

Eine Einführung in



653358 UE Biostatistik (UE)  
Wintersemester 2005/06

Franz König  
Werner Brannath



Institut für Medizinische Statistik  
BE für Medizinische Statistik und Informatik  
Medizinische Universität Wien  
Spitalgasse 23, A – 1090 Wien  
Tel.: 01-40400-7484  
[franz.koenig@meduniwien.ac.at](mailto:franz.koenig@meduniwien.ac.at)

# Homepage und Download

<http://www.r-project.org/>

## Die Umgebung R

- R ist sehr ähnlich zu S-Sprache, die auch Grundlage von S-Plus ist.
- R beinhaltet eine base Package und 25 Standard-Packages.
- Viele weitere Packages gibt es im Internet zum herunterladen.
- Die Sprachen R und S sind objektorientiert.
- Statischen Tools liefern meist wenig Output, jedoch Objekte die mit anderen Tools exploriert und weiter verarbeitet werden können.

## R-Syntax

- R unterscheidet zwischen Groß- und Kleinbuchstaben.
- Zum benennen von Objekten können alle alphanumerischen Zeichen sowohl wie "." und "\_" verwendet werden.
- Ein Objektname muss mit einem Buchstaben oder "." beginnen.
- Beginnt eine Name mit ".", dann darf der zweite Buchstabe keine Zahl sein.

## Erste Schritte

### Arbeitsverzeichnis

```
getwd()      # Angabe es aktuellen Arbeitsverzeichnisses  
setwd()      # Setzen des Pfades des Arbeitsverzeichnisses
```

### R als Taschenrechner

#### Zuweisungsoperatoren:

```
=, <-, ->
```

```
> x=1  
> x  
[1] 1  
> x <- 2+2  
> x  
[1] 4  
> 1+2+3-> x  
> x  
[1] 6
```

#### Arithmetic operators

```
+      plus  
-      minus  
*      times  
/      divided by
```

^ power

## Common functions

sqrt  
log  
log10  
exp  
abs  
round round to nearest integer  
ceiling round up  
floor round down  
sin,cos,tan sinus,cosines, tangent  
asin,acos,atan arc sine,...

## Logical operators

< less than  
> greater than  
<= less than or equal to  
>= greater than or equal to  
== equal to  
!= not equal to  
& && and  
| || or  
! not

## Typ einer Variable – Struktur der Daten

Vektoren, Matrizen, Listen, Arrays, Data Frames

Abfrage: str()

is.vector(), is.list(), is.matrix(),...

## Vektoren

## Zufallszahlen

### Normalverteilung

```
> x <- rnorm(100) # Erzeugung von 100 standardnormalverteilten
Zufallsvariablen
> x
 [1]  1.53385409  0.55099278  1.63037769 -0.19634053  0.03417379 -0.24071719
 [7] -0.33483119  0.41995630 -0.16789009 -0.79822774  1.62030278 -0.42184149
[13]  1.59802329  0.72125787  1.04513506 -0.77951419 -1.20326500  0.82119323
...
> x <- rnorm(100,mean=10,sd=20) # Erzeugung von 100
normalverteilten Zufallszahlen mit Mittelwert 10 und einer
Standardabweichung von 20.
> mean(x)
[1] 11.20779
```

```
> sd(x)
[1] 19.24144
```

weitere Verteilungen

## Ausschluss von fehlenden Werten

```
> x<-c(1,2,3,NA,5,6)
```

```
> x[!is.na(x)]
[1] 1 2 3 5 6
```

```
> mean(x)
```

```
[1] NA
```

```
> mean(x[!is.na(x)])
```

```
[1] 3.4
```

```
> mean(x,na.rm=T)
```

```
[1] 3.4
```

