Serie 2

Aufgabe 2.1

Im Wikipedia-Artikel

https://en.wikipedia.org/wiki/Heights_of_presidents_and_presidential_candidates_of_the_United_States

sind die Körpergrössen der US-Präsidenten und ihrer Herausforder bei den Wahlen aufgeführt. Es wurde festgestellt, dass der grössere Präsidentschaftskandidat gewöhnlich die Wahlen gewinnt.

In dieser Übung untersuchen wir die Daten der Präsidentschaftswahlen, seit diese im Fernsehen übertragen werden. In Tabelle 1 sind die Körpergrössen aufgeführt.

Year	Winner	Height	Opponent	Height
2016	Donald Trump	191 cm	Hillary Clinton	165 cm
2012	Barack Obama	185 cm	Mitt Romney	187 cm
2008	Barack Obama	185 cm	John McCain	175 cm
2004	George W. Bush	182 cm	John Kerry	193 cm
2000	George W. Bush	182 cm	Al Gore	185 cm
1996	Bill Clinton	188 cm	Bob Dole	187 cm
1992	Bill Clinton	188 cm	George H. W. Bush	188 cm
1988	George H. W. Bush	188 cm	Michael Dukakis	173 cm
1984	Ronald Reagan	185 cm	Walter Mondale	180 cm
1980	Ronald Reagan	185 cm	Jimmy Carter	177 cm
1976	Jimmy Carter	177 cm	Gerald Ford	183 cm
1972	Richard Nixon	182 cm	George McGovern	185 cm
1968	Richard Nixon	182 cm	Hubert Humphrey	180 cm
1964	Lyndon B. Johnson	193 cm	Barry Goldwater	180 cm
1960	John F. Kennedy	183 cm	Richard Nixon	182 cm
1956	Dwight D. Eisenhower	179 cm	Adlai Stevenson	178 cm
1952	Dwight D. Eisenhower	179 cm	Adlai Stevenson	178 cm
1956	Harry S. Truman	175 cm	Thomas Dewey	173 cm

Table 1: Körpergrössen der Präsidenten und ihrer Herausforderer seit 1948

Die Werte sind in der Datei `president.csv` abgelegt.

a) Lesen Sie die Datei ein:

```
import pandas as pd
import matplotlib.pyplot as plt
pres = pd.read_csv(r"*/president.csv", index_col=0)
```

Der $*$ steht wieder für den Pfad, wo die Datei gespeichert ist.

Erklären Sie was die Option **`index_col=0`** erreicht. Verwenden Sie dazu die Methode **`.head()`**.

b) Bestimmen die Anzahl der Zeilen mit dem Attribut **`.shape`**

c) Die Washington Post hat festgestellt, dass die Einträge für Bill Clinton (Jahr 1992 und 1996) zu klein sind. Er misst 189 cm. Ändern Sie die Einträge im Vektor **`Height_win`** entsprechend ab und geben Sie die neue Tabelle nochmals aus.

d) Die Behauptung ist, dass die Gewinner grösser sind als die Herausforderer. Überprüfen Sie dies, indem Sie die Mittelwerte der Vektoren miteinander vergleichen.

e) Bestimmen Sie den durchschnittlichen Grössenunterschied.

Aufgabe 2.2

In einer Klasse wurden in einer Statistik-Prüfung folgende Noten geschrieben:

4.2, 2.3, 5.6, 4.5, 4.8, 3.9, 5.9, 2.4, 5.9, 6, 4, 3.7, 5, 5.2, 4.5, 3.6, 5, 6, 2.8, 3.3, 5.5, 4.2, 4.9, 5.1

a) Ändern Sie drei Noten im Datensatz so ab, dass der Median gleich bleibt, aber der Mittelwert sich stark ändert.

b) Erstellen Sie zu den beiden Datensätzen je ein Histogramm und einen Boxplot. Verwenden Sie **`plt.subplot(...)`**.

Aufgabe 2.3

Wir haben aus eigener Erfahrung das Gefühl, dass bei Ehepaaren der Mann eher älter als die Frau ist. Nun wollen wir statistisch untersuchen, ob dem so ist.

In einer Untersuchung in England wurden das Alter (in Jahren) und die Körpergrösse (in cm) von 170 Ehepaaren untersucht.

a) Lesen Sie die Datei `husband_wive.csv` ein.

```
import pandas as pd
import matplotlib.pyplot as plt

couples = pd.read_csv(r"*/husband_wive.csv")
```

Der $*$ steht wieder für den Pfad, wo die Datei abgespeichert ist.

