Coursera R Programming

Kevin Ostheimer

14.07.2014

Contents

Basistypen

vector()

```
v_vec<-c(1:4,11:14); v_vec
```

```
## [1]  1  2  3  4 11 12 13 14
```

```
v_vec_b<-seq(from = -pi, to = pi, length.out = 10); v_vec
```

```
## [1]  1  2  3  4 11 12 13 14
```

list()

```
l_list<-list(ganzzahl=c(1:4), c("Hallo","Lala")); l_list
```

```
## $ganzzahl
## [1] 1 2 3 4
##
## [[2]]
## [1] "Hallo" "Lala"
```

matrix()

```
m_mat<-matrix(v_vec, nrow=5, ncol=4); m_mat
```

```
## Warning: Datenlänge [8] ist kein Teiler oder Vielfaches der Anzahl der
## Zeilen [5]
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1   12    3   14
## [2,]    2   13    4    1
## [3,]    3   14   11    2
## [4,]    4    1   12    3
## [5,]   11    2   13    4
```

```
dim(m_mat)
```

```
## [1] 5 4
```

```
# Ändern der Dimension einer Matrix
dim(m_mat)<-c(10,2); m_mat
```

```
##       [,1] [,2]
##  [1,]    1    3
##  [2,]    2    4
##  [3,]    3   11
##  [4,]    4   12
##  [5,]   11   13
##  [6,]   12   14
##  [7,]   13    1
##  [8,]   14    2
##  [9,]    1    3
## [10,]    2    4
```

```
# Generierung aus zwei Vectoren
v1<-1:4
v2<-11.1:14.1
cbind(v1,v2)
```

```
##      v1   v2
## [1,]  1 11.1
## [2,]  2 12.1
## [3,]  3 13.1
## [4,]  4 14.1
```

```
rbind(v1,v2)
```

```
##    [,1] [,2] [,3] [,4]
## v1  1.0  2.0  3.0  4.0
## v2 11.1 12.1 13.1 14.1
```

factor()

```
f_fac<-c("pos","neg","pos","pos","neg")
table(f_fac)
```

```
## f_fac
## neg pos
##   2   3
```

data.frame()

```
a <- c(10,20,15,43,76,41,25,46)                          # numerisch
b <- factor(c("m", "w", "m", "w", "m", "w", "m", "w")) # Faktor Geschlecht: m=männlich, w=weiblich
c <- c(2,5,8,3,6,1,5,6)                                  # numerisch
myframe <- data.frame(a,b,c);
names(myframe)<-c("Age","Gender","Dose"); myframe
```

```
##   Age Gender Dose
## 1  10      m    2
## 2  20      w    5
## 3  15      m    8
## 4  43      w    3
## 5  76      m    6
## 6  41      w    1
## 7  25      m    5
## 8  46      w    6
```

Missing values in a vector

```
x<-c(1,2,NA,10)
good<-!is.na(x); good
```

```
## [1]  TRUE  TRUE FALSE  TRUE
```

```
y<-x[complete.cases(x)]; y
```

```
## [1]  1  2 10
```

```
# Bei einer Matrix geht das so:
m<-matrix(rbinom(10,1,0.2),5,2)
m[3,1]<-NA
# Variante 1
m[complete.cases(m),]
```

```
##      [,1] [,2]
## [1,]    1    0
## [2,]    1    0
## [3,]    0    0
## [4,]    0    1
```

```r
# Variante 2
subset(m,subset = complete.cases(m),select = c(1,2))
```

```
##      [,1] [,2]
## [1,]    1    0
## [2,]    1    0
## [3,]    0    0
## [4,]    0    1
```

Daten I/O

Lesen eines komprimierten Formats, Mindestkonfiguration

```r
dat<-read.table(unz("rprog-data-quiz1_data.zip","hw1_data.csv"), sep = ",",header = TRUE,quote = "\"")
head(dat)
```

```
##   Ozone Solar.R Wind Temp Month Day
## 1    41     190  7.4   67     5   1
## 2    36     118  8.0   72     5   2
## 3    12     149 12.6   74     5   3
## 4    18     313 11.5   62     5   4
## 5    NA      NA 14.3   56     5   5
## 6    28      NA 14.9   66     5   6
```

```r
class(dat)
```

```
## [1] "data.frame"
```

```r
colnames(dat)
```

```
## [1] "Ozone"   "Solar.R" "Wind"    "Temp"    "Month"   "Day"
```

Abspeichern von Workspace Variablen

Clear Workspace

```r
rm(list=ls())
```

SingleData I/O

```r
dat<-airquality
dput(x = dat, file = "dat.R")
rm(list=ls())
dat<-dget(file = "dat.R")
```

dput, dget

MultiData I/O

```r
v_vec<-c(1:4,11:14);
m_mat<-matrix(v_vec, nrow=4, ncol=4);
# Die Variablen müssen in Anführungszeichen
dump(c("v_vec","m_mat"),file = "dat.R")
source("dat.R")
```