LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Statistische Software (R)

Paul Fink, M.Sc., Eva Endres, M.Sc.

Institut für Statistik
Ludwig-Maximilians-Universität München

Vektoren

Grundlegende Operatoren und Funktionen

Aufruf der Hilfeseiten zu grundlegende Operatoren und Funktionen:

`?Arithmetic`	Grundlegende Operatoren für numerische Vektoren
`?Logic`	Operatoren für logische Vektoren
`?log`	Logarithmus und Exponens
`?Trig`	Trigonnometrische Funktionen
`?Special`	Z.B. Binomialkoeffizienten, Fakultät, etc.

Konstanten

Übersicht einiger Konstanten:

`pi`	Die Zahl π
`Inf`, `-Inf`	$\infty, -\infty$
`NaN`	Not a Number: z.B. $0/0$
`NA`	Not available: fehlende Werte
`NULL`	leere Menge
`letters`	Kleinbuchstaben von a bis z
`LETTERS`	Großbuchstaben von A bis Z

Vektoren in R

DAS Datenobjekt ist ein Vektor mit Elementen des Typs

- **numeric**: ganzzahlige oder Gleitkomma-Werte,
- **character**: beliebige Zeichen,
- **logical**: die Zustände `TRUE` und `FALSE`,
- **list**: ein Objekt beliebigen Typs (rekursive Datenstruktur!). Mehr dazu später.

Jeder Vektor besitzt Elemente **eines!!** Typs und hat eine Länge (`length()`).

Erstellung:
`c(`*`Werte EINES Typs durch Komma getrennt`*`)`

(Daten-)Typen von Vektoren

- Vektor vom Typ **numeric**.

```
> numvec <- c(2.54, 4.22, 2.99, 3.14, 3.44)
> numvec
[1] 2.54 4.22 2.99 3.14 3.44
```

- Vektor vom Typ **character**.

```
> charvec <- c("Statistische", "Software")
> charvec
[1] "Statistische" "Software"
```

- Vektor vom Typ **logical**.

```
> logicvec <- c(TRUE, FALSE, FALSE, TRUE)
> logicvec
[1]  TRUE FALSE FALSE  TRUE
```

Automatische Umwandlung

- Abfrage des Typs eines Vektors: `is.⟨Typ⟩()`

```
> is.numeric(numvec)
[1] TRUE
> is.character(numvec)
[1] FALSE
```

- Struktur eines R-Objekts erfragen: `str()`

```
> str(charvec)
 chr [1:2] "Statistische" "Software"
```

- R wandelt den Typ eines Objektes *automatisch* um, wenn dies notwendig und möglich ist:

```
> TRUE + 2
[1] 3
> c("Hallo", sqrt(3))
[1] "Hallo"            "1.73205080756888"
> c(2, "Hallo", TRUE)
[1] "2"     "Hallo" "TRUE"
```

Automatische Umwandlung

- Beispiele für häufige Umwandlungen sind

logisch	$\Longrightarrow$	numerisch
logisch, numerisch	$\Longrightarrow$	Text
numerisch	$\Longrightarrow$	logisch

Umwandlung erzwingen mit `as.⟨neuerTyp⟩()`

```
> as.numeric(logicvec)
[1] 1 0 0 1
> as.logical(c(0, pi))
[1] FALSE  TRUE
> as.character(c(1, 2, 4, TRUE))
[1] "1" "2" "4" "1"
> as.numeric(charvec)
Warning: NAs introduced by coercion
[1] NA NA
```

Sequenzen – `seq()`

Funktions-Schema: `seq(from, to, by, length)`

- Absteigende Sequenz mit gleicher Schrittweite:

```
> seq(from = 3, to = -2, by = -0.5)
 [1]  3.0  2.5  2.0  1.5  1.0  0.5  0.0 -0.5 -1.5 -1.5 -2.0
```

- Standardschrittweite ist $+1$ oder -1:

```
> 2:4 # entspricht seq(from = 2, to = 4)
[1] 2 3 4
> seq(from = 4, to = 2)
[1] 4 3 2
```

- Sequenzen mit vorgegebener Länge:

```
> seq(to = 10, length = 10, by = 0.5)
 [1]  5.5  6.0  6.5  7.0  7.5  8.0  8.5  9.0  9.5 10.0
```

