STAT 260 - Nonparametric Statistics
Ying Sun
Statistics (STAT) Program
Computer, Electrical and Mathematical Sciences & Engineering (CEMSE) Division
King Abdullah University of Science and Technology (KAUST)

HOMEWORK I

Henrique Aparecido Laureano

Spring Semester 2018

Contents

Problem 1

Question Q1.1-1.3 on page 15 of Topic 2.

Q1.1

$$\mathbb{V}\hat{\beta}_0 = \sigma^2 \frac{\sum_{i=1}^n x_i^2}{nS_{xx}} \quad ?$$

Solution:

$$\mathbb{V}\hat{\beta}_0 = \mathbb{V}(\bar{y} - \hat{\beta}_1\bar{x}) = \mathbb{V}\bar{y} + \bar{x}^2\mathbb{V}\hat{\beta}_1 - 2\bar{x}\mathrm{Cov}(\bar{y}, \hat{\beta}_1).$$

The variance terms are

$$\mathbb{V}\bar{y} = \frac{1}{n^2}\sum_{i=1}^n \mathbb{V}y_i = \frac{\sigma^2}{n}, \qquad \mathbb{V}\hat{\beta}_1 = \frac{\sigma^2}{\sum_{i=1}^n (x_i - \bar{x})^2} = \frac{\sigma^2}{S_{xx}}.$$

The covariance term is

$$\mathrm{Cov}(\bar{y}, \hat{\beta}_1) = \mathbb{E}\bar{y}\hat{\beta}_1 - \mathbb{E}\bar{y}\mathbb{E}\hat{\beta}_1 = \beta_1(\beta_0 + \beta_1\bar{x}) - (\beta_0 + \beta_1\bar{x})\beta_1 = 0.$$

So

$$\begin{aligned}
\mathbb{V}\hat{\beta}_0 = \mathbb{V}(\bar{y} - \hat{\beta}_1\bar{x}) &= \mathbb{V}\bar{y} + \bar{x}^2\mathbb{V}\hat{\beta}_1 - 2\bar{x}\mathrm{Cov}(\bar{y}, \hat{\beta}_1) \\
&= \frac{\sigma^2}{n} + \frac{\bar{x}^2\sigma^2}{S_{xx}} - 0 = \sigma^2\frac{S_{xx} + n\bar{x}^2}{nS_{xx}} = \sigma^2\frac{\sum_{i=1}^n (x_i - \bar{x})^2 + n\bar{x}^2}{nS_{xx}} \\
&= \sigma^2\frac{\sum_{i=1}^n x_i^2 - n\bar{x}^2 + n\bar{x}^2}{nS_{xx}} = \sigma^2\frac{\sum_{i=1}^n x_i^2}{nS_{xx}}.
\end{aligned}$$

Therefore

$$\boxed{\mathbb{V}\hat{\beta}_0 = \sigma^2\frac{\sum_{i=1}^n x_i^2}{nS_{xx}}.}$$

□

Q1.2

$$\mathrm{Cov}(\hat{\beta}_0, \hat{\beta}_1) = -\sigma^2\frac{\bar{x}}{S_{xx}} \quad ?$$

Solution:

$$\begin{aligned}\mathrm{Cov}(\hat\beta_0,\hat\beta_1)=\mathbb{E}(\hat\beta_0-\mathbb{E}\hat\beta_0)(\hat\beta_1-\mathbb{E}\hat\beta_1)&=\mathbb{E}(\hat\beta_0-\beta_0)(\hat\beta_1-\beta_1)\\&=\mathbb{E}(\bar y-\hat\beta_1\bar x-\beta_0)(\hat\beta_1-\beta_1)\\&=\mathbb{E}(\beta_0+\beta_1\bar x-\hat\beta_1\bar x-\beta_0)(\hat\beta_1-\beta_1)\\&=\mathbb{E}(-(\hat\beta_1-\beta_1)\bar x)(\hat\beta_1-\beta_1)\\&=-\bar x\mathbb{E}(\hat\beta_1-\beta_1)^2\\&=-\bar x\mathbb{E}(\hat\beta_1^2-2\beta_1\hat\beta_1+\beta_1^2)\\&=-\bar x[\mathbb{E}\hat\beta_1^2-2\beta_1\mathbb{E}\hat\beta_1+\beta_1^2]\\&=-\bar x[\mathbb{V}\hat\beta_1+\mathbb{E}^2\hat\beta_1-\beta_1^2]\\&=-\bar x\left[\frac{\sigma^2}{S_{xx}}+\beta_1^2-\beta_1^2\right].\end{aligned}$$

Therefore,

$$\boxed{\mathrm{Cov}(\hat\beta_0,\hat\beta_1)=-\frac{\sigma^2\bar x}{S_{xx}}.}$$

□

Q1.3

What does mean $\mathbb{V}\hat\beta_1\neq 0$?

Solution:

We don't know the real slope, β_1, so $\hat\beta_1$ is an estimative with mean β_1 and with a respective variance. If the variance of $\hat\beta_1$ is zero this means that, is this case, $\hat\beta_1$ is exactly equal to β_1.

□

What is the source of randomness in $\hat\beta_1$?

Solution:

$$\hat\beta_1=\frac{\sum_{i=1}^n y_i(x_i-\bar x)}{\sum_{i=1}^n(x_i-\bar x)^2}.$$

The source of randomness is the data itself. In the numerator and denominator of the estimator we have the difference of each observation of the independent variable with its mean. This we can't control, depends from the data. This is the source of randomness.

□

How can you reduce $\mathbb{V}\hat\beta_1$?

Solution:

$$\mathbb{V}\hat{\beta}_1 = \frac{\sigma^2}{S_{xx}} = \frac{\sigma^2}{\sum_{i=1}^{n}(x_i - \bar{x})^2}.$$

Collecting more data. With a bigger sample size the component S_{xx} is bigger and consequently the variance is smaller.