## Textmanipulation

```
> Vornamen <- c("Franz","Werner","Peter")
```

```
> nchar(Vornamen)
```

```
[1] 5 6 5
```

```
> paste("Lieber",Vornamen)
```

```
[1] "Lieber Franz" "Lieber Werner" "Lieber Peter"
```

```
> paste(1:4,Vornamen)
```

```
[1] "1 Franz" "2 Werner" "3 Peter" "4 Franz"
```

```
> substring(Vornamen)
```

```
Fehler in max(1t <- length(text), length(first), length(last)) :  
  Argument "first" fehlt (ohne Standardwert)
```

```
> substring(Vornamen,1,1)
```

```
[1] "F" "W" "P"
```

```
> abbreviate(Vornamen)
```

```
 Franz Werner Peter
```

```
"Frnz" "Wrnr" "Petr"
```

```
> abbreviate(Vornamen,3)
```

```
 Franz Werner Peter
```

```
"Frn" "Wrn" "Ptr"
```

```
> paste("Alle Vornamen",Vornamen,collapse="*",sep=":")
```

```
[1] "Alle Vornamen:Franz*Alle Vornamen:Werner*Alle Vornamen:Peter"
```

```
> paste("Alle Vornamen",paste(Vornamen,collapse="*"),sep=":")
```

```
[1] "Alle Vornamen:Franz*Werner*Peter"
```

## Mode

### Abfrage:

```
mode()
```

### Überprüfung des Typs einer Variablen:

```
is.numeric()
```

```
is.character()
is.integer()
is.single()
is.double()
is.logical()
```

## Umwandlung

```
as.numerical(), as.integer(), as.single(), as.character(), as, matrix(), ...
```

## Beispiel

```
> x<-1:4
> x
[1] 1 2 3 4
> y <- c("a","b","c","d")
> y
[1] "a" "b" "c" "d"
> is.numeric(x)
[1] TRUE
> x1 <- as.character(x)
> x1
[1] "1" "2" "3" "4"
> mode(x)<-mode(y)
> x
[1] "1" "2" "3" "4"
> is.numeric(x)
[1] FALSE
```

# Matrizen

## Wichtige Funktionen

`round()`

## Funktionen der deskriptive Statistik

`sort`  
`order`  
`rank`  
`sum`  
`diff`  
`prod`  
`cumsum, cumprod`  
`mean, median, quantile`  
`var, sd, mad`  
`min, max`  
`cummin, cummax`  
`IQR`                    `Interquartilsabstand (3.Quartil-1.Quartil)`  
`range`                    `Spannweite (Max-Min)`  
`summary`  
`table`                    `Kreuztabellen für kategorielle Variablen`  
`length`  
`table`                    `Tabelle der Häufigkeitsverteilung(ein- und mehrdimensional)`

## Statistische Tests

`t.test()`                    `# t-Tests (einseitig, zweiseitig; gepaart vs ungepaart,...)`  
`chisq.test()`                `# Verwende table() zur Ausgabe der absoluten Häufigkeiten`  
`aov()`                      `# Varianzanalyse`

## Grafiken

```
> demo(graphics)      # Überblick über mögliche Grafiken

> win.graph()         #öffnen eines Grafikfensters
> dev.off()           #schließt das entsprechende Grafikfenster
> graphics.off()      #schließt alle geöffneten Grafikfenster
```

## Überblick

```
plot      Streudiagramm
boxplot
hist      Histogramm
barplot   Stabdiagramm
pie       Kreisdiagramm
curve     Plotten einer Funktion
```

```
> curve(x^2,-2,2)
> curve(x/2,-2,2,add=T) #Plottet diese Funktion zusätzlich in
die vorherige Grafik
```

## Aufteilung des Grafikfensters

```
> par(mfrow=c(3,2) # Aufteilung des Grafikfenster für mehrere
Grafiken. Hier 6 Grafiken in 3 Zeilen und 2 Spalten.

> par(mfrow=c(1,1) # Zurücksetzung auf eine Grafik pro Fenster
```

### Anmerkung: Aufruf der Hilfe für eine Funktion

```
> help(par)
> ?par
```

## Identifizierung von einzelnen Datenpunkten

```
> boxplot(iris$Sepal.Length ~iris$Species)
identify(iris$Sepal.Length ~iris$Species) # Im Grafikfenster können nun
die zu identifizierenden Punkte mit der linken Maustaste ausgewählt werden. Ende mit rechter
Maustaste.
```