Wiederholungen – `rep()`

Funktions-Schema: `rep(times, each, length)`

- n-malige Wiederholung eines Objekts:

```
> rep(3.5, times = 10)
 [1] 3.5 3.5 3.5 3.5 3.5 3.5 3.5 3.5 3.5 3.5
> rep(1:4, times = 2)
[1] 1 2 3 4 1 2 3 4
```

- Jedes Vektorelement wird mehrmals hintereinander wiederholt:

```
> rep(1:4, each = 2)
[1] 1 1 2 2 3 3 4 4
> rep(1:4, each = 2, times = 3)
 [1] 1 1 2 2 3 3 4 4 1 1 2 2 3 3 4 4 1 1 2 2 3 3 4 4
```

Faktoren

Nominale / ordinale Merkmale werden in R als „Faktoren“ codiert

- Reihenfolge der Levels festlegen mit Argument `levels`:

```
> x <- factor(c("Saft", "Saft", "Limonade", "Saft", "Wasser"),
+             levels = c("Saft", "Wasser", "Limonade"))
> x
[1] Saft     Saft     Limonade Saft     Wasser
Levels: Saft Wasser Limonade
> levels(x)
[1] "Saft"     "Wasser"   "Limonade"
```

- Typumwandlung eines `character`-Vektors in Faktor mittels `as.factor()`:

```
> x <- c("Apfel", "Birne", "Apfel", "Traube", "Traube", "Kiwi")
> x <- as.factor(x)
> x
[1] Apfel  Birne  Apfel  Traube Traube Kiwi
Levels: Apfel Birne Kiwi Traube
```

Rechnen mit Vektoren

- **Wichtig:** Die meisten Operationen von 2 Vektoren werden komponentenweise durchgeführt!!

- Addition, Subtraktion, Multiplikation, Division, ...

```
> x <- 1:4
> y <- c(4,10,2,0)
> x + y
[1]  5 12  5  4
> x * y
[1]  4 20  6  0
```

- Logische Vergleiche

```
> x < y
[1]  TRUE  TRUE FALSE FALSE
```

Recycling-Regel

R erlaubt auch Rechnen mit Vektoren unterschiedlicher Länge.

```
> x
[1] 1 2 3 4
> x + c(1, 2)
[1] 2 4 4 6
```

entspricht

```
> x + c(1, 2, 1, 2)
[1] 2 4 4 6
```

Fehlende Werte werden aus bestehenden „recycled“.

Recycling-Regel

Funktioniert auch wenn Vektorlängen nicht Vielfache sind, allerdings mit Warnung

```
> x + c(1, 2, 4) # x + c(1, 2, 4, 1)
Warning in x + c(1, 2, 4): longer object length is not a multiple of
shorter object length
[1] 2 4 7 5
```

Zugriff auf Vektorelemente

1. Vektor von positiven Zahlen

```
> x <- 1:10
> x[1:3]
[1] 1 2 3
> x[c(2, 4, 6)]
[1] 2 4 6
```

2. Vektor von negativen Zahlen

```
> x[-(1:5)]
[1]  6  7  8  9 10
```

3. Leerer Index (Spezialfall von negativem Zugriff)

```
> x[]
 [1]  1  2  3  4  5  6  7  8  9 10
```

Zugriff auf Vektorelemente

4. Logischer Vektor

```
> x[(x > 5)]
[1]  6  7  8  9 10
> x[((x %% 2) == 0)]
[1]  2  4  6  8 10
```

5. Vektor von Zeichenketten

   Die Elemente eines Vektors kann man mit Namen versehen. Zugriff auf einzelen Elemente über diesen Namen möglich

```
> xn <- c(Wasser = 1, Saft = 2, Limonade = 3 )
> names(xn)
[1] "Wasser"   "Saft"     "Limonade"
> xn["Saft"]
Saft
   2
```

Aufgaben

1.) Erstellen Sie einen Vektor `ung`, welcher die ersten 15 ungeraden Zahlen enthält die echt größer als 107 sind!

2.) Geben Sie vom Vektor `ung` diejenigen Zahlen aus, die durch 3 teilbar sind und bilden sie die Summe daraus!

3.) Erstellen Sie einen Faktor `faecher`, der 24 mal das Wort `Statistik`, 1 mal `Informatik` und 5 mal `Mathematik` enthält. Statistik, soll dabei an erster Stelle der Faktorlevels stehen.