PhD Workshop Series in Statistics and Applied Data Science II (DSP.007)

Regression basics I+II

Ass.-Prof. PD Dr. Wolfgang Trutschnig

Research group for Stochastics/Statistics
Department for Mathematics
University Salzburg
www.trutschnig.net

Salzburg, Mai 2018

Plan for today:

- Pearson vs. Spearman (rank) correlation
- The general idea behind regression
- First steps (multivariate) linear regression
- Nonparametric regression
- Fitting parametric models
- Exercises
- No sophisticated models (glm, lmm, etc.), just the very basics
- R-Code and slides can be found on www.trutschnig.net/courses

Please

- Ask whenever something is unclear
- Solve the exercises

Quick Reminder: Pearson correlation coefficient ρ

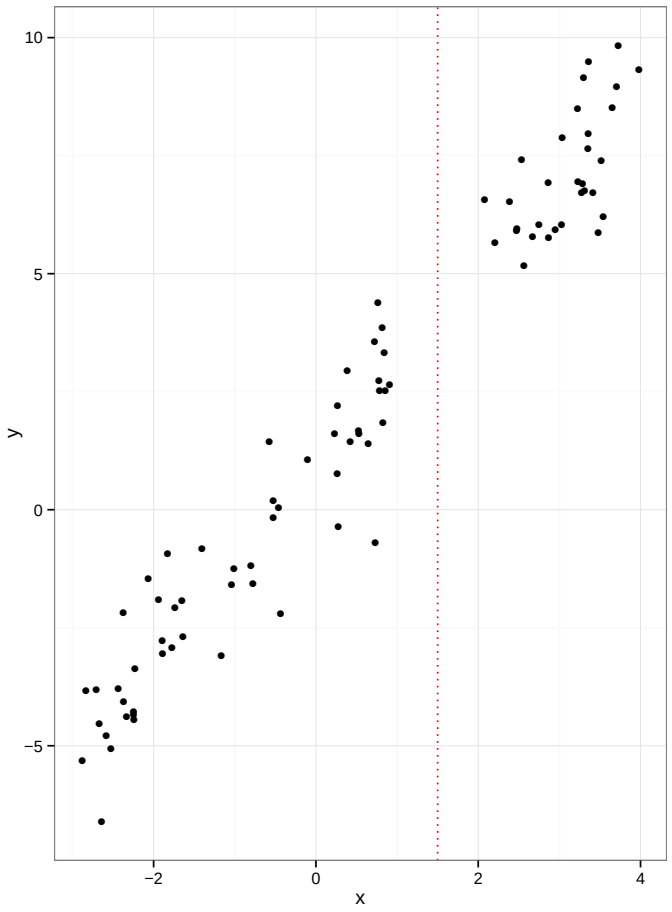


Figure: What is the correlation coefficient of the drawn sample?

- The graphic depicts a sample $(x_1, y_1), \ldots, (x_n, y_n)$.
- Give a rough estimate of the correlation coefficient ρ of the sample
- How can ρ be calculated?
- Let s_x (resp. s_y) denote the standard deviation of the x-coordinates (y-coordinates) of the sample, i.e.

$$s_x = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \overline{x}_n)^2}$$

$$s_y = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(y_i - \overline{y}_n)^2}$$

Quick Reminder: Pearson correlation coefficient ρ

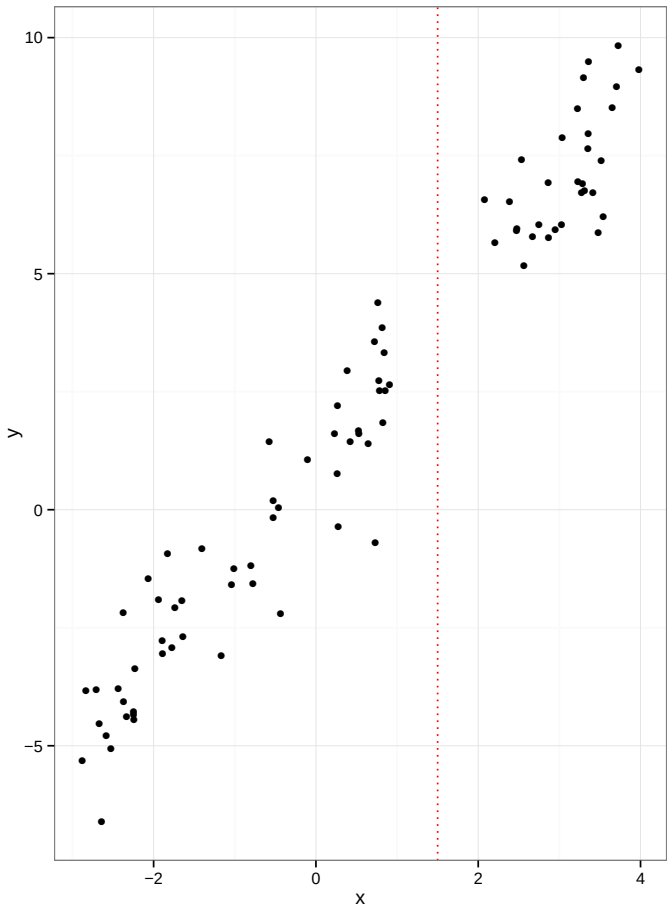


Figure: What is the correlation coefficient of the drawn sample?

- Let s_{xy} denote the (empirical) covariance of the sample, i.e.

$$s_{xy} = \frac{1}{n-1} \sum_{i=1}^{n} (x_i - \overline{x}_n)(y_i - \overline{y}_n)$$

- The (Pearson) correlation coefficient ρ_{xy} is defined as

$$\rho_{xy} = \frac{s_{xy}}{s_x s_y}$$

  if $s_x, s_y > 0$.

- In our case we get $\rho_{xy} = 0.97464$.
- How can this value be interpreted?

Properties of ρ

- Whenever ρ_{xy} exists (i.e. whenever $s_x, s_y > 0$) we have $-1 \leq \rho_{xy} \leq 1$.
- We have $\rho_{xy} = \rho_{yx}$. As a consequence we will simply write ρ in the sequel.
- $\rho = 1$ if and only if $(x_1, y_1), \ldots, (x_n, y_n)$ lie on a straight line with positive slope.
- $\rho = -1$ if and only if $(x_1, y_1), \ldots, (x_n, y_n)$ lie on a straight line with negative slope.
- In case of $\rho = 0$ we call the sample $(x_1, y_1), \ldots, (x_n, y_n)$ uncorrelated.
- $\rho = 0$ is not a measure of dependence - it only measures *linear dependence*.
- $\rho = 0$ means that there is no linear dependence.
- If instead of $(x_1, y_1), \ldots, (x_n, y_n)$ we consider $(2x_1, 3y_1), \ldots, (2x_n, 3y_n)$, what happens to ρ?
- If instead of $(x_1, y_1), \ldots, (x_n, y_n)$ we consider $(-2x_1, -3y_1), \ldots, (-2x_n, -3y_n)$, what happens to ρ?

Quick Reminder: Pearson correlation coefficient ρ

- If instead of $(x_1, y_1), \ldots, (x_n, y_n)$ we consider $(-2x_1, -3y_1), \ldots, (-2x_n, -3y_n)$, what happens to ρ?

```
file <- url("http://www.trutschnig.net/geo_reg1.RData")
load(file)
A<-geo_reg1
head(geo_reg1)

cor(A$x,A$y)
cor(2*A$x,3*A$y)
cor(-2*A$x,-3*A$y)
cor(-2*A$x,3*A$y)
```

- ρ does not change under *linear* transformations with the same sign.
- ρ changes, however, under non-linear transformations:
- If instead of $(x_1, y_1), \ldots, (x_n, y_n)$ we consider $(x_1^3, y_1^3), \ldots, (x_n^3, y_n^3)$ then we get $\rho = 0.9$.

Spearman rank correlation ρ_S

- Assume we want to have a measure quantifying if there is a monotonic relationship between the x- and the y-coordinates of a sample $(x_1, y_1), \ldots, (x_n, y_n)$.
- 'Monotonic relationship' (or concordance) in the sense that if the x-coordinates increase then also the y-coordinates (grow or fall together).
- There is no need for the relationship to be linear.
- One natural idea is to work with ranks - best explained by some simple examples:

```
x1 <- c(3, 1, 4, 15, 13)
r1 <- rank(x1)
x1
#[1] 3  1  4 15 13
r1
#[1] 2 1 3 5 4
```

Spearman rank correlation ρ_s

```
x1 <- c(3, 1, 3, 15, 13)
r1 <- rank(x1)
x1
#[1] 3  1  3 15 13
r1
#[1] 2.5 1.0 2.5 5.0 4.0
```

- The values are sorted - the rank $rk(x_i)$ of observation x_i is the position after the ranking.
- In case of ties averages of the ranks will be calculated (other choices are optional in the function).
- From $(x_1, y_1), \ldots, (x_n, y_n)$ we get the sample ranks $(rk_x(x_1), rk_y(y_1)), \ldots, (rk_x(x_n), rk_y(y_n))$.
- $rk_x(x_i)$ is the rank of observation x_i among $x_1, \ldots, x_n$.
- $rk_y(y_i)$ is the rank of observation y_i among $y_1, \ldots, y_n$.
- The Spearman rank correlation is defined as the Pearson correlation of these ranks.

Example

- Considering the following sample of size $n = 5$

x	y
3.05	10.21
1.38	2.19
4.32	19.31
15.51	241.08
7.08	50.81

x	y	rk.x	rk.y
3.05	10.21	2.00	2.00
1.38	2.19	1.00	1.00
4.32	19.31	3.00	3.00
15.51	241.08	5.00	5.00
7.08	50.81	4.00	4.00

- What can be seen?
- For ρ_S we get $\rho_s = 1$

```
cor(rank(E$x),rank(E$y))
cor(E$x,E$y,method="spearman")
```

Properties of ρ_S:

- Whenever ρ_S exists we have $-1 \leq \rho_{xy} \leq 1$.
- ρ_S is symmetric too.
- $\rho_S = 1$ if and only if: for each pair $(x_i, y_i), (x_j, y_j)$ we have $x_i \leq x_j$ and only if $y_i \leq y_j$.
- $\rho_S = -1$ if and only if: for each pair $(x_i, y_i), (x_j, y_j)$ we have $x_i \leq x_j$ if and only if $y_i \geq y_j$.
- $\rho_S = 0$ is not a measure of dependence - it only measures *monotonic dependence* (aka concordance).
- $\rho_S = 0$ means that there is no monotonic relationship dependence.
- If instead of $(x_1, y_1), \ldots, (x_n, y_n)$ we consider $(2x_1, 3y_1), \ldots, (2x_n, 3y_n)$, what happens to ρ_S?
- If instead of $(x_1, y_1), \ldots, (x_n, y_n)$ we consider $(-2x_1, -3y_1), \ldots, (-2x_n, -3y_n)$, what happens to ρ_S?
- If instead of $(x_1, y_1), \ldots, (x_n, y_n)$ we consider $(x_1^3, y_1^3), \ldots, (x_n^3, y_n^3)$, what happens to ρ_S?

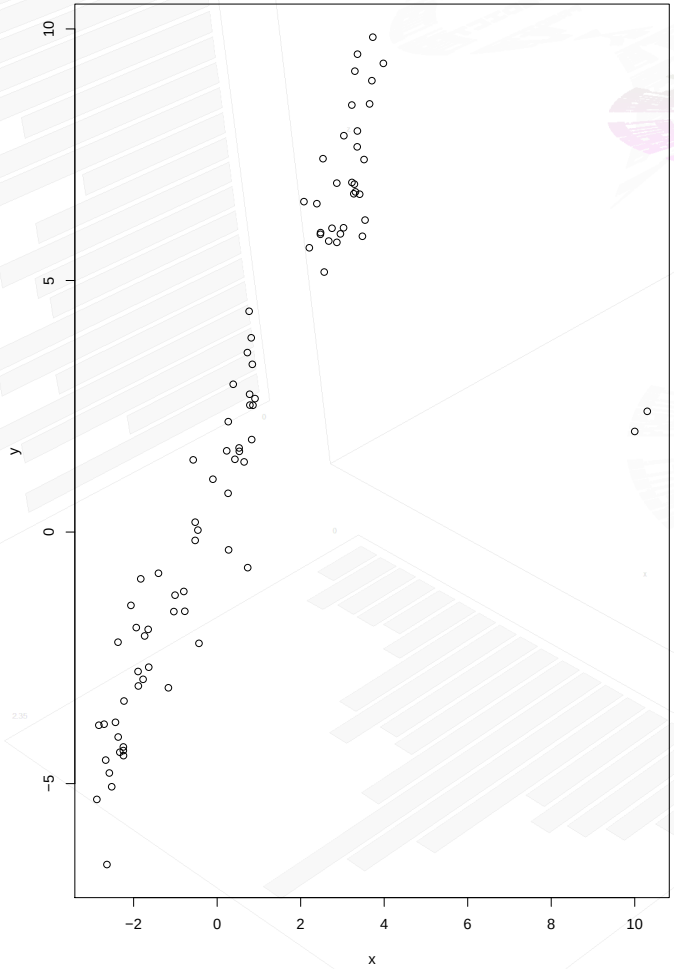
Spearman rank correlation ρ_S

```
file <- url("http://www.trutschnig.net/geo_reg1.RData")
load(file)
A<-geo_reg1
head(geo_reg1)

cor(A$x,A$y,method = "spearman")
cor(2*A$x,3*A$y,method = "spearman")
cor(-2*A$x,-3*A$y,method = "spearman")
cor(A$x^3,A$y^3,method = "spearman")
```

- For all four cases we get $\rho_S = 0.9633945$.
- Easy to verify: ρ_S is invariant under monotonic transformations (both increasing or both decreasing).
- Let's add two outliers to A and see how ρ and ρ_S change.

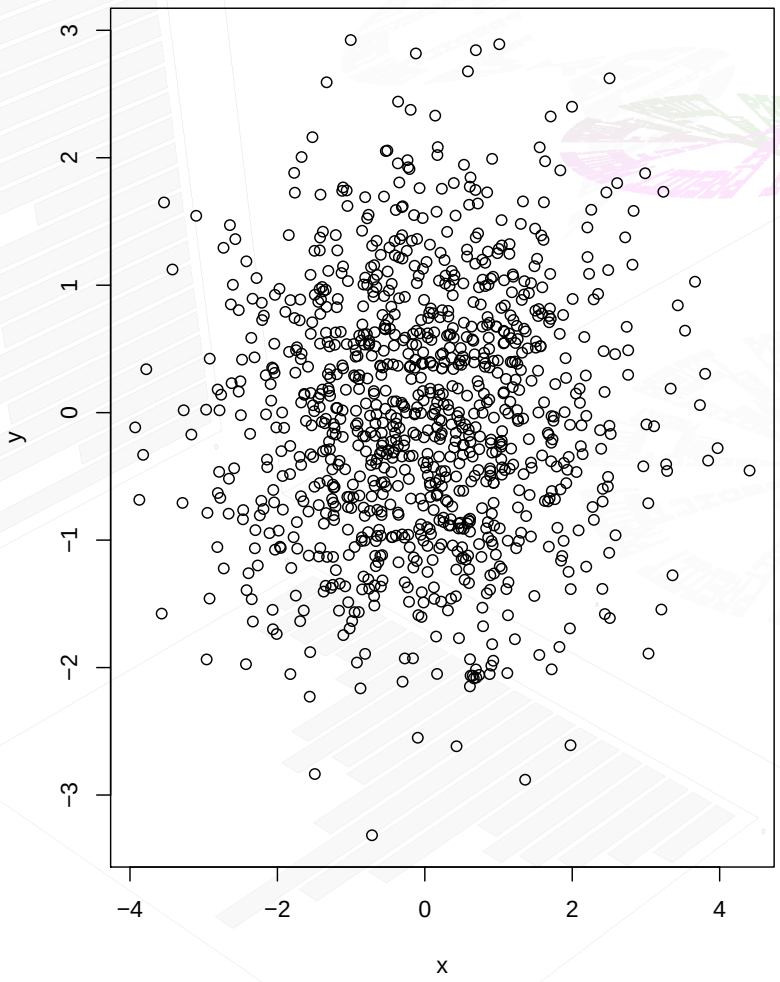
Spearman rank correlation ρ_S

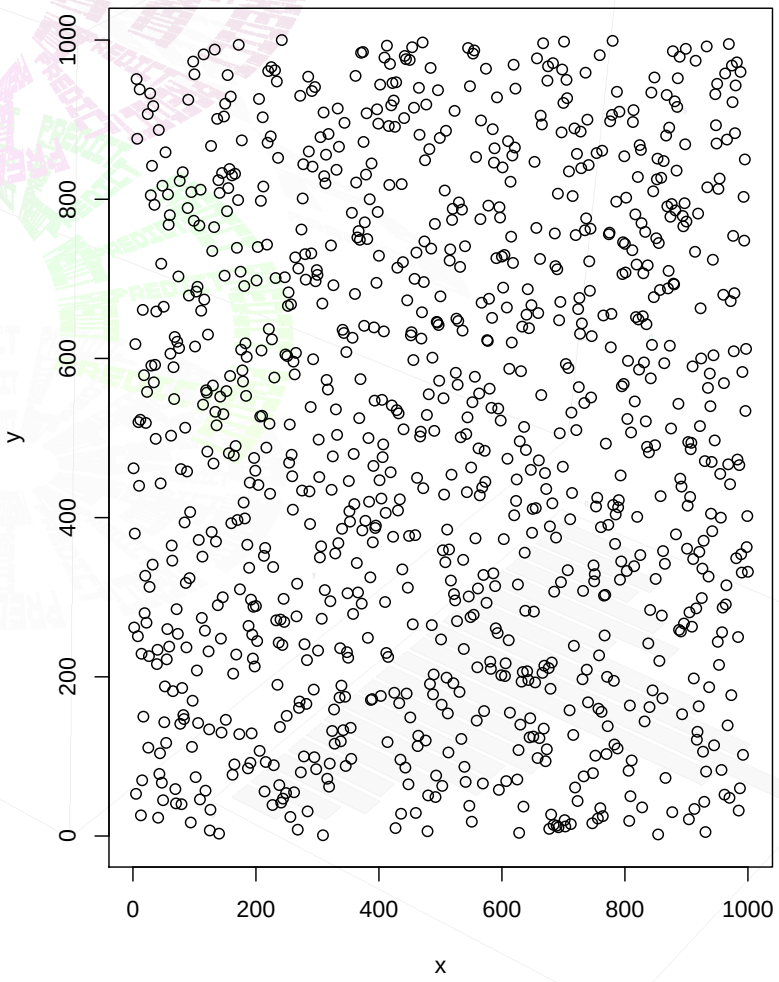



```
Dazu<-data.frame(x=c(10,10.3),y=c
    (2,2.4))
A1<-rbind(A,Dazu)
plot(A1)
cor(A1$x,A1$y)
cor(A1$x,A1$y,method = "spearman")
```

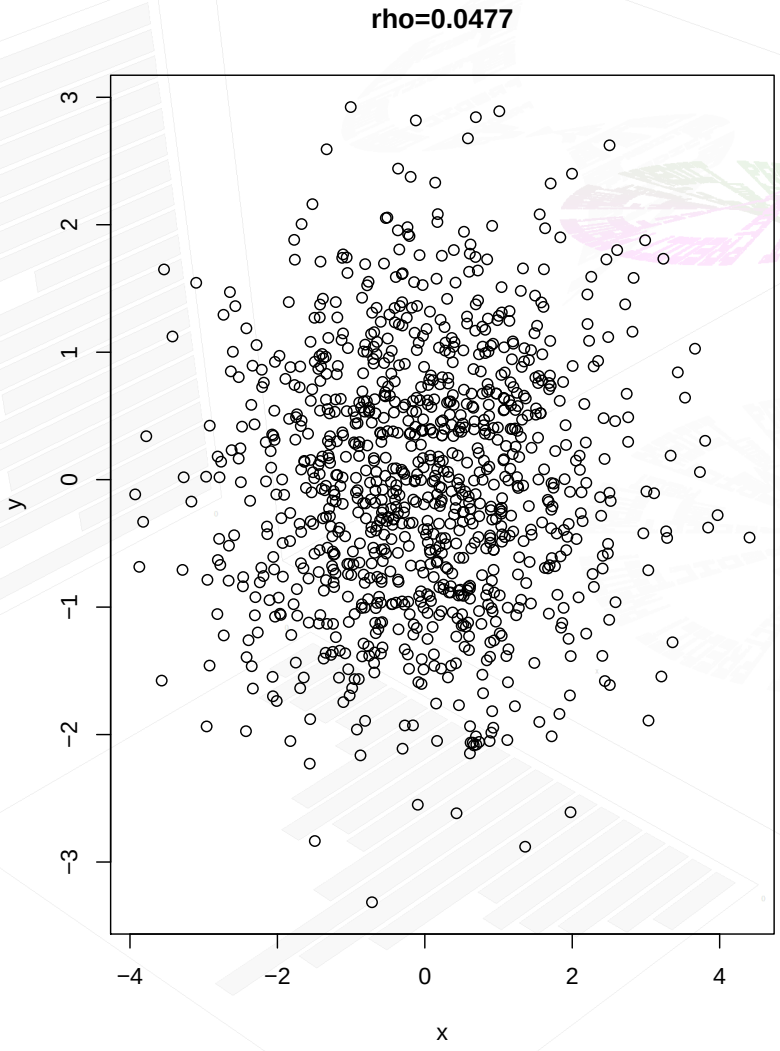
- Which is more influenced by the two new points?
- We get $\rho = 0.8187617$ (before $\rho = 0.97464$)
- Moreover $\rho_S = 0.9349794$ (before $\rho_S = 0.9633945$)
- ρ is less robust against outliers than ρ_S
- Rank-based quantities are generally robust

