Einführung in 

Sven Hartenstein

Friedrich-Schiller-Universität Jena

Wintersemester 2007/2008

Gliederung

Gliederung

Was ist R?

R ist eine sehr mächtige Sprache und frei verfügbare Umgebung für statistische Datenbearbeitung und graphische Darstellungen. Der Quelltext wird unter der GNU General Public License der Free Software Foundation veröffentlicht.

R kann als Implementation der Sprache S angesehen werden. S bildet auch die Basis für die Software S-Plus.

R wird in verschiedensten Fachbereichen verwendet. Warum? Siehe „Vorteile von R“ unten.

Wo bekomme ich die Software R für Windows?

Im PC-Pool im Institut für Psychologie ist R installiert.

Um es auf dem eigenen Rechner zu installieren, kann R im Internet heruntergeladen werden:

1. http://www.r-project.org öffnen.
2. Auf der linken Seite auf CRAN klicken.
3. Einen nahen Mirror wählen (Germany).
4. Windows (95 and later) wählen.
5. base wählen.
6. Die Datei R-[VERSIONSNUMMER]-win32.exe herunterladen.
7. Die Datei auf dem lokalen PC ausführen, um die Installation zu starten.

Gliederung

Ausgabe, Objekte erzeugen, Vektoren

```
> 3+sqrt(16) # Taschenrechner
[1] 7
> x <- 200 # Ein Objekt wird erzeugt
> x # Zeige Objekt an
[1] 200
> x <- c(3, 5, 4, 2, 1); x # Semikolon trennt Befehle
[1] 3 5 4 2 1
> 1:24
 [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17
[18] 18 19 20 21 22 23 24
> seq(from=8, to=16, by=2)
[1]  8 10 12 14 16
> seq(8, 16, 2)
[1]  8 10 12 14 16
> seq(b=2, f=8, t=16)
[1]  8 10 12 14 16
```

Deskriptive Statistik, Sortierung

```
> x <- c(4, 5, 3, 4, 6, -2, 15)
> c(min(x), max(x), mean(x), median(x), sum(x), var(x))
[1] -2 15  5  4 35 26
> quantile(0:20)
  0%  25%  50%  75% 100%
   0    5   10   15   20
> sort(x)
[1] -2  3  4  4  5  6 15
> rank(x)
[1] 3.5 5.0 2.0 3.5 6.0 1.0 7.0
> order(x)
[1] 6 3 1 4 2 5 7
```

Auswahl

```
> x <- 11:20
> x[3:6]
[1] 13 14 15 16
> x[x < 15]
[1] 11 12 13 14
> x[x %% 2 == 0]
[1] 12 14 16 18 20
> x %% 2 == 0
 [1] FALSE  TRUE FALSE  TRUE FALSE  TRUE FALSE  TRUE FALSE  TRUE
[10] TRUE
```

Gliederung

Alles ist ein Objekt

Objekte können bspw. ausgegeben, transformiert, Funktionen als Parameter übergeben oder Teil eines arithmetischen Ausdrucks werden.

Typen von Objekten: vector, matrix, list, data.frame

Datentypen: numeric, character, logical (TRUE vs. FALSE)

`mode()` zeigt den Typ eines Objektes
`str()` zeigt seine Struktur
`ls()` listet die im Workspace vorhandenen Objekte auf
`rm()` löscht Objekte.

Ein Verzeichnis pro Projekt

R schaut beim Start, ob im Verzeichnis, in dem es gestartet wird, Objekte und eine History gespeichert sind („Arbeitsverzeichnis“).

Wenn ich Objekte in dieser Form speichere oder an mehreren getrennten R-Projekten gleichzeitig arbeite, empfiehlt sich, für jedes Projekt ein eigenes Verzeichnis zu nutzen.

„The source is real“

- Dein Quellcode dokumentiert die Analyse (chronologisch).
- Er kann in jeder Session neu ausgeführt werden.
- Arbeite mit *.R-Dateien, nicht in der R-Konsole.
- Workspace und History speichern ist keine gute Idee.

