[4] `http://blogkow.wordpress.com/2010/06/18/die-apotheose-der-fussnote/`: *Die Fußnote als Gott. Oder: Zur Apotheose des Literaturverweises im Zeitalter seiner Interdisziplinarität.*

[5] Stefan Breuer: *Anmerkungen zum* `\footnote`*-Befehl.* `http://www.dante.de/DTK/Ausgaben/2007-4.pdf`.

[6] Günter Grass: *»Oralverkehr mit Vokalen«, Spiegel Gespräch.* In: Der Spiegel 33/2010, S. 118–122.

[7] Matthias Kalle Dalheimer & Karsten Günther: *LaTeX kurz & gut.* O'Reilly Verlag: Köln, 3. erweiterte Auflage, 2008.

[8] Anthony Grafton: *Die tragischen Ursprünge der deutschen Fußnote.* Berlin Verlag: Berlin 1995.

[9] Markus Kohm & Jens-Uwe Morawski: *KOMA-Script. Eine Sammlung von Klassen und Paketen für LaTeX 2ε.* DANTE e. V., Lehmanns Media: Berlin; 3. erweiterte Auflage; 2008.

[10] Frank Mittelbach & Michel Goossens: *Der LaTeX-Begleiter.* Pearson Studium: München, 2. überarbeitete und erweiterte Auflage; 2005.

[11] Elke & Michael Niedermair: *LaTeX Das Praxisbuch.* Franzis' Verlag: Poing 2003.

[12] Petra Schlager & Manfred Thibud: *Wissenschaftlich mit LaTeX arbeiten.* Pearson Studium: München 2005.

[13] Ursula Stephany & Claudia Froitzheim: *Arbeitstechniken Sprachwissenschaft.* UTB 3259; Wilhelm Fink: Paderborn 2009.

[14] Manuel René Theisen: *Wissenschaftliches Arbeiten. Technik – Methodik – Form.* WiSt-Taschenbücher, Vahlen: München; 10. Auflage, 2000.

[15] Ralf Turtschi: *Praktische Typografie.* Verlag Niggli: Sulgen 1994.

[16] Herbert Voß: *Fußnoten in Tabellen mit* `blkarray`. Die TeXnische Komödie; (21)4 2007; S. 42–45.

Datenanalyse mit Sweave, LaTeX und R

Uwe Ziegenhagen

`R` ist eine leistungsfähige Open-Source-Sprache für alle Aspekte der statistischen Datenanalyse. Mit Sweave stellt `R` ein Werkzeug bereit, das es ermöglicht, sowohl die Quellcodes für `R`-Programme als auch den beschreibenden Text der Arbeit in einem Dokument zu halten.

R in Kürze

Die Geschichte von R geht auf das Jahr 1969 zurück, als John M. Chambers und seine Kollegen von den Bell Labs eine Sprachbeschreibung von S veröffentlichten, einer Programmiersprache für Statistik und Datenanalyse, die dann 1975 zum ersten Mal auf Honeywell-Rechnern implementiert wurde. Ross Ihaka und Robert Gentleman von der Universität Auckland in Neuseeland haben dann 1992 damit begonnen, mit R eine freie Implementation der Sprache zu schaffen. Heute ist R für viele Wissenschaftler das Werkzeug der Wahl, wenn es um die Visualisierung und Auswertung von Daten geht. Das R-Projektteam hat mehr als 500 Mitglieder, CRAN (das Pendant zu CTAN) zählt mehr als 1500 Pakete, die Zusatzfunktionen und Algorithmen für R bereitstellen.

R als Taschenrechner

Da der Fokus dieses Artikels mehr auf der Interaktion mit LaTeX liegt, soll an dieser Stelle keine ausführliche Einführung in R gegeben werden. Interessierte Leser seien daher auf die Bibliografie dieses Artikels verwiesen. Um jedoch wenigstens ein Grundverständnis für die Arbeit mit R zu vermitteln, möchte ich auf einige Aspekte der Sprache eingehen. Listing 1 zeigt einige einfache Rechenfunktionen in R.