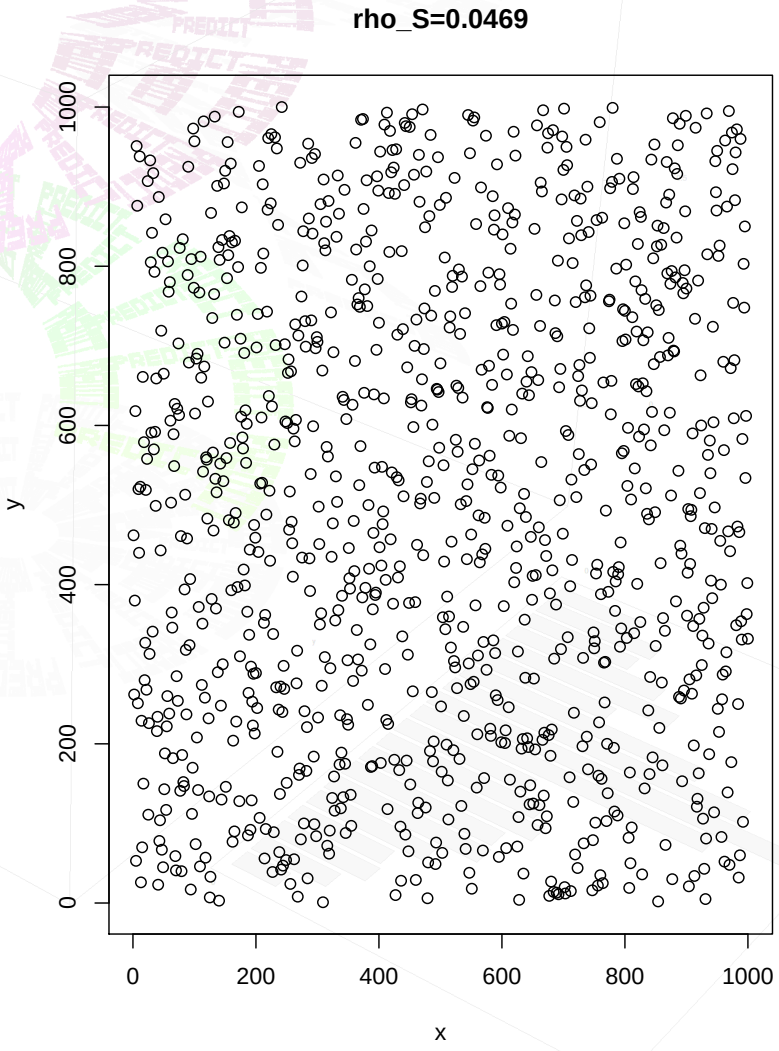
Some examples and exercises



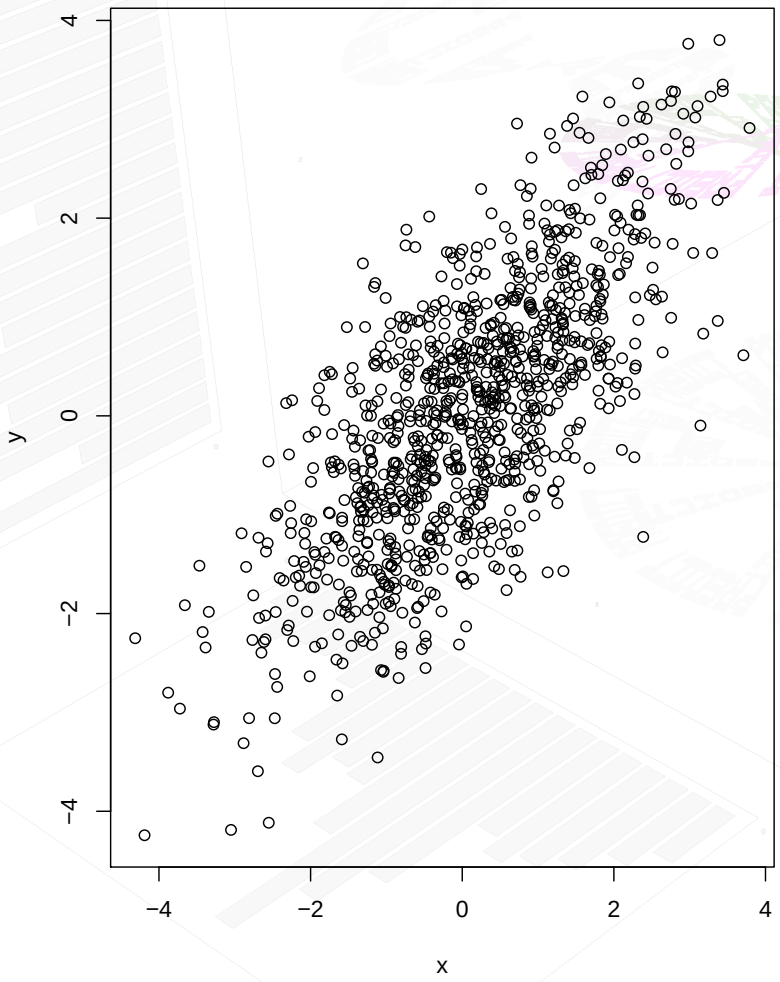


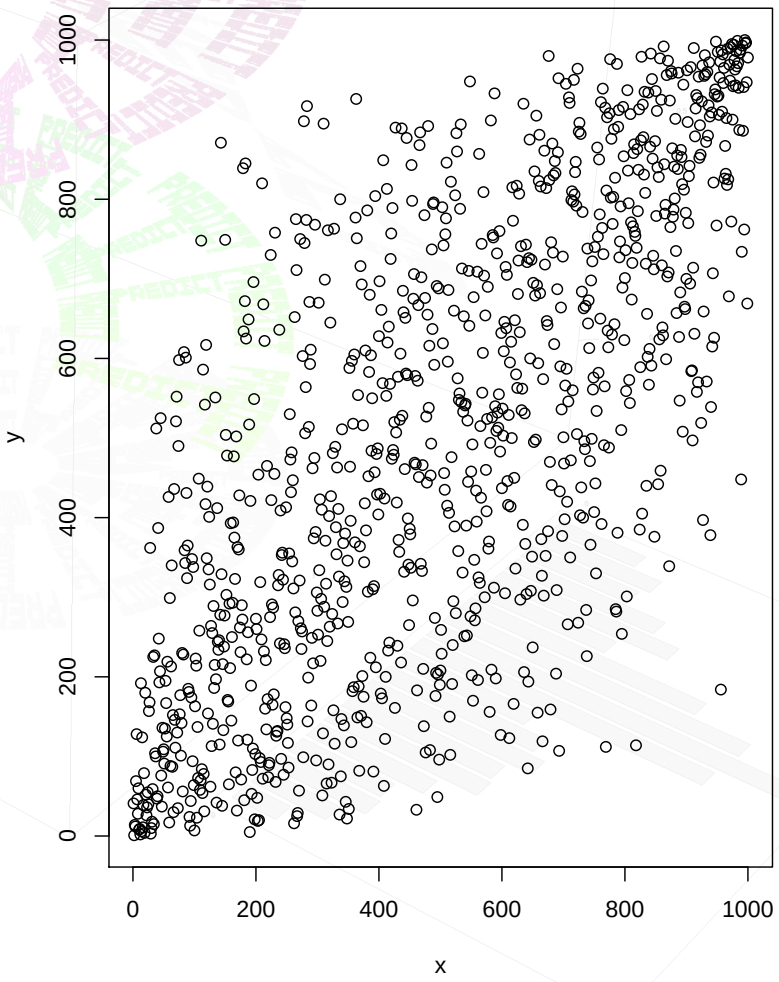
Some examples and exercises



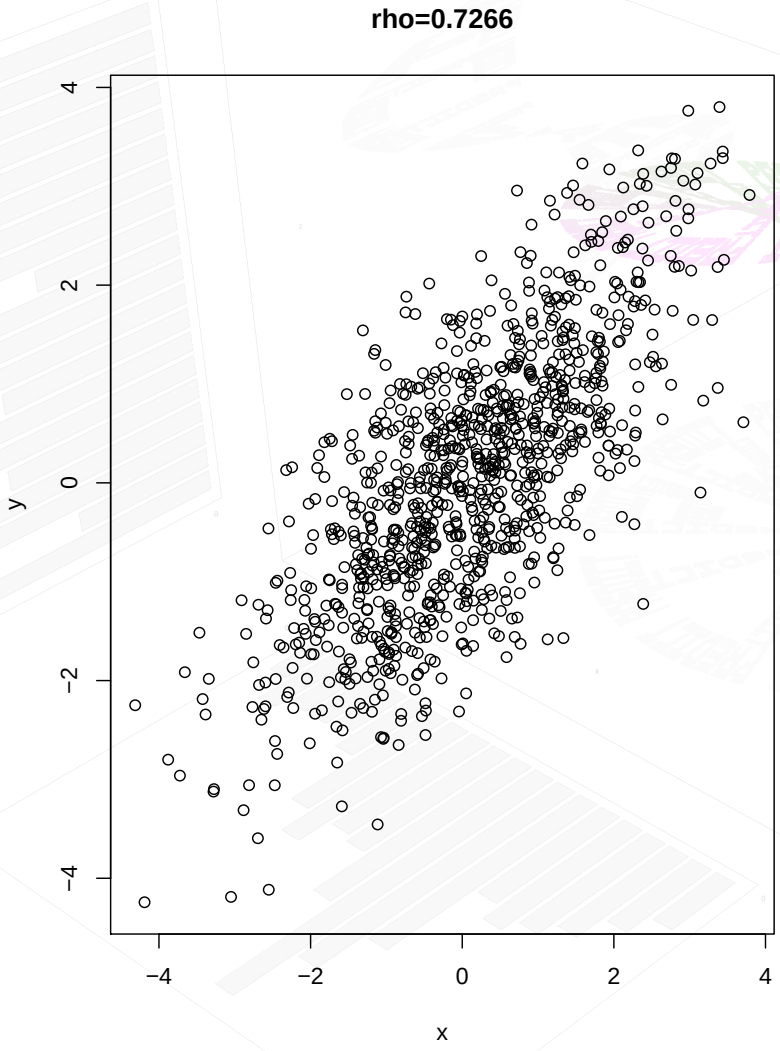


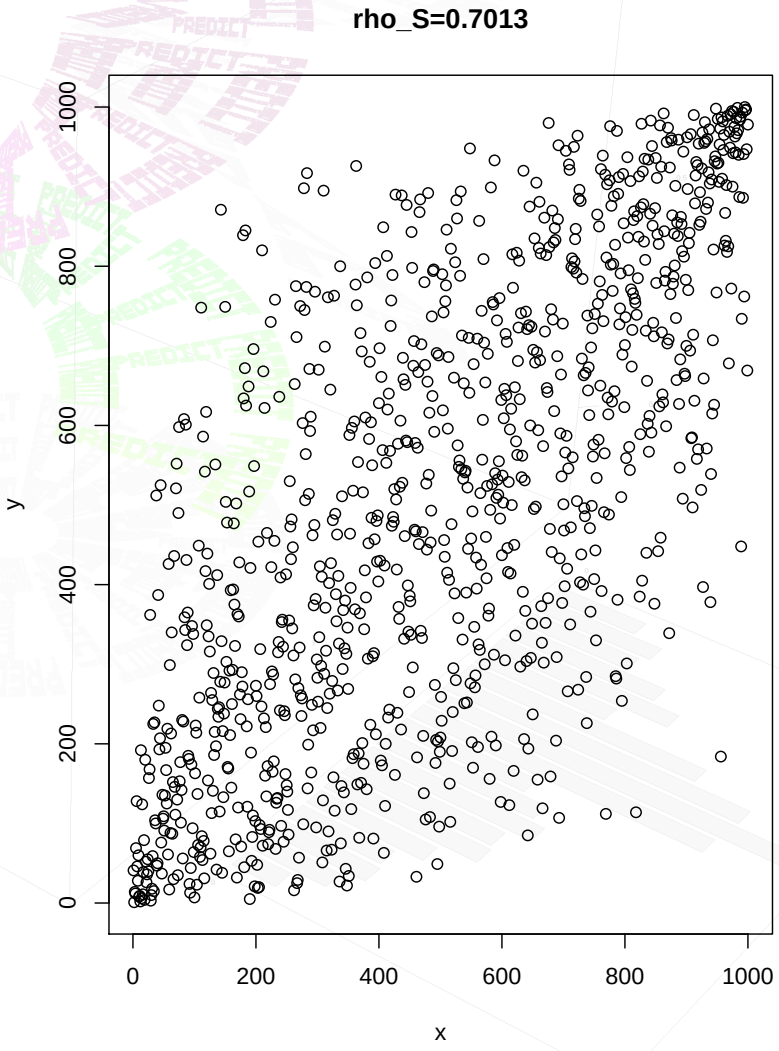
Some examples and exercises



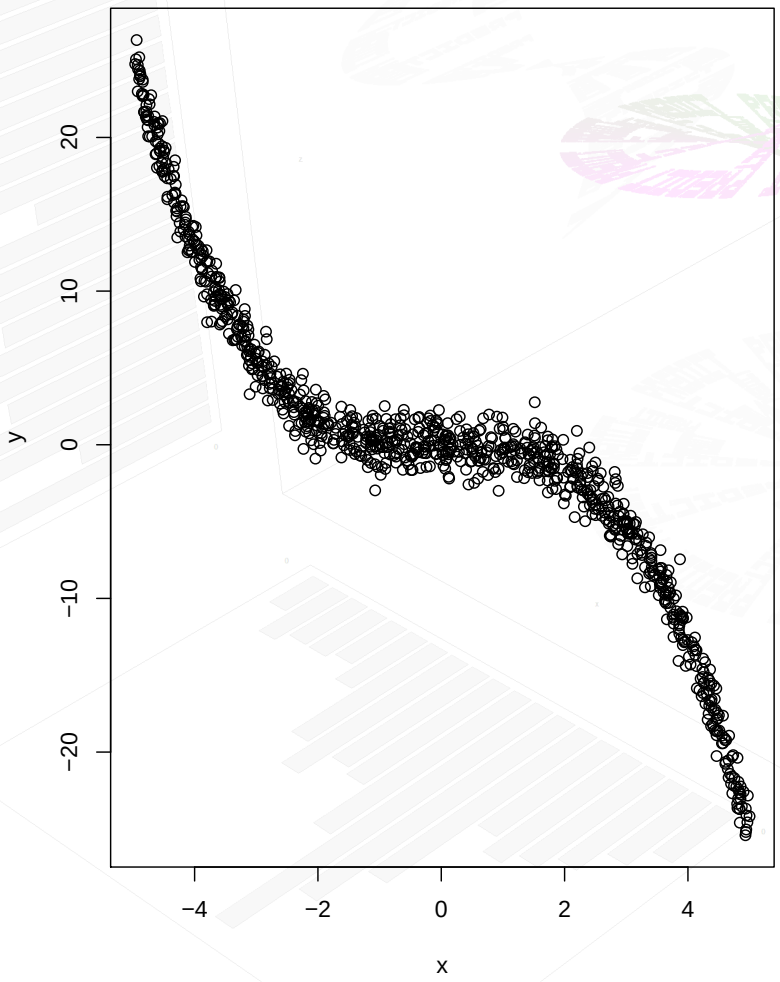


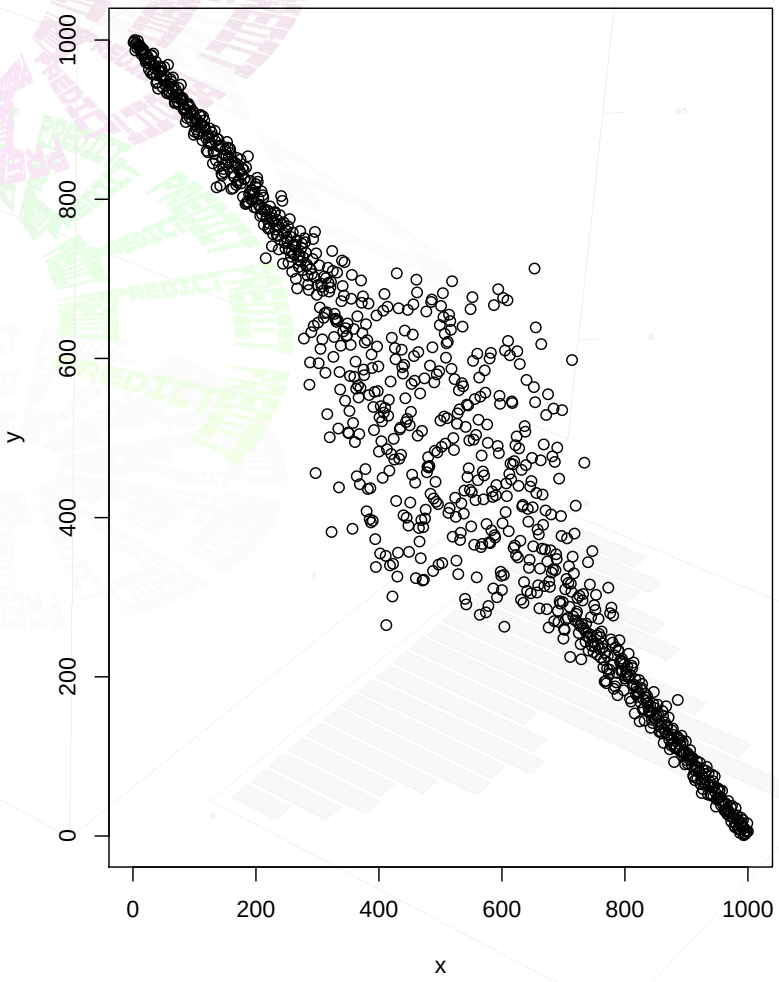
Some examples and exercises



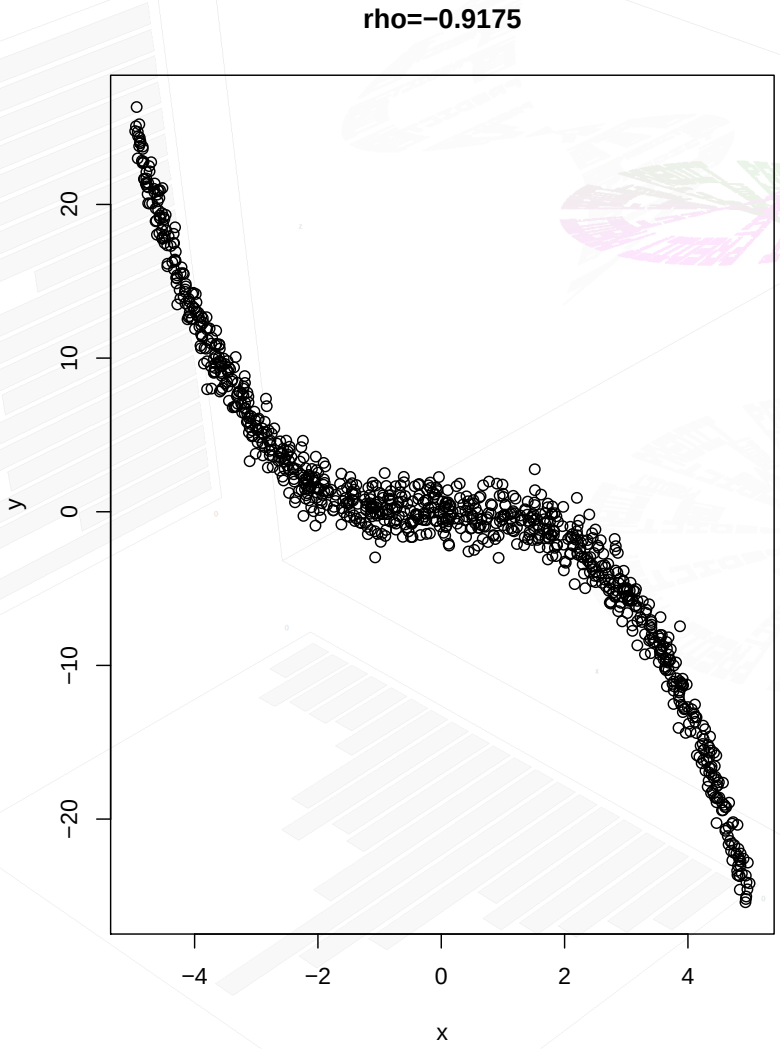


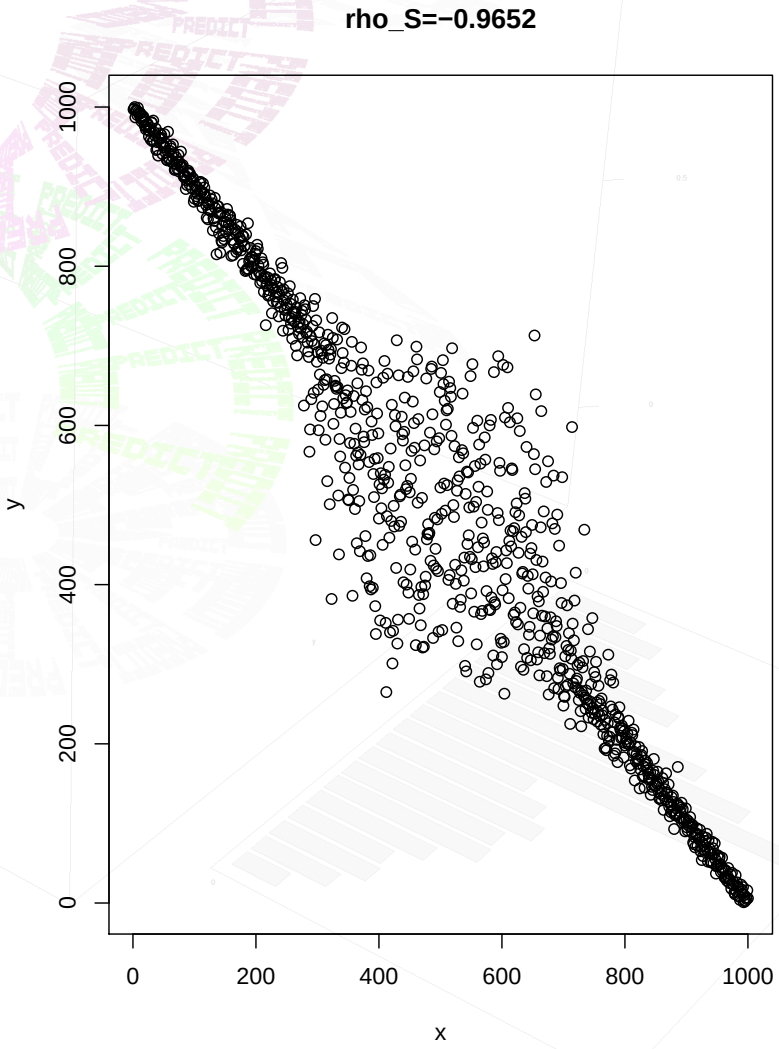
Some examples and exercises



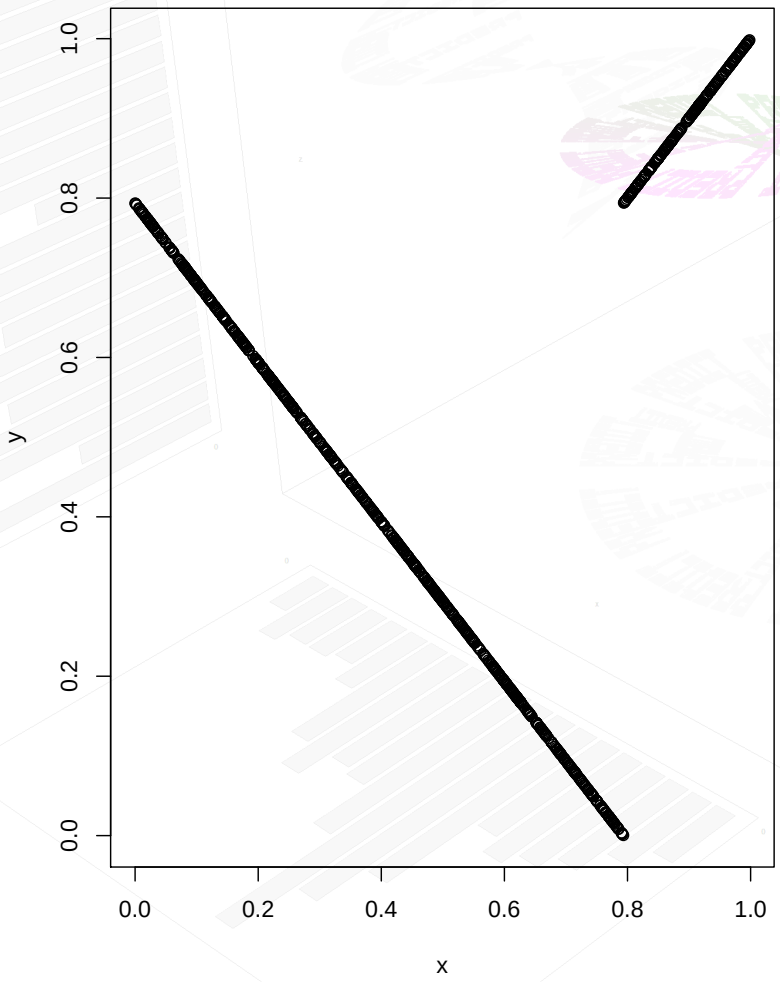


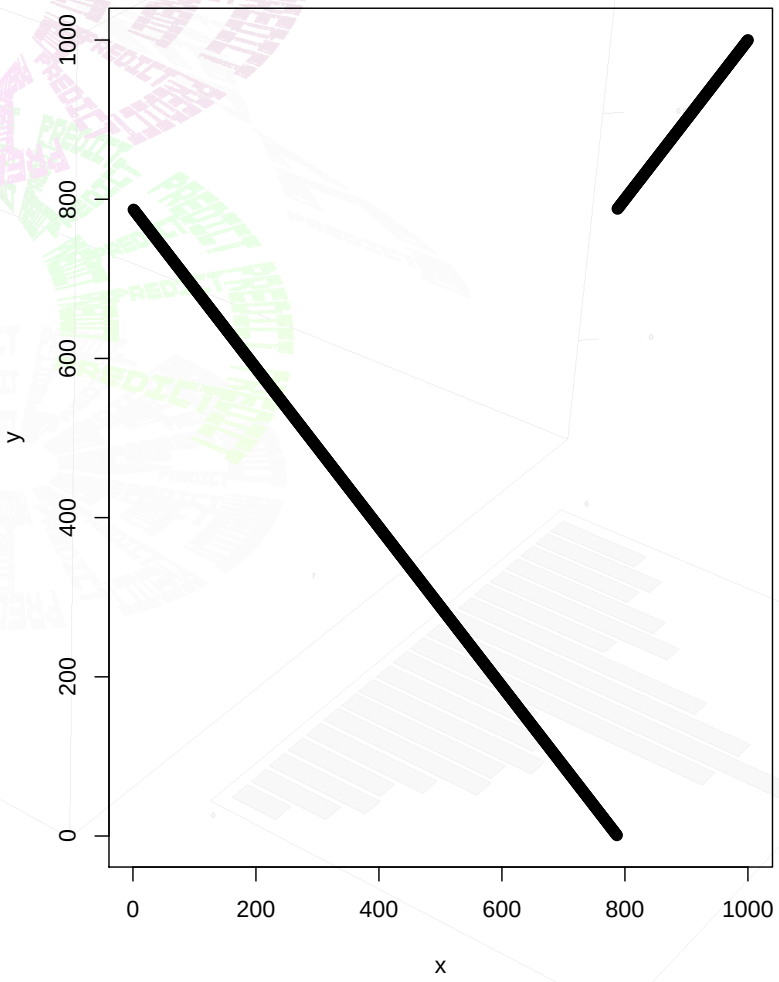
Some examples and exercises



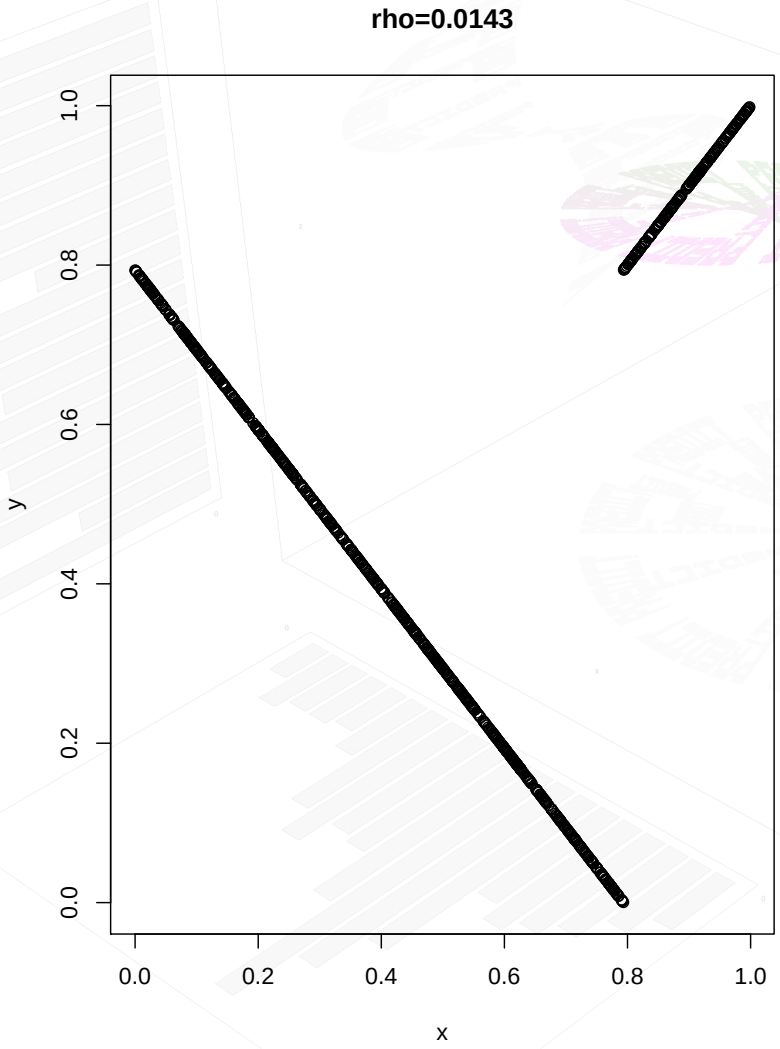


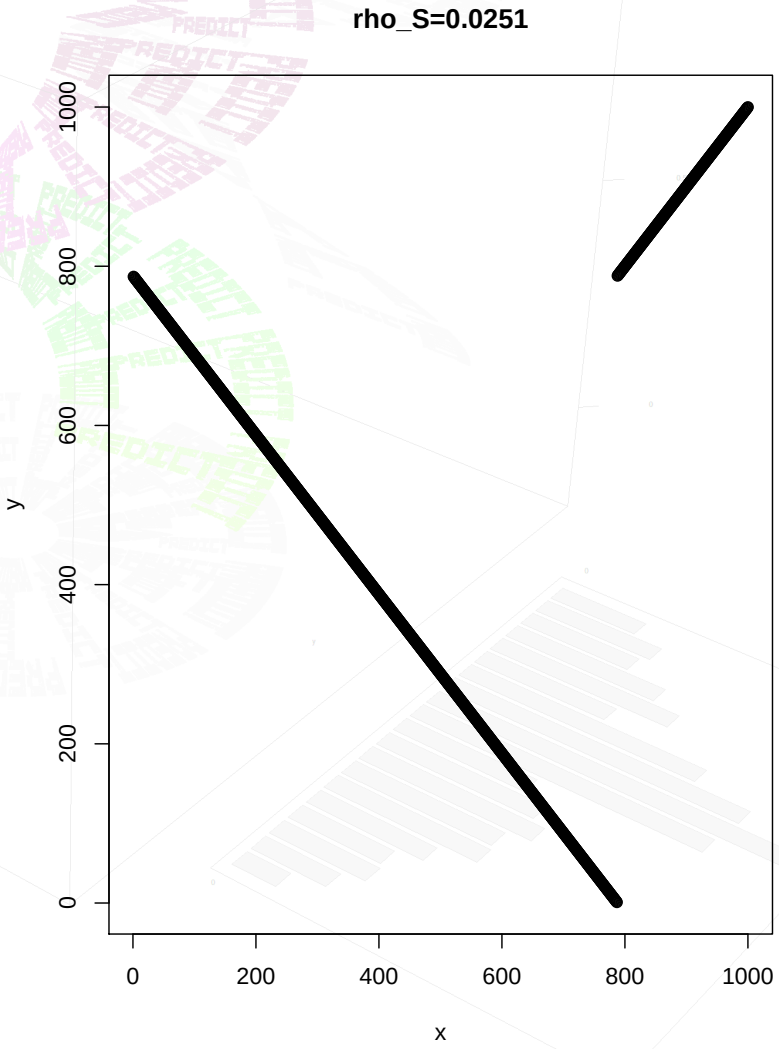
Some examples and exercises



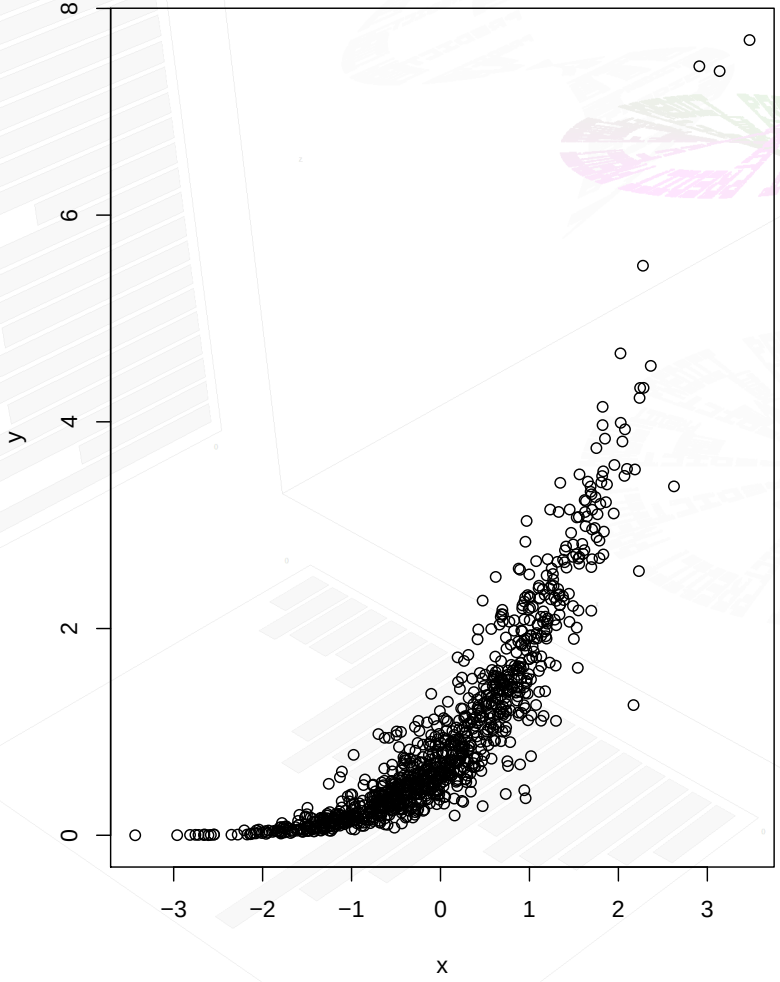


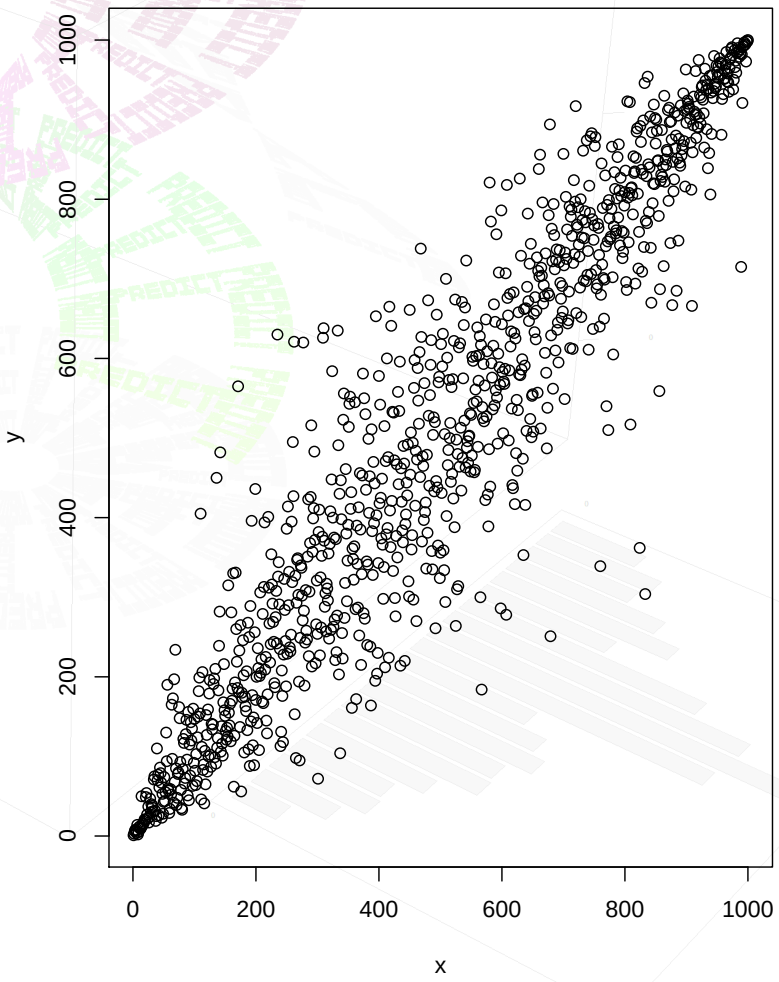
Some examples and exercises



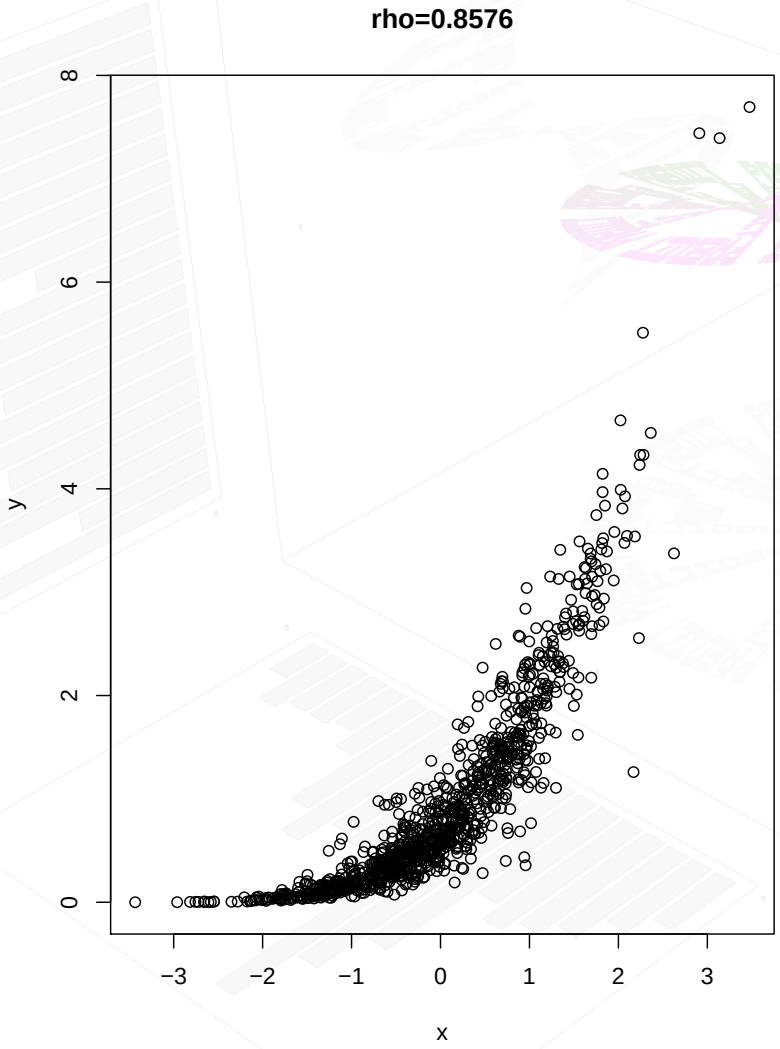


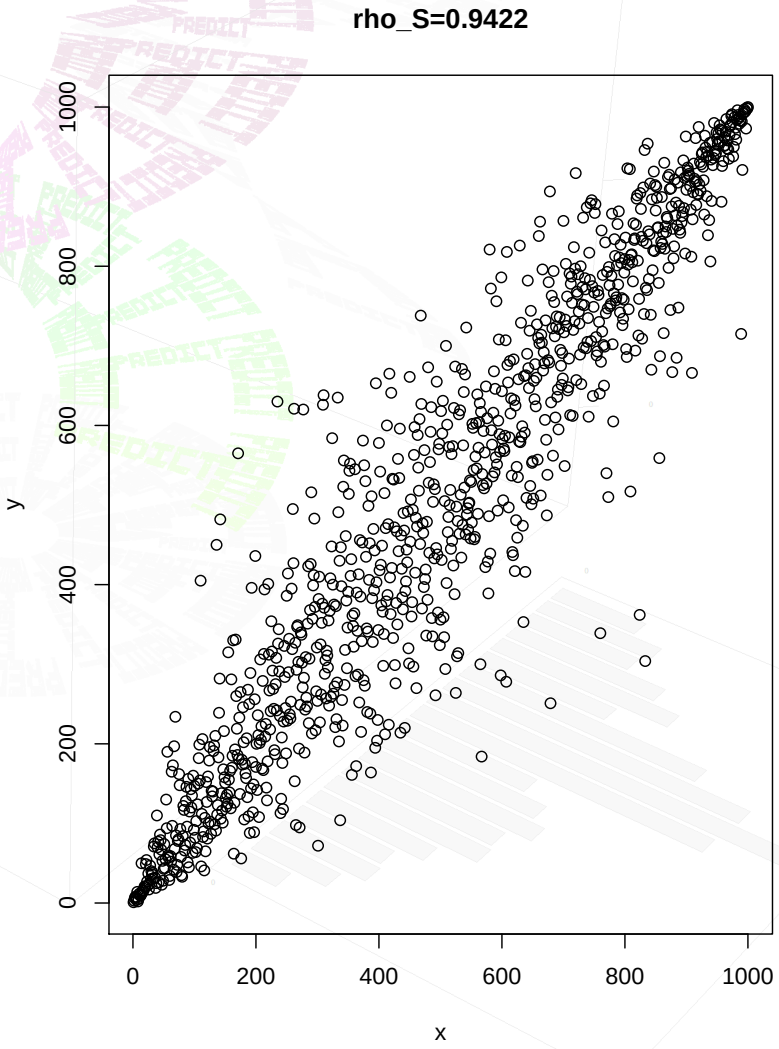
Some examples and exercises





Some examples and exercises





Solve Exercise 01 and Exercise 02 in the R-script StADS201805.R.