Überprüfen Sie mit der Methode **`head`**, ob die Datei richtig eingelesen wurde.

b) Führen Sie die **`describe`**-Methode aus. Erklären Sie, was der Befehl macht und interpretieren Sie die Werte für das Alter von Ehemann und Ehefrau.

```
couples.describe()
```

c) Erstellen Sie einen Boxplot für die *Differenz* des Alters zwischen Ehemännern und Ehefrauen. Erzeugen Sie dazu einen Vektor **`diff`** für die Differenz.

d) Interpretieren Sie im Boxplot den Median und die Quartile. Was können Sie über die Ausreisser sagen?

Aufgabe 2.4

21 Labors bestimmten den Kupfergehalt von 9 verschiedenen Klärschlammproben. Die Daten stehen in der auf Ilias abgelegten Datei `klaerschlamm.dat` zur Verfügung. Die erste Spalte bezeichnet das Labor, die restlichen 9 Spalten sind die verschiedenen Klärschlammproben. Die Daten (in mg/kg) können mit dem Befehl

```
schlamm = pd.read_csv("*klaerschlamm.dat", sep=" ", index_col=0)
```

eingelesen werden (für * wieder der Dateipfad). Die erste Spalte `Labor` wollen wir noch entfernen, da sie uns nicht interessiert.

```
schlamm = schlamm.drop("Labor",1)
schlamm.head()
```

a) Erstellen Sie für jede Probe einen Boxplot, und berechnen Sie jeweils das arithmetische Mittel und den Median. Bei welchen Proben gibt es Ausreisser, und wo unterscheiden sich arithmetisches Mittel und Median wesentlich? Bei welchen der 9 Proben ist es plausibel, dass die wahre Konzentration unter 400 mg/kg liegt?

`Python`-Hinweise:

```
schlamm.describe()
schlamm.plot(kind="box")
```

b) Erstellen Sie für jedes Labor einen Boxplot der Messfehler. Unter dem Messfehler eines Labors bei einer Probe verstehen wir den gemessenen Wert minus den Median über alle Labors. Welche der 21 Labors haben systematische Fehler in ihrem Analyseverfahren? Welche haben grosse Zufallsfehler, und bei welchen Labors ist die Qualität der Analysen besonders gut?

`Python`-Hinweise: Wir ziehen zunächst von jeder Spalte den Median ab

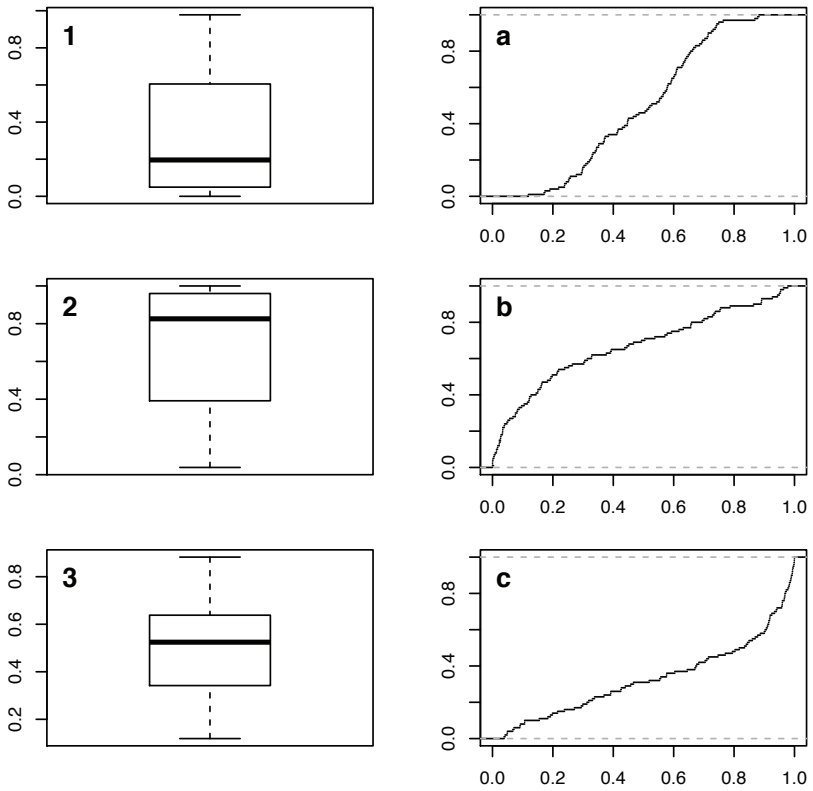
```
schlamm_centered = schlamm - schlamm.median()
```

und zeichnen den Boxplot mit vertauschten Zeilen und Spalten. Dies geschieht mit dem Attribut **`.T`**

```
schlamm_centered.T.plot(kind="box")
```

Aufgabe 2.5

Für drei Stichproben vom Umfang $n = 100$ wurden je ein Boxplot und die empirische Verteilungsfunktion gezeichnet. Ordnen Sie die drei Boxplots den entsprechenden empirischen Verteilungsfunktionen zu:



Kurzlösungen vereinzelter Aufgaben

Musterlösungen zu Serie 2

Lösung 2.1

a)
```
import pandas as pd
import matplotlib.pyplot as plt
pres = pd.read_csv("../../../Themen/Deskriptive_Statistik/Uebungen_de/Daten/president.csv",index_col =
pres.head()

##             Winner   ...
## Year                 ...
## 2016   Donald Trump  ...
## ...              ...  ...
##
## [5 rows x 4 columns]

print(pres.head())

##             Winner   ...
## Year                 ...
## 2016   Donald Trump  ...
## ...              ...  ...
##
## [5 rows x 4 columns]
```

Ohne **`index_col=0`** erhalten wir

```
import pandas as pd
import matplotlib.pyplot as plt
pres = pd.read_csv("../../../Themen/Deskriptive_Statistik/Uebungen_de/Daten/president.csv")
pres.head()

##     Year   ...
## 0    2016  ...
## ..    ...  ...
##
## [5 rows x 5 columns]

print(pres.head())

##     Year   ...
## 0    2016  ...
## ..    ...  ...
##
## [5 rows x 5 columns]
```

Das heisst, mit **`index_col=0`** wird erreicht, dass die erste Spalte (Indexierung beginnt bei 0!) als Index gewählt wird.

b)
```
import pandas as pd
import matplotlib.pyplot as plt
pres = pd.read_csv("../../../Themen/Deskriptive_Statistik/Uebungen_de/Daten/president.csv", index_col =
pres.shape

## (18, 4)

print(pres.shape)

## (18, 4)
```

Die Tabelle enthält 18 Präsidentschaftswahlen.

c)
```
import pandas as pd
import matplotlib.pyplot as plt
pres = pd.read_csv("../../../Themen/Deskriptive_Statistik/Uebungen_de/Daten/president.csv", index_col =
pres.loc[[1996, 1992], "Height_win"] = 189
pres

##             Winner   ...
## Year                 ...
## 2016   Donald Trump  ...
## ...              ...  ...
##
## [18 rows x 4 columns]

print(pres)

##             Winner   ...
## Year                 ...
## 2016   Donald Trump  ...
## ...              ...  ...
##
## [18 rows x 4 columns]
```

d)
```
import pandas as pd
import matplotlib.pyplot as plt
pres = pd.read_csv("../../../Themen/Deskriptive_Statistik/Uebungen_de/Daten/president.csv", index_col =
pres.loc[[1996, 1992], "Height_win"] = 189
pres.mean()

## Height_win    183.944444
##                   ...
## Length: 2, dtype: float64

print(pres.mean())

## Height_win    183.944444
##                   ...
## Length: 2, dtype: float64
```

Die Gewinner haben mit 184 cm einen leicht grösseren Durchschnitt als die Herausforderer mit 180.5 cm. Das heisst aber noch lange nicht, dass die Gewinner auch wirklich grösser sind. Es wäre ja möglich, dass es einige wenige sehr grosse Gewinner hatte, die den Schnitt nach oben ziehen, aber bei allen anderen der Herausforderer grösser ist.

e)
```
import pandas as pd
import matplotlib.pyplot as plt
pres = pd.read_csv("../../../Themen/Deskriptive_Statistik/Uebungen_de/Daten/president.csv", index_col =
pres.loc[[1996, 1992], "Height_win"] = 189
diff = pres["Height_win"] - pres["Height_opp"]
diff.mean()

## 3.4444444444444446

print(diff.mean())

## 3.4444444444444446
```

Der Gewinner ist also durchschnittlich etwa 3.5 cm grösser als der Herausforderer.

Es *scheint*, dass die Gewinner in der Tat grösser sind als die Herausforderer. Wir werden allerdings später sehen, dass dieser Unterschied eher zufällig ist.

Lösung 2.2

a) (zu **R**) Der ursprüngliche Datensatz hat für den Median und Mittelwert folgende Werte:

```
from pandas import Series
n1 = Series([4.2, 2.3, 5.6, 4.5, 4.8, 3.9, 5.9, 2.4, 5.9,
             6, 4, 3.7, 5, 5.2, 4.5, 3.6, 5, 6, 2.8, 3.3,
             5.5, 4.2, 4.9, 5.1])
n1.median()

## 4.65

n1.mean()

## 4.5125

print(n1.median())

## 4.65

print(n1.mean())

## 4.5125
```

Zuerst ordnen wir die Datenwerte der Grösse nach: (zu **R**)

```
from pandas import Series
n1 = Series([4.2,2.3,5.6,4.5,4.8,3.9,5.9,2.4,5.9,
             6,4,3.7,5,5.2,4.5,3.6,5,6,
             2.8,3.3,5.5,4.2,4.9,5.1])

n2 = n1.sort_values()
n2

## 1    2.3
##     ...
## Length: 24, dtype: float64

print(n2)

## 1    2.3
##     ...
## Length: 24, dtype: float64
```