□

Problem 2

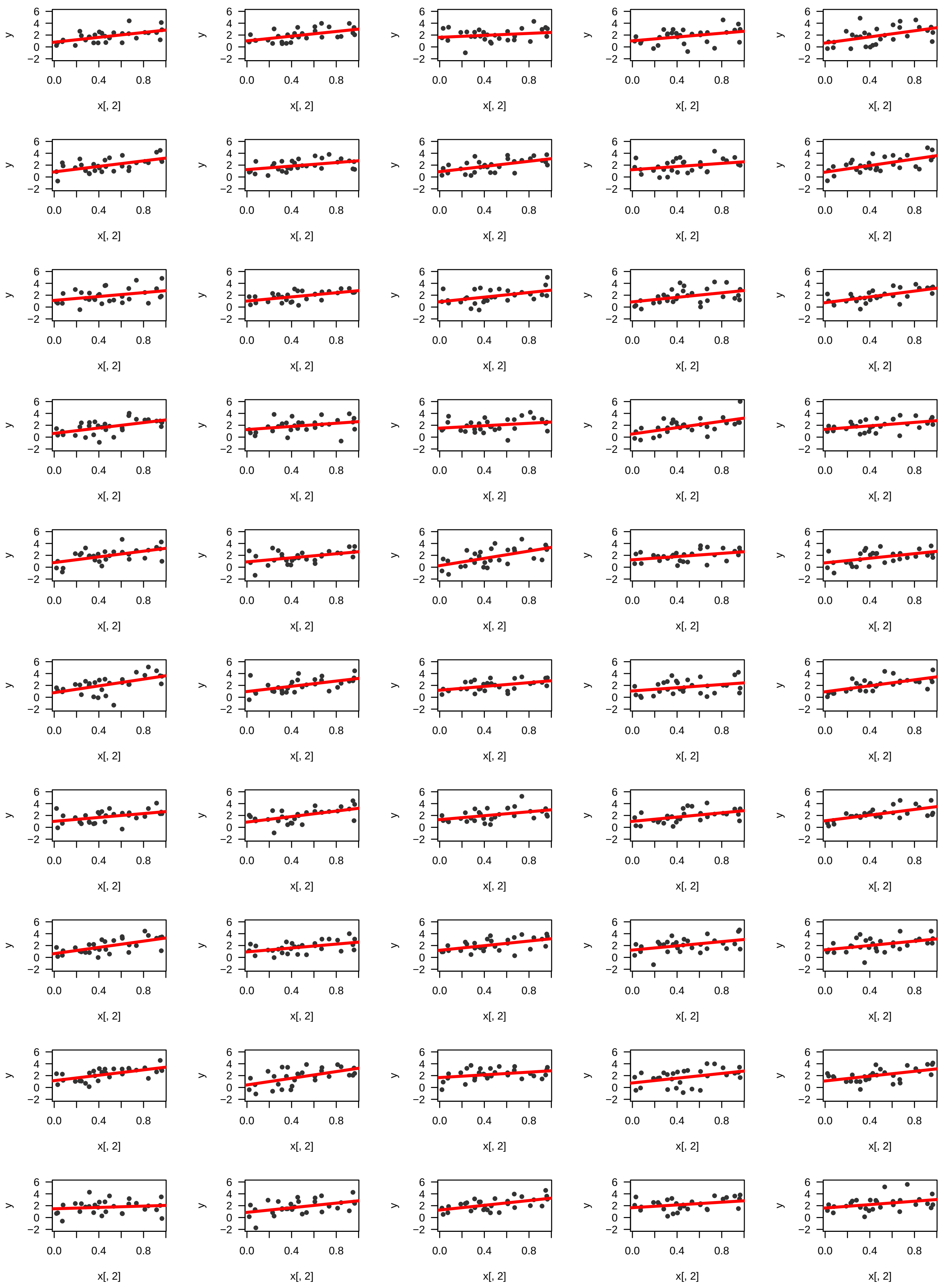
Question Q2.1-2.2 on pages 16-17 of Topic 2.

Q2.1

Change the values of n, sigma, see what happens.

```r
# <code r> ==================================================================== #
n <- 30                                                   # picking a sample size
x <- cbind(x0 = rep(1, n), x1 = runif(n))            # making up a predictor matrix
sigma <- 1                                         # assuming a 'true' error variance
beta <- c(b0 = 1, b1 = 2)                      # assuming a 'true' coefficient vector
y <- x %*% beta + rnorm(n, 0, sigma)                 # simulating a response vector
b <- solve(t(x) %*% x) %*% t(x) %*% y                                # ols estimator

# simulate and animate parallel universes/possible worlds/...
par(mfrow = c(10, 5), mar = c(4, 4, 2, 2) + .1)
for (i in 1:50) {
  y <- rnorm(n, x %*% beta, sigma)                   # simulating a response vector
  b <- solve(t(x) %*% x) %*% t(x) %*% y
  plot(x[ , 2], y, pch = 16, col = "gray20", ylim = c(-2, 6), las = 1)
  abline(b, lwd = 3, col = "red")
}
# </code r> =================================================================== #
```


y
6
4
2
0
−2
0.0
0.4
0.8
x[, 2]

Solution:

In the fifth graphs presented before we see that all the samples for y are very similar, given the small `sigma`, 1, that was used. In consequence, the fitted lines was also vey similar in all the scenarios (parallel universes/possible worlds).