Exercise 03: Can you find a sample $(x_1, y_1), \ldots, (x_n, y_n)$ for which the Pearson correlation ρ and the Spearman correlation ρ_S have different sign?
Hint: Running simulations is never a bad idea; simulate five x-coordinates and five y-coordinates from $\mathcal{U}(0, 1)$ and calculate ρ and ρ_S; repeat several times

Solve Exercise 04 in the R-script StADS201805.R.

Known:

- We know that there is a relationship between quantities X and Y of the following form:

$$Y = r(X) + \varepsilon \tag{1}$$

- r is an **unknown** function and ε is a random error fulfilling $\mathbb{E}(\varepsilon) = 0$.
- Usually we also assume that ε is not influenced by X (might be a too restrictive condition in various situations).
- We call X the **predictor** and Y the **response**.

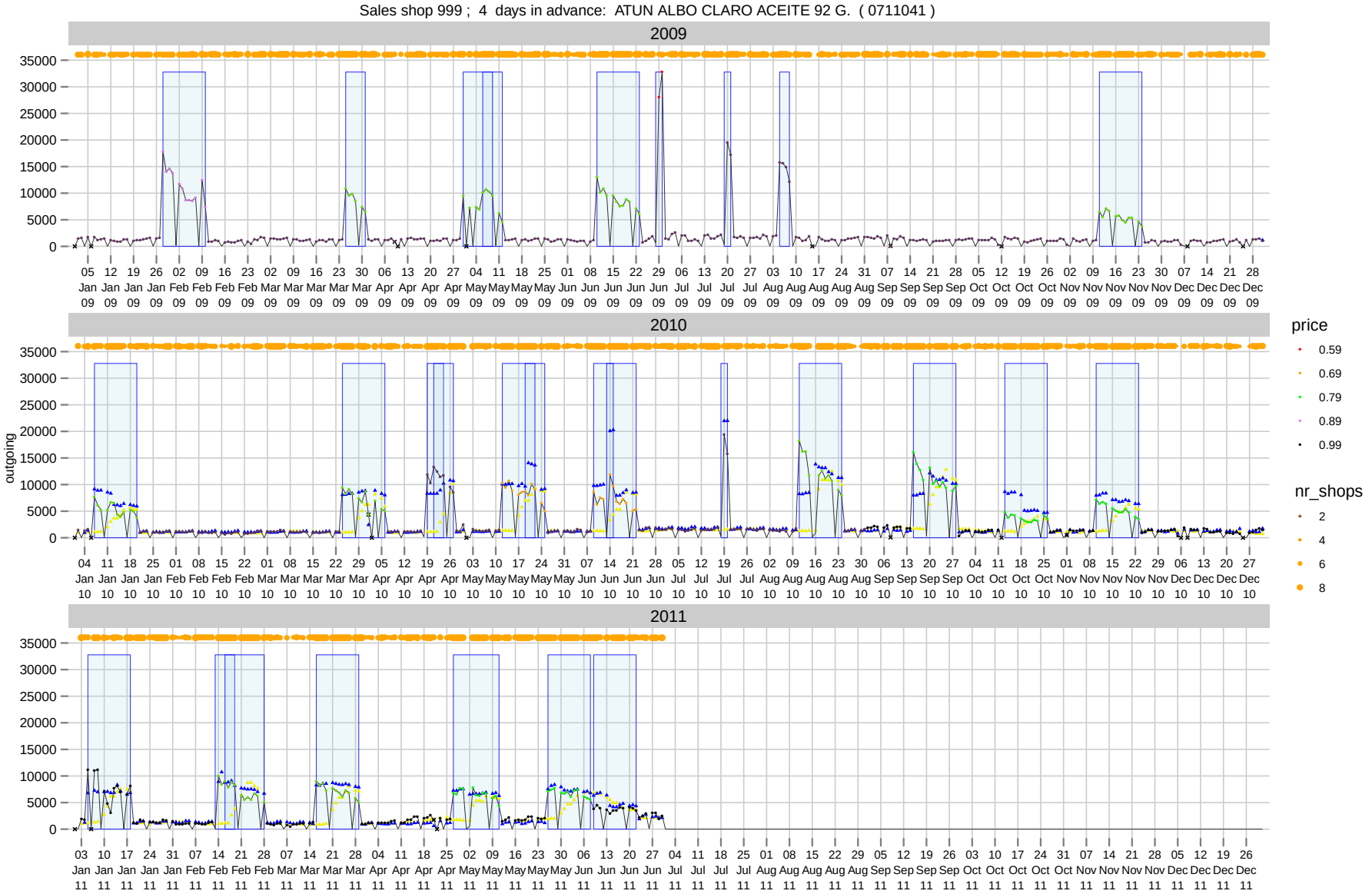
Wanted:

- Based on observations $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$ from (1) we want to determine/estimate the function r (why?).
- If we have a good estimator $\hat{r}$ of r then we can predict Y for arbitrary values of X by considering $\hat{r}(X)$.