Listing 1: Grundlegende Berechnungen mit R

```
1+2
1*2
1/2
1-2
2^2
```

```
sqrt(2)
sin(pi) # cos, tan
trunc(-pi) # -3
round(pi)  # 3
```

Die Datenstrukturen in R sind Vektoren, Matrizen und Dataframes. Vektoren und Matrizen können jeweils nur einen Datentyp aufnehmen, Dataframes (die nichts anderes als Listen von Objekten darstellen) hingegen können verschiedene Datentypen enthalten. Listing 2 zeigt einige Wege, wie Vektoren in R erstellt werden.

Listing 2: Erzeugung von Vektoren

```
a <- 1:3 # speichere Vektor 1..3 in a
b = 2:4 # speichere 2..4 in b
c(a,b) # [1] 1 2 3 2 3 4 # cat a & b
# generiere Sequenz
seq(1,2,by=0.1) # [1] 1.1 1.2 1.3 ...
# wiederhole 1..4 zweimal
rep(1:4,2) # [1] 1 2 3 4 1 2 3 4
```

Das letzte Beispiel in Listing 3 zeigt, wie man eine einfache lineare Regressionsrechnung in R durchführt. Der Vektor der unabhängigen Variable x enthält die Zahlen 1 bis 10, für den Vektor y der abhängigen Variable multiplizieren wir die einzelnen Komponenten des x-Vektors mit einem normalverteilten, zufälligen Faktor. Das lineare Modell wird über die `lm`-Funktion errechnet, die dann die Koeffizienten des Modells ausgibt.

Listing 3: Ein lineares Modell mit R

```
> x<-1:10
> y=rnorm(10)*x
> lm(y~x)

Call:
lm(formula = y ~ x)

Coefficients:
(Intercept)            x
     0.1079       1.0697
```

R Grafik

R verwaltet seine grafischen Ausgaben (siehe Listing 4 und die entsprechende Ausgabe in Abbildung 1) über sogenannte Graphics Devices, die das Grafikobjekt nehmen und in eine darstell- oder druckbare Form bringen. Die Liste der verfügbaren Devices ist sehr umfangreich. So gibt es Devices für PDF, Postscript, X11, Java und SVG, um nur ein paar zu nennen. Listing 5 zeigt beispielsweise, wie das PDF-Device angesteuert werden kann, um direkt aus R PDF-Dateien zu erzeugen.

Listing 4: Ein einfacher Plot

```
a<- c(1:10)
plot(a)
```

Listing 5: Beispielcode für die PDF Ausgabe

```
pdf(file = "c:/punkte.pdf",width = 6,
height = 6, onefile = FALSE,
family = "Helvetica",
title = "R Graphics Output",
fonts = NULL, version = "1.4",
paper = "special")

a<- c(1:10)
plot(a)
# auf Bildschirm-Device umschalten
dev.off()
```

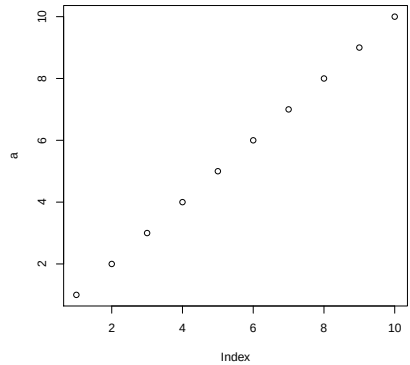

Abbildung 1: Durch Listing 4 erzeugte Grafik.