☞ Dringende Empfehlung!

Prinzip der Wiederholung bei ungleichen Längen

```
> a <- 1:5
> b <- 1:3
> a+b
[1] 2 4 6 5 7
Warning message:
Länge des längeren Objektes
  ist kein Vielfaches der Länge des kürzeren Objektes in:
  a + b
```

Gliederung

Hilfe

`help(FUN)` ruft die Hilfe-Seite zur Funktion FUN auf

`?FUN` tut das gleiche

`help.search("STR")` sucht Hilfe-Seiten, die STR enthalten

`help.start()` öffnet HTML-Dokumentation im Browser

- ☞ Das Hilfe-System ist Dein Freund!
  Nicht erst bei Verzweiflung benutzen!

Wo finde ich mehr Informationen zu R?

- Offizielle Website: http://www.r-project.org. Die Lektüre des Manuals „An Introduction to R“ ist empfehlenswert.
- R-Wiki: http://wiki.r-project.org
- R-Reference-Card: http://www.rpad.org/Rpad/R-refcard.pdf
- Sachs, L. & Hedderich, J. (2006). *Angewandte Statistik. Methodensammlung mit R*. Berlin: Springer.
- Bücher explizit zu R: in der ThULB (z. B. nach „R Programm“ suchen) oder am Lehrstuhl
- Suchmaschinen (sehr hilfreich bei konkreten Fragestellungen)
- Zahlreiche R-Beispiel-Graphiken: http://addictedtor.free.fr/graphiques/
- Pakete: CRAN (Link auf http://www.r-project.org)

Gliederung

Arithmetik

`+  -  *  /  ^  %%  %/%  sqrt()`

```
> 1:10 * 2
 [1]  2  4  6  8 10 12 14 16 18 20
> (6:2)^2
[1] 36 25 16  9  4
> 6:(2^2)
[1] 6 5 4
> 1:7 %% 2 # modulo
[1] 1 0 1 0 1 0 1
> 1:7 %/% 2
[1] 0 1 1 2 2 3 3
> sqrt(c(1, 2, 4, 9))
[1] 1.000000 1.414214 2.000000 3.000000
```

Vergleiche

`<` `>` `<=` `>=` `==` `!=`

```
> 3 < 4
[1] TRUE
> (1:8) %% 2 == 0
[1] FALSE  TRUE FALSE  TRUE FALSE  TRUE FALSE  TRUE
> (11:20)[1:10 != 5]
[1] 11 12 13 14 16 17 18 19 20
```

Logik

`!  &  |  xor()`

```
> a <- c(TRUE, TRUE, FALSE, FALSE)
> b <- c(TRUE, FALSE, TRUE, FALSE)
> !a
[1] FALSE FALSE  TRUE  TRUE
> a&b
[1]  TRUE FALSE FALSE FALSE
> a|b
[1]  TRUE  TRUE  TRUE FALSE
> xor(a, b)
[1] FALSE  TRUE  TRUE FALSE
```

Daten erzeugen

`rnorm()`, `rexp()`, `rgamma()`, `rpois()`, `rweibull()`, `rcauchy()`, `rbeta()`, `rt()`, `rf()`, `rchisq()`, `rbinom()`, `rgeom()`, `rhyper()`, `rlogis()`, `rlnorm()`, `rnbinom()`, `runif()`, `rwilcox()`

`sample()`

```
> iq <- rnorm(10, mean=100, sd=10)
> mean(iq)
[1] 102.2320
> sd(iq)
[1] 9.379149
> sample(c(2, 4, 6, 8), size=5, replace=TRUE)
[1] 8 2 8 6 6
```

Daten einlesen, schreiben

Vektor: `scan()`

Dataframe: `read.table()`, `write.table()`

SPSS-Datendatei: `library("foreign"); read.spss("datei.sav")`, `write.foreign()`

Siehe auch das Manual „R Data Import/Export“

Übungen

1. Führe eine Lotto-Ziehung („6 aus 49“) durch!
2. Lese eine SPSS-Datendatei ein und berechne den Mittelwert einer darin enthaltenen Variable!
3. Was könntest Du tun, damit Du mit dem Mittelwert in einer späteren R-Sitzung weiterrechnen kannst?
4. Verändere ein paar Werte einer Variable und speichere den Datensatz.
5. Lies den soeben gespeicherten Datensatz neu ein.

