03 R Einführung

Einführung in die quantitativen Forschungsmethoden

Heute

- Quiz zu letzter Sitzung
- Einführung in RStudio
- Einfache Befehle & Funktionen
- Weitere Befehle & Funktionen
- Umgang mit dataframes

Quiz zu letzter Sitzung

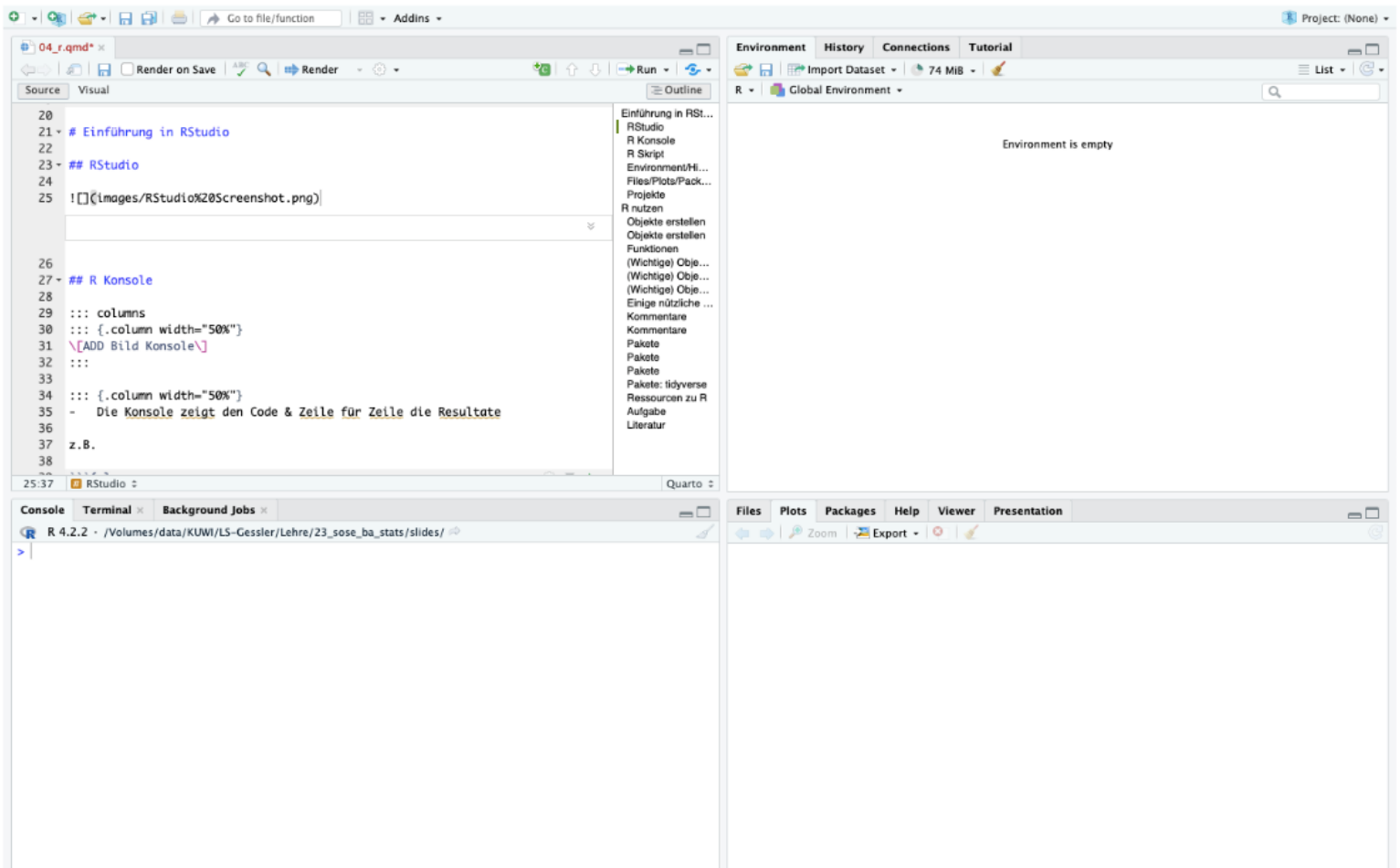
Quiz Link

Einführung in RStudio

RStudio

R Skript

Konsole



Environment

Plots &
Help Tabs

R Konsole

- Die Konsole zeigt den Code & Zeile für Zeile die Resultate

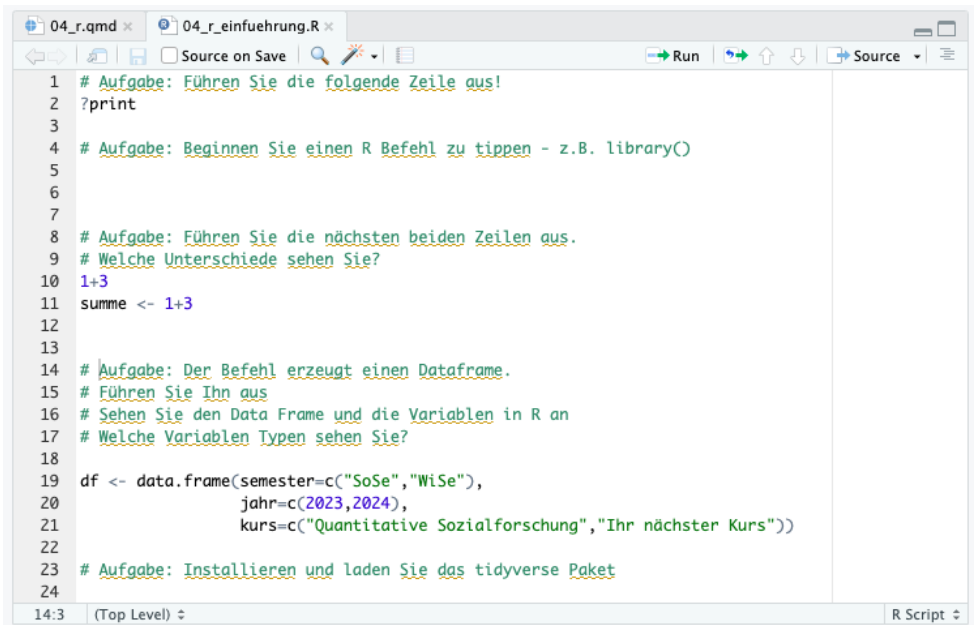
z.B.

```
1  1+1
```

```
[1] 2
```

- Befehle lassen sich in der Konsole schnell tippen, bleiben aber nicht erhalten → langfristig mehr Arbeit!
  - Reproduzierbarkeit

R Skript



- 'Skript' mit Folge von R Befehlen
  - abspeicherbar als .R-Datei

- Tipps:
  - Sie können eine oder mehrere Zeilen markieren und mit `Strg` und `Enter` ausführen (Apple: `Cmd`)
  - eine einzelne Zeile können Sie auch ausführen, wenn Sie nach einem Klick in die Zeile `Strg` und `Enter` ausführen (Apple: `Cmd`)
  - wenn Sie im Skript Befehle tippen, schlägt RStudio vor, wie es weitergeht

