– R + LaTeX –

-Grundlegendes-

`Datei:` Datei(-Format) bzw. –Endung
`R-Code:` direkte R-Kommandos für Konsole/Code-Chunk
`R-Paket:` Installation: `install.packages()` Laden: `require()`
`R-Argumente:` Argument in der angegebenen R-Funktion
Umgebung: Spezieller Textbaustein mit eigenen Befehlen (\begin{Umgebung}...\end{Umgebung})
Option: Chunk-Option (beeinflusst Ausgabe/Behandlung des Codes)
Kommentar: Erläuterungen zu Befehlen oder Optionen

-Sweave (per standardmäßig geladenem `utils`)-

Stichwort	Erläuterung/Befehle
Verarbeitung	`X.Rnw` → R: `Sweave("X.Rnw")` → LaTeX: `X.tex`
`Sweave.sty`	benötigt für den LaTeX-Lauf, zu finden im R-Installationspfad unter `R/share/texmf/tex/latex/` oder kompilieren per `Sweave("X.Rnw", stylepath=TRUE)`
Blöcke/Chunks	<<label, Optionen>>= `x <- pi/3 ...` # R-Befehle @ % << und @ müssen in erster Spalte stehen!
Einzelwert	`\Sexpr{x}` % im LaTeX-Teil und x einzelner Wert
Cache	Zwischenspeicher für wiederholte Kompilierungen per `cacheSweave`-Paket in Verbindung mit `Sweave("X.Rnw", driver=cacheSweaveDriver)` und Chunk-Option cache=TRUE/FALSE Achtung: Abhängigkeiten beachten!
Source-Datei	`Stangle("X.Rnw")` erzeugt lauffähige `X.R`
Wiederaufruf	per Labelname in einem anderen Chunk: `<<label>>` # führt Code erneut aus!
Unterdokument	`\SweaveInput{Unter.Rnw}` in einer `Haupt.Rnw`
Prompt/Forts.	`options(prompt="> ", continue="+")`
globale Opt.	`\SweaveOpts{Option=, ...}` % 1. Spalte, eine Zeile
`.Stex`-Sweave	`Sweave("X.Stex")` und Code-Chunks über \begin{Scode}{label, Option= ...} `R-Kommandos` \end{Scode} % \begin/\end jeweils in 1. Spalte!

Chunk-Option	Werte/Erläuterungen
label=	string Chunk-Label Standard: erste Chunkoption
echo=	TRUE/FALSE Ausgabe des Quellcodes in die `.tex`
keep.source=	TRUE/FALSE Kommentare im Quellcode erhalten
results=	hide R-Ausgabe unterdrücken, tex unformatiert übernehmen, verbatim als Maschinenschrift formatieren
eval=	TRUE/FALSE
fig=	TRUE/FALSE (obligatorische) Plot-Befehle im Chunk erzeugen Graphik-Datei (`.eps` & `.pdf`)
eps=	TRUE/FALSE mit fig=TRUE `.eps`-Graphikdatei
pdf=	TRUE/FALSE mit fig=TRUE `.pdf`-Graphikdatei
png=	TRUE/FALSE mit fig=TRUE `.png`-Graphikdatei
jpeg=	TRUE/FALSE mit fig=TRUE `.jpeg`-Graphikdatei
include=	TRUE/FALSE schreibt in die `.tex`-Datei: `\includegraphics{Dateiname-Chunklabel}` tatsächliche Größe im Dokument ist bestimmt durch: `\setkeys{Gin}{width=0.8\textwidth}` % variabel oder eigenes `\includegraphics[width=,height=]{}` in Verbindung mit include=FALSE
width=	6 Originalbreite der erzeugten Graphik(en) in Inch
height=	6 Originalhöhe der erzeugten Graphik(en) in Inch
prefix.string=	`Pfad/Dateinamenprefix` (Standard: `Dateiname-`)