Da die Anzahl Noten gerade ist, (zu **R**)

```
from pandas import Series
n1 = Series([4.2,2.3,5.6,4.5,4.8,3.9,5.9,2.4,5.9,
             6,4,3.7,5,5.2,4.5,3.6,5,6,
             2.8,3.3,5.5,4.2,4.9,5.1])

n2 = n1.sort_values()
n2.size

## 24

print(n2.size)

## 24
```

wird der Median aus dem Mittelwert von $x_{(12)}$ und $x_{(13)}$ gebildet. Wenn wir also Noten kleiner als $x_{(12)}$ abändern, wird sich der Median nicht ändern. Dementsprechend ändern wir die Notenwert $x_{(9)}, x_{(10)}, x_{(11)}$ zu einer eins. Dies lässt den Median unverändert, lässt den Mittelwert aber maximal schrumpfen.

Die **`Series`** `n2` sieht wie folgt aus:

```
from pandas import Series
import numpy as np
n1 = Series([4.2,2.3,5.6,4.5,4.8,3.9,5.9,2.4,5.9,
             6,4,3.7,5,5.2,4.5,3.6,5,6,
             2.8,3.3,5.5,4.2,4.9,5.1])
n2 = n1.sort_values()
n2.head()

## 1    2.3
##     ...
## Length: 5, dtype: float64

print(n2.head())

## 1    2.3
##     ...
## Length: 5, dtype: float64
```

Der Index wurde also mit sortiert. Damit wir auf die 9., 10. und 11. Werte von `n2` zugreifen können, müssen wird den Index von `n2` ändern.

```
from pandas import Series
import numpy as np
n1 = Series([4.2,2.3,5.6,4.5,4.8,3.9,5.9,2.4,5.9,
             6,4,3.7,5,5.2,4.5,3.6,5,6,
             2.8,3.3,5.5,4.2,4.9,5.1])
n2 = n1.sort_values()
n2.index = np.arange(1, n2.size+1)
n2.head()

## 1    2.3
##     ...
## Length: 5, dtype: float64

print(n2.head())

## 1    2.3
##     ...
## Length: 5, dtype: float64
```

Nun können wir den 9., 10. und elften Wert auf 1 setzen:

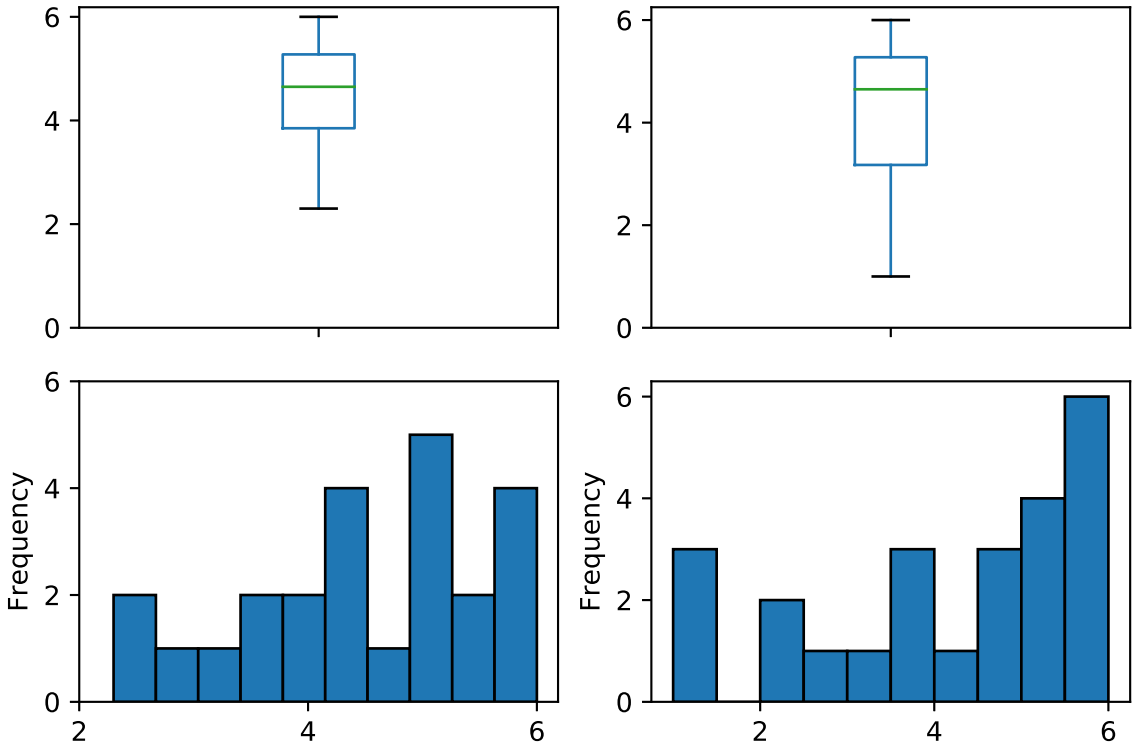
```
from pandas import Series
import numpy as np
n1 = Series([4.2,2.3,5.6,4.5,4.8,3.9,5.9,2.4,5.9,
             6,4,3.7,5,5.2,4.5,3.6,5,6,
             2.8,3.3,5.5,4.2,4.9,5.1])
n2 = n1.sort_values()
n2.index = np.arange(1, n2.size+1)
n2[11], n2[10], n2[9] = 1, 1, 1
n2.head(n=12)

## 1    2.3
##     ...
## Length: 12, dtype: float64
```

```
print(n2.head(n=12))

## 1    2.3
##     ...
## Length: 12, dtype: float64
```