## Data frames

Ein Data Frame ist eine Spezielle Variante einer Matrix, hier können unterschiedliche Spalten unterschiedliche Modes haben. Alle Elemente einer Spalte müssen denselben Mode haben, zum Beispiel: integer, factor oder character

Data Frames sind als Listen implementiert. Um ein DataFrame bestehend nur aus numerischen Werten in eine Matrix umzuwandeln verwende die Funktion `as.matrix()`.

- 2-dimensionales Array
- Aufbau zur „klassischen“ Datenmatrix
- Spalten entsprechen Variablen
- Zeilen entsprechen den Beobachtungen an einer Beobachtungseinheit (z.B Patient, Versuchstier, Proband)
- Spalten sind Vektoren, die einen unterschiedlichen mode haben können

### Beispiel

```
> groesse<-c(170,180,191)
> gewicht<-c(53,81,94)
> geschlecht<-c("weiblich","maennlich","maennlich")
> geschlecht
[1] "weiblich" "maennlich" "maennlich"
> geschlecht<-as.factor(geschlecht)
> geschlecht
[1] weiblich maennlich maennlich
Levels: maennlich weiblich
> daten<-data.frame(groesse,gewicht,geschlecht)
> daten
  groesse gewicht geschlecht
1     170      53 weiblich
2     180      81 maennlich
3     191      94 maennlich
>
```

### Möglichkeiten um das Gewicht auszugeben

```
> daten$gewicht
[1] 53 81 94
> daten[,2]
[1] 53 81 94
> daten[,"gewicht"]
[1] 53 81 94
> daten[[2]]
[1] 53 81 94
```

### Standarddatensätze im R

```
> data()
```

## Ein weiterer Datensatz: Iris-Daten

```
> ?iris #description of the data frame

> iris
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1           5.1           3.5           1.4           0.2   setosa
2           4.9           3.0           1.4           0.2   setosa
3           4.7           3.2           1.3           0.2   setosa
4           4.6           3.1           1.5           0.2   setosa
5           5.0           3.6           1.4           0.2   setosa
...

> mode(iris)
[1] "list"
> str(iris)
`data.frame`: 150 obs. of 5 variables:
 $ Sepal.Length: num  5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
 $ Sepal.Width : num  3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
 $ Petal.Length: num  1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
 $ Petal.Width : num  0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
 $ Species      : Factor w/ 3 levels "setosa","versicolor",...: 1 1 1 1 1 1 1
1 1 1 ...
> dim(iris) #Anzahl Beobachtungen, #Anzahl Variablen
[1] 150 5
> length(iris) #Anzahl Variablen
[1] 5
```

## Anwenden einer Funktion auf ein Data Frame

```
apply
lapply
sapply

> apply(iris[,1:4],2,mean)
> sapply(iris[1:4],mean)
```

## aggregate and tapply

```
> aggregate(Sepal.Width,by=list(Species),FUN=mean)
  Group.1      x
1   setosa 3.428
2 versicolor 2.770
3  virginica 2.974
> tapply(Sepal.Width,Species,mean)
  setosa versicolor virginica
  3.428    2.770    2.974
```

## Multiple Vergleiche

### ***Beispiel für Dunnett***

```
> x<-read.table("c:\\rloschen\\walking.txt",header=T)
> is.data.frame(x)
[1] TRUE
```

```
boxplot(x$age~x$group)
> boxplot(x$age~x$group,xlab="Treatment Groups",ylab="Age (in months)",main="Data
from Zelazo et al (1972)")
```

```
identify(x$age~x$group)
> sim.x<-(simint(age~group,data=x,type="Dunnett",base=2)) # Vorsicht, base gibt die
Kontrollgruppe an!
> summary(sim.x)
> plot(sim.x)
```