☞ Zeitrahmen: 15 Minuten

Matrizen

`matrix()`, `rbind()`, `cbind()`

Algebra: %*%, `crossprod()`, `t()`, ...

siehe auch http://gauss.metpsy.uni-jena.de/metheval/wiki-metheval-hp/index.php/R_Matrixalgebra

Matrizen II

```
> m <- matrix(1:6, ncol=2, byrow=TRUE); m
     [,1] [,2]
[1,]    1    2
[2,]    3    4
[3,]    5    6
> m <- rbind(t(m), 9); m
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
[3,]    9    9    9
> m[2,3]
[1] 6
> m[,1]
[1] 1 2 9
```

Listen

list(), unlist(), [[]], $

```
> listi <- list(a=4, b="hallo", z=3:8); listi
$a
[1] 4

$b
[1] "hallo"

$z
[1] 3 4 5 6 7 8
> listi$b
[1] "hallo"
> listi[["b"]]
[1] "hallo"
```

Statistische Analysen

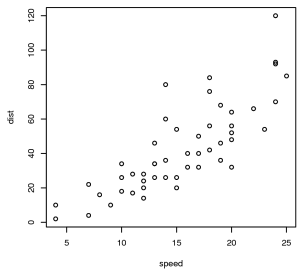
- Session mit Norman Rose
- siehe VL-Folien Inferenzstatistik vom WiSe06/07

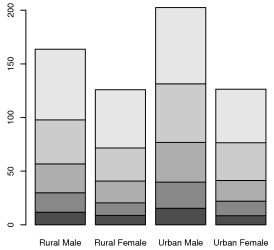
Formeln in Modellen:

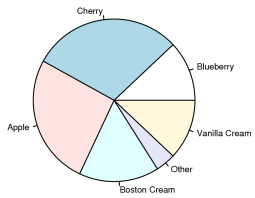
`a+b`	additive Effekte von a und b
`X`	aditive Effekte der Spalten von X
`a:b`	Wechselwirkung (Interaktion) von a und b
`a*b`	entspricht `a + b + a:b`
`poly(a,n)`	Polynome von a bis Grad n
`^n`	alle Interaktionen bis Grad n z.B. `(a+b+c)^2`
`a-b`	z.B. `(a+b+c)^2 - a:b`
`-1`	Modell ohne Abschnitt
`1`	Modell mit nur Abschnitt
`I()`	Arithmetik, z.B. `I(a*b)`

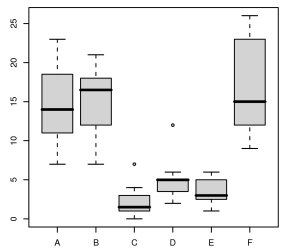
Einfache Graphiken

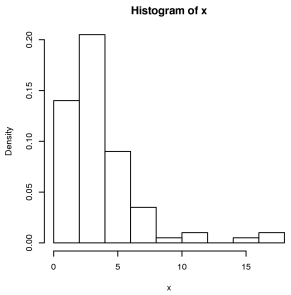
`plot()`, `barplot()`, `pie()`, `boxplot()`, `hist()`, `image()`

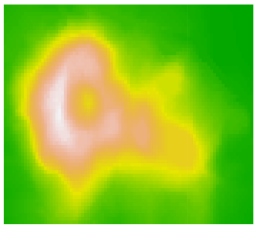













```
> postscript("im.eps",paper="special",width=24,height=4)
> par(mar=c(0, 3.5, 0, 3.5)) # Rand
> plot(cars) # high-level
> lines(lowess(cars)) # low-level
> dev.off() # Graphik erzeugen
null device
          1
```

Weitere wichtige Funktionen

`cat()` gibt einen Text aus

`paste()` verbindet Zeichenketten

`q()` beendet 

```
> cat(paste("Sieben mal vier sind ",7*4,".\n", sep=""))
Sieben mal vier sind 28.
> q()

Process R finished at Wed Nov 14 23:36:57 2007
```

Gliederung

Relevante Features von Text-Editoren

- syntax highlighting
- auto indentation
- auto completion
- bracket matching
- code folding
- Anzeige von Parametern von Funktionen
- Code (-Auswahl) an den R-Prozess senden
- ...