Das TikZ-Device

Besonders interessant in der Liste der verfügbaren Graphics Devices ist das TikZ-Device, das direkt aus R-Objekten Quellcode für Till Tantaus exzellentes Grafikpaket erzeugt. Das TikZ-Device erzeugt dabei je nach Vorgabe LaTeX-Code, der eigenständig übersetzt werden kann, oder aber Code, der in ein anderes LaTeX-Dokument eingefügt wird. Der Vorteil der Nutzung dieses Ausgabe-Device liegt darin, dass die Fonts des LaTeX-Dokuments genutzt werden und mathematische Symbole auch in der Überschrift und der Legende genutzt werden können. Listing 7 zeigt einen Ausschnitt aus der Datei, die mittels TikZ-Device aus Listing 6 erstellt wurde.

Listing 6: Beispielcode für das TikZ Device

```
tikz(file = "c:/test2.tex",standAlone=F)
# StandAlone=T
plot(1:10)
dev.off()
```

Listing 7: Ausschnitt aus dem erzeugten TikZ-Code

```
% Created by tikzDevice
\begin{tikzpicture}[x=1pt,y=1pt]
\draw[color=white,opacity=0] (0,0)
rectangle (505.89,505.89);
\begin{scope}
\path[clip] ( 49.20, 61.20) rectangle (480.69,456.69);
\definecolor[named]{drawColor}{rgb}{0.56,0.96,0.51}
\definecolor[named]{fillColor}{rgb}{0.13,0.09,0.52}
\definecolor[named]{drawColor}{rgb}{0.00,0.00,0.00}
\draw[color=drawColor,line cap=round,line join=round,
```

```
fill opacity=0.00,] ( 65.18, 75.85) circle (  2.25);
\draw[color=drawColor,line cap=round,line join=round,
fill opacity=0.00,] (109.57,116.54) circle (  2.25);
\end{scope}
```

Sweave und R

Einführung

Im zweiten Teil des Artikels möchte ich das von Friedrich Leisch entwickelte Paket `Sweave` vorstellen. `Sweave` ist Teil der Standard-R-Installation, muss also nicht separat installiert werden. Mit Hilfe von `Sweave` lassen sich LaTeX- und R-Code in einem Dokument halten, der R-Code wird dabei von »noweb«-Tags eingeschlossen. Noweb ist ein frei verfügbares Tool, das Donald Knuths Ansatz des »literate programming« implementiert. Noweb ist auch für andere Sprachen erhältlich, mehr zu diesem Thema findet der interessierte Leser in der Wikipedia. Für unsere Zwecke ist es nur wichtig zu wissen, dass am Anfang eines R-Schnipsels `<<>>=` steht und `@` am Ende.

In R ruft man dann den Befehl `Sweave("<Dateiname>")` auf, der die R-Schnipsel aus der Datei extrahiert, die Ergebnisse berechnet und – eingebettet in `verbatim`-Umgebungen – wieder in die Datei einfügt und diese als »Dateiname.tex« abspeichert. Diese Datei kann dann durch LaTeX übersetzt werden. Neben `Sweave` existiert noch ein weiterer Befehl, `Stangle`, der nur die R-Code-Teile aus der Noweb-Datei extrahiert. Dies ist dann sinnvoll, wenn nur die R-Codes bearbeitet oder weitergegeben werden sollen. Listing 8 zeigt ein sehr einfaches Beispiel für eine vollständige Noweb-Datei.

Listing 8: Einfaches `Sweave` Beispiel

```
\documentclass{article}
\begin{document}
<<>>=
1+1
@
\end{document}
```

Wenn die Datei in Listing 8 mit `Sweave` übersetzt wird, erhalten wir den in Listing 9 abgedruckten LaTeX-Code. Dieses Dokument kann dann in eine PDF- oder PostScript-Datei übersetzt werden, für Listing 8 ist diese in Abbildung 2 abgebildet. Wie wir im LaTeX-Dokument sehen können, benötigt `Sweave` das LaTeX-Paket gleichen Namens im Suchpfad, da dieses die notwendigen Input- und Outputbefehle für LaTeX definiert.