### ***Beispiel für Tukey***

```
> sim.tukey.x<-(simint(age~group,data=x,type="Tukey"))
> sim.tukey.x
> plot(sim.tukey.x)
```

## Übungsbeispiele

1. Man schreibe eine Funktion, welche eine Matrix von absoluten Häufigkeiten als Zeilen-Spalten,- oder Totalprozent zurückgibt (Tipp: sweep, apply)

2. Man schreibe ein R-Funktion, welcher eine Matrix beliebiger Dimension (I\*J) übergeben wird und die eine erweiterte Matrix (I+1\*J+1) zurückgibt. Durch einen optionalen Parameter (Default sind Summen) soll gesteuert werden, ob die Randelemente der Matrix die Zeilen, bzw Spaltensummen –mittelwerte oder Maxima der jeweiligen Matrixelemente enthalten. Weiters sollen die Zeilen und Spalten der erweiterten Matrix optional mit Bezeichnen versehen werden können. (Tipp: apply, rbind, cbind, colnames, rownames)

Beispiel

```
> a
      [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
mat.change(a, rand="sum", zeilen.bez="Z", spalten.bez="S")
      [S1] [S2] [S3] [S4]
[Z1]    1    3    5    9
[Z2]    2    4    6   12
[Z3]    3    7   11   21
```

3. Beispiel ANOVA, multiple Vergleiche

Distribution of Ages(in months) at Which Infants First Walked Alone. Data from Zelaza et al. (1972)

ActiveGroup	PassiveGroup	No-ExerciseGroup	8WeekControlGroup
9	11	11,5	13,25
9,5	10	12	11,5
9,75	10	9	12
10	11.75	11,5	13,5
13	10.5	13,25	11,5
9.5	15	13	

- Erstellen Sie ein entsprechendes Data Frame.
- Erstellen Sie Boxplots getrennt nach Gruppe
- Formulieren Sie den Null und Alternativhypothese für eine ANOVA. Führen Sie anschließend eine Anova durch. Interpretieren Sie das Ergebnis.
- Multiple Vergleiche: Verwenden Sie als post-hoc Test die Methode nach Dunnett. Die Kontrollgruppe ist die Gruppe „8WeekControlGroup“.
- Angenommen es gibt keine Kontrollgruppe und Sie sind an allen paarweisen Vergleichen interessiert. Wählen Sie eine geeignete multiple Testprozedur.

4. Schreiben Sie ein Programm zur blockweisen Randomisierung mit variabler Blocklänge für  $k$  Behandlungen. Als Übergabeparameter an die Funktion sollen die Anzahl der Behandlung  $k$ , der Stichprobenumfang pro Gruppe  $n$ , und die maximale Blocklänge angegeben werden. (Tipp: sample)

## 5. Beispiel ANOVA (mit fehlenden Werten)

In der Zahnmedizin wurden 3 Kleber bezüglich ihrer Haftstärke (Bonding Strength in MPa) verglichen. Aufgrund schlechter Dokumentation kam es dabei zu fehlenden Werten in der Datenmatrix.

	haft	kleber
1	6.42	1
2	8.22	NA
3	10.24	NA
4	9.22	1
5	7.15	1
6	7.03	1
7	10.42	NA
8	9.26	2
9	9.62	2
10	3.65	2
11	7.63	2
12	8.06	2
13	11.04	3
14	13.35	3
15	12.30	3
16	NA	3
17	15.01	3
18	14.44	3

- Schreiben Sie Programm, das die Mittelwerte, Standardabweichung getrennt für alle Kleber berechnet.
- Formulieren Sie den Null und Alternativhypothese für eine ANOVA. Führen Sie anschließend eine Anova durch. Interpretieren Sie das Ergebnis.
- Multiple Vergleiche
- Wiederholen Sie Analyse mit simulierten Daten und größerem Stichprobenumfang pro Gruppe. Beachten Sie, dass im Experiment ein balanciertes Design verwendet wurde. (Kleber 1: Mean=8,Sd=2; Kleber2 M=8.3,Sd=2; Kleber3 M=14, Sd=2)  
(Tipp: Erzeugung der normalverteilten Zufallszahlen mit rnorm)

## 6. Simulation

Simulieren Sie für das Szenario aus Beispiel 5d mit einem Stichprobenumfang von n=20 pro Gruppe die Überdeckungswahrscheinlichkeit dass ALLE Mittelwerte in den entsprechenden naiven (unadjustierten) 95%Konfidenzintervall liegen.