Editoren mit R-Unterstützung

- Tinn-R, https://sourceforge.net/projects/tinn-r/
  - ☞ Viele Features, nur Windows
- JGR, http://rosuda.org/JGR/
  - ☞ Auch schön, Java (cross platform)
- Emacs Speaks Statistics, http://ess.r-project.org
  - ☞ Für Emacs-Nutzer. Viele Features, hohe Einarbeitungszeit, wenn man Emacs noch nicht kennt
- (WinEdit u.a.)
  - ☞ Wer schon einen Editor gewöhnt ist kann schauen, ob es für ihn R-Unterstützung gibt
- Siehe auch http://www.r-project.org/GUI

Gliederung

Vorteile von R

- Syntax ist einfach zu lesen und Funktionen einfach zu kombinieren (vgl. SPSS, SAS)
- Transparenz (und Edukation) durch interpretierten, offenen Code. Keine Black-Box-Software
- Effektiver arbeiten
- Flexibel, anpassbar, individuell, ideenreich, ...
- Gibt es auch für ordentliche Betriebssysteme
- Kostengünstig (bzw. legal)
- Freie Software
- „One can teach WITH R. One can teach SAS, but one cannot teach WITH SAS (or LISREL).“ (De Leeuw)

☞ Was als Vorteil angesehen wird ist individuell je nach Bedürfnissen sehr verschieden.

Nachteile von 

- Interface: ungewohnt (neue Art, mit dem Rechner zu arbeiten)
- GUIs: uneinheitlich
- Englischkenntnisse erforderlich
- Namen von Funktionen und Parametern behalten: heißt es `col.names`, `colnames` oder `header`?
- Dokumentation richtet sich eher an Experten: „Was ist S3 und warum ist das wichtig?“