→ **probieren Sie es selbst**

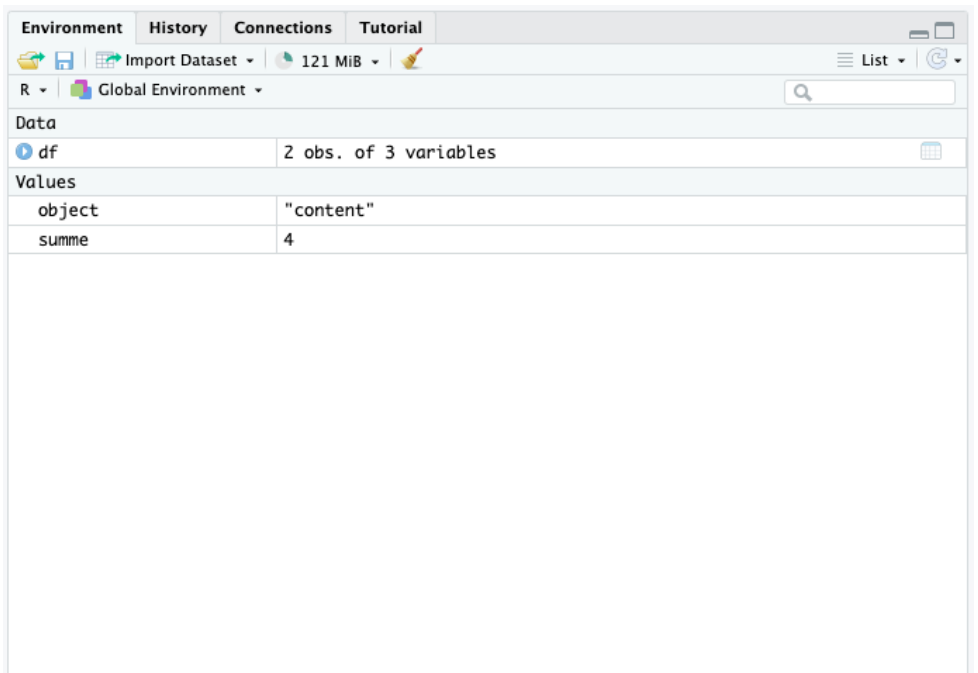
R Skript

- Erstellen Sie ein neues Skript (Datei → neue Datei → R-Skript)
- schreiben Sie zwei Zeilen, z.B.

```
1+1
print("hallo")
```

- führen Sie die Zeilen aus

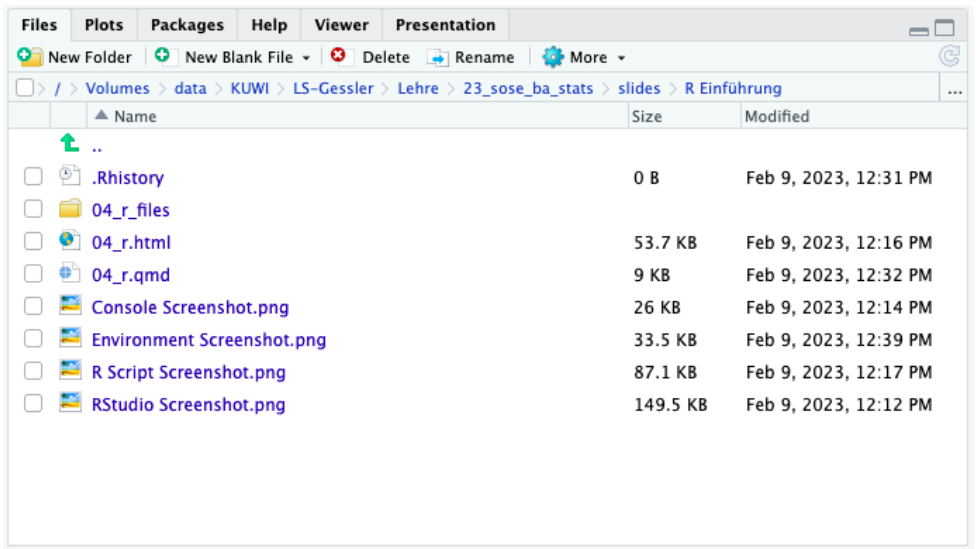
Environment/History



- **Environment** / Umgebung: Aktuelle Datensätze, Variablen, Funktionen, …
  - gut um Variablennamen nachzusehen & den Überblick zu behalten
- **History**: Chronologie der ausgeführten Befehle

- Praktisch:
  - bei einigen Objekt-Typen können Sie auf den Pfeil oder ein Symbol klicken, um in das Objekt zu sehen (z.B. für Variablennamen in einer Tabelle)