Nun ist (zu R)

```
from pandas import Series
import numpy as np
n1 = Series([4.2,2.3,5.6,4.5,4.8,3.9,5.9,2.4,5.9,
             6,4,3.7,5,5.2,4.5,3.6,5,6,
             2.8,3.3,5.5,4.2,4.9,5.1])
n2 = n1.sort_values()
n2.index = np.arange(1, n2.size+1)
n2[11], n2[10], n2[9] = 1, 1, 1
n2.median()

## 4.65

n2.mean()

## 4.1000000000000005

print(n2.median())

## 4.65

print(n2.mean())

## 4.1000000000000005
```

b) Der Plot (die Skizzen wurden hier noch skaliert).




```
plt.subplot(221)
n1.plot(kind="box")

plt.subplot(222)
n2.plot(kind="box")

plt.subplot(223)
n1.plot(kind="hist",edgecolor="black")

plt.subplot(224)
n2.plot(kind="hist",edgecolor="black")

plt.show()
```

Lösung 2.3

a)
```
import pandas as pd
import matplotlib.pyplot as plt
couples = pd.read_csv("../../../Themen/Deskriptive_Statistik/Uebungen_de/Daten/husband_wive.csv")
couples.head()

##     age.husband  ...
## 0            49  ...
## ..          ...  ...
##
## [5 rows x 4 columns]

print(couples.head())
```

```
##     age.husband  ...
## 0            49  ...
## ..          ...  ...
##
## [5 rows x 4 columns]
```

b)

```
import pandas as pd
import matplotlib.pyplot as plt
couples = pd.read_csv("../../../Themen/Deskriptive_Statistik/Uebungen_de/Daten/husband_wife.csv")
couples.describe()

##        age.husband  ...
## count        170.0  ...
## ...            ...  ...
##
## [8 rows x 4 columns]

print(couples.describe())

##        age.husband  ...
## count        170.0  ...
## ...            ...  ...
##
## [8 rows x 4 columns]
```

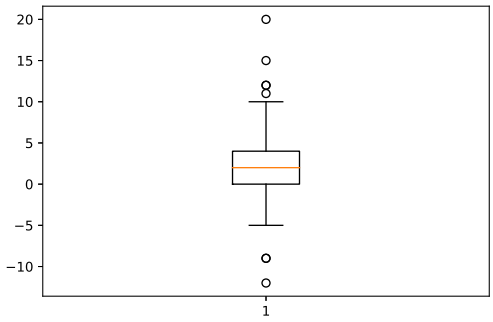
Mit der Methode **`.describe()`** erhalten wir eine kurze statistische Übersicht über die Daten. Es wird die Anzahl der Daten aufgeführt (**`count`**), der Mittelwert (**`mean`**), die Standardabweichung (**`std`**), der kleinste Wert (**`min`**), das untere Quartil (25 %), der Median (50 %), das obere Quartil (75 %) und der maximale Wert (**`max`**).

Für die 170 Ehemänner ist 20 das kleinste und 64 das grösste Alter. Das Durchschnittsalter ist knapp 43 Jahre mit einer Standardabweichung von knapp 12 Jahren. Das heisst, das Alter liegt „durchschnittlich" zwischen 31 und 55 Jahren. Das untere Quartil ist 33 Jahre, also sind 25 % der Ehemänner jünger als 33 Jahre und 75 % älter als 33 Jahre. Der Median ist 43.5 Jahre, also sind 50 % der Ehemänner jünger als 43.5 Jahre und 50 % älter als 43.5 Jahre. Das obere Quartil ist 53 Jahre, also sind 75 % der Ehemänner jünger als 53 Jahre und 25 % älter als 53 Jahre.

