Sortierung und Vektor Arithmetik

Der `sort` - Befehl

Sortierung mittels des `sort` Befehls:

- ✚ Sortiert die von uns nachgefragte Variable in aufsteigender Reihenfolge

```
x <- c(15, 43, 12, 19)
sort(x)
```

```
[1] 12 15 19 43
```

Problem: Oft möchten wir unseren kompletten Datensatz auf der Basis einer Variablen sortieren, nicht nur die Variable selbst.

Der `order` - Befehl

Sortierung mittels des `order`-Befehls:

- ✚ Gibt den Indexwert wieder, welche den Eingabevektor sortiert

```
x <- c(15, 43, 12, 19)
index <- order(x)
x
```

```
[1] 15 43 12 19
```

```
index
```

```
[1] 3 1 4 2
```

```
x[index]
```

```
[1] 12 15 19 43
```

Der `order` - Befehl

Indizierung nutzen um einen Data Frame nach einer gegebenen Variablen zu sortieren:

- Beispielsweise die Lebenserwartung gegeben dem BIP pro Kopf

```
ind <- order(gapminder$gdpPercap)
gapminder$lifeExp[ind]
```

- Sortierung immer aufsteigend, wenn eine absteigende gewünscht ist, dann kann dies spezifiziert werden

```
ind <- order(gapminder$gdpPercap, decreasing = TRUE)
gapminder$lifeExp[ind]
```

Die Befehle `max` and `which.max`

- Größter Wert einer Variablen herausfinden mit `max`
- Größter Indexwert mit `which.max`

```
data("gapminder")
max(gapminder$gdpPercap)
```

```
[1] 113523.1
```

```
i_max <- which.max(gapminder$gdpPercap)
gapminder$gdpPercap[i_max]
```

```
[1] 113523.1
```

- Gleiches gilt für das Minimum mit `min` und `which.min`.

> Welches ist der maximale/minimale Wert für die Lebenserwartung in den Daten?

Vektor Arithmetik

Gesamtes BIP?

Wie groß ist die Bevölkerung für das Land mit dem höchsten BIP pro Kopf?

```
gapminder$pop[which.max(gapminder$gdpPercap)]
```

```
[1] 212846
```

Das Land mit dem BIP pro Kopf von 113523.1329 Dollar hat eine Bevölkerung von nur 212 846 Personen.

> Wir können uns hier für alle Länder das gesamte BIP berechnen.

In R können wir die Information zum BIP eines Landes direkt berechnen, da arithmetische Operationen in R auf einem Vektor *elementweise* ausgeführt werden.

Rechnen mit zwei Vektoren

Wenn wir zwei Vektoren der gleichen Länge in R multipliziert, dann werden diese *elementweise* multipliziert.

$$\begin{pmatrix} a \\ b \\ c \\ d \end{pmatrix} * \begin{pmatrix} e \\ f \\ g \\ h \end{pmatrix} = \begin{pmatrix} a * e \\ b * f \\ c * g \\ d * h \end{pmatrix}$$

Dies können wir uns zunutze machen und das BIP für jedes Land durch folgenden Befehl berechnen:

```
BIP <- gapminder$gdpPercap * gapminder$pop
#Wir wollen auf 0 Stellen hinter dem Komma runden
gapminder$BIP <- round(gapminder$gdpPercap * gapminder$pop,0)
```

- Anschließend können wir nach dem BIP sortieren:

```
gapminder$pop[order(BIP)]
```

Unterteilung mit logischen Operatoren

BIP vorhanden, was nun?

- Filtern nach BIP größer oder gleich $1.000.000.000.000

```
h <- BIP >= 1000000000000
```

- Wie oft sehen wir Länder mit einem BIP größer oder gleich 1 Billion Dollar in den Daten?

```
sum(h)
```

```
[1] 66
```

Logische Operatoren

Durch logische Operatoren können wir mehrere Bedingungen miteinander verknüpfen.

- Durch `&` können wir mehrere Bedingungen verbinden
- Durch `|` mehrere entweder, *oder* Bedingungen erzeugen

Beispiel:

- Wie viele Länder haben in 2007 ein BIP größer als 1 Billion Dollar?

```
hohes_BIP <- gapminder$BIP >= 1000000000000
jahr <- gapminder$year == 2007
kombi <- hohes_BIP & jahr
length(gapminder$BIP[kombi])
```

```
[1] 13
```

> Wie ist die Bevölkerungs-Verteilung für Länder mit einem BIP >= 1 Billion Dollar in 2007?

Der Befehl `which`

Effizientere Filterung, sollte nur genau ein Wert gesucht werden:

```
kleinestes_BIP <- which(gapminder$BIP == 52784691)
kleinestes_BIP
```

```
[1] 1297 1298
```

```
gapminder$pop[kleinestes_BIP]
```

```
[1] 60011 61325
```

Der Befehl `match`

Sollen nun mehrere Einträge abgefragt werden kann `match` verwendet werden:

```
einige_laender <- match(c(52784691, 10094200603, 9648014150), gapminder$BIP)
einige_laender
```

```
[1] 1297   19    4
```

Wie groß ist die Bevölkerung für diese Länder?

```
gapminder$pop[einige_laender]
```

```
[1]    60011  2780097 11537966
```

Die Funktion `match` gibt nur den *ersten* Treffer für einen Vektor zurück!

Der Befehl `%in%`

Alternative zu `match` wäre `%in%`:

Vorteil: In Kombination mit `which` können *alle* Treffer gefunden werden, nicht nur der Erste.

```
c(52784691, 10094200603, 9648014150) %in% gapminder$BIP
```

```
[1] TRUE TRUE TRUE
```

```
which(gapminder$BIP %in% c(52784691, 10094200603, 9648014150))
```

```
[1]    4   19 1297 1298
```