Formel für das unadjustierte Konfidenzintervall:

$$\left[ \bar{X} - z_{1-\alpha/2} \frac{\sigma}{\sqrt{n}}; \bar{X} + z_{1-\alpha/2} \frac{\sigma}{\sqrt{n}} \right]$$

wobei  $z_{1-\gamma}$  das  $1-\gamma$  Quantil der Standardnormalverteilung ist (Tipp: qnorm).

## 7. Beispiel Multipler Fehler

Zur Erinnerung: es werden  $k$  unabhängigen Signifikanztests zum lokalen Niveau  $\alpha$  durchgeführt. Wenn nun in Wirklichkeit bei jedem Test die Nullhypothese gilt, dann ist der multiple Fehler die Wahrscheinlichkeit mindestens eine der  $k$  Nullhypothesen zu verwerfen. Schreiben Sie eine Funktion zur Berechnung des multiplen Fehlers bei  $k$  unabhängigen Tests, wobei das lokale Signifikanzniveau  $\alpha$  beträgt. (Optional:  $\alpha$  und  $k$ ). Plotten sie den multiplen Fehler in Abhängigkeit der durchgeführten Tests ( $k=1,2,\dots,20$ ) für ein Signifikanzniveau von  $\alpha=0.5$  und  $\alpha=0.01$ .

## 8. Zentraler Grenzwertsatz

Simulieren sie

- a) 1000 Stichproben der Grösse 2
- b) 1000 Stichproben der Grösse 10
- c) 1000 Stichproben der Grösse 100

mit jeweils standardnormalverteilten Beobachtungen. Berechnen Sie jeweils für a-c die Stichprobenmittelwerte, sowie Mittelwert und Varianz der 1000 Mittelwerte. Zeichnen Sie jeweils für a-c die Histogramme der Stichprobenmittelwerte und Boxplots. Vergleichen Sie die Ergebnisse und interpretieren Sie die Unterschiede.

(Schreiben Sie eine Funktion, der man optional folgende Parameter übergibt: wh für die Zahl der Stichprobenwiederholungen, n für den Stichprobenumfang.)

## 9. Regression:

Verwenden Sie den Datensatz „fat\_long.data“ aus der Übung (falls Sie ihn benötigen, einfach ein Email schicken bzw. <http://www.amstat.org/publications/jse/v4n1/datasets.johnson.html>). Verwenden Sie in allen multiplen Regression als abhängige Variable „bodyfat“ und als unabhängige Kandidatenvariablen age weight height neck chest abdomen hip thigh knee ankle biceps forearm wrist.

- 1) Führen Sie vor der Analyse ein „Datacleaning“ der Daten auf Plausibilität durch. Exkludieren Sie gegebenenfalls unplausible Werte.
- 2) Entscheiden Sie sich für ein automatisiertes Variablen-Selektions-Verfahren (siehe zB packages MASS aus der Übung mit der Funktion „stepAIC“) und rechnen Sie eine Variante mit.
  - a. Vorwärtselektion
  - b. Rückwärtselektion
  - c. stufenweise

(Anmerkung: Als Alternative zur Funktion stepAIC können Sie gerne auch andere Methoden zB aus den Packages leaps oder wle verwenden.)

Ein vollständiges Skriptum gibt es am 19.1.2006!

## Zusätzliche sehr nützliche Packages

lattice            Zusätzliche Grafiken (entspricht der trellis Library in S-Plus)

Installieren in Windows: Click„Pakete“,Click„Installiere Paket(e)“,..„lattice“

library(lattice)        #Laden des Packages in R, muss bei jeder Sitzung wiederholt werden.