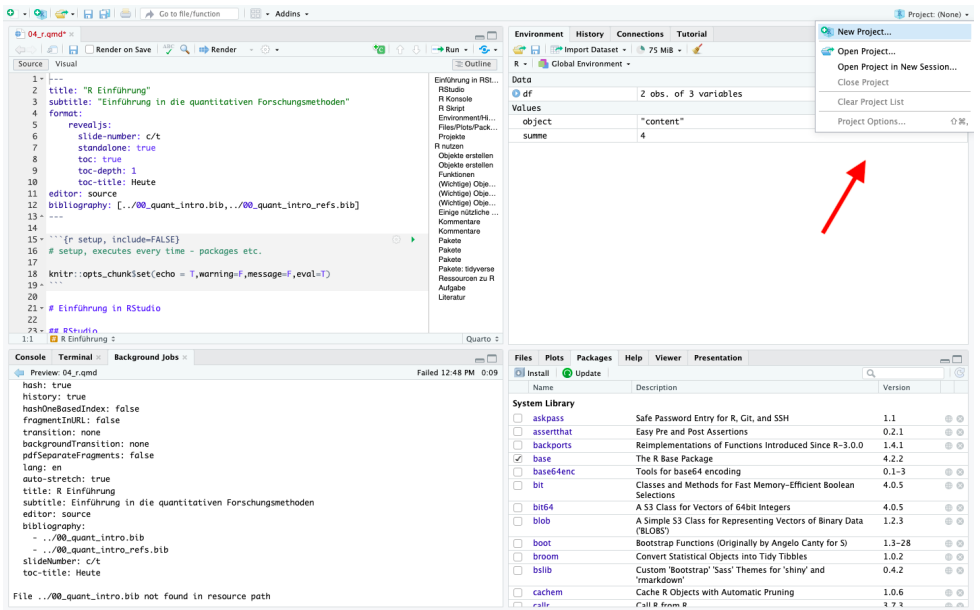
Files/Plots/Packages/Help/Viewer



- **Files**: Dateien, z.B. im aktuellen Ordner
  - z.B. ihre Skripte und Daten
- **Packages** - Übersicht über installierte und geladene Pakete
  - hier sollten Sie bereits einige Pakete sehen
- **Help**: R Dokumentation
  - wenn Sie im Skript oder in der Konsole `?rbefehl` tippen öffnet sich hier die Hilfe zu `rbefehl()`, z.B. `?print`

→ **probieren Sie es selbst**

Projekte



- **Projekte** helfen in R ‘Ordnung zu halten’
  - fester Ordner für Skripte, Datensätze etc. → keine langen Dateipfade
  - speichern der history für dieses spezifische Projekt
  - v.a. hilfreich, wenn Sie mehrere Projekte haben
    - **Probieren Sie es selbst**: erstellen Sie ein Projekt für den Kurs im Ordner, in dem Sie die Kursinhalte speichern

Einfache Befehle & Funktionen

Objekte erstellen

Um in R etwas (z.B. einen Datensatz, ein Ergebnis, …) zu speichern, müssen Sie es mit

`<-`

an ein Objekt zuweisen

→ Den Namen des Objekts können Sie selber wählen

→ Durch aufrufen des Objektnamens können Sie auf das Objekt (z.B. den Datensatz, das Ergebnis, …) zugreifen & damit weiterarbeiten

Objekte erstellen

Führen Sie diese beiden Befehle aus. Welche Unterschiede sehen Sie? → **probieren Sie es selbst**

```
1  1+3
2  summe <- 1+3
3  summe
```

→ Im ersten Fall berechnen wir `1+3`, aber weisen das Ergebnis keinem Objekt zu

→ Im zweiten Fall erscheint in unserer Umgebung ('environment') ein neues Objekt namens `summe`, in dem das Ergebnis gespeichert ist

Objekte erstellen

Objekte haben verschiedene Typen

- z.B. **numerisch**: `object1 <- 1+5`
- z.B. Text (**character**): `object2 <- "text"`

- Gut zu wissen:
  - Sie sehen bei manchen Objekttypen (Vektoren) auch im environment, welche Art von Objekt sie haben
  - mit `class(object2)` können Sie den Typ des Objekts `object2` abfragen
  - Besonderheit bei Text: Anführungszeichen `"` zeigen R, dass es Text, aber kein auszuführender Befehl ist!

Objekte erstellen

Objekte können oft mehrere Werte haben:

- **Vektoren** sind der einfachste Typ Objekt: ein oder mehrere Objekte des gleichen Typs, verkettet durch `c()`
  - z.B. numerische Vektoren: `vektor1 <- c(1,15,7)`
    - als Spanne: `vektor1 <- 1:15`
  - z.B. Text (‘character’) Vektoren: `vektor2 <- c("text","xx","bausteine")`

Funktionen

Funktionen sind Befehle, die R ausführt - z.B. etwas drucken

```
1 print(x="Hallo!")
```

```
[1] "Hallo!"
```

```
1 print(summe)
```

```
[1] 4
```

- Das Objekt der Funktion (hier: “Hallo”) nennen wir ein **Argument**
  - Argumente können durch Ihren Namen (hier: x) oder ihre Position (hier: erstes Argument) spezifiziert werden → Llaudet and Imai (2023), S. 12