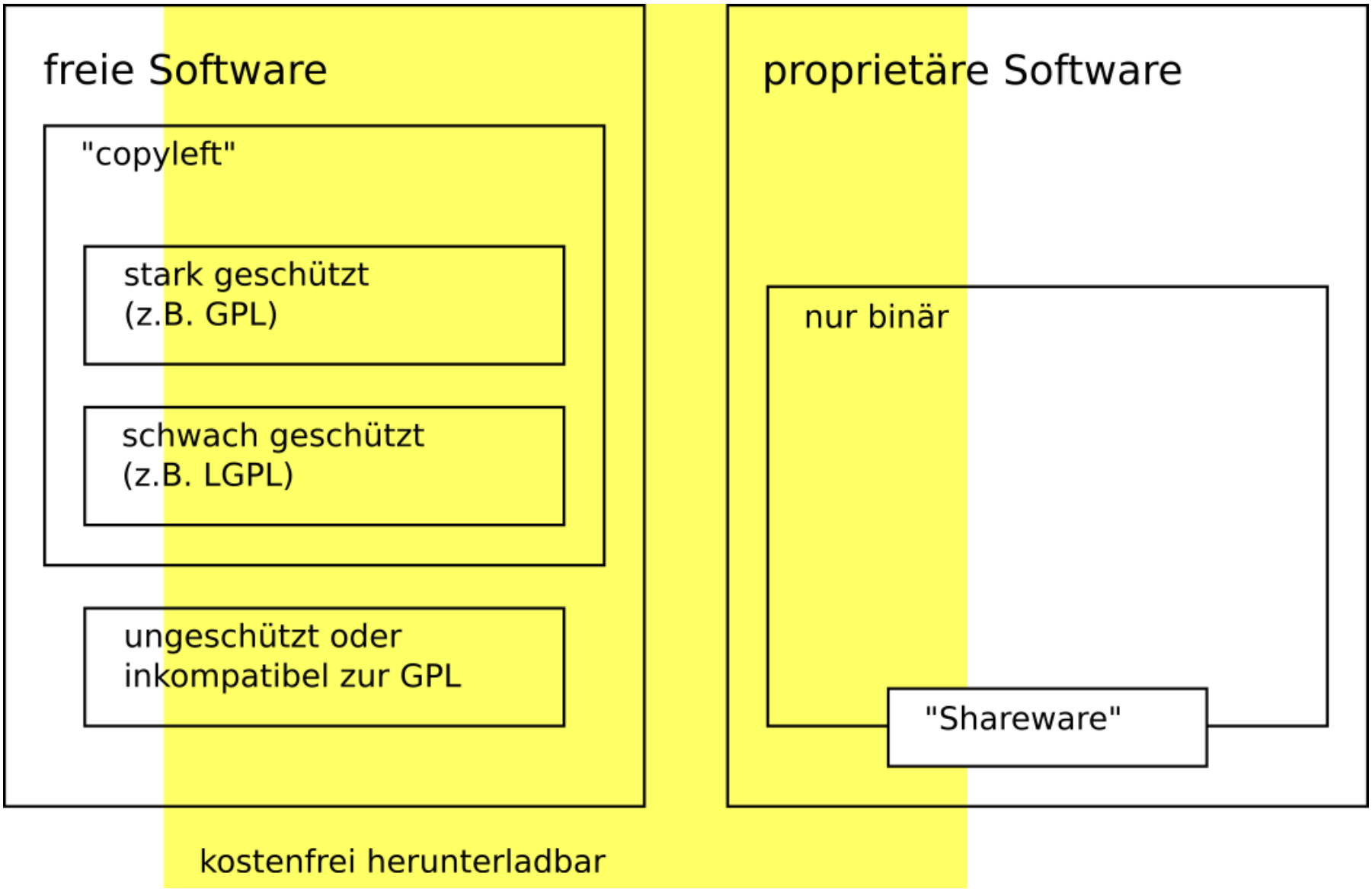
Exkurs „Freie Software“: Was ist das?

4 Freiheiten:

- Freiheit der Nutzung zu jedem Zweck
- Freiheit des Studiums und Verständnisses wie das Programm funktioniert und der Anpassung an eigene Bedürfnisse
- Freiheit der Weitergabe von Kopien
- Freiheit der Weitergabe von verbesserten Versionen

„Free software“ is a matter of liberty, not price. To understand the concept, you should think of „free“ as in „free speech,“ not as in „free beer.“ (Free Software Foundation)

Exkurs „Freie Software“: Verortung



SPSS versus R

- Lizenz: proprietär versus freie Software
- Verbreitung: stark versus (noch) gering
- Interface: Menus, Maus, Syntax versus Syntax, GUIs verfügbar
- Einfachheit: Standard-Analysen sehr einfach versus Einarbeitungszeit (und Englischkenntnisse) erforderlich
- Kosten: viel Geld versus kostenlos
- Flexibilität: relativ niedrig versus extrem hoch
- Graphiken: naja versus flexibel, hohe Qualität, viele Dateiformate
- Funktionsumfang: vergleichsweise gering versus Erweiterbarkeit durch Pakete und eigene Programmierung
- Transparenz: gering versus kompletter Quelltext offen
- Plattformunabhängigkeit: erst SPSS 16.0 gibt es für GNU/Linux versus für viele Plattformen verfügbar

Gliederung

Bedingungen

```
if (a == b) c <- 2*a

if (p < .05) sig <- TRUE else sig <- FALSE

if (is.na(var) && a == b) {
   ...
} else {
   ...
}

ifelse(vector.condition, vector.a, vector.b)
```