-knitr + LaTeX-

Stichwort	Erläuterung/Befehle
Verarbeitung	`X.Rnw` → R: `knit("X.Rnw")` → LaTeX: `X.tex`
Blöcke/Chunks	<<label, Optionen>>= `x <- pi/3 ...` # R-Befehle @ % << und @ nicht zwingend in 1. Spalte!
Einzelwert	`\Sexpr{x}` % im LaTeX-Teil und x einzelner Wert
Source-Datei	`purl("X.Rnw")` erzeugt lauffähige `X.R`
ext. Chunks	einlesen per `read_chunk("extChunks.R")`; Format: `## -- label --` # 1. Chunk-Variante `R-Befehle` # kein Chunk-Ende-Zeichen `## @knitr label` # 2. Chunk-Variante `R-Befehle` # kein Chunk-Ende-Zeichen Aufruf wie bei Wiederaufruf:
Wiederaufruf	Variante 1: leerer Chunk mit wiederholtem Label Variante 2: Option ref.label="label" in leerem Chunk
Themes	`knit_theme$set(knit_theme$get()[1])` oder `css <- eclipse_theme(13331)` # eclipsecolortheme `knit_theme$set(knit_theme$get(css))`
Unterdokument	\Sexpr{`knit_child("Unter.Rnw", options=list())`} in einer `Haupt.Rnw` oder dort Chunk anlegen mit entsprechender child-Option: <<label, child="Unter.Rnw">>= für separates Kompilieren des Unterdokuments, dort: `set_parent("Haupt.Rnw")` einfügen
globale Opt.	`opts_chunk$set(Option=, ...)`

Option	Werte/Erläuterungen zur Chunk-Option
label=	string/"string" Chunk-Label Standard: 1. Chunkopt.
eval=	TRUE/FALSE Evaluieren des Code-Chunks

Option	Werte/Erläuterungen zur **Quellcode-Option**
echo=	TRUE/FALSE/3:5 Quellcodezeilen in die `.tex`
tidy=	TRUE/FALSE Quellcode ggf. umformatieren
highlight=	TRUE/FALSE Quellcode-Syntaxhervorhebung (je nach Theme)
prompt=	TRUE/FALSE Konsolensymbol > für Quellcode

Option	Werte/Erläuterungen zur **Ausgabe-Option**
include=	TRUE/FALSE generelle Unterdrückung der Ausgabe
results=	"markup" Ausgabe mit Syntax-Hervorhebung, "hide" R-Ausgabe unterdrücken, "asis" unformatiert übernehmen, "hold" alle Ausgaben nach Quellcode
message=	TRUE/FALSE Nachrichten anzeigen/unterdrücken, analog warning= (Warnungen) und error= (Warnung)
comment=	"##"/NA R-Ausgaben auskommentiert

Option	Werte/Erläuterungen zur **Cache-Option**
cache=	TRUE/FALSE/0/1/2/3 R-Objekte des Chunks in den Cache und bei Bedarf per (lazy-) laden verfügbar
dependson=	"label"/c(1,3)/c(-1,-3) gecacheten Chunk in Abhängigkeit der angegebenen Chunks ausführen
autodep=	FALSE/TRUE autom. Erkennen von Abhängigkeiten

Option	Werte/Erläuterungen zur **Graphik-Option**
fig.keep=	"high" erstelle nur High-Level-Plots, "none" keine Plots, "all" alle Zwischenstufen, "first" nur ersten und "last" nur letzten Plot
fig.show=	"asis" Graphik direkt nach Plotbefehl eingebunden, "hold" alle Graphiken nach dem Quellcode des Chunks, "hide" erzeugte Graphiken werden nicht eingebunden, "animate" erzeugt Animation ("aniopts=")
dev=	"pdf"/"png"/... Graphik-Device für die Plot-Ausgabe
dev.args=	list(pdf=list(colormodel="cmyk", ...), png=...)
fig.path=	"figure/" Pfad & Prefix der Graphikdateien
fig.width=	7 originäre Breite der Graphik-Datei(en) in Inch
fig.height=	7 originäre Höhe der Graphik-Datei(en) in Inch
out.width=	"0.9\textwidth" tatsächliche Breite im Dokument
out.height=	"" tatsächliche Höhe im Dokument
fig.align=	"default" keine Ausrichtung, "left", "center", "right"
fig.env=	"figure" Fließumgebung für Graphiken
fig.pos=	"htbp" Positionsargument für Fließumgebung
fig.cap=	"Meine Graphik ..." Bildunterschrift/Caption
fig.lp=	"fig:" LaTeX-Label-Prefix (`\label{fig:Chunklabel}`)