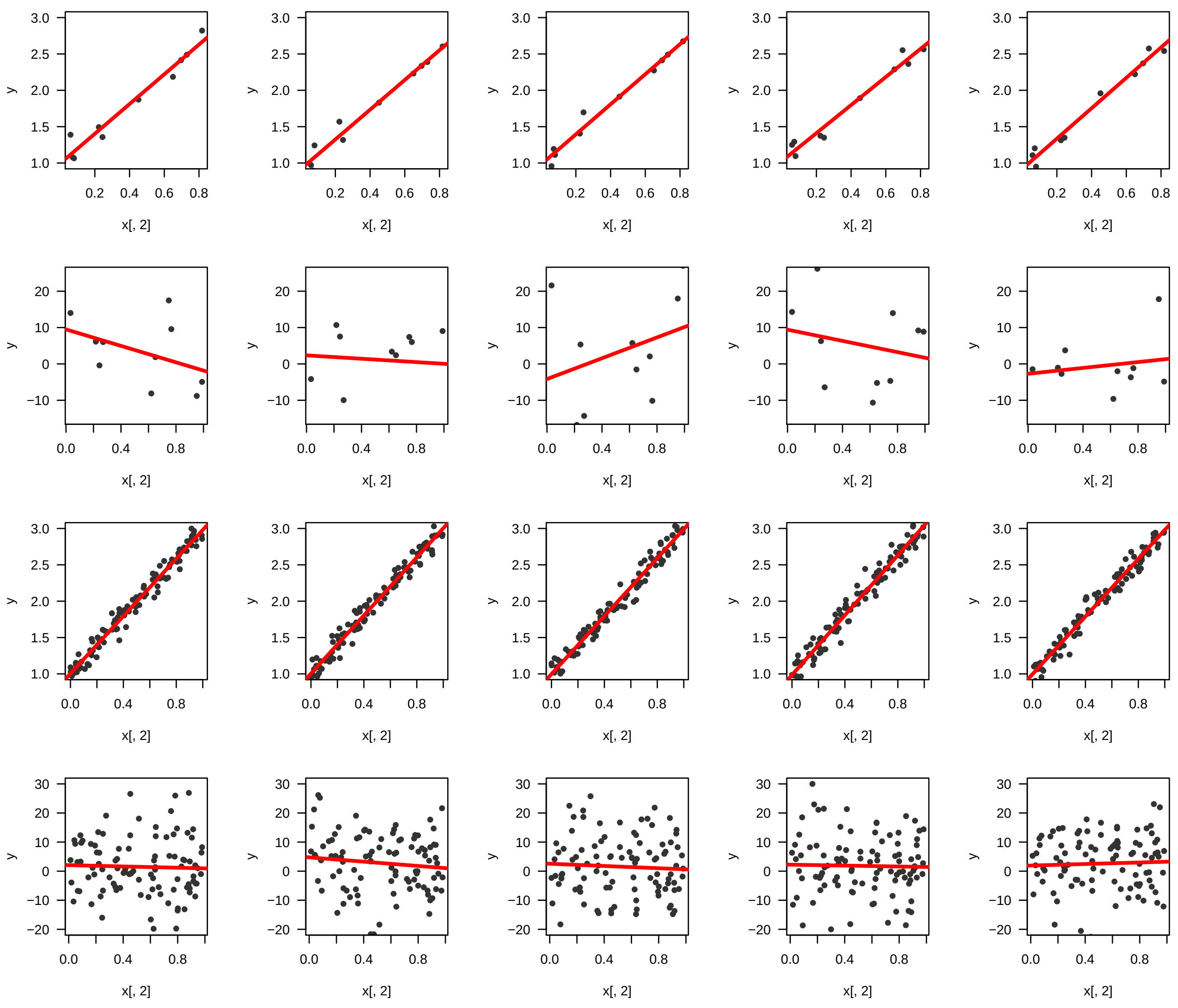
To see what changes we elaborate four scenarios and for each one we run five different samples for y. Each scenario correspond to one line in the Figure presented in the next page.

- Scenario 1 (line 1): sample size 10 and standard deviation, `sigma`, 0.1;
- Scenario 2 (line 2): sample size 10 and standard deviation 10;
- Scenario 3 (line 3): sample size 100 and standard deviation 0.1;
- Scenario 4 (line 4): sample size 100 and standard deviation 10.

In the scenarios 1 and 3 we have an equal behaviour, all the fitted lines are basically the same, given the small variance (0.1^2). This show that with a small variance the sample size doesn't seem so important.

In the scenarios 2 and 4 we have a bigger variance, and in consequence we see different fitted lines. However, with a small sample size the difference in the lines is much bigger. In conclusion, the biggest differences are seen with a small sample size and with a bigger variance.

```r
# <code r> ===================================================================== #
par(mfrow = c(4, 5), mar = c(4, 4, 2, 2) + .1)
q2.1 <- function(n, sigma, lim) {
  x <- cbind(x0 = rep(1, n), x1 = runif(n))         # making up a predictor matrix
  beta <- c(b0 = 1, b1 = 2)                      # assuming a 'true' coefficient vector
  y <- x %*% beta + rnorm(n, 0, sigma)           # simulating a response vector
  b <- solve(t(x) %*% x) %*% t(x) %*% y                          # ols estimator
  # simulate and animate parallel universes/possible worlds/...
  for (i in 1:5) {
    y <- rnorm(n, x %*% beta, sigma)             # simulating a response vector
    b <- solve(t(x) %*% x) %*% t(x) %*% y
    plot(x[ , 2], y, pch = 16, col = "gray20", ylim = lim, las = 1)
    abline(b, lwd = 3, col = "red")
  }
}
q2.1(n = 10,  sigma = .1, lim = c(1, 3))
q2.1(n = 10,  sigma = 10, lim = c(-15, 25))
q2.1(n = 100, sigma = .1, lim = c(1, 3))
q2.1(n = 100, sigma = 10, lim = c(-20, 30))
# </code r> ==================================================================== #
```



