7.025e-09
```

To compare these to methods we can see in the plot that it's actually quite good.
Also the prediction for body weights of 150 and 200 kg we found with predict:

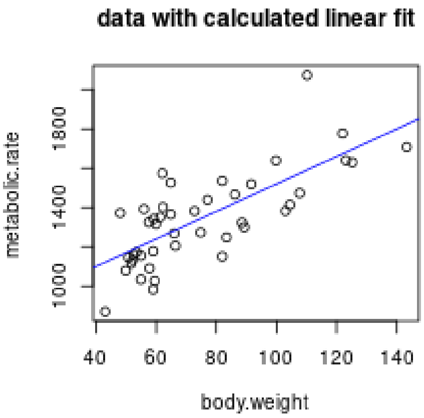
```
150 kg: 1870.156 with error estimation of 77.1995
200 kg: 2223.132 with error estimation of 124.6124
```

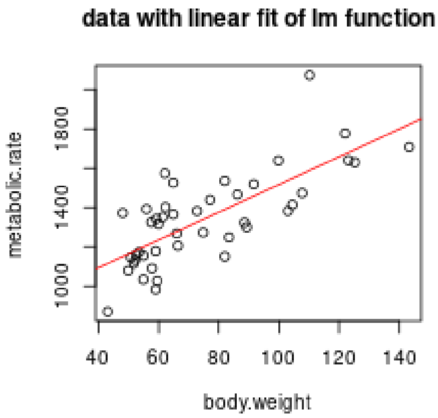
Code:

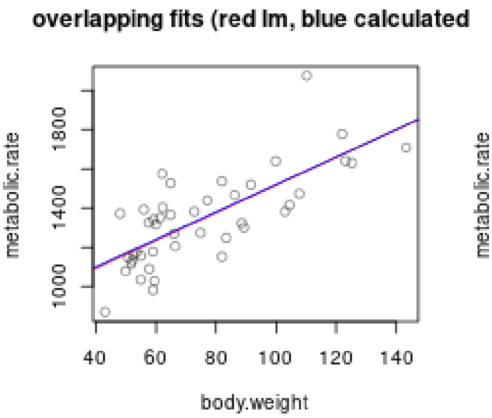
```r
data <- read.table("rmr_ISwR.dat", header = TRUE, sep = " ")
#print(data)

#linear fitting as in the lecture described
#define vectors for slope and intercept
a = c(0.6, 0.8, 1.0, 1.2, 1.4, 1.6, 1.8, 2.0, 2.2, 2.4, 2.6, 2.8,
    3.0, 3.2, 3.4, 3.6, 3.8, 4.0, 4.2, 4.4, 4.6, 4.8, 5.0, 5.2, 5.4, 5.6, 5.8,
    6.2, 6.4, 6.6, 6.8, 7.0, 7.2, 7.4, 7.6, 7.8, 8.0)
b = c(800, 810, 820, 830, 840, 850, 860, 870, 880, 890,
    900, 910, 920, 930, 940, 950, 960, 970, 980, 990,
    1000, 1010, 1020, 1030, 1040, 1050, 1060, 1070, 1080, 1090,
    1100, 1110, 1120, 1130, 1140, 1150, 1160, 1170, 1180, 1190, 1200)

lengthData = length(data$body.weight)
lengthA = length(a)
lengthB = length(b)
sum = 1000000000
temp = 0
x = data$body.weight
y = data$metabolic.rate
a_final = 0
b_final = 0

#calculate the sigma as given in the sheet
for(i in 1:lengthA) {
    for(j in 1:lengthB) {
        for(k in 1:lengthData) {
            y_fit = a[i] * data$body.weight[k] + b[j]
            temp = temp + (y[k] - y_fit)^2 / (lengthData - 2)
        }
        temp = sqrt(temp)
        if(temp < sum) {
            sum = temp
            a_final = a[i]
            b_final = b[j]
        }
    }
}

print(a_final)
print(b_final)