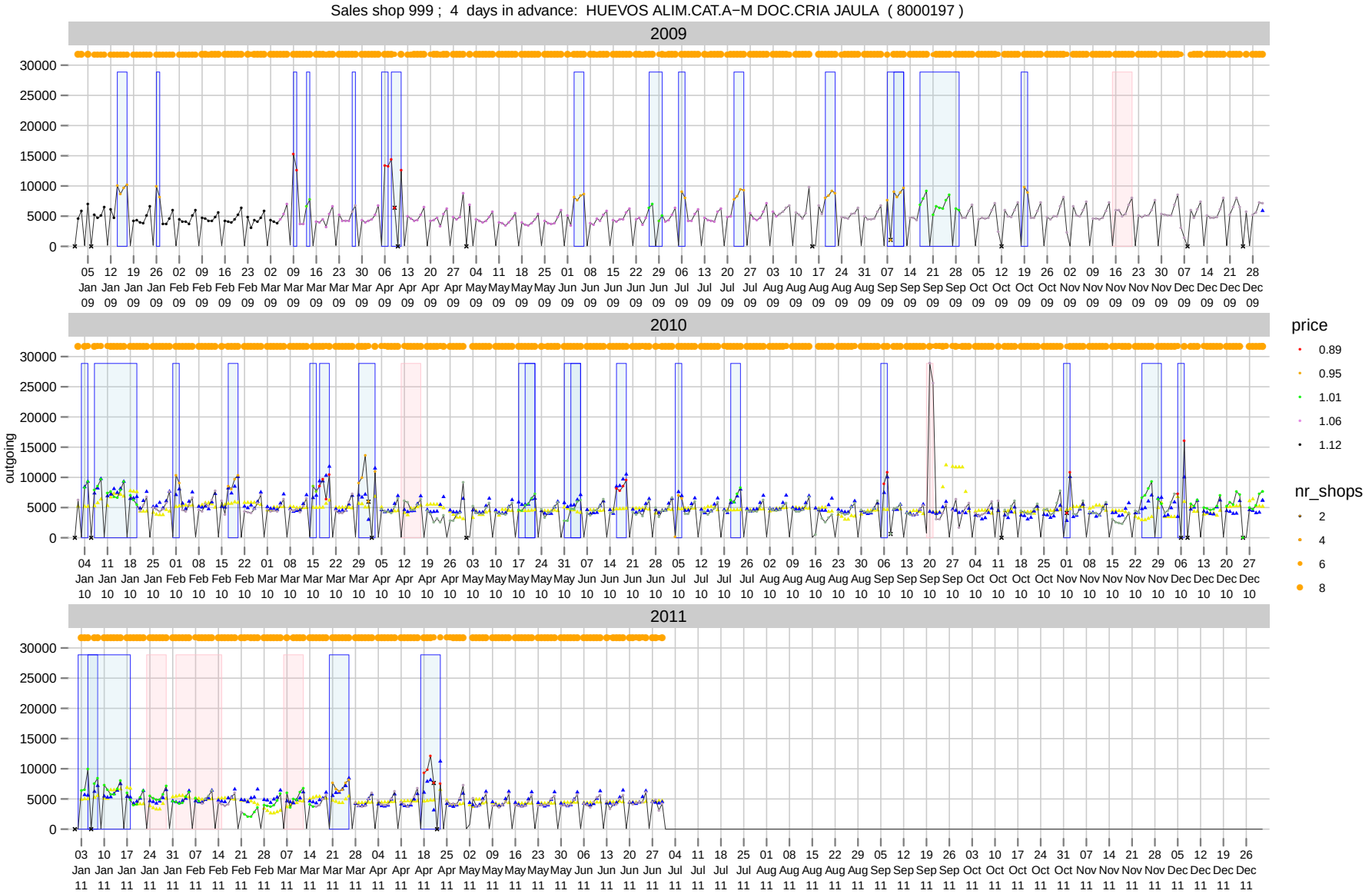
Example (Offer optimization in supermarkets)

- A supermarket chain wants to optimize their offers.
- If the price is only reduced by 5% then the sales numbers will only go up a bit.
- If the price is reduced by 50% then the sales numbers will go up a lot but the company might earn less because the margin is too small.
- Objective: Determine the optimal price reduction in the sense that the supermarket's profit is maximal.
- X...price reduction (absolute or percentage) of a certain product.
- Y...net earnings (based on this product).
- $Y = r(X) + \varepsilon$.
- What do you think: Is the model solely based on price reduction as predictor good?
- Which other predictors would you choose?

A real-life example



A real-life example



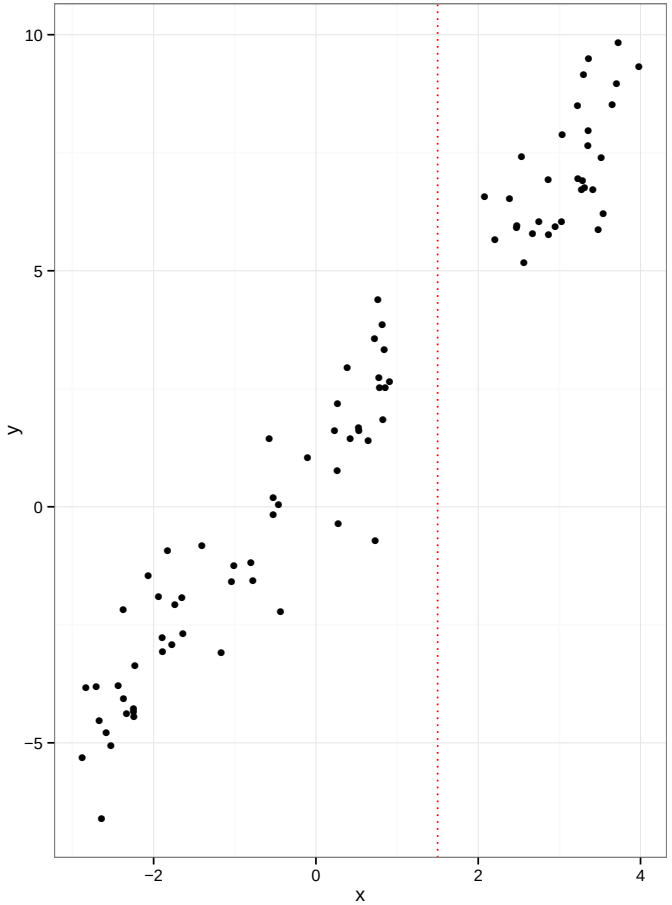


Figure: Prediction at the point $x = 1.5$?

- The graphic depicts measurements $(x_1, y_1), \ldots, (x_n, y_n)$.
- It is known that the data comes from the following linear model

$$Y = \underbrace{aX + b}_{r(X)} + \varepsilon.$$

- In other words: $y_i = ax_i + b + \varepsilon_i$ for $i \in \{1, \ldots, n\}$.
- ε_i...samples of the random error ε fulfilling $\mathbb{E}(\varepsilon) = 0$ that do not influence each other and are not influenced by x_i.
- No normality assumption for ε!
- Wanted: Forecast the y-value at the point $x = 1.5$.

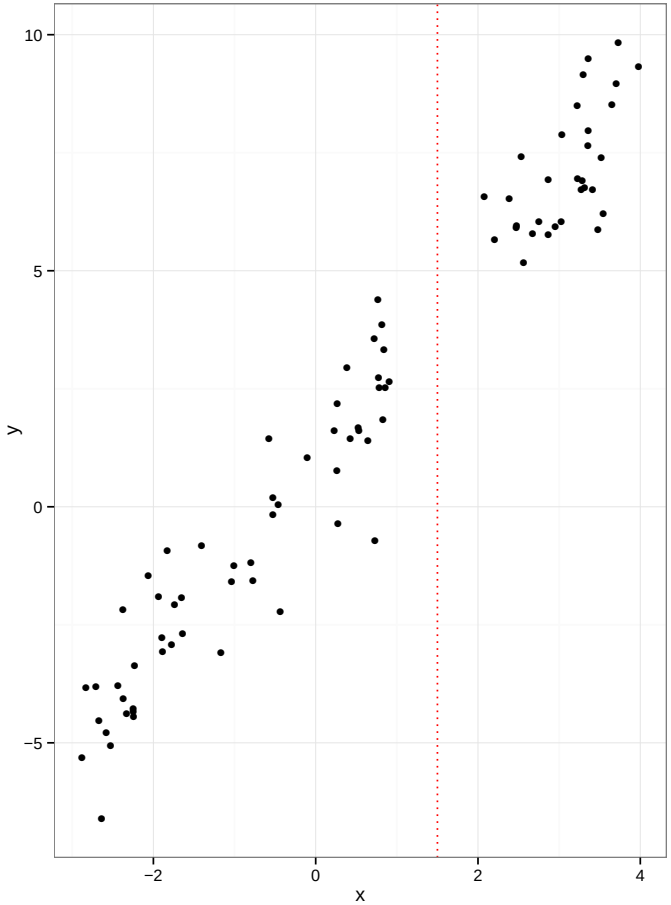
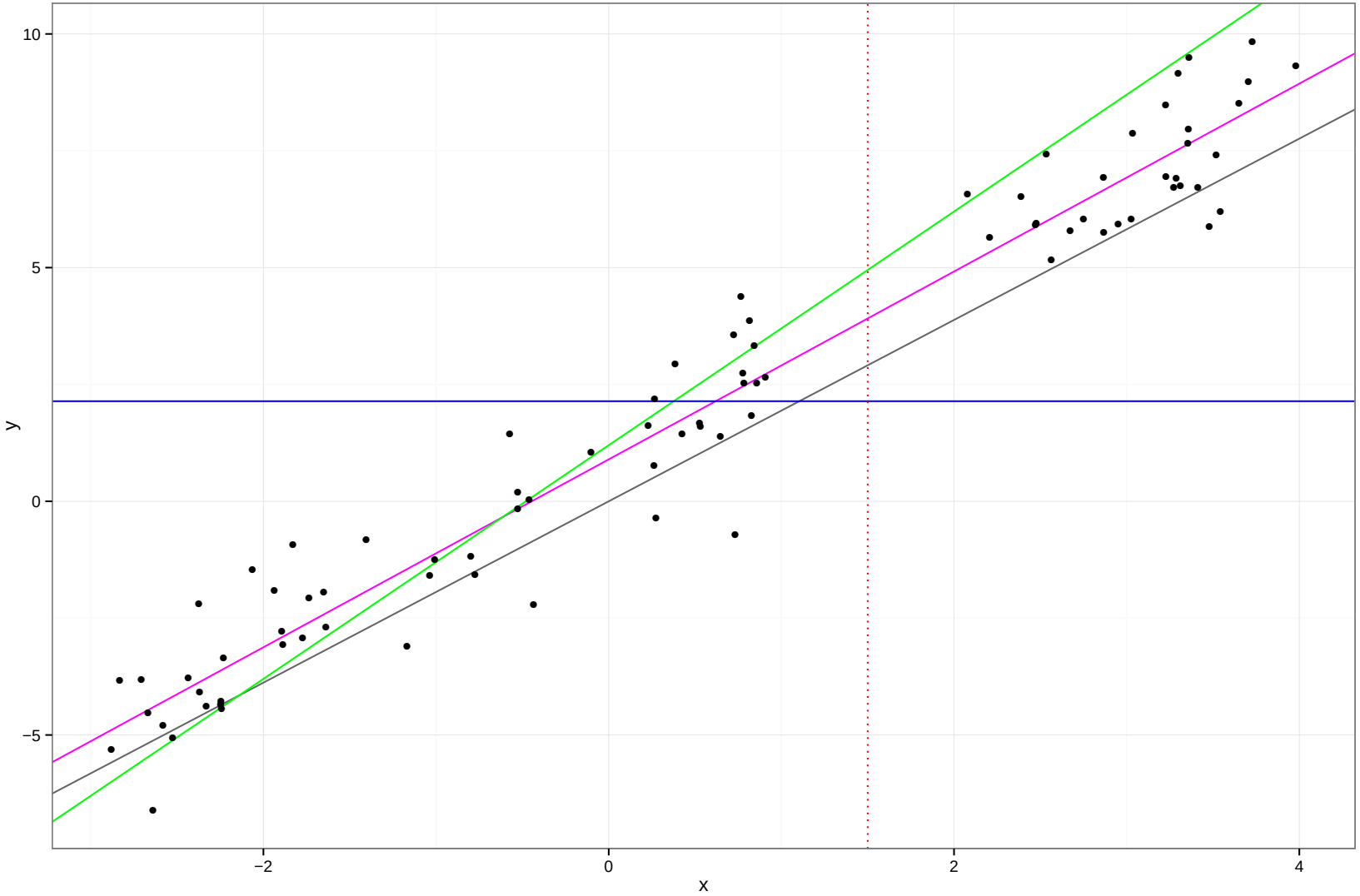


Figure: Prediction at the point $x = 1.5$?

- How would you predict the value at the point $x = 1.5$?
- Problem: We do not know the parameters a and b.
- Choose a and b in such a way that the straight line $y = ax + b$ fits the data in the best possible way.
- Denote the optimal values by $\hat{a}$ and $\hat{b}$.
- Given $\hat{a}$ and $\hat{b}$, predict $\hat{y} = \hat{a}\,1.5 + \hat{b}$.
- Which of the following straight lines fits best?

Linear regression



- Choose those values for $\tilde{a}$ and $\tilde{b}$ that minimize the prediction errors at the points in the sample.
- Choosing $\tilde{a}$ and $\tilde{b}$ as parameters we would forecast $\tilde{a}x_i + \tilde{b}$ for x_i.
- The error r_i we make is $r_i = y_i - (\tilde{a}x_i + \tilde{b}) = y_i - \tilde{a}x_i - \tilde{b}$. Plot r_i
- The sum of all squared errors is given by

$$F(\tilde{a}, \tilde{b}) := \sum_{i=1}^{n} \left(y_i - \tilde{a}x_i - \tilde{b}\right)^2 \tag{2}$$

- Choose $\tilde{a}$ and $\tilde{b}$ in such a way that $F(\tilde{a}, \tilde{b})$ is minimal.
- Analytic calculation yields the following optimal values

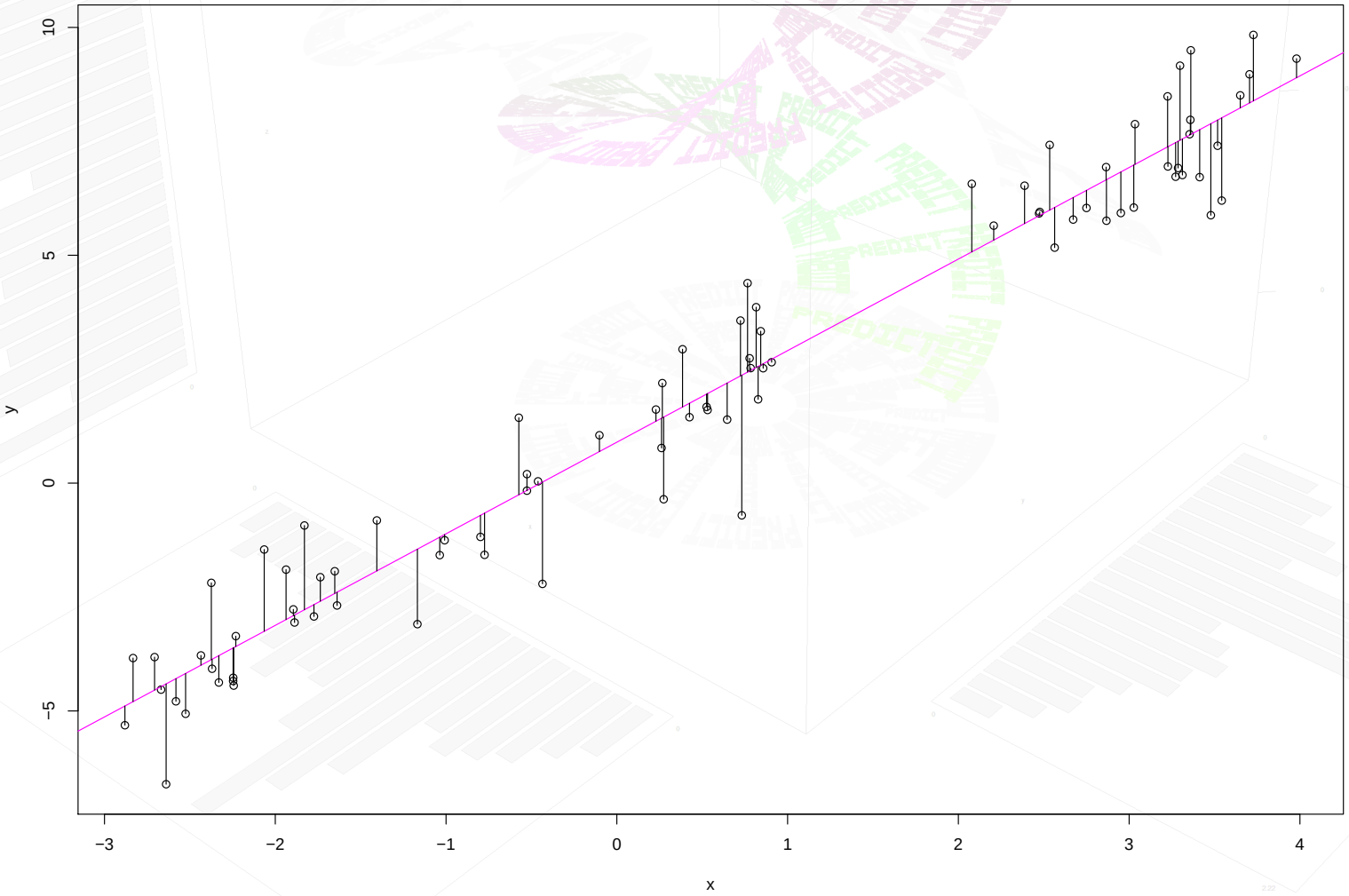
$$\hat{a} = \frac{\sum_{i=1}^{n}(x_i - \overline{x}_n)(y_i - \overline{y}_n)}{\sum_{i=1}^{n}(x_i - \overline{x}_n)^2} = \frac{s_{xy}}{s_x^2} \tag{3}$$

$$\hat{b} = \overline{y}_n - \hat{a}\,\overline{x}_n. \tag{4}$$

- For our given sample we get $\hat{a} = 2.010$ and $\hat{b} = 0.897$.
- The forecast at the point $x = 1.5$ therefore is $y = 2.01 \cdot 1.5 + 0.897 = 3.912$

Linear regression

Back



- Before fitting linear models in R some additional observations:
- The estimate slope $\hat{a} = \frac{s_{xy}}{s_x^2}$ looks a bit like the Pearson correlation $\rho = \frac{s_{xy}}{s_x s_y}$.
- Using both expressions we get

$$\hat{a} = \rho \frac{s_y}{s_x}$$

- Increasing x by one standard deviation s_x increases y by ρ standard deviations s_y, in fact

$$\hat{r}(x + s_x) = \hat{a}(x + s_x) + \hat{b} = \underbrace{\hat{a}x + \hat{b}}_{y} + \hat{a}s_x = y + \rho \frac{s_y}{s_x} s_x = y + \rho s_y .$$

- How do we quantify if our optimal model offers a good explanation of the model?

- A natural idea is the *coefficient of determination* R^2
- Easy to show:

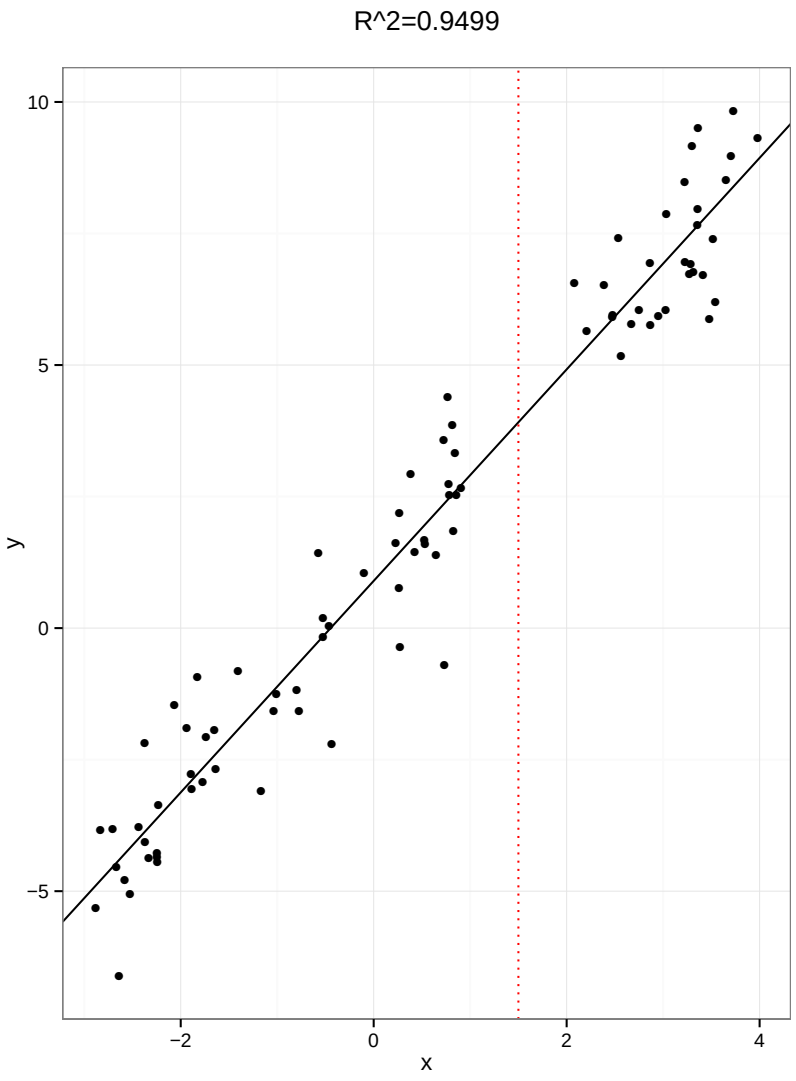
$$\sum_{i=1}^{n}(y_i - \overline{y}_n)^2 = \sum_{i=1}^{n}\underbrace{(y_i - \hat{y}_i)^2}_{r_i^2} + \sum_{i=1}^{n}(\hat{y}_i - \overline{\hat{y}}_n)^2$$

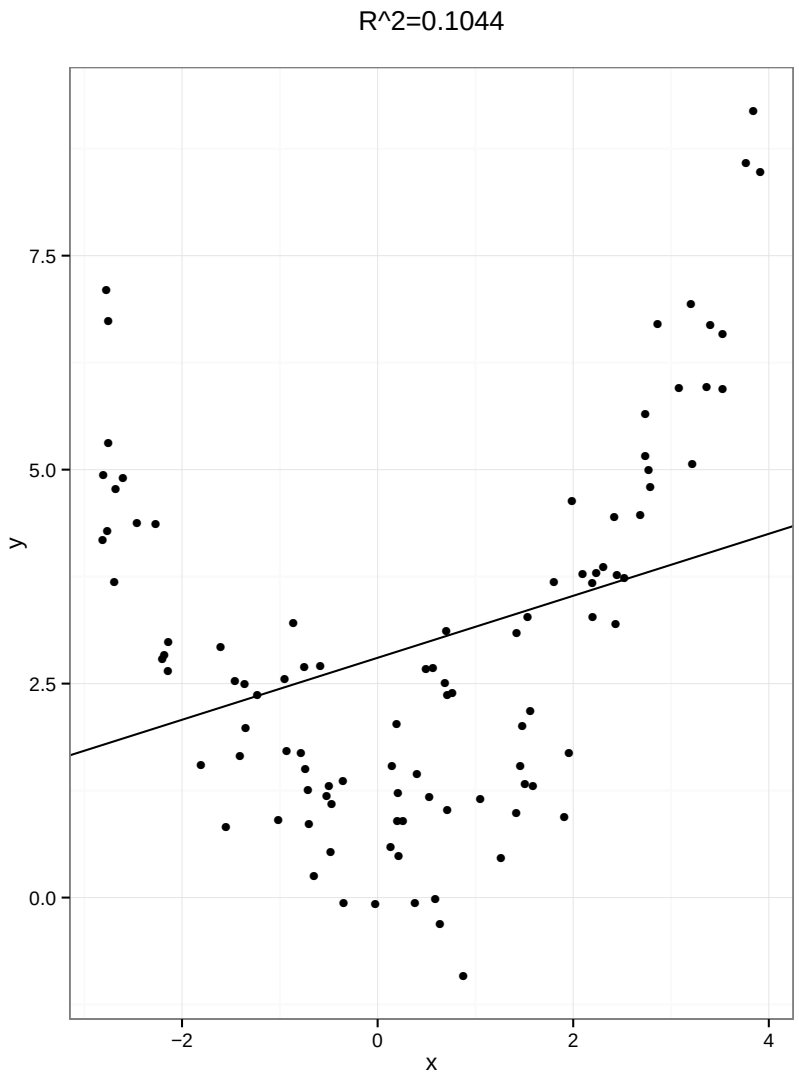
- Variance of $y_1, \ldots, y_n$ equals the variance of the residuals plus the variance of the forecasts $\hat{y}_1, \ldots, \hat{y}_n$.
- Calculate

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \overline{y}_n)^2} = \frac{\sum_{i=1}^{n}(\hat{y}_i - \overline{\hat{y}}_n)^2}{\sum_{i=1}^{n}(y_i - \overline{y}_n)^2} \tag{5}$$

- R^2 is the portion of y-variance explained by the model.