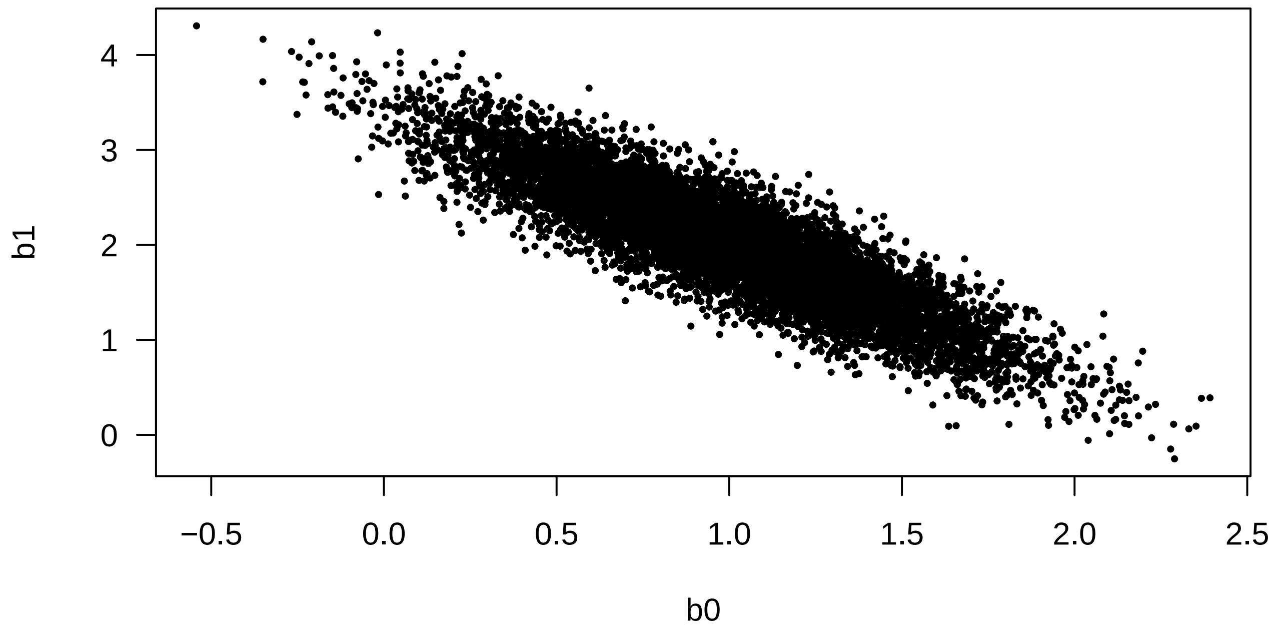
□

Q2.2

Change the values of n, sigma, see what happens.

```
# <code r> ==================================================================== #
n <- 30                                                     # picking a sample size
x <- cbind(x0 = rep(1, n), x1 = runif(n))            # making up a predictor matrix
sigma <- 1                                         # assuming a 'true' error variance
beta <- c(b0 = 1, b1 = 2)                        # assuming a 'true' coefficient vector
# next, simulate many response vectors (possible worlds, parallel universes,...)
#         and store the results:
nsim <- 1e4
bs <- matrix(NA, nrow = nsim, ncol = length(beta))          # storage for coefficient
                                                            #                estimates
```

```
colnames(bs) <- paste0("b", 0:(length(beta)-1))                      # cosmetics
for (isim in 1:nsim) {
  y <- x %*% beta + rnorm(n, 0, sigma)             # simulating a response vector
  bs[isim, ] <- solve(t(x) %*% x) %*% t(x) %*% y
  # if (isim %% 100 == 0) cat(isim, "...")
}
# plot slopes versus intercepts to illustrate the 'sampling' or
# 'dataset-to-dataset' variability of b:
par(mar = c(4, 4, 1, 2) + .1) ; plot(bs, pch = 16, cex = .5, las = 1)
# => negative correlation between b0 and b1 !!!
# </code r> ================================================================== #
```