-LaTeX-Matrizen und Vektoren-

Stichwort	R-Funktion für `\begin{pmatrix}\Sexpr{...`
Matrix-Funk.	`MATRIX <- function(M) cat(paste(apply(M, 1, function(i) paste(i, collapse="& ")), collapse="\\\\\n"), "\\\\")`
Vektor	`\begin{pmatrix}\Sexpr{MATRIX(1:3)}\end{pmatrix}`
Matrix	`\begin{bmatrix}\Sexpr{MATRIX(M)}\end{bmatrix}`

-LaTeX-Graphiken per `tikzDevice`-

Argument	für Graphikdevice in `tikz(file="Fig.tex", ...)`
`width=`	`7` Graphikbreite, `height=7` -höhe in Inch
`bg=`	`"transparent"` Hintergrundfarbe (`fg="black"`)
`standAlone=`	`FALSE/TRUE` separat lauffähige `.tex`-Datei erzeugen
`bareBones=`	`FALSE/TRUE` tikz-Befehle ohne `tikzpicture`-Umg
LaTeX-Engine	tikzDefaultEngine, tikzDocumentDeclaration
Pakete ergän.	tikzLatexPackages, tikzMetricPackages

Dr. André König
akoenig@statistik.tu-dortmund.de

-Tabellen mit `xtable` und `kable()` aus `knitr`-

Für Tabellen-Chunks Option: results="asis"/tex (knitr/Sweave)!

Argument	der R-Funktion `xtable(T, ...)` aus `xtable`
`type=`	`"latex"/"html"` Tabellenausgabeformat
`caption=`	`NULL` Tabellenbeschriftung / 2-dim. Vektor (lang, kurz)
`label=`	`NULL` LaTeX-Label oder HTML-Anker
`align=`	`c("l", "p\{5em\}")` Spaltenausrichtungsvektor der Länge `ncol(T)+1` # Extraspalte Zeilennummern
`digits=`	2, Vektor der Nachkommastellen je Spalte (`ncol(T)+1`)
`display=`	`c("d"` (Integer), `"f"` (n.dd), `"e"` (n.dde+nn), `"E"` (n.ddE+nn), `"g"` (kürzer{f,e}), `"G"` (kürzer{f,E}), `"fg"` (f mit signif(x, digits)), `"s"` (Strings))

Argument	der R-Funktion `print.xtable(X, ...)` aus `xtable`
`scalebox=`	1 Skalierungsfaktor für Tabelle
`booktabs=`	`FALSE/TRUE` Tabelle mit `booktabs`-Linien
`latex.environments=`	`"center"` zusätzliche LaTeX-Umgebung
`tabular.environment=`	`"tabular"` LaTeX-Tabellenumgebung
`width=`	`\textwidth` Tabellenbreite (z.B. xtabular)
`floating=`	`TRUE/FALSE` Fließumgebung anschalten
`floating.environment=`	`"table"/"sidewaystable"` Fließumgebung
`format.args=`	`list(big.mark="'", decimal.mark=",")` für `formatC`
`NA.string=`	`""` Darstellung fehlender Werte
`include.rownames=`	`TRUE/FALSE` Zeilenbezeichner in erste Spalte
`include.colnames=`	`TRUE/FALSE` Spaltenbezeichner als Tab.-kopf
`only.contents=`	nur Tabelleninneres (inkl. Linien!)
`add.to.row=`	`list(pos=list(zeile=1), command="\\colorred")`
`comment=`	`TRUE/FALSE` LaTeX-Zeitstempel
`sanitize.text.function=function(x){`	LaTeX-Sonderzeichen ersetzen