Listing 9: LaTeX Dokument erzeugt aus Listing 8

```
\documentclass{article}
\usepackage{Sweave}
\begin{document}
\begin{Schunk}
\begin{Sinput}
> 1 + 1
\end{Sinput}
\begin{Soutput}
[1] 2
\end{Soutput}
\end{Schunk}
\end{document}
```

```
> 1 + 1

[1] 2
```

Abbildung 2: Durch Listing 8 erzeugte Ausgabe.

Optionen für Sweave

Im `<<>>=` Kopf eines R-Schnipsels lassen sich verschiedene Optionen für die Transformation setzen. So unterdrückt `echo=false` zum Beispiel die Ausgabe des R-Codes, während `results=hide` die Ausgabe der Ergebnisse verhindert. Eine Kombination beider Befehle, die vielleicht auf den ersten Blick sinnlos erscheinen mag, ist jedoch vorteilhaft, wenn nur Daten gelesen werden oder Umgebungsvariablen gesetzt werden sollen. Da es auch R-Pakete gibt, die gültigen LaTeX-Code direkt erzeugen, kann man mit `results=tex` verhindern, dass die in sweave.sty definierte `verbatim`-Umgebung zum Einsatz kommt. Ergebnisse werden dann genauso ausgegeben, wie sie von R angeliefert werden.

Wenn der R-Schnipsel ein Bild erzeugt, muss die Option `fig=true` gesetzt sein. In der Standardeinstellung werden sowohl PS- als auch PDF-Versionen einer Grafik erstellt, mit `pdf=true/false` beziehungsweise `eps=true/false` kann dies jedoch angepasst werden. Die Größe des Plots kann über `width` und `height` gesetzt werden, beide Parameter erwarten die Größenangabe in Inch. Optionen können auch global für das Dokument gesetzt werden, wozu `Sweave` den `\SweaveOpts{<Option>}`-Befehl bereitstellt. Für Details sei auf das Handbuch verwiesen.

Einzelne Quellcodeteile lassen sich auch wiederverwenden, wenn man ihnen einen Namen mit `<<` *name*, `opt=... >>=` zuweist; die einzelnen Teile werden dann über `<<` *name*`>>` adressiert. Skalare Werte, wie zum Beispiel die Spalten- oder

Zeilenzahl einer Matrix, lassen sich auch direkt im Fließtext ausgeben. Dazu stellt R den `Sexpr{<\R-code>}` bereit. Die einzige Anforderung an den Rückgabewert ist, dass es sich um einen String handeln muss oder zumindest um ein R-Objekt, das sich in einen einzelnen String umwandeln lässt. In Listing 10 wird dieser Befehl genutzt, um die Anzahl von Zeilen und Spalten eines Datensatzes auszugeben.

Auswertung der Iris-Daten

Listing 10 zeigt ein kurzes Beispiel für eine Datenanalyse von IRIS-Daten, die einen der klassischen Datensätze unter anderem für die Clusteranalyse darstellen. Der Datensatz besteht aus 50 Beobachtungen verschiedener Pflanzensorten, die Variablen enthalten Angaben zur Länge und zum Durchmesser der Blütenblätter, etc. Im ersten Code-Schnipsel wird der Datensatz geladen. Da dieser Schritt recht uninteressant für die weitere Analyse ist, wird sowohl die Eingabe als auch die Ausgabe unterdrückt.

Um die Anzahl von Zeilen und Spalten auszugeben, wird der `\Sexpr()`-Befehl genutzt, bevor eine kurze deskriptive Analyse der Daten erfolgt. Im Anschluss berechnet R das lineare Regressionsmodell für zwei Variablen, die Ergebnisse werden über das `xtable` Kommando als LaTeX-Tabelle formatiert ausgegeben.

Der letzte Code-Schnipsel erzeugt dann ein Streuungsdiagramm für die Variablen *Petal.length* und *Sepal.width*; man beachte hier das `fig=true`-Kommando.