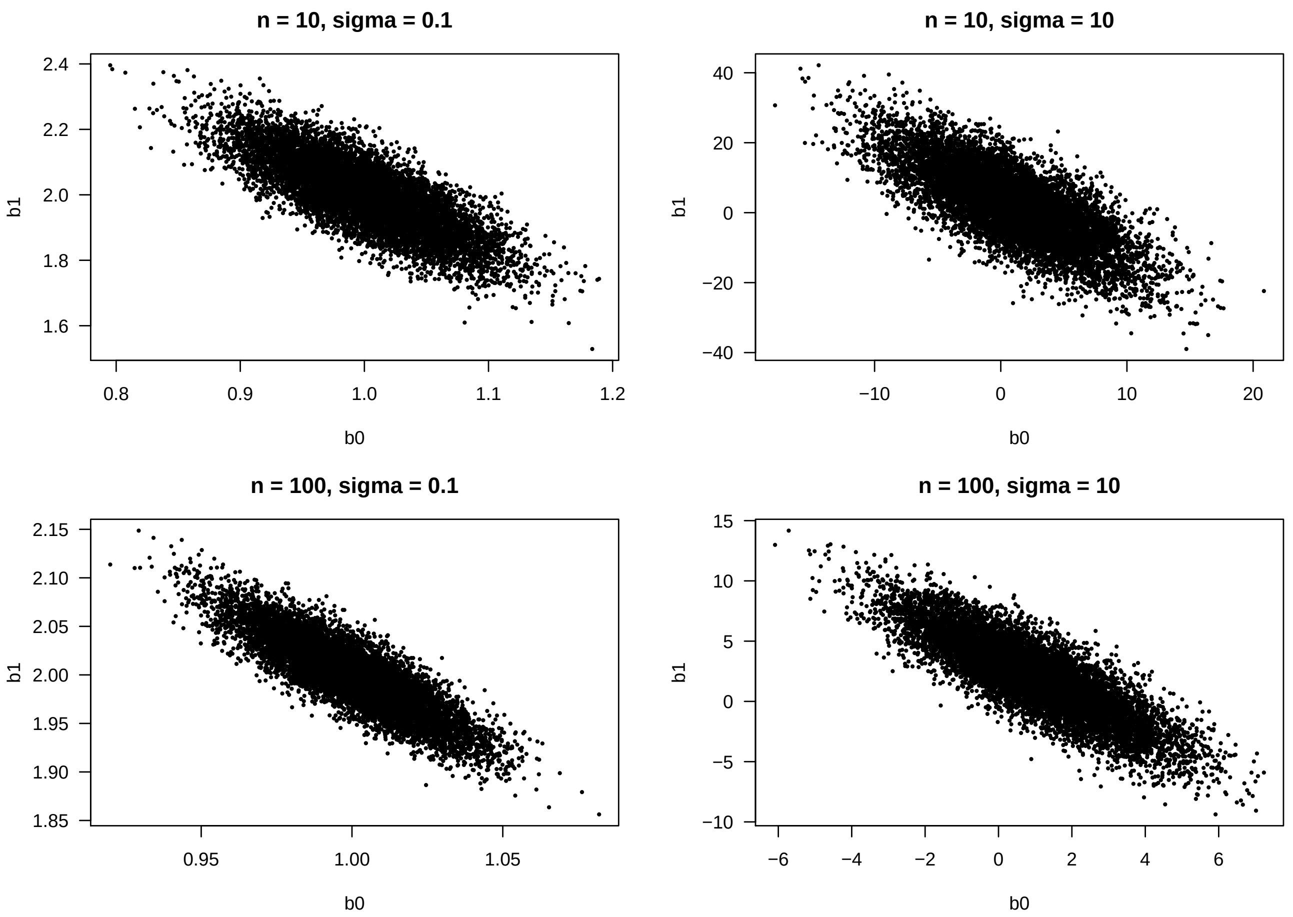


Solution:

To see what changes we elaborate four scenarios, the results are presented in the Figure in the next page.

- Scenario 1 (line 1): sample size 10 and standard deviation, `sigma`, 0.1;
- Scenario 2 (line 2): sample size 10 and standard deviation 10;
- Scenario 3 (line 3): sample size 100 and standard deviation 0.1;
- Scenario 4 (line 4): sample size 100 and standard deviation 10.

```
# <code r> =================================================================== #
par(mfrow = c(2, 2), mar = c(4, 4, 3, 2) + .1)
q2.2 <- function(n, sigma, title) {
  x <- cbind(x0 = rep(1, n), x1 = runif(n))         # making up a predictor matrix
  beta <- c(b0 = 1, b1 = 2)                      # assuming a 'true' coefficient vector
  # next, simulate many response vectors (possible worlds, parallel universes,...)
  #       and store the results:
  nsim <- 1e4
  bs <- matrix(NA, nrow = nsim, ncol = length(beta))     # storage for coefficient
                                                          #                estimates
  colnames(bs) <- paste0("b", 0:(length(beta)-1))                       # cosmetics
  for (isim in 1:nsim) {
    y <- x %*% beta + rnorm(n, 0, sigma)             # simulating a response vector
    bs[isim, ] <- solve(t(x) %*% x) %*% t(x) %*% y
  }
  # plot slopes versus intercepts to illustrate the 'sampling' or
  # 'dataset-to-dataset' variability of b
  plot(bs, pch = 16, cex = .5, las = 1, main = title)
}
q2.2(n = 10,  sigma = .1, title = "n = 10, sigma = 0.1")
q2.2(n = 10,  sigma = 10, title = "n = 10, sigma = 10")
q2.2(n = 100, sigma = .1, title = "n = 100, sigma = 0.1")
q2.2(n = 100, sigma = 10, title = "n = 100, sigma = 10")
# </code r> ==================================================================== #
```



With the small sample size together with the bigger variance, scenario 2, we see the biggest variance between the coefficients. The smallest variance between the coefficients is seen with a bigger sample size and small variance, scenario 3.

Between the scenarios 1 and 4 the scenario 1 (with the small variance) is better because present a small variance between the coefficients.

$\square$

Problem 3

Question Q3 on page 19 of Topic 2.

Prove the following ANOVA Decomposition:

$$\underbrace{\text{TSS}}_{\text{Total variations in } Y} = \underbrace{\text{SS}_{\text{reg}}}_{\text{Variations in } Y \text{ explained by model}} + \underbrace{\text{RSS}}_{\text{Variations in } Y \text{ unexplained by model}}$$

Solution:

Notation:

$$\begin{aligned}
\text{TSS} &= \sum_{i=1}^{n}(y_i - \bar{y})^2, && \text{Total sum of squares} \\
\text{RSS} &= \sum_{i=1}^{n}(y_i - \hat{y})^2, && \text{Residual sum of squares} \\
\text{SS}_{reg} &= \sum_{i=1}^{n}(\hat{y}_i - \bar{y})^2, && \text{Sum of squares in regression}
\end{aligned}$$

with $\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x_i$.

Proof:

$$\begin{aligned}
y_i - \bar{y} &= \hat{y}_i - \bar{y} + y_i - \hat{y}_i \\
y_i - \bar{y} &= (\hat{y}_i - \bar{y}) + (y_i - \hat{y}_i) \\
\sum_{i=1}^{n}(y_i - \bar{y})^2 &= \sum_{i=1}^{n}\Big[(\hat{y}_i - \bar{y}) + (y_i - \hat{y}_i)\Big]^2 \\
\sum_{i=1}^{n}(y_i - \bar{y})^2 &= \sum_{i=1}^{n}\Big[(\hat{y}_i - \bar{y})^2 + (y_i - \hat{y}_i)^2 + 2(\hat{y}_i - \bar{y})(y_i - \hat{y}_i)\Big] \\
\sum_{i=1}^{n}(y_i - \bar{y})^2 &= \sum_{i=1}^{n}(\hat{y}_i - \bar{y})^2 + \sum_{i=1}^{n}(y_i - \hat{y}_i)^2 + 2\sum_{i=1}^{n}(\hat{y}_i - \bar{y})(y_i - \hat{y}_i),
\end{aligned}$$