Wiederholungen

```
for (i in 1:100) {
  cat(paste("Jetzt hat i den Wert ",i,"!\n", sep=""))
}

while (abs(a) < 100 && a != 0) {
  a <- a*2
  if (a == 40) next
  cat(a)
  if (a == 60) break
}

repeat {
  ...
  if (...) break
}
```

Eleganter und schneller

```
x <- list(a=1:10, beta=exp(-3:3), logic=c(T, F, F))
y <- lapply(x, quantile, na.rm=TRUE) # Gibt Liste zurück
z <- sapply(x, mean) # Gibt Vektor (o. Matrix) zurück

hist(replicate(100, mean(rexp(10))))
```

Siehe auch `apply()`, `tapply()`, `mapply()`, `rapply()` und `replicate()`.

Eigene Funktionen schreiben

```
> kreis.sektor <- function(r, winkel=360) {
+   if (r < 0) stop("Radius must be positive.")
+   if (winkel < 0 || winkel > 360) {
+     stop("Winkel must be between 0 and 360.")
+   }
+   A <- pi*r^2*winkel/360
+   A # oder: return(A)
+ }
> kreis.sektor(4, 180)
[1] 25.13274
> kreis.sektor(4, -2)
Fehler in kreis.sektor(4, -2) : Winkel must be between
    0 and 360.
```

Warum Funktionen schreiben?

- Strukturierung des Codes
- Code-Teile an verschiedenen Stellen verwendbar
- Einfachere Fehlerkontrolle

Übungen

1. Erstellen Sie einen Vektor mit den Quadratzahlen bis 400!
2. Schreiben Sie eine Funktion, die die Fakultät einer Ganzzahl errechnet.
3. Erweitern Sie die Funktion so, dass sie als Argument auch einen Vektor zulässt und für jedes Element die Fakultät zurückgibt!

Übungen: Mögliche Lösungen

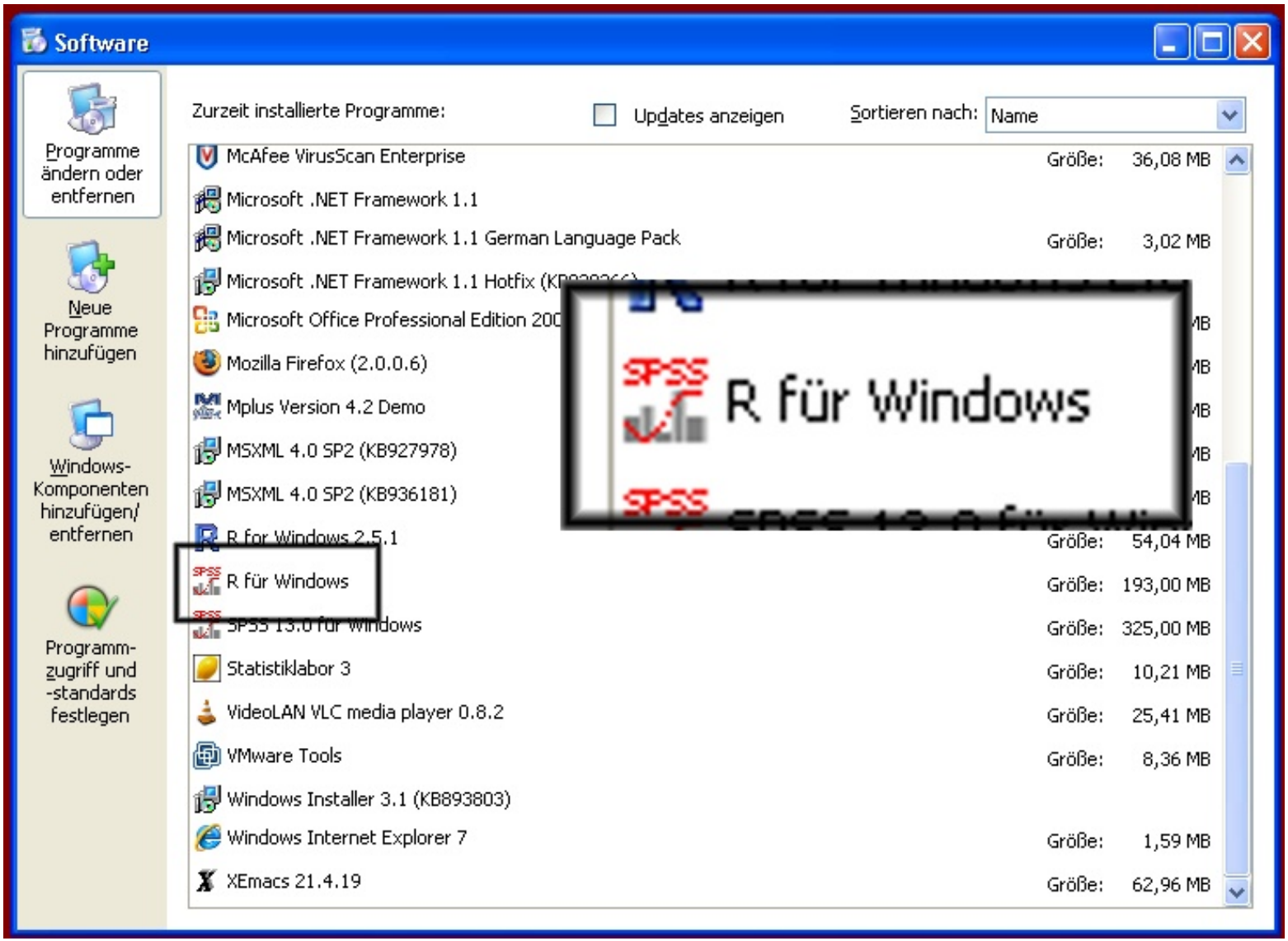
```
### Aufgabe 1 ###
(1:20)^2

### Aufgabe 2 ###
fakultaet <- function(i) {
  if (i == 0) return(1)
  fak <- 1
  for (j in 1:i) fak <- fak * j
  fak
}

### Aufgabe 3 ###
vec.fakultaet <- function(vec) {
  sapply(vec, fakultaet)
}
```