Linear regression





Properties of R^2:

- We have $0 \leq R^2 \leq 1$.
- The higher R^2 the higher the percentage of variance explained by the model.
- If R^2 is close to 1 then the model explains the data very well.
- If R^2 is close to 0 the model does not help much to explain the data.
- There should be a strong interrelation between R^2 and the correlation ρ of the original sample $(x_1, y_1), \ldots, (x_n, y_n)$...
- Calculations in R will make this clear.

```
file <- url("http://www.trutschnig.net/geo_reg1.RData")
load(file)
head(geo_reg1)
A<-geo_reg1

model<-lm(data=A,y~x)      #use whatever name you want instead of model
summary(model)
```

- yields

```

Call:
  lm(formula = y ~ x, data = A)

Residuals:
   Min      1Q  Median      3Q     Max
-3.07477 -0.63681 -0.03544  0.70030  1.95308
```

- and

```
Coefficients:
  Estimate   Std. Error t value   Pr(>|t|)
(Intercept)  0.89704    0.11406   7.865   1.13e-11 ***
  x          2.00965    0.05035  39.917   < 2e-16 ***
  ---
  Signif. codes:  0   ***   0.001   **   0.01   *   0.05   .
          0.1        1

Residual standard error: 1.03 on 84 degrees of freedom
Multiple R-squared:  0.9499, Adjusted R-squared:  0.9493
F-statistic:  1593 on 1 and 84 DF,  p-value: < 2.2e-16
```

- Calculate the prediction for $x = 1.5$

```
ND<-data.frame(x=c(1.5))
p<-predict(model,new=ND)
p
3.91152
```

Exercise 05:

- Load the dataset geo_reg1.RData (see R-Code, end of part 01 in linear regression).
- Produce a scatterplot of the data including the regression line.
- Add the values of the estimated parameters $\hat{a}$ and $\hat{b}$ in the title of the plot.
- Produce a boxplots of the residuals $r_1, \ldots, r_n$.
- Calculate ρ and ρ_S of the data.
- Forecast $r(x)$ for $x \in \{0, 0.1, 0.2, \ldots, 0.9, 1\}$.

Exercise 06:

- The datset 'brainhead.txt' (see R-Code)contains Brain weight (grams) and head size (cm3) for 237 adults.
- Fit a linear regression with 'weight' as response and 'cm3' as explanatory variable.
- Plot the data together with the regression line.
- Calculate the corresponding R^2.
- Calculate the biggest ten residuals ('biggest' in the sense of absolute value) - how many men and how many woman are in the 'top-ten'?

Summary @univariate linear regression

- $(x_1, y_1), \ldots, (x_n, y_n)$ are observations from the model $Y = aX + b + \varepsilon$.
- Thereby ε was a random error fulfilling $\mathbb{E}(\varepsilon) = 0$; set $\sigma^2 = \mathbb{V}(\varepsilon)$.
- In other words: $y_i = ax_i + b + \varepsilon_i$ for every $i \in \{1, \ldots, n\}$.
- Using least squares we got the following estimators $\hat{a}$ of a and $\hat{b}$ of b

$$\hat{a} = \frac{\sum_{i=1}^{n}(x_i - \overline{x}_n)(y_i - \overline{y}_n)}{\sum_{i=1}^{n}(x_i - \overline{x}_n)^2} = \frac{s_{xy}}{s_x^2} \tag{6}$$

$$\hat{b} = \overline{y}_n - \hat{a}\,\overline{x}_n. \tag{7}$$

- We hope to get $\hat{a} \approx a$ and $\hat{b} \approx b$, i.e. we hope that the estimates are close to the true values.
- Will this always be the case?
- When can we expect to get good estimates?

@Performance

```
#one simulation
a<-2;b<-1
n<-100
x<-runif(n,-3,4)        #generate random x values
error<-rnorm(n,0,1)     #error from normal distribution N(0,1)
y<-a*x+b+error
A<-data.frame(x=x,y=y)
plot(A)
model<-lm(data=A,y~x)
abline(model)
summary(model)
```

- yields

```

Call:
lm(formula = y ~ x, data = A)

Residuals:
     Min       1Q   Median       3Q      Max
-2.20647 -0.66814 -0.09888  0.77627  1.95348
```

```
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.76146    0.10149   7.503 2.87e-11 ***
x            2.09902    0.04686  44.790  < 2e-16 ***
---
Signif. codes:  0  ***  0.001  **  0.01  *  0.05  .  0.1    1

Residual standard error: 0.9631 on 98 degrees of freedom
Multiple R-squared:  0.9534, Adjusted R-squared:  0.9529
F-statistic:  2006 on 1 and 98 DF,  p-value: < 2.2e-16
```

```
sum(model$residuals^2)/(n-2)
```

- yields

```

[1] 0.9275251
```

@Performance

```
#several runs
R<-1000
E<-data.frame(a=rep(0,R),b=rep(0,R))

a<-2;b<-1
n<-100
for(i in 1:R){
  x<-runif(n,-3,4)    #generate random x values
  error<-rnorm(n,0,1)
  y<-a*x+b+error
  A<-data.frame(x=x,y=y)
  model<-lm(data=A,y~x)
  E[i,]<-as.numeric(coefficients(model))[2:1]
}
```

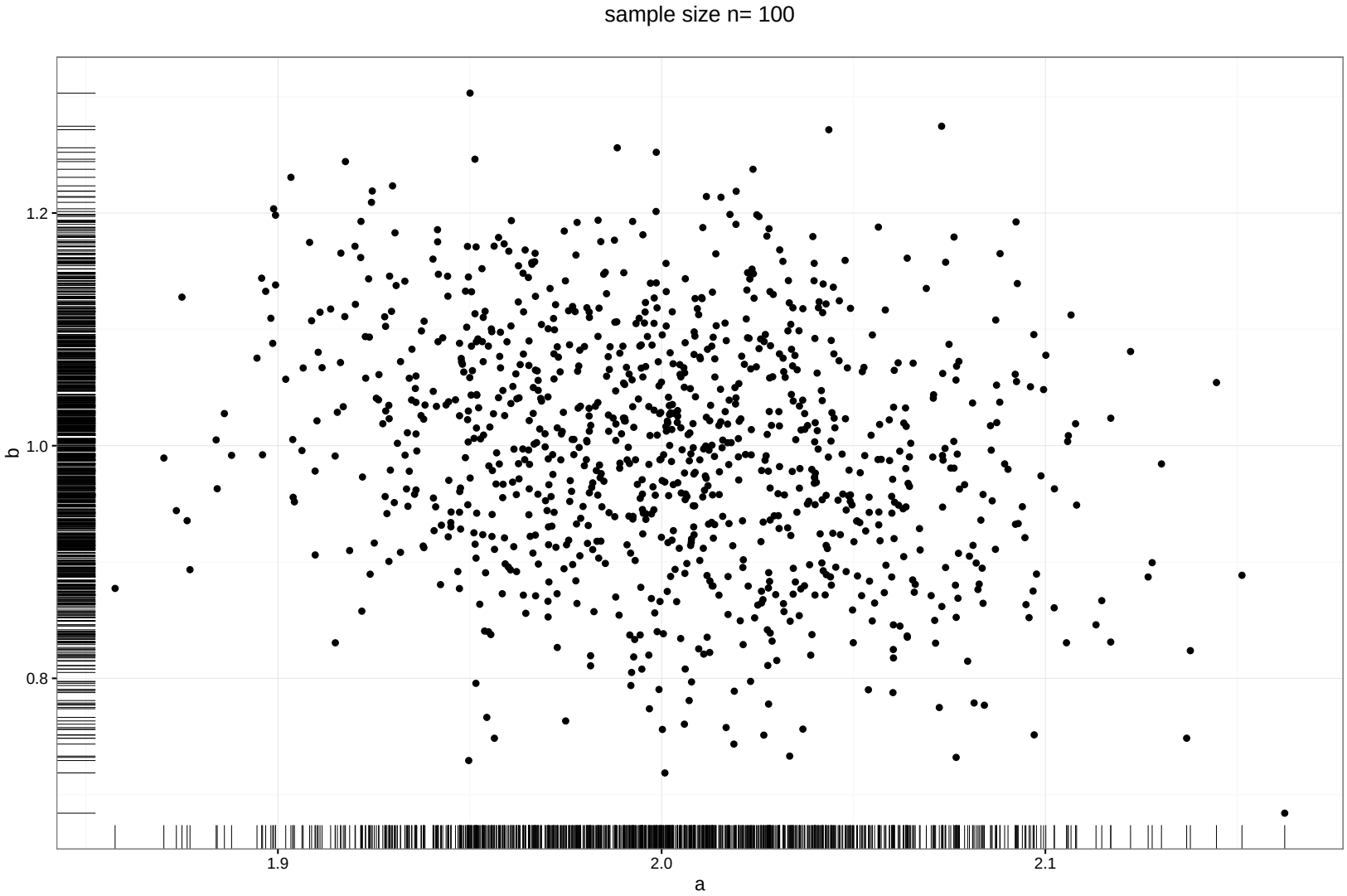
- yields

```
          a         b
1 1.994841 0.8079434
2 1.987354 1.0237531
3 1.951075 0.9251133
4 1.999110 1.0721703
5 1.996653 0.8200005
6 1.968700 1.0383586
```

```
       a                b
 Min.   :1.858   Min.   :0.6841
 1st Qu.:1.966   1st Qu.:0.9280
 Median :2.001   Median :0.9994
 Mean   :2.002   Mean   :0.9989
 3rd Qu.:2.035   3rd Qu.:1.0722
 Max.   :2.162   Max.   :1.3031
```

- What does the table tell us?
- A graphical overview also helps to interpret the results.

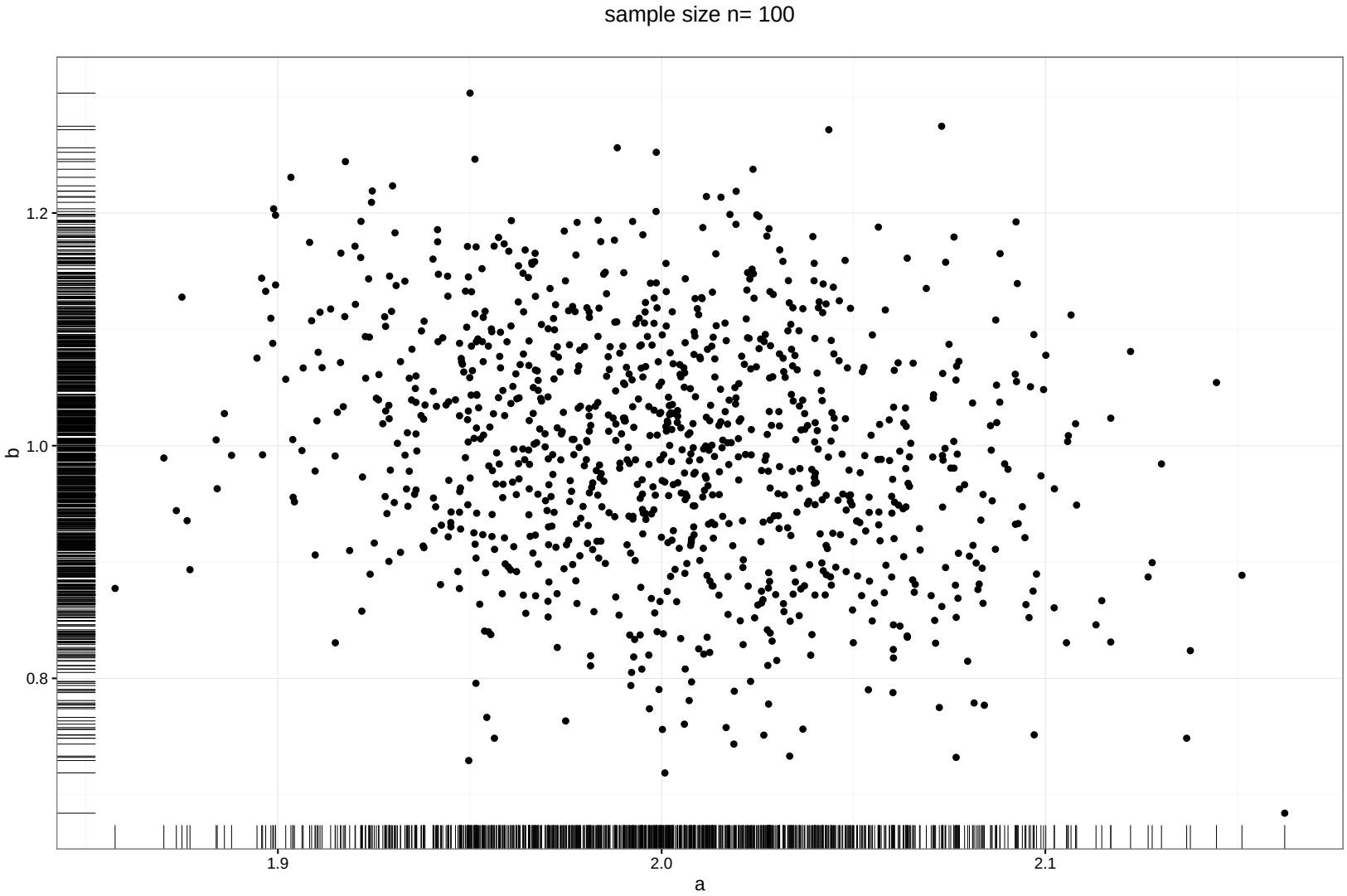
@Performance



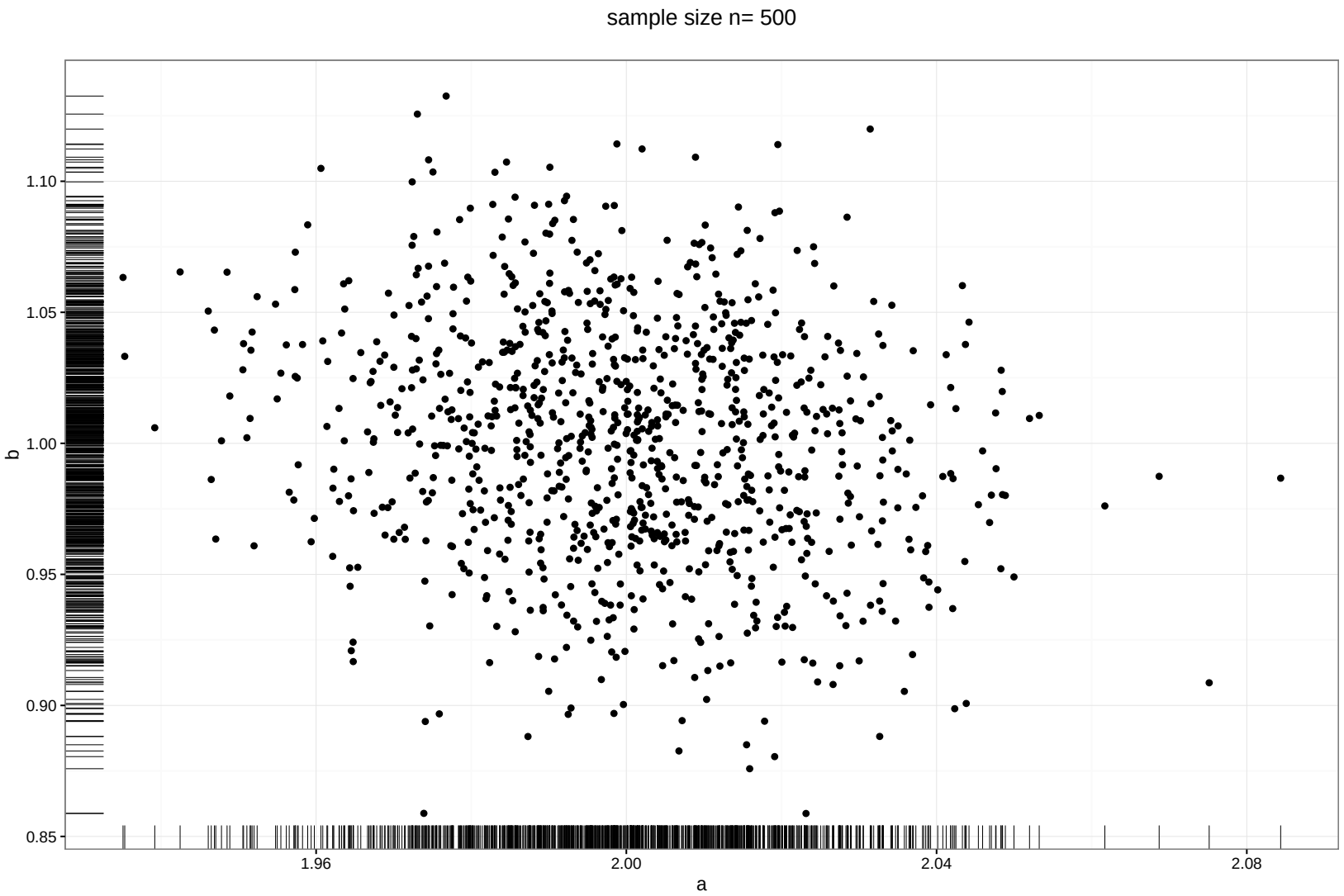
Natural related questions:

- What happens if the sample size n is increased?
- **The more info the better the estimates should (on average) be!**
- What other parameter in the simulation could have an influence on the quality of the estimates?
- Answer: The variance σ^2 of ε is important.
- **The higher the variance the poorer the estimates**.
- Repeat the simulation (several runs) for higher and lower sample size and vary the variance of the error.