Die Zahlen für die Ehefrauen lassen sich analog interpretieren. Ein kurzer Blick zeigt allerdings, dass die Frauen eher ein bisschen jünger sind als die Männer. Dies heisst allerdings *nicht*, dass die Ehemänner allgemein auch älter sind als ihre Frauen.

c)

```
diff = couples["age.husband"] - couples["age.wive"]

plt.boxplot(diff)
```

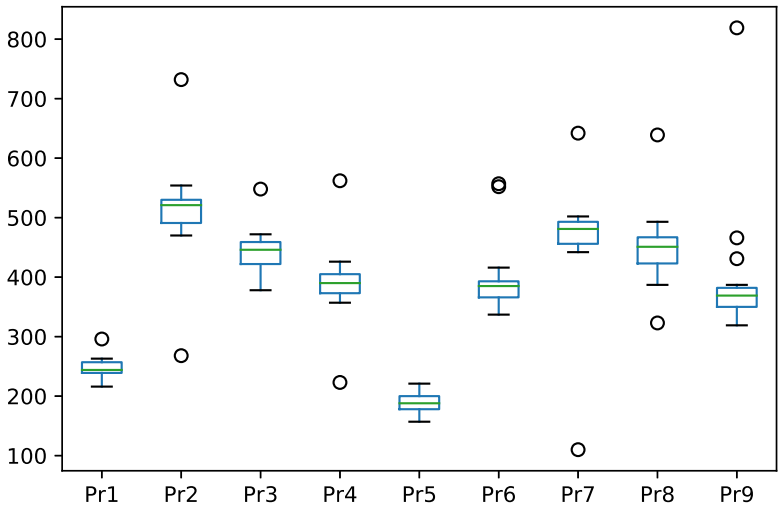


d) Das untere Quartil ist etwa bei 0 Jahren, der Median ist etwa 2 und das obere Quartil etwa 4 Jahre. Das heisst bei 50 % der Ehepaare ist der Ehemann bis zu 4 Jahre älter als die Ehefrau. Bei 25 % der Ehepaare ist die Ehefrau älter als der Ehemann (negative Differenz).

Es gibt einige Ausreisser. In einem Fall ist die Ehefrau über 10 Jahre älter als der Ehemann und in einem Fall ist der Ehemann 20 Jahre älter als die Ehefrau.

Lösung 2.4

a) Aus den Boxplots erkennen wir, dass es vor allem bei den Proben 2, 4, 6, 7, 8 und 9 Ausreisser gibt. Das arithmetische Mittel und der Median unterscheiden wesentlich bei den Proben 2, 6, 7 und 9.



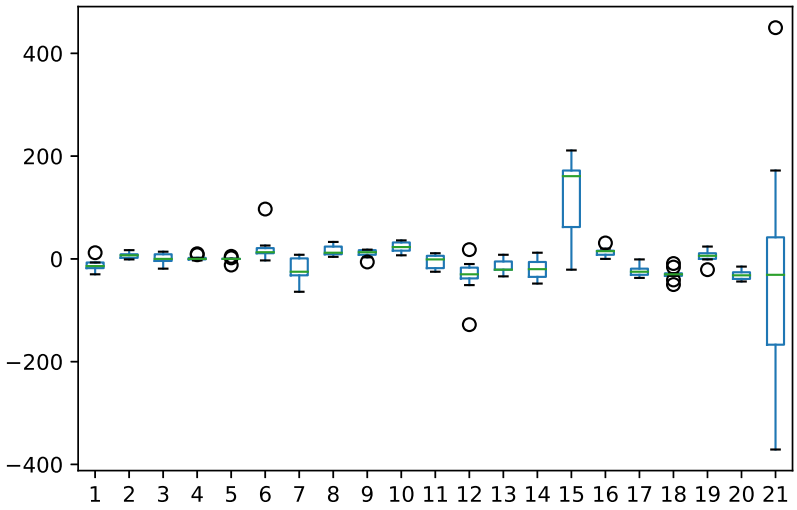
(zu **R**)

```
schlamm.describe()
```

Bei den Proben 1 und 5 ist es plausibel, dass die Konzentration unter 400 mg/kg liegt, während wir bei Probe 2, 3, 7 und 8 eher dazu tendieren, den Grenzwert 400 mg/kg als überschritten zu betrachten.