but

$$\begin{aligned}\sum_{i=1}^{n}(\hat{y}_i-\bar{y})(y_i-\hat{y}_i) &= \sum_{i=1}^{n}\Big[\hat{y}_i y_i-\hat{y}_i^2-\bar{y}y_i+\bar{y}\hat{y}_i\Big]\\ &= \sum_{i=1}^{n}\hat{y}_i y_i-\sum_{i=1}^{n}\hat{y}_i^2-\bar{y}\sum_{i=1}^{n}y_i+\bar{y}\sum_{i=1}^{n}\hat{y}_i\\ &= \sum_{i=1}^{n}\hat{y}_i(y_i-\hat{y}_i)-\bar{y}\sum_{i=1}^{n}(y_i-\hat{y}_i)\\ &= 0,\end{aligned}$$

because

$$\begin{aligned}i^{\text{th}}\text{ residual}\qquad\Rightarrow\qquad e_i &= y_i-\hat{y}_i\\ \sum_{i=1}^{n}e_i &= \sum_{i=1}^{n}(y_i-\hat{y}_i)\\ &= \sum_{i=1}^{n}(y_i-\hat{\beta}_0-\hat{\beta}_1x_i)\\ &= \sum_{i=1}^{n}y_i-n\hat{\beta}_0-\hat{\beta}_1\sum_{i=1}^{n}x_i\\ &= \sum_{i=1}^{n}y_i-n\left(\bar{y}-\hat{\beta}_1\sum_{i=1}^{n}\frac{x_i}{n}\right)-\hat{\beta}_1\sum_{i=1}^{n}x_i\\ &= \sum_{i=1}^{n}y_i-\sum_{i=1}^{n}y_i+\hat{\beta}_1\sum_{i=1}^{n}x_i-\hat{\beta}_1\sum_{i=1}^{n}x_i\\ &= 0.\end{aligned}$$

Therefore,

$$\begin{aligned}\sum_{i=1}^{n}(y_i-\bar{y})^2 &= \sum_{i=1}^{n}(\hat{y}_i-\bar{y})^2+\sum_{i=1}^{n}(y_i-\hat{y}_i)^2+2\sum_{i=1}^{n}(\hat{y}_i-\bar{y})(y_i-\hat{y}_i)\\ &= \sum_{i=1}^{n}(\hat{y}_i-\bar{y})^2+\sum_{i=1}^{n}(y_i-\hat{y}_i)^2+0\\ \underbrace{\sum_{i=1}^{n}(y_i-\bar{y})^2}_{\text{TSS}} &= \underbrace{\sum_{i=1}^{n}(\hat{y}_i-\bar{y})^2}_{\text{SS}_{\text{reg}}}+\underbrace{\sum_{i=1}^{n}(y_i-\hat{y}_i)^2}_{\text{RSS}}.\end{aligned}$$

$\square$

Problem 4

Question Q4 on page 20 of Topic 2.

Prove that for simple linear regression, R^2 equals to r^2, the sample correlation between X and Y

$$R^2 = r^2 = \frac{\Big(\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})\Big)^2}{\sum_{i=1}^n (x_i - \bar{x})^2 \sum_{i=1}^n (y_i - \bar{y})^2}.$$

<u>Solution:</u>

First,

we know that $\quad R^2 = \frac{\text{SS}_{\text{reg}}}{TSS} = \frac{\sum_{i=1}^n (\hat{y}_i - \bar{y})^2}{\sum_{i=1}^n (y_i - \bar{y})^2}, \quad \hat{\beta}_0 = \bar{y} - \hat{\beta}_1 \bar{x} \quad$ and $\quad \hat{\beta}_1 = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^n (x_i - \bar{x})^2}.$

So

$$\begin{aligned} \text{SS}_{\text{reg}} = \sum_{i=1}^n (\hat{y}_i - \bar{y})^2 &= \sum_{i=1}^n (\hat{\beta}_0 + \hat{\beta}_1 x_i - \bar{y})^2 \\ &= \sum_{i=1}^n (\bar{y} - \hat{\beta}_1 \bar{x} + \hat{\beta}_1 x_i - \bar{y})^2 \\ &= \hat{\beta}_1^2 \sum_{i=1}^n (x_i - \bar{x})^2 \\ &= \frac{\Big(\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})\Big)^2}{\Big(\sum_{i=1}^n (x_i - \bar{x})^2\Big)^2} \sum_{i=1}^n (x_i - \bar{x})^2 \\ &= \frac{\Big(\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})\Big)^2}{\sum_{i=1}^n (x_i - \bar{x})^2}. \end{aligned}$$

Therefore,

$$\boxed{R^2 = \frac{\text{SS}_{\text{reg}}}{TSS} = \frac{\sum_{i=1}^n (\hat{y}_i - \bar{y})^2}{\sum_{i=1}^n (y_i - \bar{y})^2} = \frac{\Big(\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})\Big)^2}{\sum_{i=1}^n (x_i - \bar{x})^2 \sum_{i=1}^n (y_i - \bar{y})^2} = r^2.}$$

□

Problem 5

Write your own version of `anova()` function on page 14 of Topic 3 using `R`. Your function should not use `lm()` function. Must be able to compare two nested models.

Replicate results on slides 14, 17, 18, 21 and 22 about the `gala` data set.