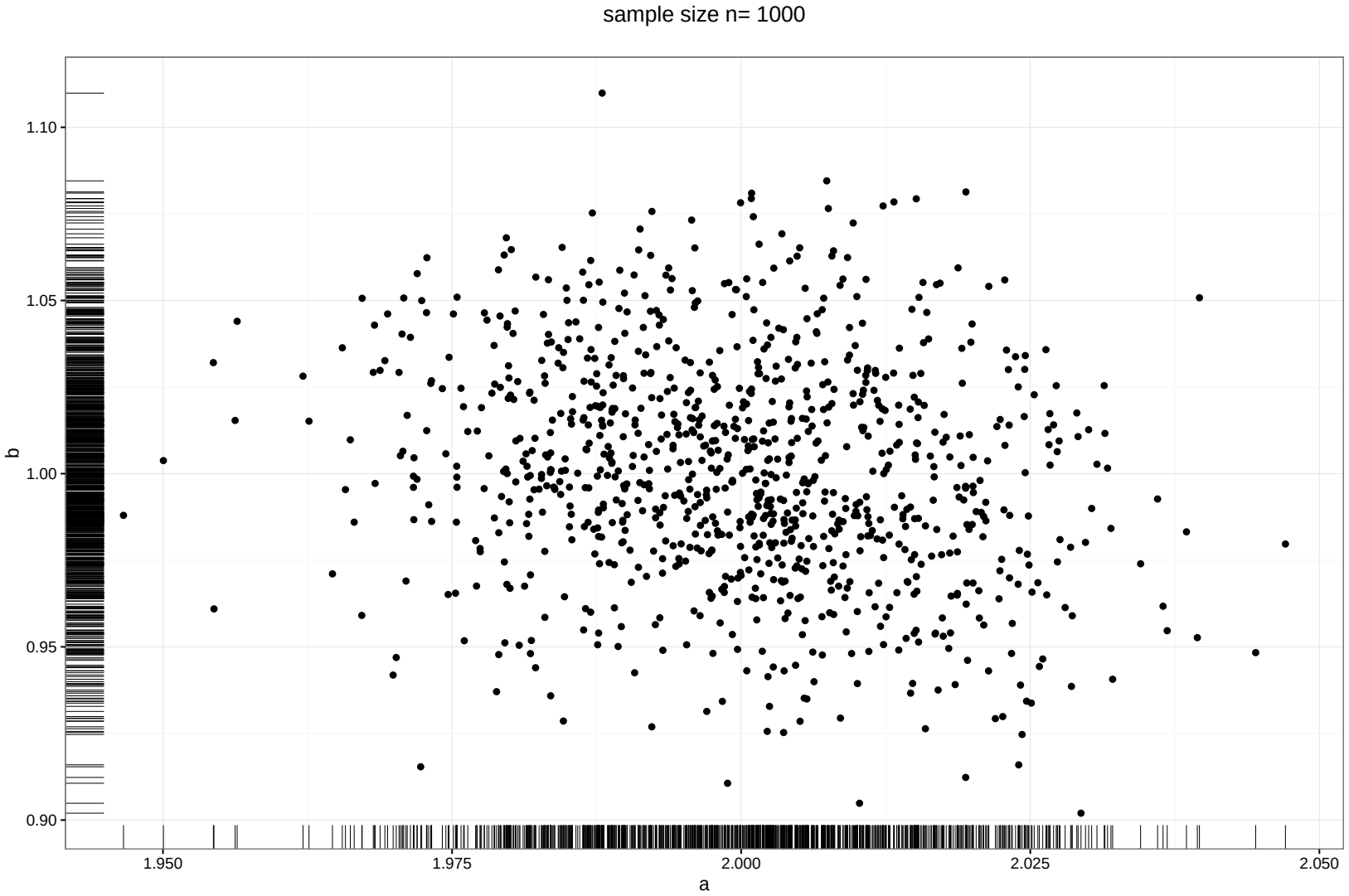
Influence of the parameters at stake



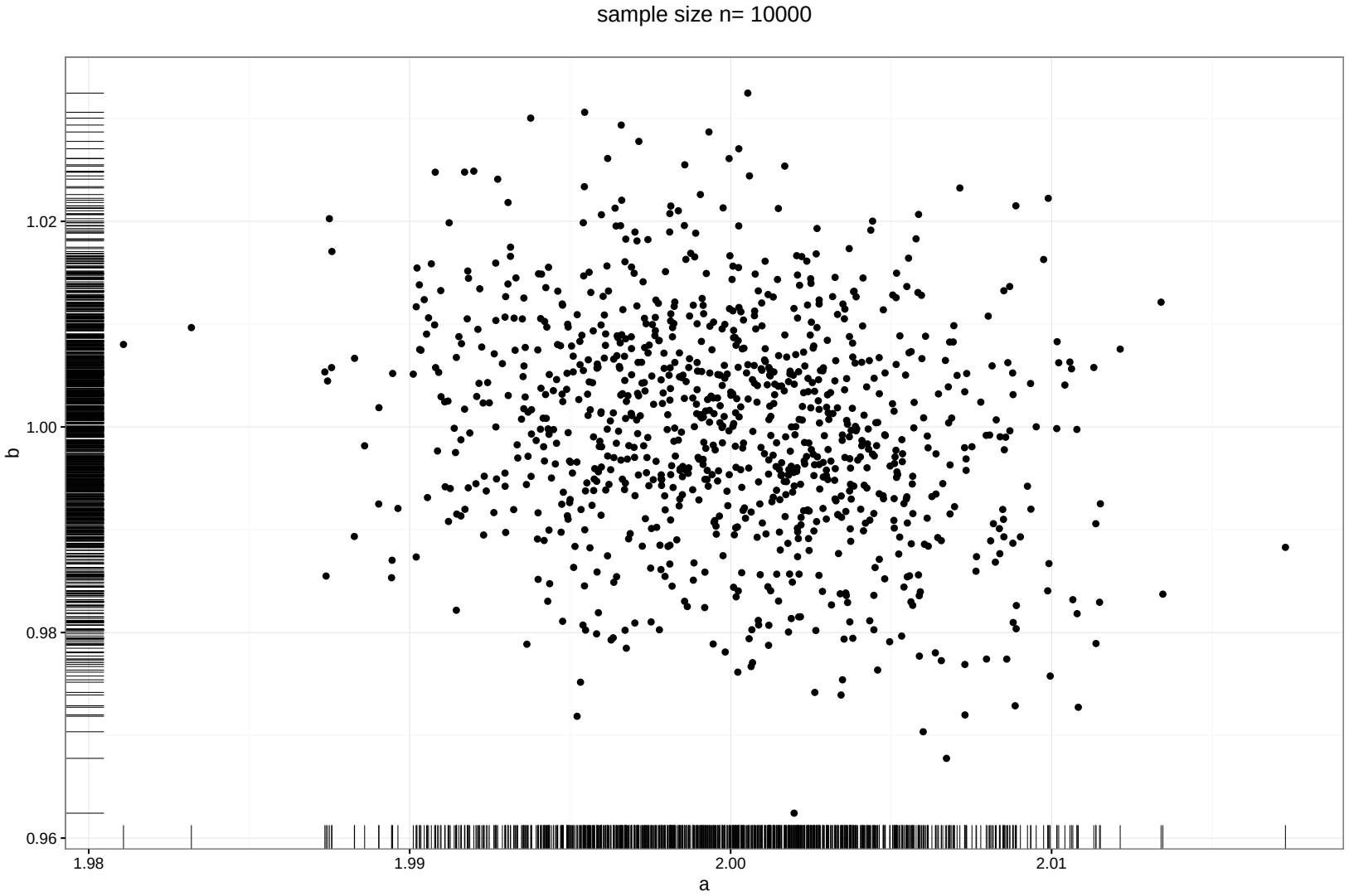
Influence of the parameters at stake



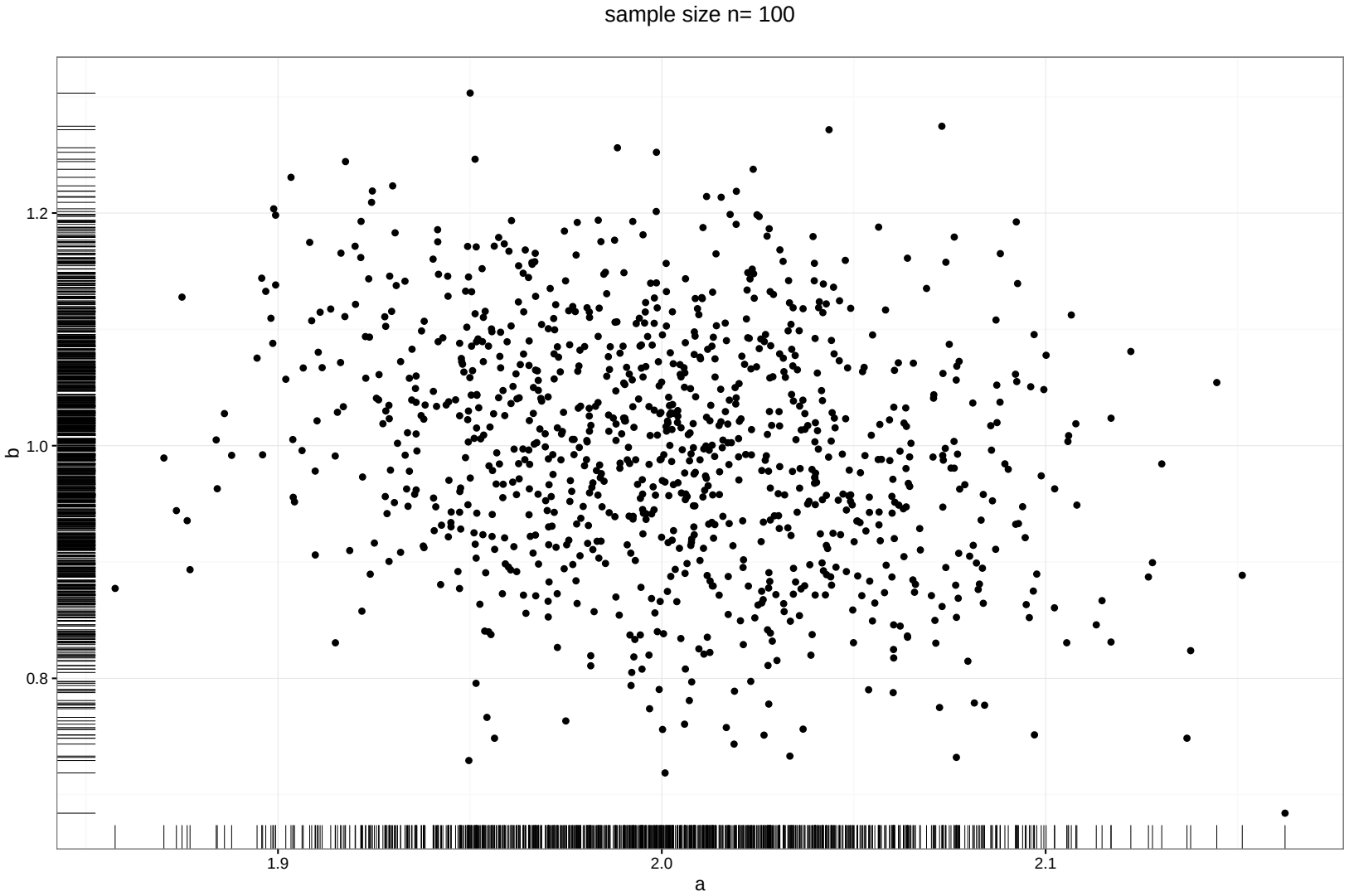
Influence of the parameters at stake



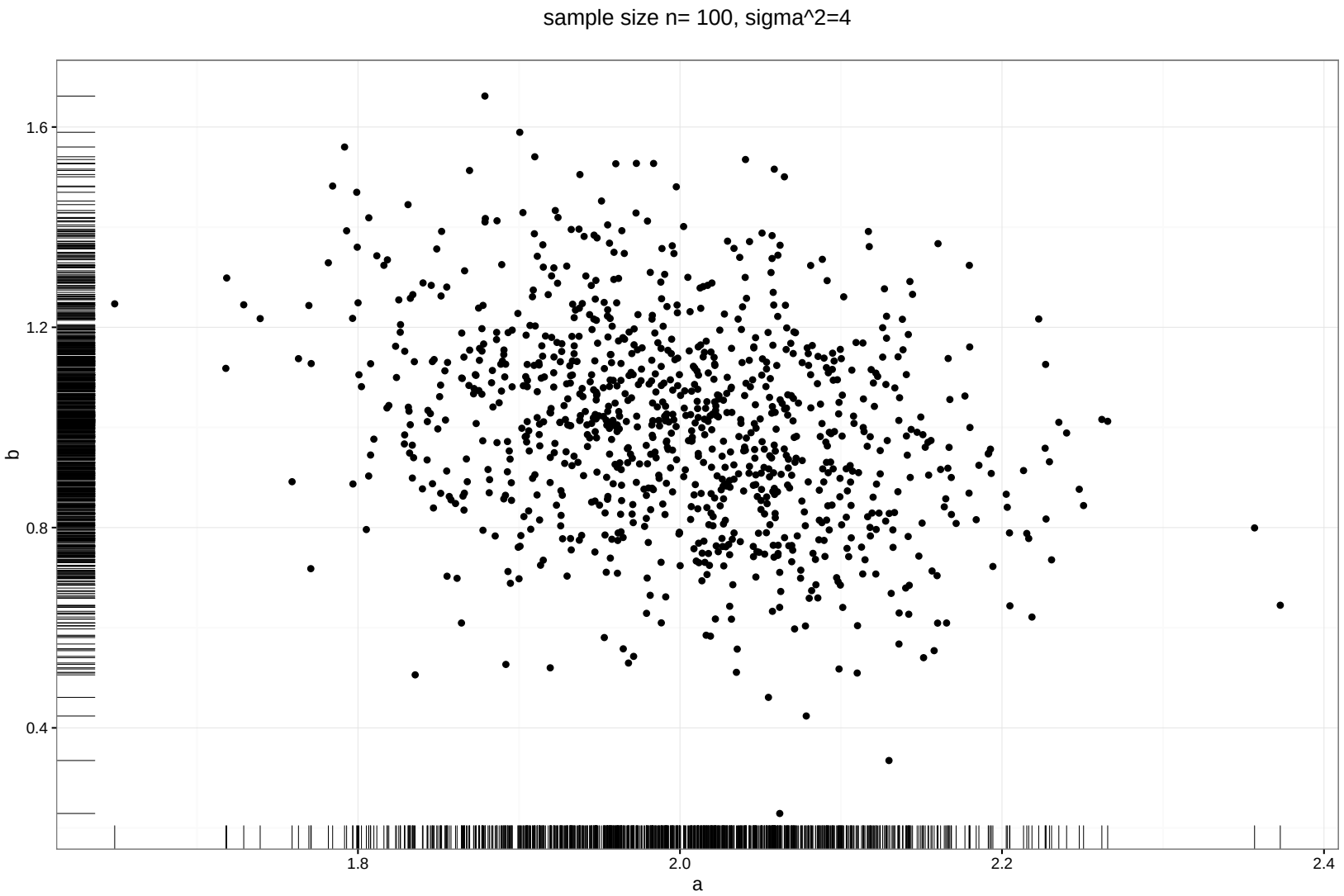
Influence of the parameters at stake



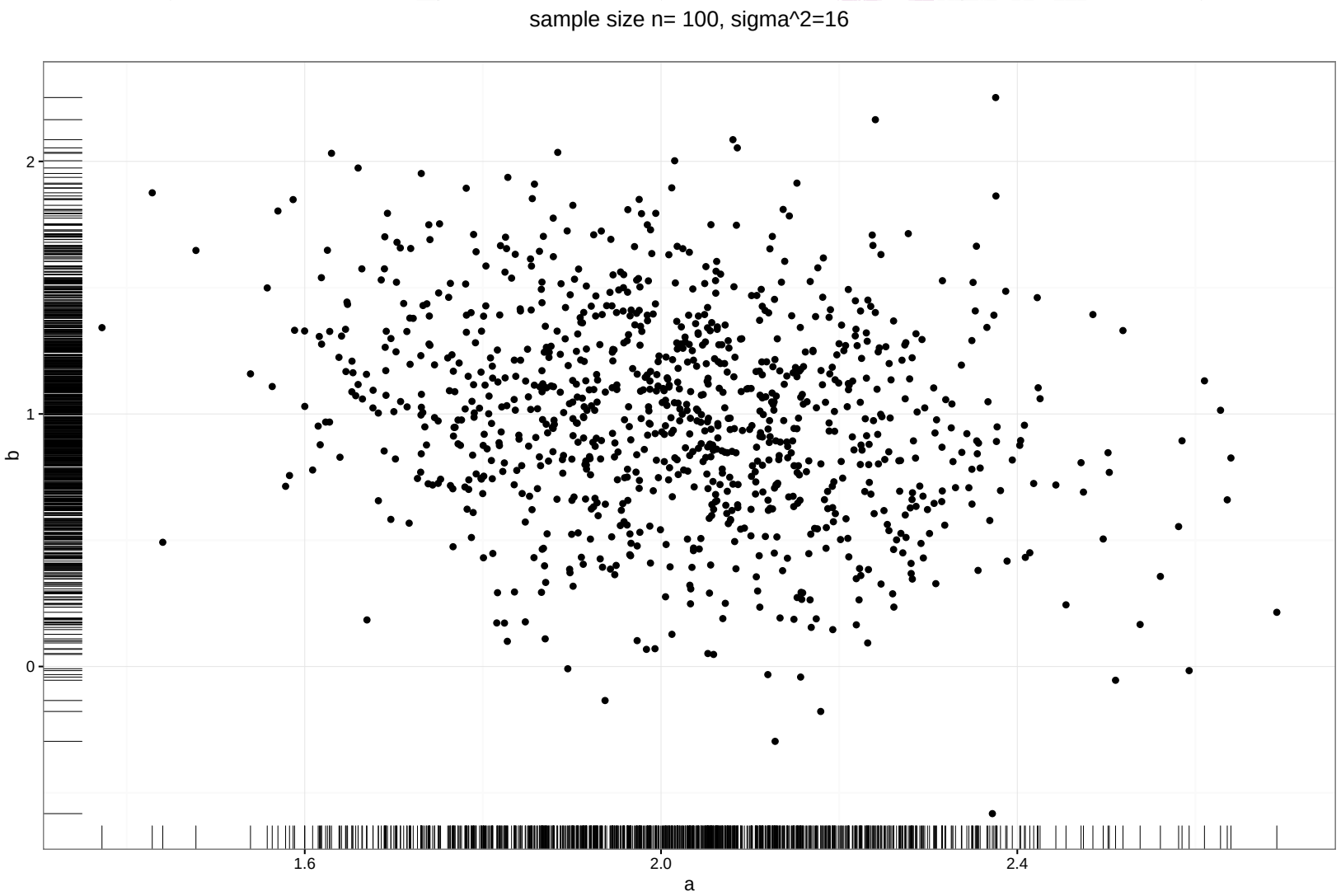
Influence of the parameters at stake



Influence of the parameters at stake



Influence of the parameters at stake



Exercise 07:
Modify the lines 141-165 of the R-Code StADS201711.R to do the following:

- Simulate a sample of size $n = 100$ from the model $Y = 0.5X - 1 + \varepsilon$ whereby $\varepsilon \sim \mathcal{N}(0, 0.5)$.
- Include a scatterplot of the data including the regression line; include the estimated parameters $\hat{a}$ and $\hat{b}$ in the title of the scatterplot.
- Produce a boxplots of the residuals $r_1, \ldots, r_n$.
- Calculate ρ and ρ_S of the data.
- Forecast $r(x)$ for $x \in \{0, 0.1, 0.2, \ldots, 0.9, 1\}$.

Exercise 08:
Modify the lines 141-165 of the R-Code StADS201711.R to do the following:

- Simulate a sample of size $n = 100$ from the model $Y = 0.5X - 1 + \varepsilon$ with $\varepsilon \sim \mathcal{N}(0, 0.5)$.
- Save the estimated parameters $\hat{a}$ and $\hat{b}$ in a data.frame A.
- Repeat the previous two steps $R = 1000$ times.
- Produce a boxplots of the estimates $\hat{a}_1, \ldots, \hat{a}_R$ and a boxplot of the estimates $\hat{b}_1, \ldots, \hat{b}_R$.
- Calculate the biggest, the smallest and the median value of $\hat{a}_1, \ldots, \hat{a}_R$.
- Calculate the biggest, the smallest and the median value of $\hat{b}_1, \ldots, \hat{b}_R$.
- Repeat the previous steps for bigger sample size and/or for bigger variance of the errors.

Exercise 09:

- In the literature and in (bad) courses one frequently sees that regression only works in case the errors have normal distribution.
- Consider $\mathcal{U}(-1,1)$-distributed errors using the command error=runif(n,-1,1) and repeat the tasks in Exercise 08 and Exercise 09 for this situation.
- Do we also get good results in this setting?

- We know that there is a relationship between quantities X and Y of the following form:

$$Y = r(X) + \varepsilon \tag{8}$$

- r is an unknown function and ε is a random error fulfilling $\mathbb{E}(\varepsilon) = 0$.
- Based on observations $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$ from (8) we want to determine/estimate the function r.
- If we have a good estimator $\hat{r}$ of r then we can predict Y for arbitrary values of X by considering $\hat{r}(X)$.
- All regression functions considered so far were *parametric*, i.e. r was fully determined by (a fixed number of) parameters.
- Example: $r(x) = ax + b$ (univariate linear gression)
- Example: $r(x) = a_1 x + a_2 x^2 + b$ (univariate quadratic regression)
- Example: $r(x_1, x_2) = a_1 x_1 + a_2 x_2 + b$ (two-dimensional linear regression)

Kernel regression - the basics

- **Problem: In practise not even the a parametric description of r is known.**
- What to do? $\rightarrow$ Use nonparametric techniques: kernel regression, local weighted linear/polynomial regression, etc.
- **(Nadaraya-Watson) Kernregression:** Given data $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$ from the regression model.
- Set

$$\hat{r}_n(x) := \frac{\sum_{i=1}^n K\left(\frac{x-x_i}{h}\right) y_i}{\sum_{i=1}^n K\left(\frac{x-x_i}{h}\right)}, \qquad \text{where} \tag{9}$$

- K...**Kernel**: Probability density, e.g. density of $\mathcal{N}(0,1)$ or density of $\mathcal{U}(-1,1)$.
- $h > 0$...**Bandwidth** (Smoothing parameter).
- x...Point at which the estimator is evaluated/for which the forecast is calculated.
- NB: $\hat{r}_n(x)$ is a weighted mean of the values $y_1, \ldots, y_n$ - the bigger $|x - x_i|$ the less weight has y_i for the calculation of $\hat{r}_n(x)$.

Example (Kernel Regression)

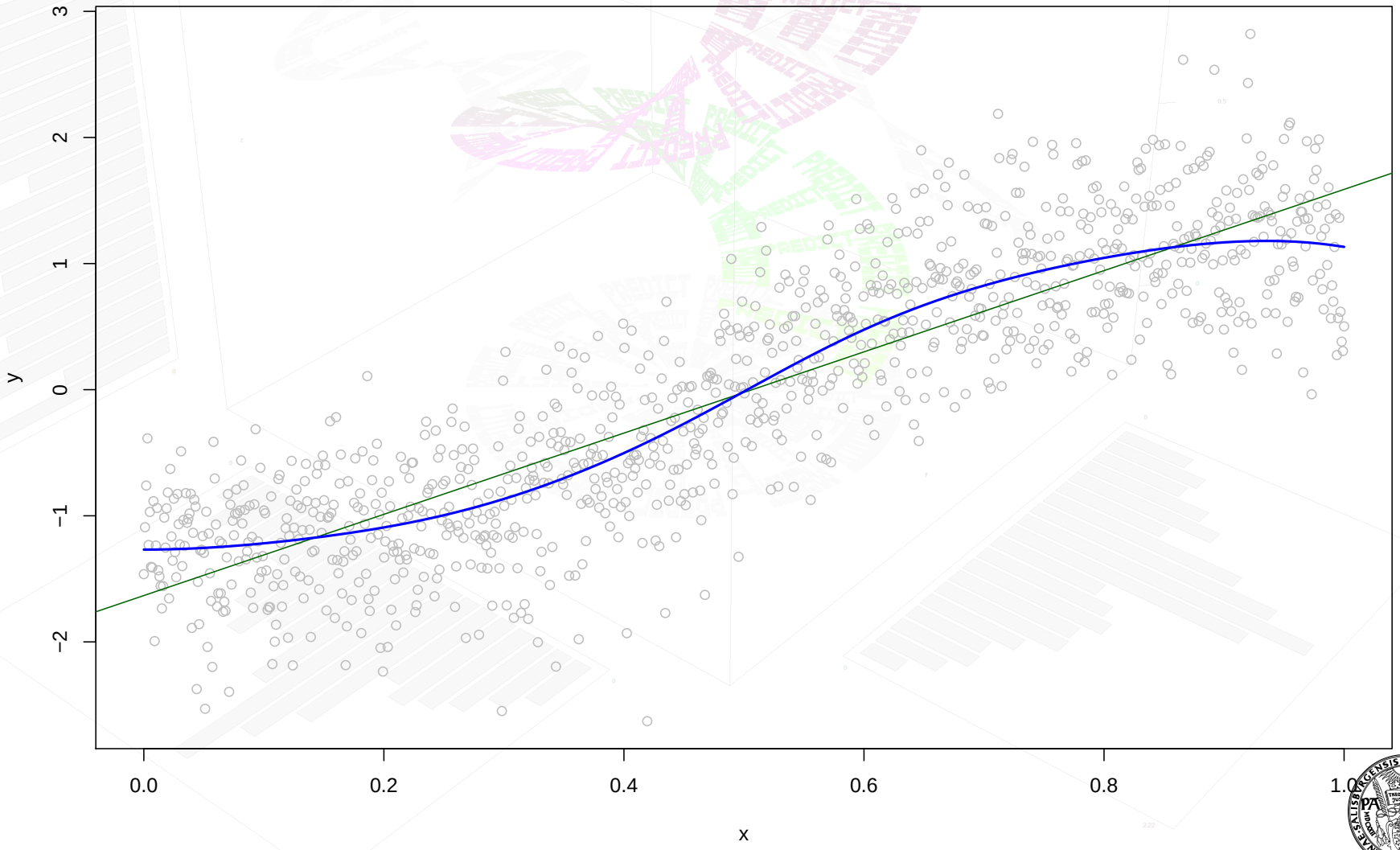
- We load the dataset reg_data.RData and fit a linear regression.
- Additionally, we calculate the kernel regression (estimator):

```
dir <- url("http://www.trutschnig.net/reg_data.RData")
load(dir)
A<-reg_data
head(A)
plot(A,col="gray")
abline(lm(data=A,y~x),col="darkgreen")

library(sm)
nreg<-sm.regression(A$x,A$y,eval.points=c(0.6),display="none")
                    #kernel regression at x=0.6
nreg$estimate
points(0.6,nreg$estimate,col="blue",cex=1)

nreg<-sm.regression(A$x,A$y,eval.points=seq(0,1,length=101),
     display="none")
lines(nreg$eval.points,nreg$estimate,type="l",col="blue",lwd=2)
```

Kernel regression - the basics



Example (Kernel regression, cont.)

- The function sm.regression selects the bandwidth h automatically.
- Nevertheless, h can also be chosen manually.
- Illustration of the effect of the bandwidth h to the kernel regression (estimator) $\rightarrow$ shiny app.
- Conclusion: The smaller h the faster the weights drop with the distance.
- Too big h yields too much smoothing.
- Too small h yields a very shaky regression estimator.
- Calculate R^2 for the calculated regression:

```
nreg<-sm.regression(A$x,A$y,eval.points=A$x,display="none")
res<-A$y-nreg$estimate
R2<-1-var(res)/var(A$y)
R2
```

- The last example was purely descriptive.
- We want to evaluate the quality of kernel regression estimates via simulations.
- We consider $\hat{r}_n$ a good estimator if it is close to the regression function r uniformly.
- We proceed as in the parametric setting:
  - choose a regression function $r(x)$.
  - generate samples $(x_1, y_1), \ldots, (x_n, y_n)$ with $y_i = r(x_i) + \varepsilon_i$.
  - calculate the kernel regression $\hat{r}_n$.
  - check how close $\hat{r}_n$ and r are.