Die übrigen Proben, Probe 4, 6 und 9 sind eher Grenzfälle. Die Konzentrationen scheinen zwar unter 400 mg/kg zu liegen, die drei Proben weisen jedoch jeweils extreme Ausreisser über dem Grenzwert auf.

b) Als erstes stechen die Messungen der Labors 15 und 21 ins Auge. Beide haben sowohl eine grosse Standardabweichung als auch systematische Fehler. Die Labors 6 und 12 haben beide Ausreisser zu verzeichnen. Die Labors 1, 7, 12, 13, 14, 17, 18, 20 und 21 geben systematisch zu kleine Werte an, während die Labors 6, 8, 10 und 15 zu grosse Werte erhalten. Die Labors 2, 3, 4, 5 und 19 scheinen zuverlässige Untersuchungen durchzuführen. Sowohl systematische wie auch Zufallsfehler scheinen sich hier in Grenzen zu halten (zu **R**)



Lösung 2.5

1b, 2c, 3a

R-Code

Aufgabe 2.2

a) (zu **Python**)

```
noten.1 <- c(4.2, 2.3, 5.6, 4.5, 4.8, 3.9, 5.9, 2.4, 5.9,
    6, 4, 3.7, 5, 5.2, 4.5, 3.6, 5, 6, 2.8, 3.3, 5.5, 4.2,
    4.9, 5.1)
median(noten.1)

## [1] 4.65

mean(noten.1)

## [1] 4.5125
```

(zu **Python**)

```
sort(noten.1)

##  [1] 2.3 2.4 2.8 3.3 3.6 3.7 3.9 4.0 4.2 4.2 4.5
## [12] 4.5 4.8 4.9 5.0 5.0 5.1 5.2 5.5 5.6 5.9 5.9
## [23] 6.0 6.0
```

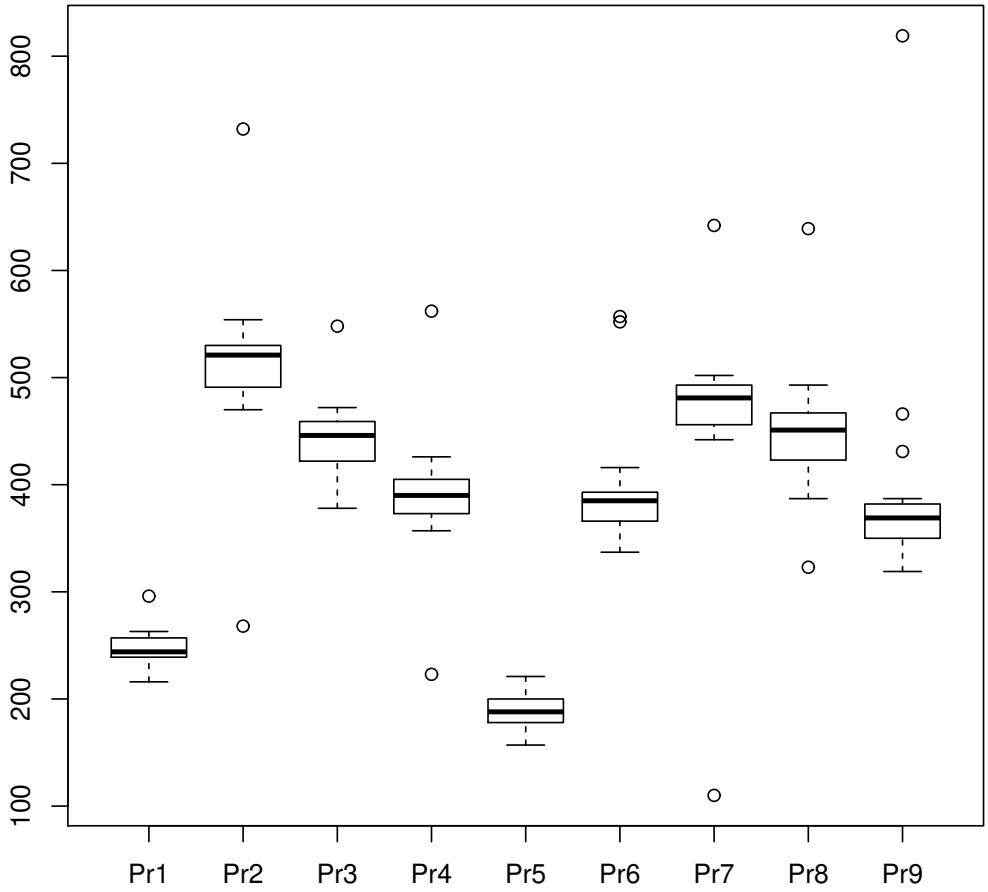
```
noten.2 <- c(1, 2.3, 5.6, 4.5, 4.8, 3.9, 5.9, 2.4, 5.9,
    6, 4, 3.7, 5, 5.2, 1, 3.6, 5, 6, 2.8, 3.3, 5.5, 1, 4.9,
    5.1)
median(noten.2)

## [1] 4.65

mean(noten.2)

## [1] 4.1
```

Aufgabe 2.4

a) (zu **Python**)