#linear fitting with lm function
fit_lin <-lm(metabolic.rate ~ body.weight, data = data)
summary(fit_lin)
#quadratic fitting with lm function
fit_quad <-lm(metabolic.rate ~ body.weight + I(body.weight^2), data = data)
#cubic fitting with lm function
fit_cubic <-lm(metabolic.rate ~ body.weight + I(body.weight^2) + I(body.weight^3), data =

#plot linear fits with and without lm function
png("exercise_10_1.png")
attach(mtcars)
par(mfrow = c(2, 2))
#data plot with calculated fit
plot(data, main = "data with calculated linear fit")
abline(b_final, a_final, col = "blue")
```

```r
#linear fit plot with lm function
plot(data, main = "data with linear fit of lm function")
abline(fit_lin, col = "red")
#overlapping linear fits
plot(data, main = "overlapping fits (red lm, blue calculated)",
        col = rgb(0, 0, 0, 0.5))
abline(fit_lin, col = "red")
abline(b_final, a_final, col = "blue")
#plot with confidence limit
plot(data, main = "confidence limit 95%")
confint(fit_lin, level = 0.95)


png("exercise_10_lm.png")
attach(mtcars)
par(mfrow = c(2, 2))
#data plot
plot(data, main = "data")
#linear fit plot with lm function
plot(data, main = "data with linear fit of lm function")
abline(fit_lin, col = "red")
#quadratic fit plot with lm function
plot(data, main = "data with quadratic fit of lm function")
lines(data$body.weight, fitted(fit_quad), col = "red")
#cubic fit plot with lm function
plot(data, main = "data with cubic fit of lm function")
lines(data$body.weight, fitted(fit_cubic), col = "red")


#prediction for 150 and 200 kg
new.data <- data.frame(body.weight = c(150, 200))
predict(fit_lin, new.data, se.fit = TRUE)
```

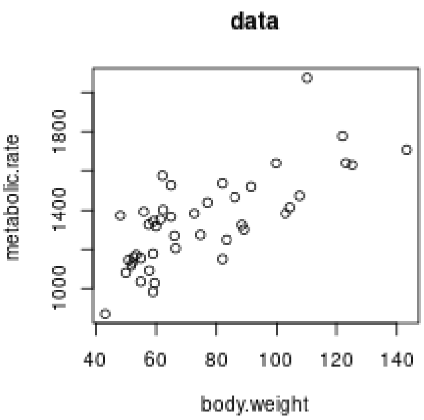
Plots:

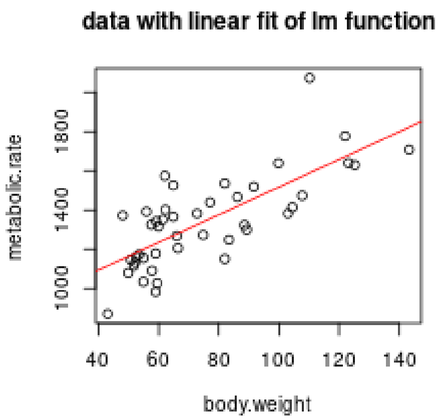

data with calculated linear fit
metabolic.rate
1000 1400 1800
40 60 80 100 120 140
body.weight

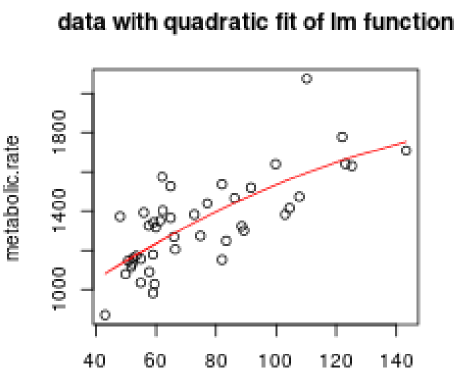

data with linear fit of lm function
metabolic.rate
1000 1400 1800
40 60 80 100 120 140
body.weight

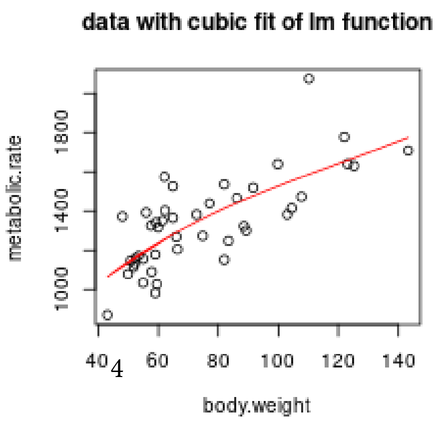

overlapping fits (red lm, blue calculated
metabolic.rate
1000 1400 1800
40 60 80 100 120 140
body.weight


confidence limit 95%
metabolic.rate
1000 1400 1800
40 60 80 100 120 140
body.weight


data
metabolic.rate
1000 1400 1800
40 60 80 100 120 140
body.weight


data with linear fit of lm function
metabolic.rate
1000 1400 1800
40 60 80 100 120 140
body.weight


data with quadratic fit of lm function
metabolic.rate
1000 1400 1800
40 60 80 100 120 140
body.weight


data with cubic fit of lm function
metabolic.rate
1000 1400 1800
40 60 80 100 120 140
body.weight

Problem 2: linear fitting with errors

Code:

```r
data <- read.table("rmr_ISwR_errors.dat", header = TRUE, sep = "\t")
print(data)

#define function for calculation of S_0, S_1, S_2, S_d, S_xd
S_n <- function(index, withd = FALSE) {
    lengthData = length(data$body.weight)
    sum = 0
    x = data$body.weight
    d = data$metabolic.rate
    sigma = data$errors
    if(withd == FALSE) {
        for(i in 1:lengthData) {
            sum = sum + x[i]^index / sigma[i]^2
        }
        #return(sum)
    } else {
        for(i in 1:lengthData) {
            sum = sum + (x[i]^index * d[i]) / sigma[i]^2
        }
    }

    return(sum)
}

#define matrices for calculation of set of linear equations
G <- matrix(c(S_n(0), S_n(1), S_n(1), S_n(2)), nrow = 2, ncol = 2, byrow = TRUE)
D <- c(S_n(0, TRUE), S_n(1, TRUE))
#compute set of equations (4.16), (4.17) as in the recent lecture notes
b_a = solve(G, D)

fit_lm <- lm(metabolic.rate ~ body.weight, data = data, weights = errors)

png("exercise10_2.png")
attach(mtcars)
par(mfrow = c(2, 2))
#data plot
plot(data$body.weight, data$metabolic.rate, main = "data",
        ylim = range(c(data$metabolic.rate - data$errors,
                    data$metabolic.rate + data$errors)), pch = ".")
arrows(data$body.weight, data$metabolic.rate - data$errors ,data$body.weight,
        data$metabolic.rate + data$errors, length = 0.05, angle = 90, code = 3)
#data plot with calculated linear fit
plot(data$body.weight, data$metabolic.rate, main = "witch calculated linear fit")
abline(b_a[1], b_a[2], col = "blue")
#data plot with lm function linear fit
plot(data$body.weight, data$metabolic.rate, main = "with lm function linear fit")
abline(fit_lm, col = "red")
#data plot witch both fit for comparison
plot(data$body.weight, data$metabolic.rate, main = "both linear fits for comparison",
        ylim = range(c(data$metabolic.rate - data$errors,
                    data$metabolic.rate + data$errors)),
        col = rgb(0, 0, 0, 0.5), pch = ".")
abline(fit_lm, col = "red")
abline(b_a[1], b_a[2], col = "blue")
```

```
arrows(data$body.weight, data$metabolic.rate - data$errors ,data$body.weight,
         data$metabolic.rate + data$errors, length = 0.05, angle = 90, code = 3,
         col = rgb(0, 0, 0, 0.5))
```