Example (Quality check I)

- Model $Y = \arctan(6x - 3) + \varepsilon$
- In other words: the regression function r is given by $r(x) = \arctan(6x - 3)$
- Use the following R-Code to generate the data and calculate the kernel regression:

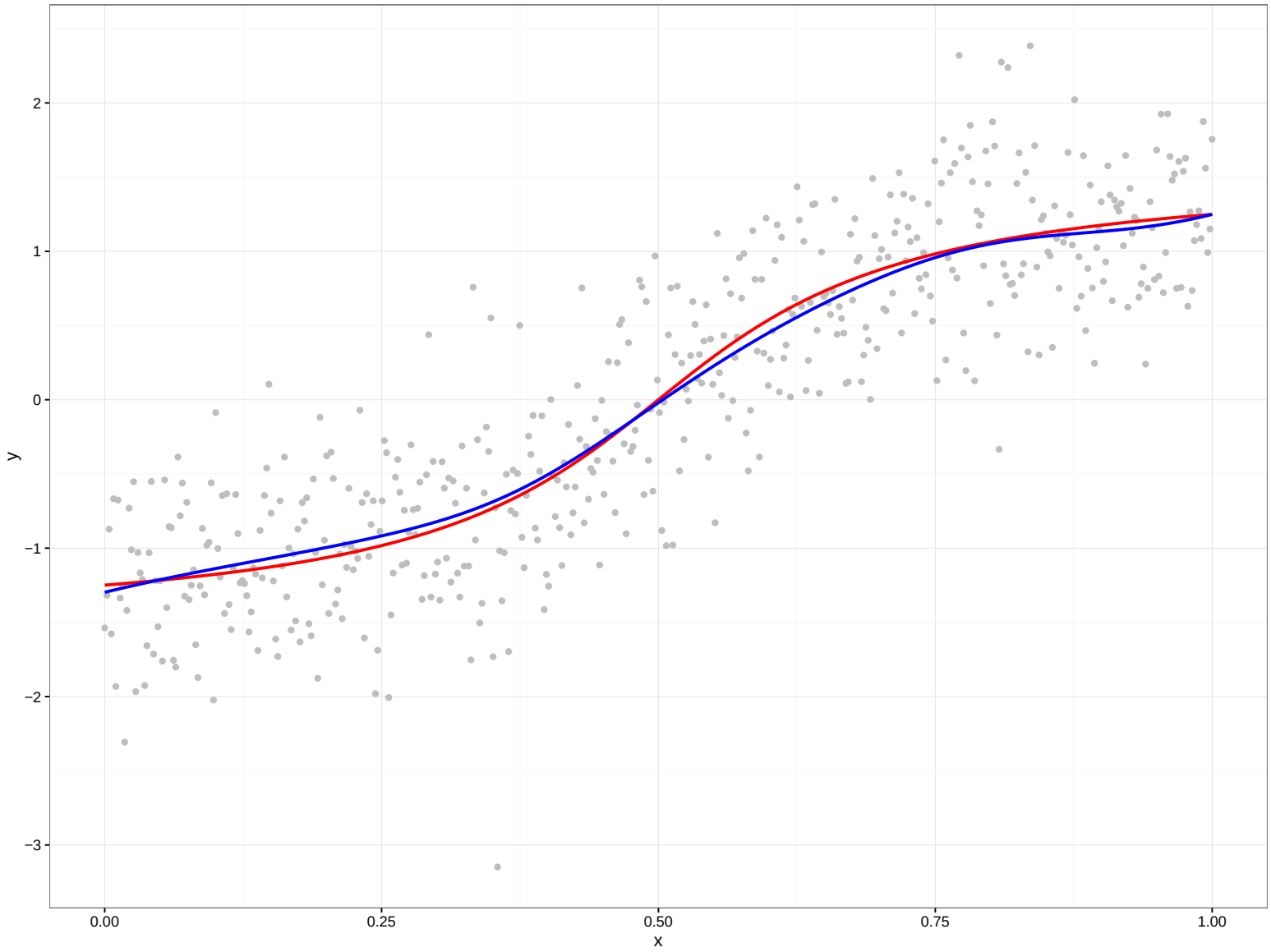
Example (Quality check I)

```
n <- 500
x <- seq(0,1,length=n)
error <- rnorm(n,0,0.5)
y <- atan(6*x-3)+error
A <- data.frame(x=x,y=y)
plot(A,col="gray")

lines(x,atan(6*x-3),type="l",col="red",lwd=2)
nreg <- sm.regression(A$x,A$y,eval.points=seq(0,1,length=101),
      display="none")
lines(nreg$eval.points,nreg$estimate,type="l",col="blue",lwd=2)
```

- yields

Kernel regression - the basics



Example (Quality check II)

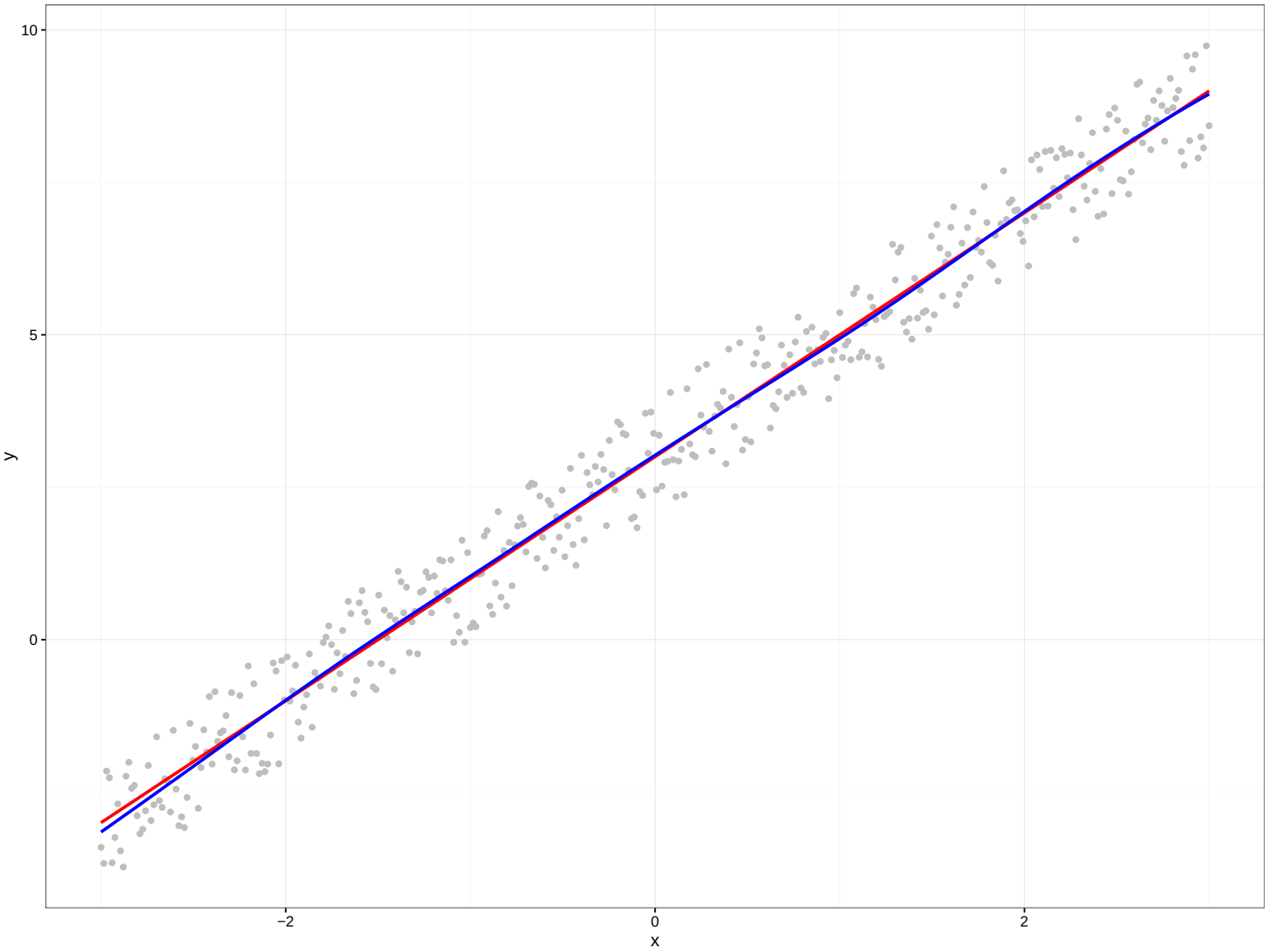
- Model $Y = 2X + 3 + \varepsilon$
- In other words: the regression function r is given by $r(x) = 2x + 3$.
- Use the following R-Code to generate the data and calculate the kernel regression:

```
a <- 2; b <- 3
n <- 400
x <- seq(-3,3,length=n)
y <- a*x+b+runif(n,-1,1)
A <- data.frame(x=x,y=y)
plot(A,col="gray")
abline(3,2,col="red")

nreg <- sm.regression(A$x,A$y,eval.points=A$x,display="none")
lines(nreg$eval.points,nreg$estimate,type="l",col="blue",lwd=2)

R2 <- 1-var(A$y-nreg$estimate)/var(A$y)
R2
```

Kernel regression - the basics



Summary:

- Kernel regression seems to yield good results for sufficiently large sample size n.
- Convergence to the true regression function (convergence of $\hat{r}_n$ to $r(x)$) can also be proved mathematically .
- Kernel regression also works in dimensions two and three, in higher dimensions things get more complicated (sample size, computing time).
- It often makes sense to play with the bandwidth h and observe what happens (i.e. not only to use the implemented bandwidth h).
- Concluding example, timeseries decomposition, shiny app.

Exercise 10 (Quality check III)

- Repeat the quality check for data from the model $Y = \frac{3}{1+2X^2} + \varepsilon$, where X fulfills $X \geq 0$.
- Work with different sample sizes n.
- Choose a fixed grid in the interval $[0, 5]$ and calculate the maximum distance of $\hat{r}_n$ to r on this grid.
- Does the maximum distance also decrease if the sample size n increases?

Exercise 11: The data set SBP.RData contains the following data for 8.000 patients: age, BMI (body mass index), SBP (systolic blood pressure). Using 238-246 lines in StADS201711.R three age groups of patients are built.

To do list:

- Using kernel regression estimate the regression function r with $SBP = r(BMI)$ individually for each of the three age groups.
- For each of the three age groups predict the (average) SBP-value for a patient having BMI 25 and a patient having BMI 30.
- For each of the three groups calculate the percentage increase from BMI 25 to BMI 30.

- Natural generalization of the univariate linear model to several predictors
- Consider the case of two predictors X_1 and X_2 (analogously for more than two) and one response variable Y.
- In this case the model is of the form: $Y = a_1 X_1 + a_2 X_2 + \varepsilon$
- In other words: we assume that the data $(x_{1,1}, x_{2,1}, y_1), (x_{1,2}, x_{2,2}, y_2), \ldots (x_{1,n}, x_{2,n}, y_n)$ fulfills $y_i = a_1 x_{1,i} + a_2 x_{2,i} + b + \varepsilon_i$
- As before: ε_i... independent random errors with fixed variance and $\mathbb{E}(\varepsilon_i) = 0$
- We proceed as in the univariate case and want to minimize

$$F(\tilde{a}_1, \tilde{a}_2, \tilde{b}) := \sum_{i=1}^{n} (y_i - \tilde{a}_1 x_{1,i} - \tilde{a}_2 x_{2,i} - \tilde{b})^2 \tag{10}$$

- Choose $\tilde{a}_1, \tilde{a}_2, \tilde{b}$ in such a way that $F(\tilde{a}_1, \tilde{a}_2, \tilde{b})$ is minimal
- We fit the two-dimensional model to the data geo_reg_d2.RData

Multivariate linear regression

```
#two dim:
file <- url("http://www.trutschnig.net/geo_reg_d2.RData")
load(file)
A<-geo_reg_d2
head(A)

model<-lm(data=A,y~x1+x2)
model
summary(model)
```

- yields

```
Call:
  lm(formula = y ~ x1 + x2, data = A)

Residuals:
   Min      1Q  Median      3Q     Max
-3.5631 -0.6376  0.0564  0.9176  2.1860
```

- and

```
Coefficients:
Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.04404    0.11306    0.39    0.698
x1           2.93824    0.04889   60.10   <2e-16 ***
x2           1.96832    0.06229   31.60   <2e-16 ***
---
 Signif. codes:  0   ***   0.001   **   0.01   *   0.05   .
 0.1   1

Residual standard error: 1.096 on 97 degrees of freedom
Multiple R-squared:  0.9791, Adjusted R-squared:  0.9787
F-statistic:  2273 on 2 and 97 DF,  p-value: < 2.2e-16
```

- Calculate the prediction for $x1 = 1.5$ and for $x = 0$

```
ND<-data.frame(x1=c(1.5),x2=c(0))
p<-predict(model,new=ND)
p
4.4514
```

Exercise 12: Analogously to the case of univariate linear regression do a simulation study to find out whether the estimates $\tilde{a}_1, \tilde{a}_2, \tilde{b}$ are close to the true values a_1, a_2, b. Proceed as follows and lines 141-155 as draft:

- Fix a_1, a_2 and b and fix the sample size n
- Consider (or generate) some values $x_{1,1}, \ldots, x_{1,n}$ and $x_{2,1}, \ldots, x_{2,n}$
- Generate random errors $\varepsilon_1, \ldots, \varepsilon_n$
- Set $y_i = b + a_1 x_{1,i} + a_2 x_{2,i} + \varepsilon_i$ for every $i \in \{1, \ldots, n\}$
- Consider the sample $(x_{1,1}, x_{2,1}, y_1), \ldots, (x_{1,n}, x_{2,n}, y_n)$ and calculate $\hat{a}_1, \hat{a}_2$ and $\hat{b}$
- Check how close $\hat{a}_1$, $\hat{a}_2$ and $\hat{b}$ are to a_1,a_2 and b
- Repeat the above steps at least $R = 1000$ times

Summarize the most important observations (influence of sample size, variance of the errors, etc.) in a knitR report.

- Syntax analogous to the univariate setting
- R-output analogous to the univariate setting
- Increasing the sample size yields better estimates (of the coefficients b, a_1, a_2)
- Increasing the variance of the errors implies (on average) worse estimates
- Anything else that might be relevant? Can we always fit a multivariate linear model without further ado?

Example (Two-dimensional linear regression)

Consider the following R-Code (also contained the R-Code).

```
n <- 100
x1 <- runif(n=n,-10,10)
x2 <- 2*x1+runif(n,-0.1,0.1)
y <- 3*x1+x2+runif(n,-1,1)
A <- data.frame(x1=x1,x2=x2,y=y)
```

Example (Two-dimensional linear regression, cont.)

```
model <- lm(data=A, y~x1+x2)
model
```

- Yields

```
Call:
  lm(formula = y ~ x1 + x2, data = A)

Coefficients:
  (Intercept)           x1           x2
      0.09016      4.74766      0.12710
```

- Rerunning the code yields

```
Call:
lm(formula = y ~ x1 + x2, data = A)

Coefficients:
  (Intercept)           x1           x2
      0.02081      6.08233     -0.53973
```

Example (Two-dimensional linear regression, cont.)

- Increasing the sample size n improves the results but the estimates still vary a lot
- What is the reason for this (unexpected) bad behavior?

- The reason is that the explanatory variables x_1 and x_2 have very high correlation

```
cor(A$x1,A$x2)
[1] 0.9999889
```

- This problem/phenomenon is usually referred to as **multicollinearity**
- Before fitting a multivariate linear model check the correlations of the explanatory variables!
- Only include variables with low correlation in the model!

How to proceed in practice (dimensions up to $d = 3$)

- Consider the univariate case:
- Given data $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$ from a model $Y = r(X) + \varepsilon$.
- r is an unknown function and ε is a random error fulfilling $\mathbb{E}(\varepsilon) = 0$.
- For sufficiently large sample size we can estimate r via kernel regression.
- If the kernel regression is of a specific form (if it looks linear, quadratic, saturation-curve-like, S-shaped, etc.) the next natural step is to fit a **parametric model**
- We already know how to fit linear and polynomial models.
- How to proceed in the general case?
- We start with a simple example.

Example (Moisture dataset)

- The dataset moisture.txt contains moisture content of core samples from mud.
- For each sample the depth (in feet) as well as the moisture (g water per 100g dried solid).

tiefe	Wassergehalt
0.00	127.20
0.00	132.03
0.00	130.44
0.00	128.07
0.00	126.95
0.00	118.15
0.00	117.82
0.00	129.23

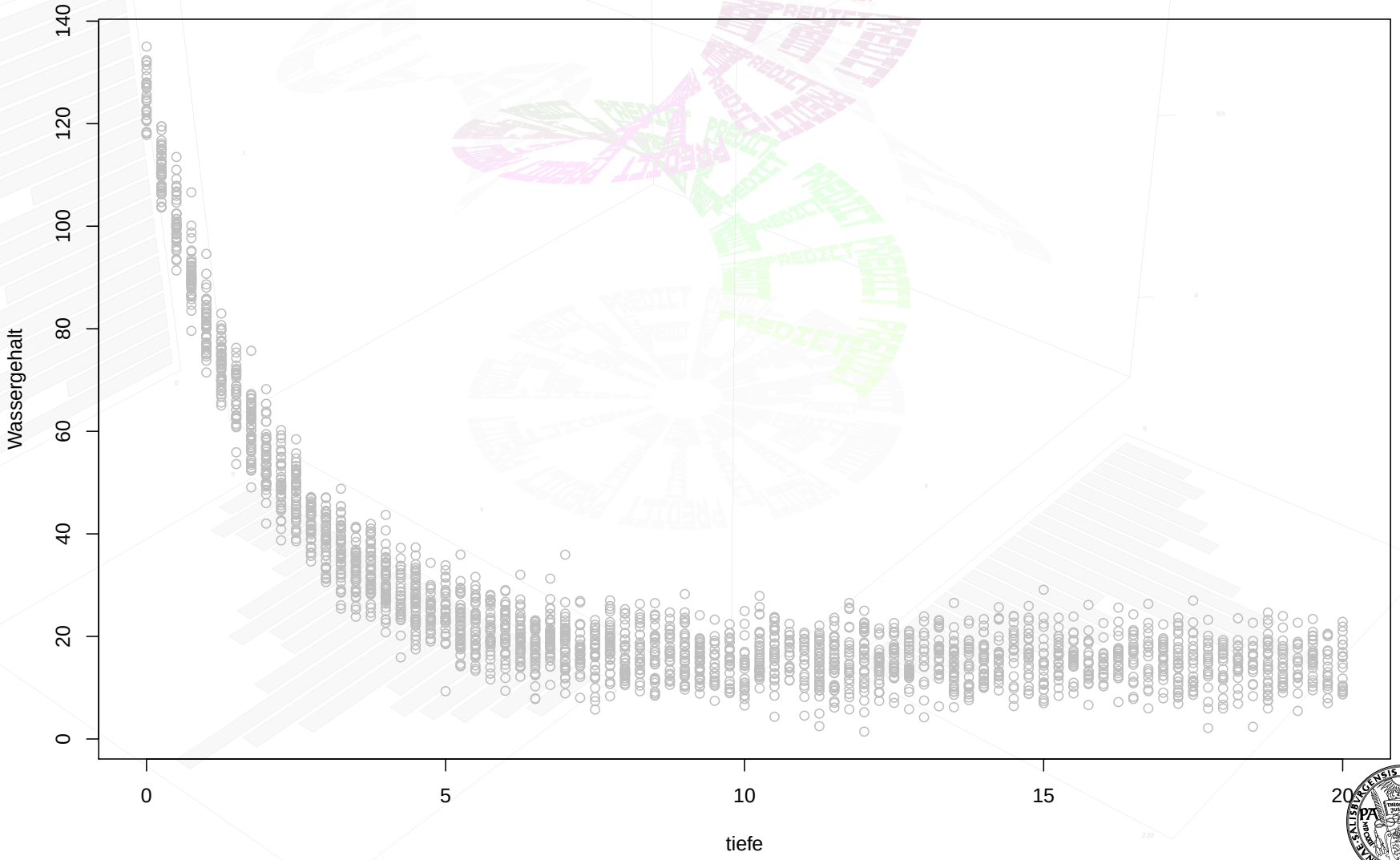
Table: First eight lines of the moisture dataset

tiefe	Wassergehalt
Min. : 0.000	Min. : 1.44
1st Qu.: 3.500	1st Qu.: 14.89
Median : 7.750	Median : 19.73
Mean : 8.522	Mean : 30.06
3rd Qu.:13.000	3rd Qu.: 33.72
Max. :20.000	Max. :135.03

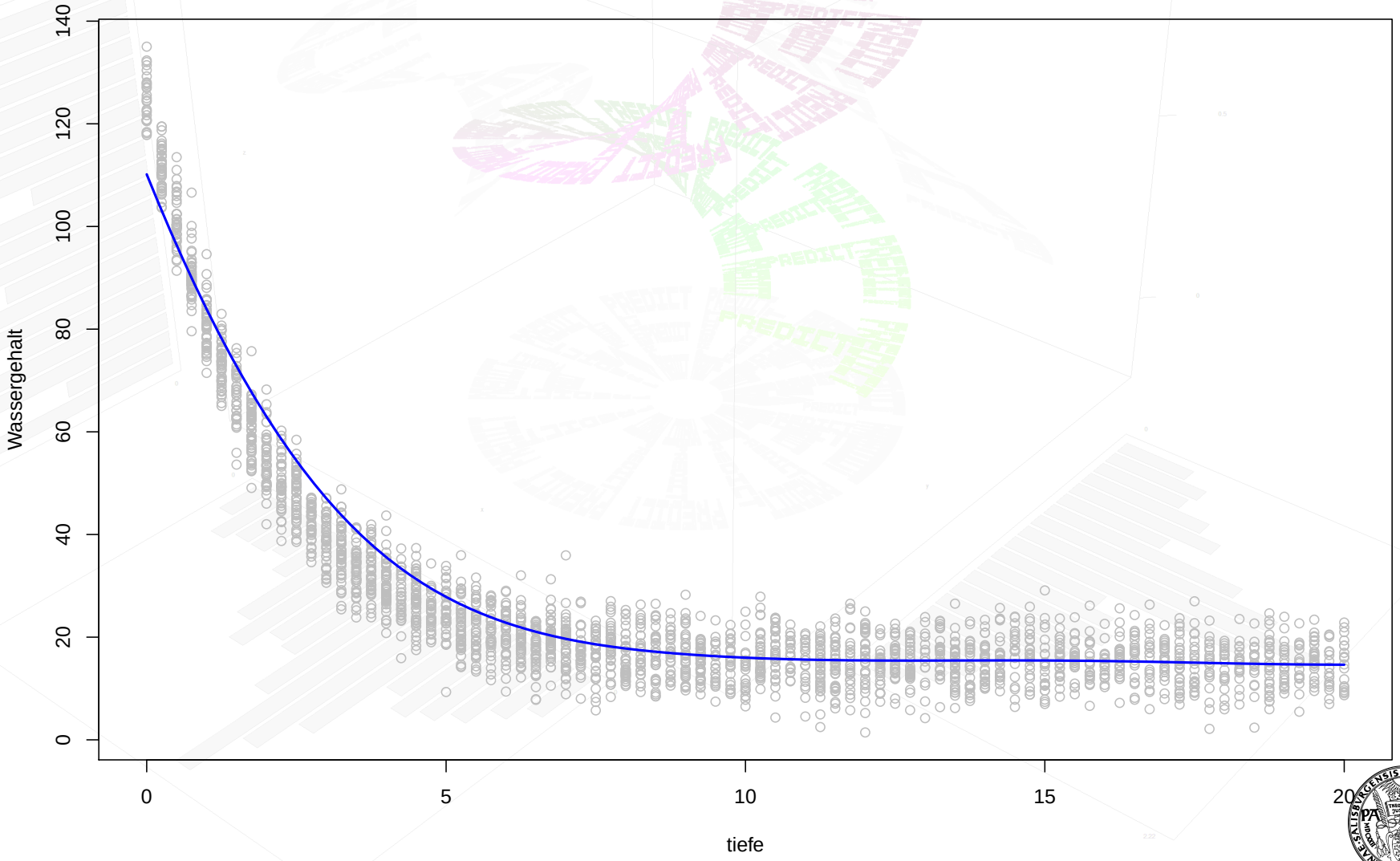
Table: Summary of the moisture dataset

- The dataset is plotted on the next slide.

The function nls



The function nls



Example (Moisture dataset, continued)

- Assume that from geological models it is known that (under certain conditions) the interrelation by depth t and moisture m is given by

$$m = 125 - 110(1 - e^{-at}). \tag{11}$$

- In other words, there is a one-parametric model describing the interrelationship.
- This parameter a needs to be estimated from the observed data $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$.
- How can this be done?
- Option 1: Simply transform eq. (11) into a linear model in t.
- Option 2: Proceed as before and choose $\hat{a}$ in such a way that the sum of all squared errors

$$F(\tilde{a}) := \sum_{i=1}^{n} \left(m_i - (125 - 110(1 - e^{-\tilde{a}t_i}))\right)^2 \tag{12}$$

is minimized.