Listing 10: Sweave-Code zur Erzeugung von Abbildung 4

```
\documentclass[a4paper]{scrartcl}
\begin{document}
<<echo=false,results=hide>>=
data(iris) # load iris data
@

Der Datensatz hat \Sexpr{ncol(iris)} Spalten
und \Sexpr{nrow(iris)} Zeilen.

<<echo=false>>=
summary(iris$Petal.Length)
@

<<echo=false,results=tex>>=
xtable(lm(iris$Sepal.Width~iris$Petal.Length),
caption="Linear Model of Sepal.Width
and Petal.Length")
@

\begin{figure}
\centering
<<fig=true,echo=false>>=
```

```
pch.vec <- c(16,2,3)[iris$Species]
col.vec <- c(16,2,3)[iris$Species]
plot(iris$Sepal.Width,iris$Petal.Length,
col = col.vec,pch=pch.vec)
@
\caption{Plot of iris\$Petal.Length vs. iris\$Sepal.Width}
\end{figure}
\end{document}
```

Dynamische Reports

Das letzte Beispiel zeigt, wie Reports mit dynamischen Daten erzeugt werden können. Nehmen wir an, wir benötigen regelmäßig eine Übersicht des USD/EURO-Wechselkurses. Die Daten können von der Europäischen Zentralbank bezogen werden, die sie im CSV- und XML-Format bereitstellt. Das Herunterladen der Daten übernimmt `wget`, das wir über das `system()`-Kommando aus `R` heraus aufrufen können. Zum Entpacken der Daten kommt das `R`-interne ZIP-Werkzeug zum Einsatz, der Datensatz wird dann in `data` abgelegt.

Wie schon im vorigen Beispiel werden mit `\Sexpr()` die Dimensionen des Datensatzes ausgegeben, anschließend wird die Kursentwicklung für den gesamten Zeitraum der Daten dargestellt.

Listing 11: Sweave-Code mit dynamischer Datenquelle

```
\documentclass{scrartcl}
\begin{document}

<<echo=f,results=hide>>=
# Breite und Höhe des Plots
windows(width = 8, height = 4)
# wget zum Herunterladen der Datei
# und der Speicherung in d.zip
system("wget -O d.zip http://www.ecb.int/stats/eurofxref/eurofxref-hist.zip")
# Nutzung des eingebauten zip-Befehls zum Entpacken
zip.file.extract(file="eurofxref-hist.csv",zip="d.zip",unzip="",dir=getwd())
# read the data
data= read.csv("eurofxref-hist.csv",sep=",",header=TRUE)
@

Der Datensatz hat \Sexpr{nrow(data)} Wechselkurse, der neueste Kurs (\Sexpr{data$
 ↪Date[1]}) lautet \Sexpr{data$USD[1]}

\begin{figure}
\centering
<<fig=true,echo=false,width=15,height=6>>=
# Ausgabe des Plots
plot(data$USD,t="l", sub=paste(nrow(data)," datasets from ",data$Date[nrow(data)],"
 ↪ until ",data$Date[1]),asp=)
```

```
@
\end{figure}
\end{document}
```

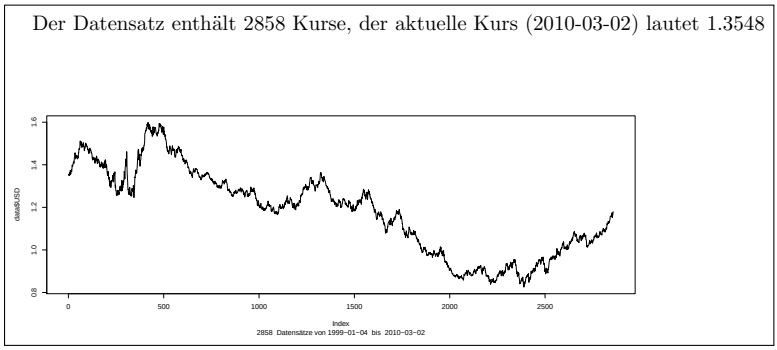

Abbildung 3: Wechselkursbeispiel

Anpassungen von Sweave

Der Quellcode von `Sweave.sty` ist auch für Nicht-TeX-Experten verständlich und lässt sich recht einfach anpassen. Um die `Sinput`- und `Soutput`-Routinen zu verändern, lässt sich beispielsweise mit `\lstnewenvironment` aus dem `listings` Paket das Aussehen der Umgebung komplett verändern. Listing 12 zeigt ein entsprechendes Beispiel.