Argument	der R-Funktion `kable(T, ...)` aus `knitr`
`format=`	`"latex"/"html"/"markdown"/"pandoc"` Tab-Format
`caption=`	`NULL` Tabellenbeschriftung
`booktabs=`	`FALSE/TRUE` LaTeX-Tabelle mit `booktabs`-Linien
`longtable=`	`FALSE/TRUE` LaTeX-longtable-Tabelle
`align=`	`c("l", "p\{5em\}")` Spaltenausrichtungsvektor der Länge `ncol(T)`
`digits=`	2, Vektor der Nachkommastellen je Spalte (`ncol(T)`)
`table.attr=`	`"id=\"mytable\""` als Beispiel für ein html-Tag

-R und html per Sweave (R2HTML) und `knitr`-

Sweave	in valider html-Datei mit `.Rnw`-Endung gewöhnliche Sweave-Syntax verwenden: <<label, Optionen>>= `x <- pi/3 ...` # R-Befehle @ % << und @ müssen in erster Spalte stehen!
Kompilieren	`require(R2HTML); Sweave("X.Rnw",driver=RweaveHTML)`
knitr	in valider html-Datei mit `.Rhtml`-Endung folgende Chunk-Syntax: <!--begin.rcode label, Optionen `x <- pi/3 ...` # R-Befehle `end.rcode-->` ;Einzelwert: `<!--rinline signif(x,2)-->`
Kompilieren	`require(knitr); knit("X.Rhtml")`

direkt über R2HTML	`HTMLStart(outdir=, filename="Test", autobrowse=FALSE, HTMLframe=TRUE, echo=FALSE)`
	`HTMLplot(Caption="")` nach Plot-Befehlen
	`HTMLStop()` # Ende der direkten html-Sitzung
	`HTML.title("Mein Bericht", file=, HR=1)`
	`HTML(summary(data), file=)`
	`HTMLInsertGraph("S.png", Caption="", file=)`
	`HTML(as.latex("f(x)=\int^x_a \log(t) dt"), file=)`

-R und Office-Dateien-

`odfWeave`	Sweave für Open Office
Verarbeitung	Open Office `In.odt` → R → Open Office `Out.odt`
Kompilieren	`odfWeave("Eingabe.odt", "Ausgabe.odt")`
Chunks	analog zu Sweave-Chunks («`label,Option=`») Chunk-Optionen: echo, eval, results, fig
Einzelwerte	\Sexpr{`mean(xvec)`}

`XLConnect` Befehle und Erläuterungen für Excel-Ausgabe:

```
wb <- loadWorkbook( "X.xls", create=TRUE) Excel-Datei-Verbin.
createSheet(wb, "RGB") Arbeitsblatt anlegen
writeWorksheet(wb,data=D,sheet="A",startRow=2,startCol=1)
saveWorkbook(wb)                          # Abspeichern der Datei
D <- readWorksheet(wb, sheet="RGB", header=TRUE)      # Laden
createName(wb, name="NR", formula="RGB!$C$2", overwrite=TRUE)
writeNamedRegion(wb, D[,3:5], name="NR")          # Daten eintr.
setCellStyle(wb, sheet="NR", row=3:4, col=4:9, cellstyle=cs)
addImage(wb, filename="G.png", name="NR", originalSize=TRUE)
```

`rtf` Befehle und Erläuterungen für `.rtf`/`.doc(x)` Ausgaben:

```
rtf <- RTF("Dok.rtf", # Datei(-pfad) anlegen (auch .doc)
font.size=12, (Schriftgröße) omi=rep(1,4)) (Seitenränder))
done(rtf) Datei schließen und schreiben
addHeader( rtf, title=, subtitle=, font.size=, TOC.level=1)
addParagraph(rtf, paste(...)) # ganzer Absatz ins rtf
addPageBreak(rtf) Seitenumbruch manuell einfügen
startParagraph(rtf) ... endParagraph(rtf) Absatzfunktionen
addText(rtf, paste("Sei &alpha;=", alpha, bold=T, italics=F))
addPlot(rtf, plot.fun=, width=6, height=4, res=300, ...)
addPng(rtf, "Bsp.png", width=6, height=4)
addTable(rtf, dat=tab,font.size=12,row.names=F,NA.string="",
         header.col.justify=,col.justify=c("R","L","C"))
```