Example (Moisture dataset, continued)

- The minimization can be done using the R-function *nls*:

```
model.nls <- nls(data=A, Wassergehalt~125-110*(1-exp(-a*tiefe)),
                 start = list(a = 1))
model.nls
```

- yields

```
Nonlinear regression model
model: Wassergehalt ~ 125 - 110 * (1 - exp(-a * tiefe))
data: A
a
0.5027
residual sum-of-squares: 50676

Number of iterations to convergence: 5
Achieved convergence tolerance: 5.126e-08
```

Example (Moisture dataset, continued)

- Moreover

```
summary(model.nls)
```

- yields

```
Formula: Wassergehalt ~ 125 - 110 * (1 - exp(-a * tiefe))

Parameters:
  Estimate   Std. Error t value Pr(>|t|)
a 0.502668   0.002241   224.3   <2e-16 ***
  ---
Signif. codes:  0   ***   0.001   **   0.01   *   0.05
    .   0.1        1

Residual standard error: 4.493 on 2510 degrees of freedom

Number of iterations to convergence: 5
Achieved convergence tolerance: 5.126e-08
```

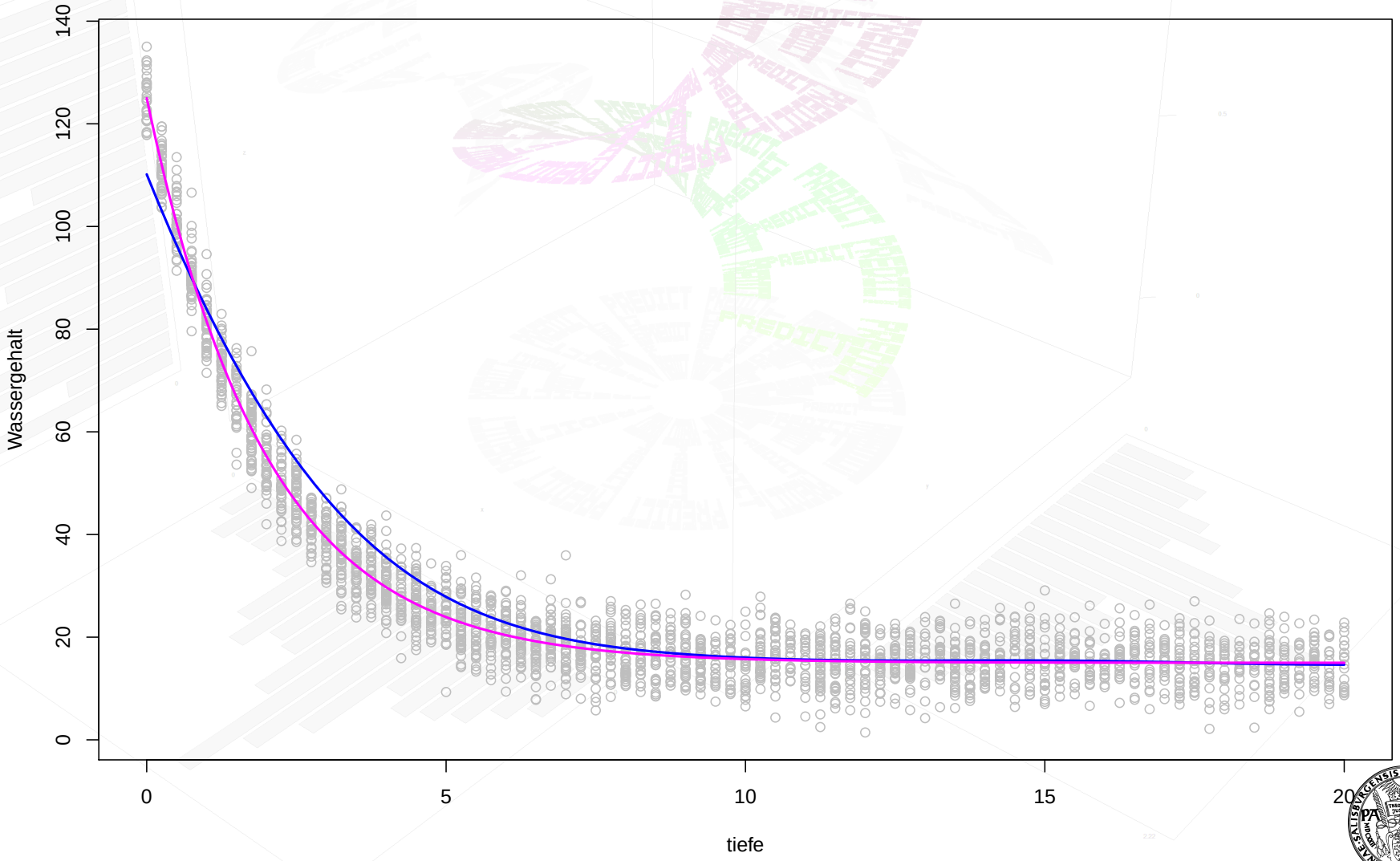
Example (Moisture dataset, continued)

- We add the function $m(t) = 125 - 110(1 - e^{-at})$ for the estimated parameter $a = 0.502668$ to the plot.

```
new <- data.frame(tiefe=seq(0,20,length=1001))
pred <- predict(model.nls,newdata = new)
new$prediction <- pred
lines(new$tiefe,new$prediction,col="red",type="l")
```

- Notice that the syntax is exactly the same as for *lm*, only the command itself is replaced by *nls* and a reasonable starting value (initial guess) for *a* has to be provided.
- The following graphic shows the sample as well as the kernel regression (blue) and the parametric regression calculated via nls (magenta).

The function nls



Example (SBP versus age and BMI)

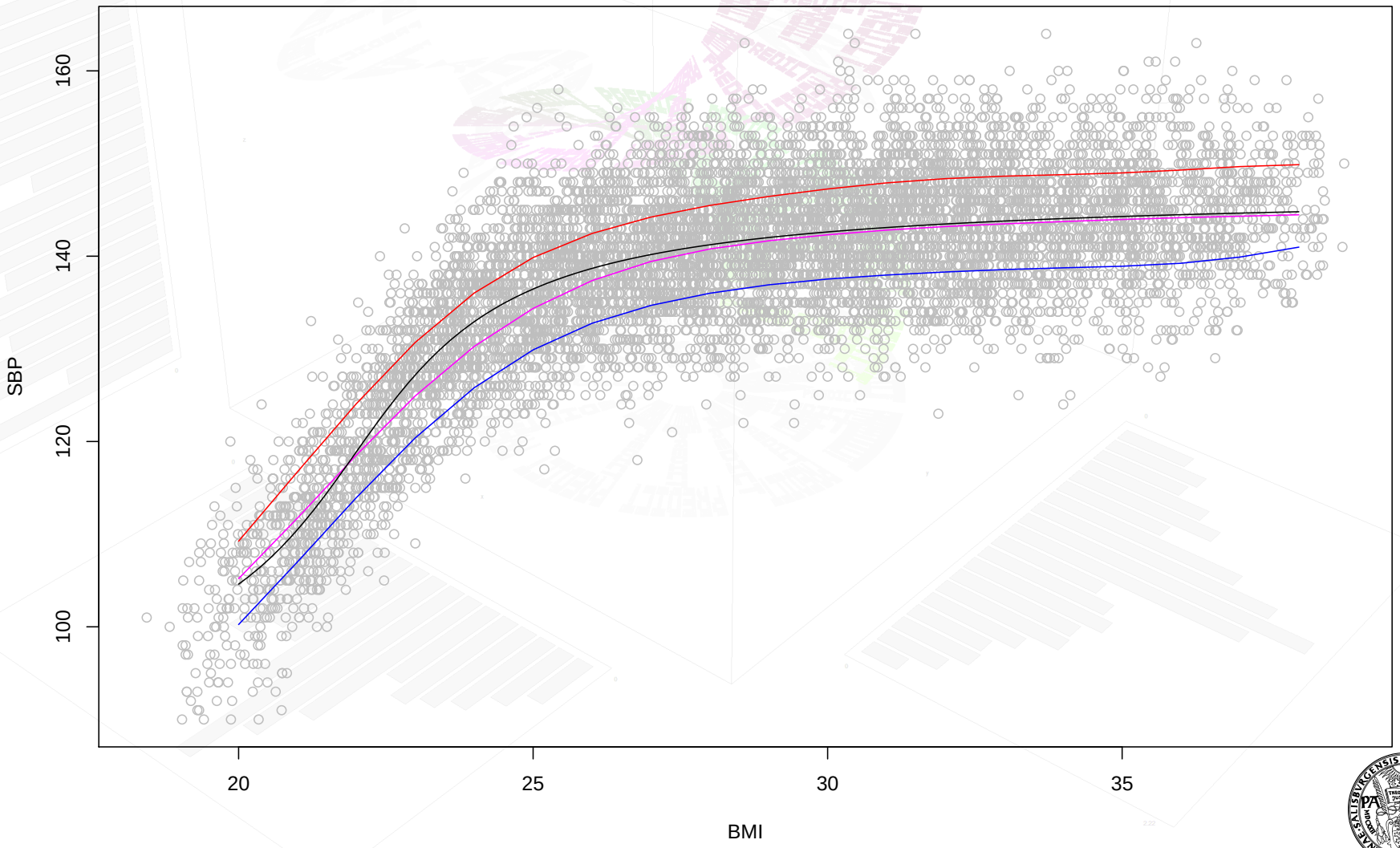
- The data set SBP.RData (see Exercise 11) contains the following data for 8.000 patients: age, BMI (body mass index), SBP (systolic blood pressure).

age	BMI	SBP
61.00	26.05	131.00
37.00	26.47	137.00
77.00	36.35	151.00
75.00	25.75	147.00
39.00	28.20	141.00
34.00	25.32	129.00

Table: First six lines of the SBP dataset

- Considered three age groups: $[30, 40]$, $[50, 60]$ and $[70, 80]$.
- For each age group we calculated the kernel regression estimate for the regression function r with $SBP = r(BMI)$ and got the following result:

The function nls, cont.



- The function *nls* can be used in arbitrary dimensions (the reason for the restriction on dimension 1-3 before was due to the kernel regression)

Example (SBP versus age and BMI, continued)

- We fit a regression model of the form

$$SBP = r(BMI, age) = 105 + a \cdot age + b \cdot atan(c \cdot BMI + d)$$

- The model contains four parameters: a, b, c, d
-
```
model.nls <- nls(data=A, SBP ~ 105+a*age + b*atan(c*BMI + d),
                 start = list(a=1, b=10, c=0.5, d=-5))
model.nls
summary(model.nls)
```

yields

Example (SBP versus age and BMI, continued)

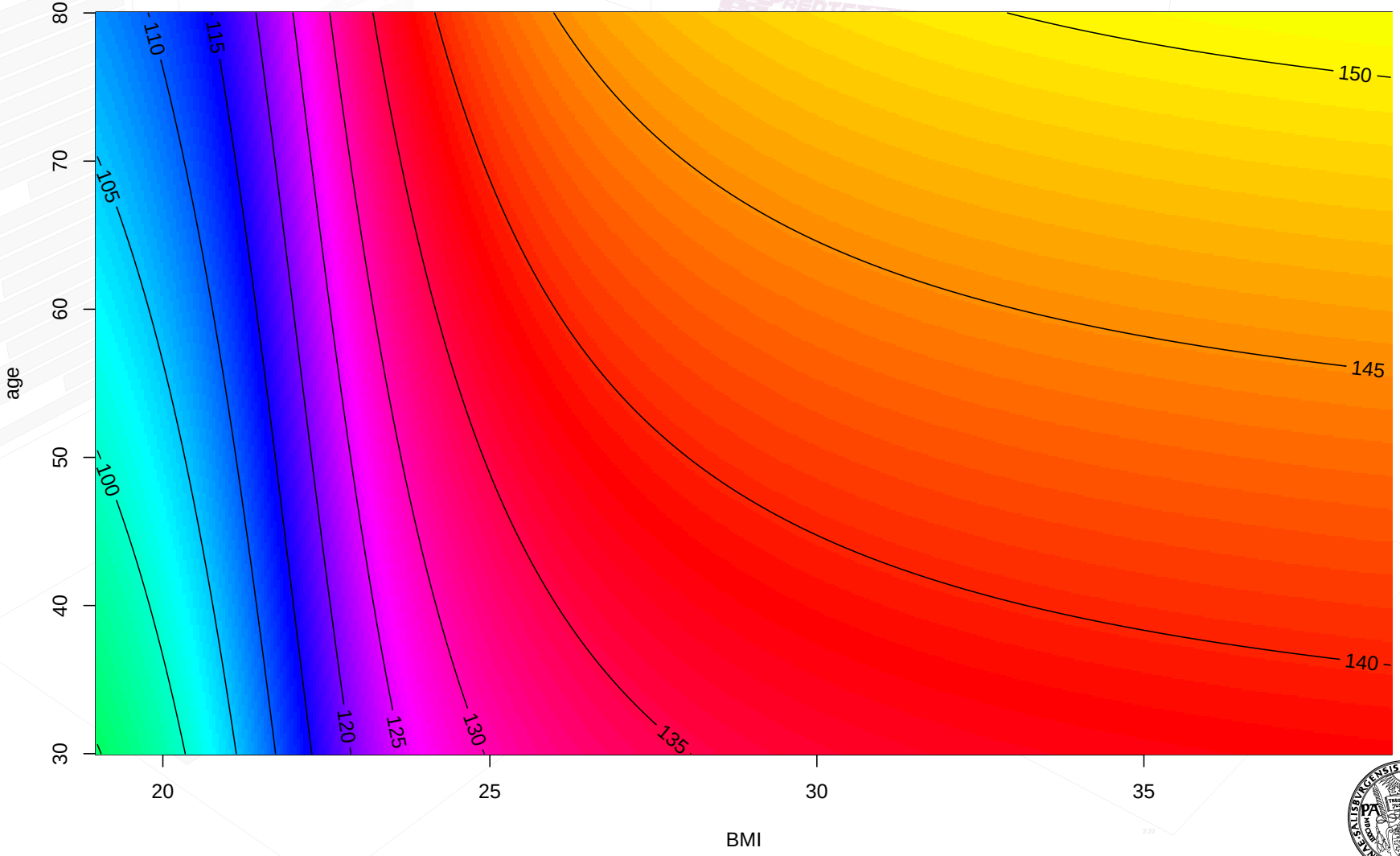
- ```
  Parameters:
        Estimate   Std. Error t value Pr(>|t|)
  a     0.252004   0.003506   71.87   <2e-16 ***
  b    17.830446   0.178180  100.07   <2e-16 ***
  c     0.510601   0.010566   48.33   <2e-16 ***
  d   -11.236677   0.240598  -46.70   <2e-16 ***
  ---
  Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1 1
  ```

- as well as

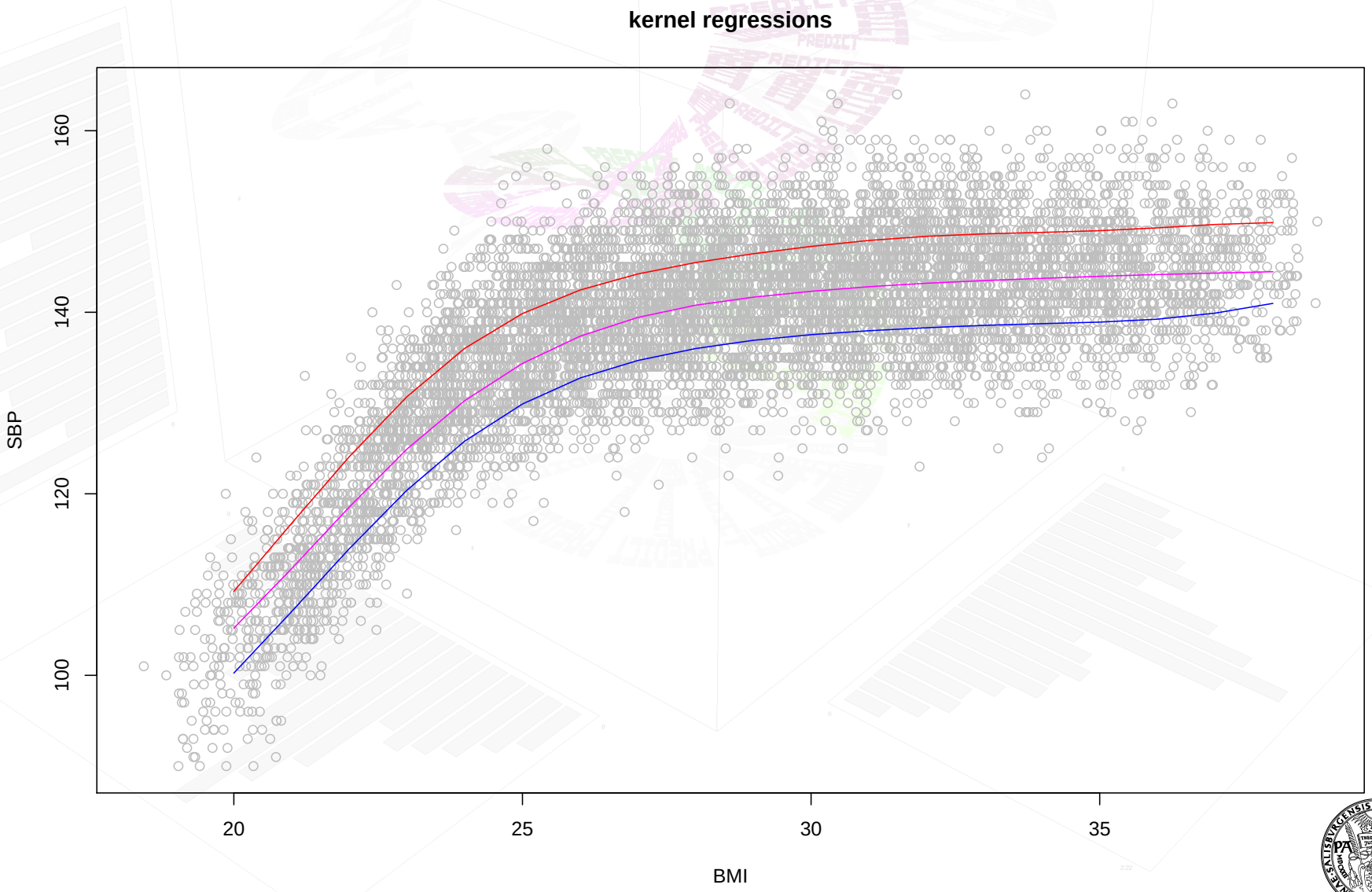
```
Number of iterations to convergence: 6
Achieved convergence tolerance: 5.443e-07


#image plot of the regression function
pars <- as.numeric(coef(model.nls))
reg <- function(BMI,age){
  r <- 105 + pars[1]*age + pars[2]*atan(pars[3]*BMI + pars[4])
  return(r)
```
```

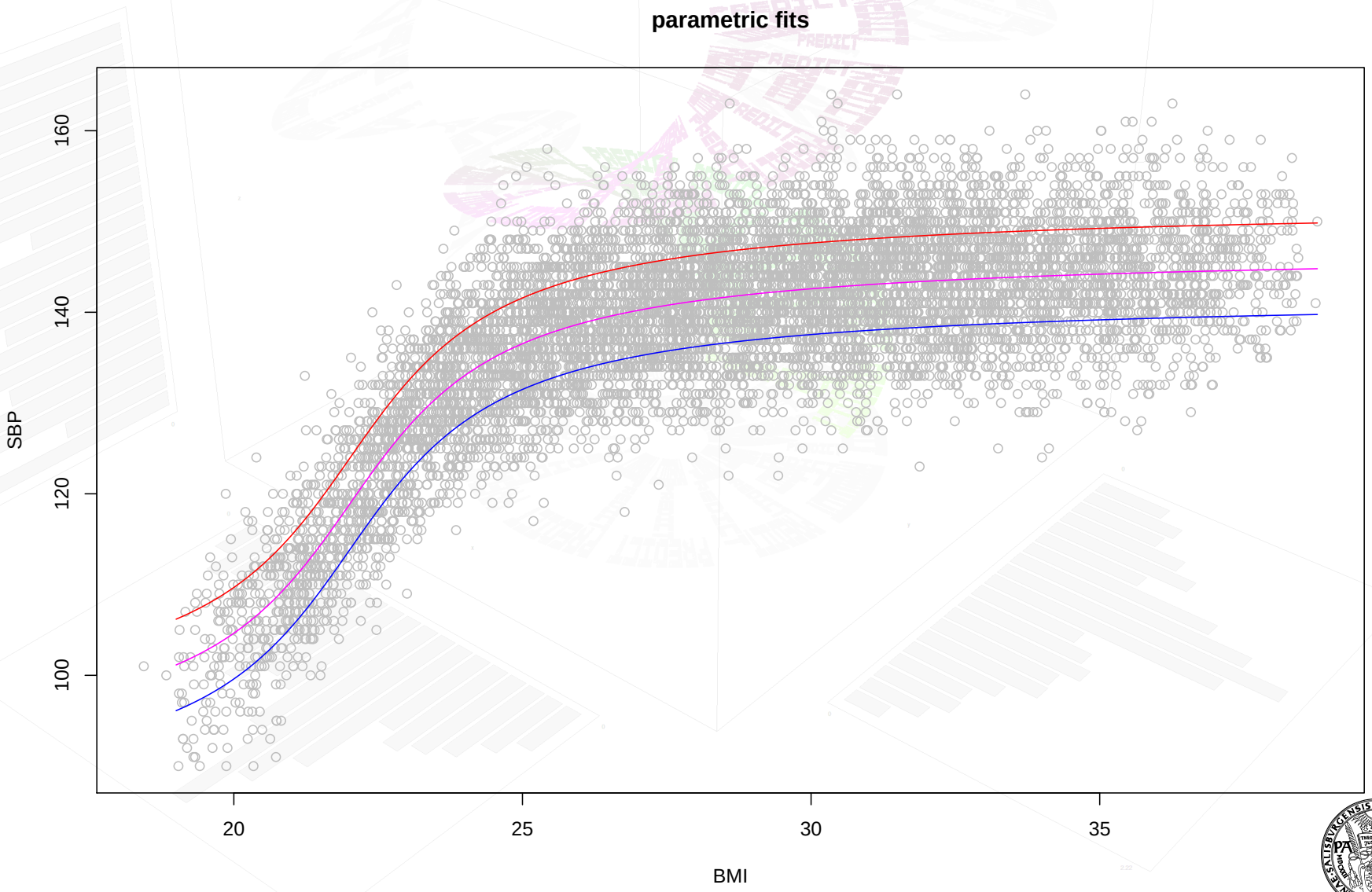
The function nls, cont.



The function nls, cont.



The function nls, cont.



A word of caution

- *nls* tries to find the absolute minimum of a function based on numerical methods.
- Hence good starting values (initial guesses) for the parameters are important - otherwise the algorithm might run into local minima or even produce no results (error message "Singular gradient matrix at initial parameter estimates").
- It often helps to first try out some parameter values and calculate the sum of the squared residuals F and then run *nls* with those initial parameters which resulted in the smallest value of F.

Outlook:

- Sometimes the parameters also have to fulfill certain boundary conditions.
- In this case the *optim* function can be used.

Exercise 13:

- The data set *beer.txt* contains foam height at various time points for three brands of beer.
- Use lines 353-354 to load the data.
- Fit an individual regression model of the form

$$H = H_0 \, e^{-at}$$

  to the data of each brand. Thereby H denotes the foam height, t the time (in seconds), and H_0 and a are parameters.
- According to the models, which brand is expected to have the highest foam at time point $t = 480$?
- Include graphics, the resulting regression models as well as the answer to the afore-mentioned question in a short knitR report.