Der Datensatz hat 5 Spalten und 150 Zeilen.

```
 Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
1.000   1.600   4.350   3.758   5.100   6.900
```

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	3.4549	0.0761	45.40	0.0000
iris$Petal.Length	-0.1058	0.0183	-5.77	0.0000

Table 1: Lineares Model von Sepal.Width und Petal.Length

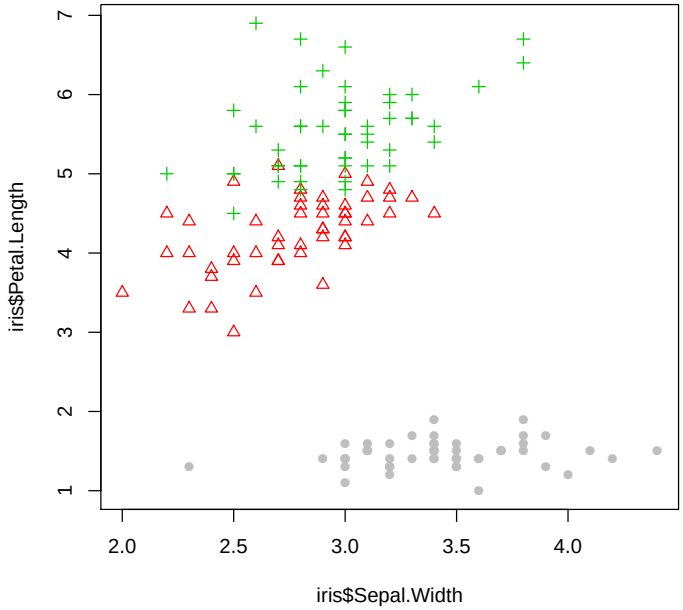

Figure 1: Plot von iris$Petal.Length vs. iris$Sepal.Width

Abbildung 4: Dokument erzeugt aus Listing 10

Listing 12: Beispiel für die Veränderung von `Soutput`

```
\lstnewenvironment{Soutput}[1][]
  {\lstset{basicstyle=\small,backgroundcolor=\color{green},language=R,#1}}
  {}
```

Zusammenfassung

R stellt eine unglaubliche Menge an Funktionen für die professionelle Auswertung von Daten bereit; zusammen mit `Sweave` lassen sich hochqualitative Reports oder Grafiken erzeugen, bei denen der Text des Dokuments an sich sowie die Quellen der eingesetzten Software in einem Dokument gehalten werden.

Literatur

[1] Michael J. Crawley: *The R Book*; Wiley; Juni 2007.

[2] Peter Dalgaard: *Introductory Statistics with R (Statistics and Computing)*; Springer; 2. Aufl.; August 2008.

[3] Friedrich Leisch: *Sweave: Dynamic Generation of Statistical Reports Using Literate Data Analysis*; in *Compstat 2002 — Proceedings in Computational Statistics* (Hg. Wolfgang Härdle und Bernd Rönz); S. 575–580. Physica Verlag, Heidelberg; 2002; ISBN 3-7908-1517-9 [image: Pick It!].

[4] Uwe Ligges: *Programmieren mit R (Statistik und ihre Anwendungen) (German Edition)*; Springer; 3. Aufl.; September 2008.

[5] Robert A. Muenchen: *R for SAS and SPSS Users (Statistics and Computing)*; Springer; Oktober 2008.

[6] John Verzani: *Using R for Introductory Statistics*; Chapman and Hall/CRC; November 2004.