Solution:

Slide 14.

$$\begin{aligned}\text{Small model}: &\quad \text{Species} = \beta_0 + \varepsilon\\ \text{Bigger model}: &\quad \text{Species} = \beta_0 + \beta_1\text{Area} + \beta_2\text{Elevation} + \beta_3\text{Nearest} + \beta_4\text{Scruz} + \beta_5\text{Adjacant} + \varepsilon\end{aligned}$$

```r
# <code r> ================================================================== #
library(faraway) ; data(gala)                          # loading package and dataset
myanova <- function(y, small, bigger) {                      # y: response variable
  # small: covariables in the small model; bigger: covariables in the bigger model
  n = nrow(gala)                                                      # sample size
  id = rep(1, n)                                         # vector of 1s, intercept
  ybar = (1/n) * id %*% t(id) %*% y                    # \bar{y} in matricial form
  ysum = t(y) %*% y                                      # sum_{i=1}^{n} y_{i}^{2}
  # computations for the small model (1)
  x1 = as.matrix(cbind(id, small))           # creating matrix x, the model matrix
  inv1 = solve(t(x1) %*% x1)                                  # (x^{\top} x)^{-1}
  h1 = x1 %*% inv1 %*% t(x1)                                  # creating h matrix
  y1hat = h1 %*% y                                     # \bar{y} in matricial form
  ss1 = t(y1hat - ybar) %*% (y1hat - ybar)          # sum of squares in regression
  rss1 = ysum - t(y) %*% y1hat                           # residual sum of squares
  df1 = n - ncol(x1)                          # degrees of freedom (n - # of parameters)
  # computations for the bigger model (2)
  x2 = as.matrix(cbind(id, bigger))          # creating matrix x, the model matrix
  inv2 = solve(t(x2) %*% x2)                                  # (x^{\top} x)^{-1}
  h2 = x2 %*% inv2 %*% t(x2)                                  # creating h matrix
  y2hat = h2 %*% y                                     # \bar{y} in matricial form
  ss2 = t(y2hat - ybar) %*% (y2hat - ybar)          # sum of squares in regression
  rss2 = ysum - t(y) %*% y2hat                           # residual sum of squares
  df2 = n - ncol(x2)                          # degrees of freedom (n - # of parameters)
  tss = t(y - ybar) %*% (y - ybar)                          # total sum of squares
  f = ((rss1 - rss2)/(df1 - df2)) / (rss2/df2)                  # f-test statistic
  # results to be outputted
  cat("Small model:", "\n", "RSS:", rss1, ", df:", df1, "\n")
  cat("Bigger model:", "\n", "RSS:", rss2, ", df:", df2, "\n")
  cat("F Statistic:", f, ", p-value", pf(f, df1-df2, df2, lower.tail = FALSE))
}
myanova(y = gala$Species, small = NULL,
        bigger = gala[ , c("Area", "Elevation", "Nearest", "Scruz", "Adjacent")])
# </code r> ================================================================= #
```

```
Small model:
 RSS: 381081.4 , df: 29
Bigger model:
 RSS: 89231.37 , df: 24
F Statistic: 15.69941 , p-value 6.837893e-07
```

Slide 17.

$$\begin{aligned}\text{Small model}: &\quad \text{Species} = \beta_0 + \beta_1\text{Elevation} + \beta_2\text{Nearest} + \beta_3\text{Scruz} + \beta_4\text{Adjacant} + \varepsilon\\ \text{Bigger model}: &\quad \text{Species} = \beta_0 + \beta_1\text{Area} + \beta_2\text{Elevation} + \beta_3\text{Nearest} + \beta_4\text{Scruz} + \beta_5\text{Adjacant} + \varepsilon\end{aligned}$$

```
# <code r> ==================================================================== #
myanova(y = gala$Species,
        small = gala[ , c("Elevation", "Nearest", "Scruz", "Adjacent")],
        bigger = gala[ , c("Area", "Elevation", "Nearest", "Scruz", "Adjacent")]
)
# </code r> =================================================================== #
```

```
Small model:
 RSS: 93469.08 , df: 25
Bigger model:
 RSS: 89231.37 , df: 24
F Statistic: 1.139792 , p-value 0.296318
```

Slide 18.

$$\begin{aligned}\text{Small model}: &\quad \text{Species} = \beta_0 + 0.5\text{Area} + \beta_2\text{Elevation} + \beta_3\text{Nearest} + \beta_4\text{Scruz} + \beta_5\text{Adjacant} + \varepsilon\\ \text{Bigger model}: &\quad \text{Species} = \beta_0 + \beta_1\text{Area} + \beta_2\text{Elevation} + \beta_3\text{Nearest} + \beta_4\text{Scruz} + \beta_5\text{Adjacant} + \varepsilon\end{aligned}$$

```
# <code r> ==================================================================== #
myanova(y = gala$Species - .5 * gala$Area,
        small = gala[ , c("Elevation", "Nearest", "Scruz", "Adjacent")],
        bigger = gala[ , c("Area", "Elevation", "Nearest", "Scruz", "Adjacent")]
)
# </code r> =================================================================== #
```

```
Small model:
 RSS: 2119270 , df: 25
Bigger model:
 RSS: 89231.37 , df: 24
F Statistic: 546.0068 , p-value 5.103823e-18
```

Slide 21.

$$\begin{aligned}\text{Small model}: &\quad \text{Species} = \beta_0 + \beta_1\text{Elevation} + \beta_2\text{Nearest} + \beta_3\text{Scruz} + \varepsilon\\ \text{Bigger model}: &\quad \text{Species} = \beta_0 + \beta_1\text{Area} + \beta_2\text{Elevation} + \beta_3\text{Nearest} + \beta_4\text{Scruz} + \beta_5\text{Adjacant} + \varepsilon\end{aligned}$$

```
# <code r> ====================================================================== #
myanova(y = gala$Species,
        small = gala[ , c("Elevation", "Nearest", "Scruz")],
        bigger = gala[ , c("Area", "Elevation", "Nearest", "Scruz", "Adjacent")]
)
# </code r> ===================================================================== #
```

```
Small model:
 RSS: 158291.6 , df: 26
Bigger model:
 RSS: 89231.37 , df: 24
F Statistic: 9.287352 , p-value 0.001029711
```