-R und Markdown-

Markdown	Syntax und Erläuterungen
Titel	Führende Rauten: `# Titel`, `##` und `###`
Hervorheben	`*kursiv*` / `_kursiv_`, `**fett**` / `__fett__`
Listen	jeweils in neuer Zeile beginnen! unsortiert: * Item1 + Item1a + Item1b * Item2 sortiert: 1. Item1 + Item1a + Item1b 2. Item2
Umbrüche	Zeile: ≥2 Leerzeichen, Seite: `******`
Links	URL: direkt oder per `[Textlink](http://url.com)` im Dokument: `[Linktext][id]` verweist auf `[id]: figures/img.png "Title"`
Formeln	`$f(x)=x^2$` in der Zeile oder `$$\frac{\pi}{2}$$`
Bilder	`![altern. Text](http://url/logo.png)` (oder Pfad!)

R `markdown`	Syntax und Erläuterungen
Verarbeitung	R + Markdown `In.Rmd` → R → Markdown `Mid.md` → Endformat (`Out.html`/`Out.docx`/`Out.pdf`/...)
Kompilieren	`knit("X.Rmd"); markdownToHTML("X.md", "X.html")`

Code-Chunks	```` ```{r label, knitr-Optionen} ````
	R-Befehle # z.B. Plots/Berechnungen
	```` ``` ````

Einzelwert	`` `r mean(x)` `` muss skalar-wertig sein
`rmarkdown`	(benötigt Pandoc als Markdown-Interpreter)
Installation	`devtools::install_github("rstudio/rmarkdown")`
Kompilieren	`render("X.Rmd", "all"/Formatname/Formatfunktion)`

Form.-fkt.	`html_document(toc=TRUE, fig_caption=T, ...)`
	`pdf_document(fig_width=6,latex_engine="pdflatex")`
	`word_document(reference_docx="default")`
	`beamer_presentation(incremental=T,theme="Dresden")`
	`ioslides_presentation(logo=, highlight="tango")`

Verarbeitung	Syntax und R-Chunks wie oben

Tabelle 1: Kurzübersicht der `knitr`-Syntax nach Dateiformat, wobei ⋆ die Position der Chunkoptionen anzeigt.

Eingabeformat	`.Rnw`	`.Rtex`	`.Rhtml`	`.Rmd`
Chunk-Beginn	`<<⋆>>`	`% begin.rcode ⋆`	`<!-- begin.rcode ⋆`	```` ```{r ⋆} ````
Chunk-Ende	@	`% end.rcode`	`end.rcode -->`	```` ``` ````
Einzelwert	`\Sexpr{x}`	`\rinline{x}`	`<!-- rinline x -->`	`` `r x` ``
Ausgabeformat	`.tex`	`.tex`	`.html`	`.md`

Tabelle 2: Übersicht ausgewählter Beispieldokumente (eingebettet, Quellcode per Klick auf Datai aufrufbar).

Datei	Erläuterung	Datei	Erläuterung	Datei	Erläuterung
`.Rnw`	LaTeX mit Sweave-Syntax	`.odt`	`odfWeave` Sweave-Syntax	`.Rnw`	html mit Sweave-Syntax
`.Rnw`	LaTeX mit knitr-Syntax	`.R`	`XLConnect` Beispiel	`.Rhtml`	html mit knitr-Syntax
`.Stex`	LaTeX mit Sweave-Syntax	`.R`	`rtf` Beispiel	`.Rmd`	`knitr` + `markdown`
`.Rtex`	LaTeX mit knitr-Syntax	`.R`	`R2HTML` html direkt	`.Rmd`	`rmarkdown`-Beispiel

akoenig@statistik.tu-dortmund.de