Gliederung

Quellen

- Sachs, L. & Hedderich, J. (2006). *Angewandte Statistik. Methodensammlung mit R*. Berlin: Springer.
- „Einführung in R“ (Folien von Ivailo Partchev, R-Workshop 2007)
- „R in Psychometrics and Psychometrics in R“ (Folien von Jan de Leeuw, useR-Konferenz 2006)
- „Teaching Social-Science Statistics Courses with R“ (Folien von John Fox, useR-Konferenz 2006)
- „UseR! vor Teaching“ (Folien von Sanford Weisberg, useR-Konferenz 2006)

The End


Software
Programme ändern oder entfernen
Neue Programme hinzufügen
Windows-Komponenten hinzufügen/ entfernen
Programm-zugriff und -standards festlegen
Zurzeit installierte Programme:
Updates anzeigen
Sortieren nach:
Name
McAfee VirusScan Enterprise
Größe: 36,08 MB
Microsoft .NET Framework 1.1
Microsoft .NET Framework 1.1 German Language Pack
Größe: 3,02 MB
Microsoft .NET Framework 1.1 Hotfix
Microsoft Office Professional Edition
Mozilla Firefox (2.0.0.6)
Mplus Version 4.2 Demo
MSXML 4.0 SP2 (KB927978)
MSXML 4.0 SP2 (KB936181)
R for Windows 2.5.1
Größe: 54,04 MB
R für Windows
Größe: 193,00 MB
SPSS 13.0 für Windows
Größe: 325,00 MB
Statistiklabor 3
Größe: 10,21 MB
VideoLAN VLC media player 0.8.2
Größe: 25,41 MB
VMware Tools
Größe: 8,36 MB
Windows Installer 3.1 (KB893803)
Windows Internet Explorer 7
Größe: 1,59 MB
XEmacs 21.4.19
Größe: 62,96 MB
R für Windows