Slide 22.

$$\begin{aligned}\text{Small model}: &\quad \text{Species} = \beta_0 + \beta_1\text{Elevation} + \beta_2\text{Nearest} + \beta_3\text{Scruz} + \beta_4(\text{Area} + \text{Adjacent}) + \varepsilon \\ \text{Bigger model}: &\quad \text{Species} = \beta_0 + \beta_1\text{Area} + \beta_2\text{Elevation} + \beta_3\text{Nearest} + \beta_4\text{Scruz} + \beta_5\text{Adjacant} + \varepsilon\end{aligned}$$

```
# <code r> ====================================================================== #
myanova(y = gala$Species,
        small = cbind(gala[ , c("Elevation", "Nearest", "Scruz")],
                      gala[ , "Area"] + gala[ , "Adjacent"]),
        bigger = gala[ , c("Area", "Elevation", "Nearest", "Scruz", "Adjacent")]
)
# </code r> ===================================================================== #
```

```
Small model:
 RSS: 109591.1 , df: 25
Bigger model:
 RSS: 89231.37 , df: 24
F Statistic: 5.476035 , p-value 0.02792556
```

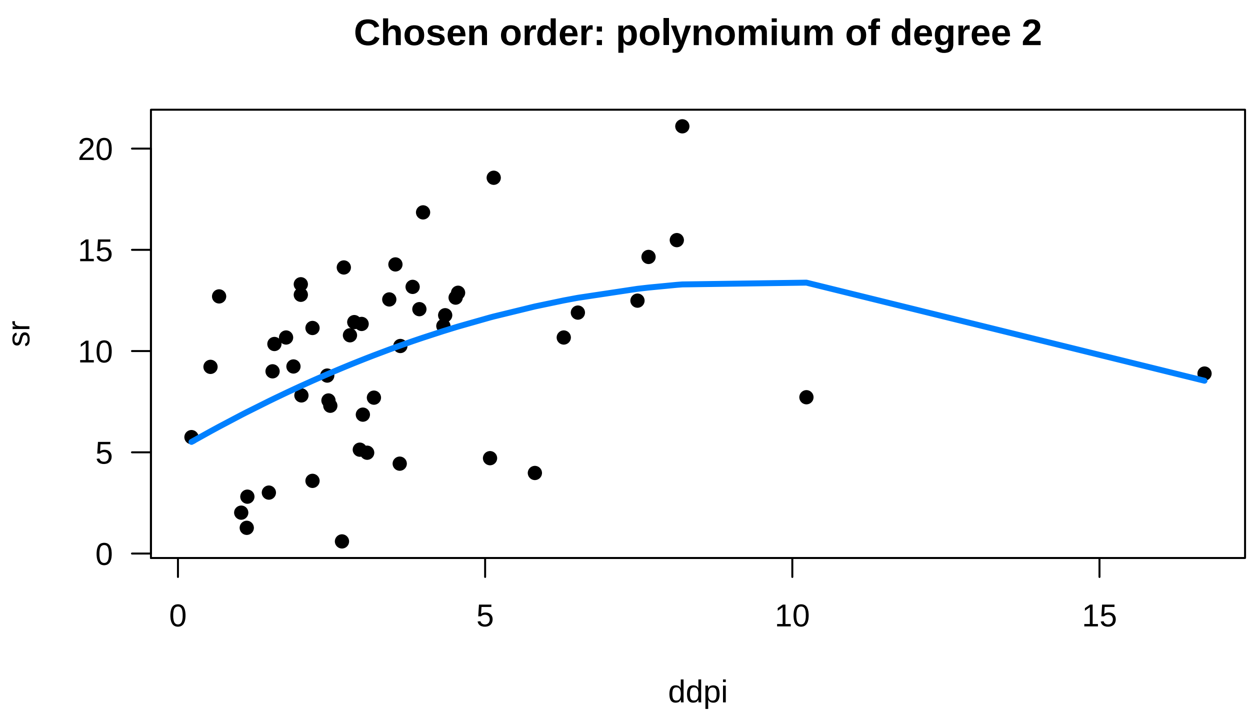
□

Problem 6

Write an `R` function that can be used to automatically determine the order *d* of the polynomial regression and replicate the results on pages 31 and 32 of Topic 3 about the `savings` data. (The output should be the chosen order *d* and the plot of the fitted polynomial function to the data.)

Solution:

```
# <code r> ================================================================== #
data("savings")                                                # loading the data
poly.reg <- function(y, x) {
  resp = as.matrix(y)                  # converting the response to a vector (matrix)
  n = length(resp)                     # computing the number of elements, sample size
  mm = cbind(x)                # creating the matrix that will receive the polynomials
  p = 1                                          # initializing the polynomial degree
  sig = 0  # initializing the p-value limit (creating the obj with a random value)
  while (sig < .05) {                                   # p-value limit: 0.05 (5%)
    model = lm(resp ~ mm)                                        # fitting the model
    sig = summary(model)$coef[ncol(mm)+1, 4]                   # storing the p-value
    if (sig < .05) {                              # if the p-value is less than 0.05:
      p = p + 1                                 # increase by 1 the polynomial degree
      mm = cbind(mm, x**p)  # compute the new degree and store in the model matrix
    }
  }
  p = p - 1                                     # taking the picked polynomial degree
  model = lm(resp ~ mm[ , 1:p])             # fitting the model with the picked degree
  plot(resp ~ x, pch = 16, las = 1, xlab = "ddpi", ylab = "sr"          # plotting
       , main = paste("Chosen order: polynomium of degree", p))
  lines(sort(x), fitted(model)[order(x)], col = "#0080ff", lwd = 3)
}
poly.reg(y = savings$sr, x = savings$ddpi)
# </code r> ================================================================= #
```



■