

Musterlösung zur R-Einführungsübung

1. Visualisierung von Daten und Berechnung von Kenngrößen

a) Siehe Aufgabenstellung.

b) Um die Daten in Tabellenform zu sehen, tippt man den Namen des Objektes ein.

```
> d.fuel
  X weight mpg  type
1  1  2560  33 Small
2  2  2345  33 Small
3  3  1845  37 Small
4  4  2260  32 Small
5  5  2440  32 Small
:  :      :    :
:  :      :    :
59 59  3185  20  Van
60 60  3690  19  Van
```

c) Auswählen der fünften Beobachtung:

```
> d.fuel[5,]
  X weight mpg  type
5  5  2440  32 Small
```

d) Auswählen der 1. bis 5. Beobachtung:

```
> d.fuel[1:5,]
  X weight mpg  type
1  1  2560  33 Small
2  2  2345  33 Small
3  3  1845  37 Small
4  4  2260  32 Small
5  5  2440  32 Small
```

e) Auswählen der 1. bis 3. und 57. bis 60. Beobachtung:

```
> d.fuel[c(1:3,57:60),]
  X weight mpg  type
1  1  2560  33 Small
2  2  2345  33 Small
3  3  1845  37 Small
57 57  3735  19  Van
58 58  3415  20  Van
59 59  3185  20  Van
60 60  3690  19  Van
```

f) Die Werte der Reichweiten stehen in der dritten Spalte, die mpg heisst. Zur Berechnung des Mittelwertes gibt es verschiedene Möglichkeiten, welche sich in der Art der Datenselektion unterscheiden:

```
> mean(d.fuel[,3])
[1] 24.58333
> mean(d.fuel[, "mpg"])
[1] 24.58333
> mean(d.fuel$mpg)
[1] 24.58333
```

g) Auch hier gibt es wieder verschiedene Möglichkeiten. Eine davon ist:

```
> mean(d.fuel[7:22, "mpg"])
[1] 27.75
```

h) Umrechnung der Miles Per Gallon in Kilometer pro Liter und der Pounds in Kilogramm:

```
> t.kml <- d.fuel[, "mpg"]*1.6093/3.789
> t.kg <- d.fuel[, "weight"]*0.45359
```

i) Mittelwert der Reichweite und des Gewichtes:

```
> mean(t.kml)
[1] 10.44127
> mean(t.kg)
[1] 1315.789
```

Der Mittelwert der Reichweite kann auch wie folgt berechnet werden (siehe Stat. Datenanalyse, Kapitel 2.6): `> mean(d.fuel[, "mpg"])*1.6093/3.789`

j) Verbrauch als Funktion des Gewichtes:

```
> plot(t.kg, 100/t.kml)
```

k) Stem-and-leaf-Plot des Benzinverbrauchs:

```
> stem(100/t.kml)
```

```
The decimal point is at the |
```

```
6 | 479
7 | 1111448
8 | 1447777
9 | 111114448888
10 | 2222222277777
11 | 22222288888
12 | 444
13 | 1111
```

```
> min(100/t.kml)
[1] 6.363351
> max(100/t.kml)
[1] 13.08022
```

l) Histogramm des Verbrauchs: `hist(100/t.kml)`

Mit 15 Klassen: `hist(100/t.kml, nclass=15)`

```
Mit x-Achse 0 bis 15: hist(100/t.kml,nclass=15,xlim=c(0,15))
Mit Titel: hist(100/t.kml,nclass=15,xlim=c(0,15),main="Verteilung der
Verbraeuche")
```

m) Boxplot der Verbräuche zeichnen: `boxplot(100/t.kml)`

n) Vergleich der Standardabweichung mit dem MAD:

```
> sd(100/t.kml)
[1] 1.783549
> mad(100/t.kml)
[1] 1.751184
> mad(100/t.kml,constant=1)
[1] 1.181157
```

Der Befehl `mad` ohne `constant=1` berechnet einen skalierten MAD, sodass der `mad` bei normalverteilten Daten gerade der Standardabweichung entsprechen würde. Im Buch wurde der MAD ohne Skalierung eingeführt.

o) Vergleich des Mittelwertes mit dem Median:

```
> mean(100/t.kml)
[1] 9.912268
> median(100/t.kml)
[1] 10.23669
```

2. Korrelationen

a) Erzeugen der Vektoren:

```
> t.x <- (-10):10
> t.x1 <- 0:10
> t.y <- t.x^2
> t.y1 <- t.x1^2
```

b) `> par(mfrow=c(1,2)) # zwei Grafiken im Grafikfenster`

```
> plot(t.x,t.y)
> plot(t.x1,t.y1)
```

c) `> cor(t.x,t.y)`

```
[1] 0
> cor(t.x1,t.y1)
[1] 0.9631427
```

Die Korrelation zwischen `t.x` und `t.y` ist 0, weil die Daten symmetrisch zur y-Achse liegen.

Im zweiten Fall ist die Korrelation hoch (0.96), obwohl die Daten keine lineare Beziehung aufweisen. Der Grund dafür ist, dass `x` und `y` monoton steigen. Wenn statt der üblichen Korrelation die Rangkorrelation verwendet worden wäre, würde der Koeffizient exakt 1.0 betragen.