

### Arbeitsblatt 3

1) Variable in R können beliebige Namen haben, die aber mit einem Buchstaben beginnen müssen. Groß- und Kleinschreibung wird unterschieden. Die Zuweisung von Werten zu Variablen erfolgt durch <-:

```
a <- 3; a
```

Die wichtigste Datenstruktur in R ist ein "Vektor", eine Liste von Elementen. Vektoren können durch den Befehl `c` erzeugt werden:

```
a <- c(1,3,5)
```

```
b <- c(a,1,a)
```

Die Länge eines Vektors (die Anzahl seiner Elemente) wird durch `length(b)` berechnet. Fast alle Befehle in R sind "vektorisert" und operieren auf allen Elementen eines Vektors.

```
sin(a)+1
```

Wenn Vektoren unterschiedlicher Länge in einem Ausdruck verwandt werden, dann werden in den Berechnungen die kürzeren Vektoren durch Wiederholung ihrer Werte verlängert, bis sie die Länge des längsten Vektors erreicht haben.

```
d <- 2*a + b + 1
```

2) Eine  $n \times m$  Matrix  $A$  besteht aus  $nm$  rechteckig angeordneten Zahlen mit  $n$  Zeilen und  $m$  Spalten:

$$A = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1m} \\ a_{21} & a_{22} & \dots & a_{2m} \\ \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & \dots & a_{nm} \end{pmatrix}$$

Analog kann man Felder beliebiger Dimension definieren.

Matrizen und Felder beliebiger Dimension werden durch

```
A <- matrix(c(1,2,3,4), nrow=2, ncol=2)
```

```
B <- array(c(1,2,3,4,1,2,3,4), dim=c(2,2,2))
```

erzeugt.

Im Gegensatz zur mathematischen Sprechweise haben in R Vektoren keine Dimension, wohl aber Matrizen und Felder.

```
a <- c(1,2,3,4)
```

```
dim(a)
```

```
dim(B)
```

Elementweise Operationen werden auch bei Matrizen und Feldern für alle Elemente gleichzeitig durchgeführt:

```
2*A + 1
```

```
C <- matrix(c(3,4,1,2),nrow=2)
```

```
A + C
```

Elemente von Vektoren, Matrizen und Feldern können durch die Angabe ihrer Indizes angesprochen werden.

```
a[2]
```

```
a[c(1,2)]
```

```
A[1,1]
```

```
A[,1]
```

```
C[C>2]
```

```
B[1,2,1]
```

Negative Indizes entfernen die entsprechenden Elemente.

```
a <- a[-2]; a
```

3) Sequenzen von ganzen Zahlen werden durch `:` erzeugt.

```
2:5
```

```
-3:3
```

```
10:1
```

`seq()` ist eine allgemeinere Funktion:

```
seq(2,10)
```

```
seq(-2,2,by=0.2)
```

```
seq(-2,length=11,by=0.4)
```

`rep()` wiederholt sein Argument:

```
rep(2,times=4)
```

```
rep(a,2)
```

```
rep(a,times=c(2,1,2))
```

4) Listen sind geordnete Mengen von verschiedenen Objekten:

```
> d <- list(a, c(-1,3,5), "abrakadabra")
```

```
> d
```

```
[[1]]
```

```
[1] 1 3 5
```

```
[[2]]
```

```
[1] -1 3 5
```

```
[[3]]
```

```
[1] "abrakadabra"
```

`unlist()` macht aus einer Liste einen Vektor. Elemente einer Liste können durch Angabe ihres Indexes ausgewählt werden.

```
d[1]
d[[1]]
```

Die erste Form ergibt eine Liste mit den ausgewählten Elementen, die zweite Form die ausgewählten Elemente. Da das erste Element von `d` ein Vektor ist, ist

```
d[[1]][2]
```

das zweite Element des Vektors `a`.

Elemente einer Liste können Namen haben. Einzelne Elemente einer Liste können auch über ihren Namen ausgewählt werden:

```
d <- list(eins=a, zwei=c(-1,3,5), drei="abrakadabra")
names(d)
names(d) <- c("alpha", "beta", "Wort")
names(d)
d$Wort
```

5) Die bisher eingeführten Datentypen lassen sich durch `is.list()`, `is.vector()` und `is.array()` abfragen. Die Speicherart erhält man durch `is.logical()`, `is.integer()` und `is.real()`. `mode()` bzw. `typeof()` liefern einen String mit der Speicherart des Objekts. Wenn es logisch möglich ist, lassen sich Speicherarten und -formen durch `as.logical()`, `as.integer()`, `as.character()`, `as.numeric()` bzw. durch `as.vector()`, `as.list()`, `as.matrix()`, `as.array()` verändern.

6) Die Funktion `runif()` liefert gleichverteilte Pseudozufallszahlen auf dem 0-1 Intervall. Die Funktion `rnorm()` berechnet normalverteilte Pseudozufallszahlen. Insbesondere liefert

```
X <- runif(100)
```

100 gleichverteilte Pseudozufallszahlen, die der Variablen mit dem Namen `X` zugewiesen werden. Man kann die Startwerte des Zufallszahlengenerators durch `set.seed(n)` festlegen, wobei `n` eine ganze Zahl ist.

a) Erzeugen Sie 100 gleichverteilte Pseudozufallszahlen auf dem Intervall  $(-1, 1)$ .

b) Erzeugen Sie 100 Pseudozufallszahlen mit Werten in  $\{1, 2\}$  und den Wahrscheinlichkeiten  $\Pr(\{1\}) = \Pr(\{2\}) = 0.5$ . Hinweis: Der Operator `X < Y` erzeugt den Wert `TRUE`, falls `X < Y` ist, `FALSE` sonst. `as.numeric(X)` verwandelt diese logischen Werte in die Zahlen 1 bzw. 0.

7) `sum()`, `mean()`, `var()`, `sd()` berechnen die Summe, den Mittelwert, die Varianz und die Standardabweichung ihrer Argumente. `cov()` und `cor()` berechnen Kovarianzen und Korrelationen. `summary()` berechnet die Quartile, den Mittelwert und die Extrema, wenn das Argument der Funktion ein numerischer Vektor ist. `plot()` interpretiert die Elemente seiner Argumente als Koordinaten und malt sie in einem Graphikfenster. `hist()` malt ein Histogramm. `density()` berechnet einen Kern-Dichte-Schätzer der Daten.

```
X <- runif(100)
mean(X)
sum(X)/length(X)
var(X)
Y <- X/2 + runif(100)
cov(X,Y)
plot(X,Y)
summary(X)
hist(X)
plot(density(X))
```

a) Erzeugen Sie zwei Variable wie in den Aufgaben 4 a-b mit je 1000 Beobachtungen.

b) Berechnen Sie deskriptive Statistiken für die beiden Variablen.

c) Berechnen Sie die Korrelation der Variablen.

d) Bilden Sie neue Variable durch

```
X <- rnorm(1000)
Y <- rnorm(1000) + X
Z <- rnorm(1000) + X
```

Was ist die Korrelation zwischen `X` und `Y`? Zwischen `Y` und `Z`?