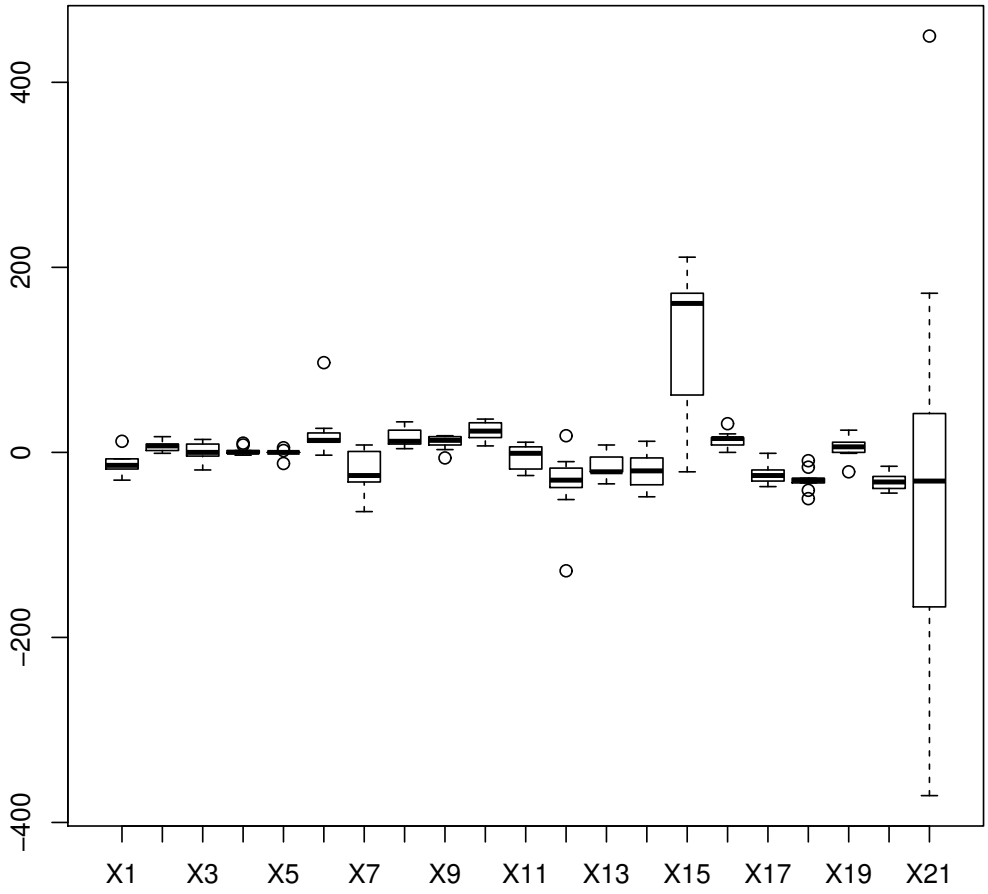
```
schlamm.all <- read.table(file = "./Daten/klaerschlamm.dat",
    header = TRUE)
schlamm <- schlamm.all[, -1]  # Labor-Spalte entfernen
summary(schlamm)

##       Pr1             Pr2             Pr3
##  Min.   :216.0   Min.   :268.0   Min.   :378.0
##  1st Qu.:239.0   1st Qu.:491.0   1st Qu.:422.0
##  Median :244.0   Median :521.0   Median :446.0
##  Mean   :246.1   Mean   :511.4   Mean   :443.4
##  3rd Qu.:257.0   3rd Qu.:530.0   3rd Qu.:459.0
##  Max.   :296.0   Max.   :732.0   Max.   :548.0
##       Pr4             Pr5             Pr6
##  Min.   :223.0   Min.   :157.0   Min.   :337.0
##  1st Qu.:373.0   1st Qu.:178.0   1st Qu.:366.0
##  Median :390.0   Median :188.0   Median :385.0
##  Mean   :389.2   Mean   :188.2   Mean   :394.9
##  3rd Qu.:405.0   3rd Qu.:200.0   3rd Qu.:393.0
```

```
##  Max.   :562.0   Max.   :221.0   Max.   :557.0
##       Pr7             Pr8             Pr9
##  Min.   :110.0   Min.   :323   Min.   :319.0
##  1st Qu.:456.0   1st Qu.:423   1st Qu.:350.0
##  Median :481.0   Median :451   Median :369.0
##  Mean   :465.5   Mean   :450   Mean   :388.9
##  3rd Qu.:493.0   3rd Qu.:467   3rd Qu.:382.0
##  Max.   :642.0   Max.   :639   Max.   :819.0

boxplot(schlamm)
```



b)

```
# Fuer jede Spalte Median berechnen
med <- apply(schlamm, 2, median)
# Median von jeder *Spalte* abziehen
schlamm.centered <- scale(schlamm, scale = FALSE, center = med)
# Boxplot zeichnen. Dazu zuerst data-frame
# transponieren
boxplot(data.frame(t(schlamm.centered)))
```


400
200
0
−200
−400
X1
X3
X5
X7
X9
X11
X13
X15
X17
X19
X21