Funktionen können keine, eine oder viele Argumente haben

Einige nützliche Funktionen

Ein paar besonders praktische Funktionen helfen uns in die Objekte ‘hineinzusehen’

```r
jahreszahlen <- 1980:2010
head(jahreszahlen)
```

```
[1] 1980 1981 1982 1983 1984 1985
```

```r
# head: erste n  Observationen
head(jahreszahlen,n=1) # Standard: n=6
```

```
[1] 1980
```

```r
# table(): Tabelle mit Häufigkeiten
table(jahreszahlen)
```

```
jahreszahlen
1980 1981 1982 1983 1984 1985 1986 1987 1988 1989 1990 1991 1992 1993 1994
1995
   1    1    1    1    1    1    1    1    1    1    1    1    1    1    1
1
1996 1997 1998 1999 2000 2001 2002 2003 2004 2005 2006 2007 2008 2009 2010
   1    1    1    1    1    1    1    1    1    1    1    1    1    1    1
```

Kommentare

Kommentare lassen sich zu R Code mit # hinzufügen - ob am Anfang der Zeile oder währenddessen

→ Text ab # wird ignoriert, hilft aber Ihrem Verständnis

```
1 print("Hallo") # druckt Hallo
```

```
[1] "Hallo"
```

```
1 # einzeiliger Kommentar
```

…hier weiß `print()` übrigens aus der Reihenfolge, was das Argument ist!

Kommentare

Viele nutzen Kommentare mit # auch um Ihren Code zu strukturieren

```
#### Daten laden ####
# Überschrift ----
```

Zeilen die mit #### oder ---- enden wandelt RStudio in Sektionen um

→ das macht die Navigation in langen Skripten leichter

Aufgabe

Bearbeiten Sie den ersten Teil des Übungsskripts

→ **probieren Sie es selbst**

Weitere Befehle & Funktionen

Pakete

- R basiert auf sog. ‘packages’: Software-Pakete, die bestimmte Funktionen definieren und damit die Funktionalität von R erweitern
  - z.B. `stats` für Statistik und `utils` für R Kommandos - vorinstalliert
  - z.B. `ggplot2` für Grafiken
  - z.B. `dplyr` für Datenmanipulation

Pakete

Pakete installieren

(Die meisten) R-Pakete müssen **vor der ersten Nutzung** aus dem Internet installiert werden, z.B.:

```
1 install.packages("tidyverse") # installiert Paket tidyverse
```

Pakete laden

Zusätzlich müssen Pakete **vor jeder Nutzung** geladen, d.h. aktiviert werden:

```
1 library(tidyverse) # läd Paket tidyverse
```

Pakete: tidyverse

Das `tidyverse` ist eine Sammlung von Paketen, die häufig genutzt werden und aufeinander aufbauen. Wir nutzen diese Pakete oft im Kurs.

```
1 install.packages("tidyverse") # installiert Paket tidyverse
```

Pakete

Tipp: Wenn sie nur einmalig einen Befehl aus dem Paket nutzen wollen können Sie ihn auch aus dem installierten Paket ohne laden des Pakets aufrufen:

→ hilfreich für Pakete, die wir für laden der Daten nutzen

```
1  # importiert Befehl read_dta() aus dem haven Paket
2  haven::read_dta("files/file.dta")
```

Objekte

- **Dataframes** sind Datensätze - also mehrere Beobachtungen, über die wir verschiedene Informationen (Variablen) haben
  - jede Zeile: Observation (z.B. 1 Person, 1 Land, …)
  - jede 'Spalte': Variable (z.B. das jeweilige Alter)
- Datensätze können verschiedene Objekttypen kombinieren - z.B. numerische und Text-Vektoren
- Sie können einzelne Variablen im Datensatz mit einem `$` Zeichen aufrufen: `Datensatz$Variable`

```
1  df <- data.frame(semester=c("SoSe","WiSe"),
2                   jahr=c(2023,2024),
3                   kurs=c("Quantitative Sozialforschung","Ihr nächster Kurs"))
4  df$semester
```

```
[1] "SoSe" "WiSe"
```

→ Machen Sie sich mit der Struktur vertraut!

(Wichtige) Objekttypen

- …es gibt aber noch viele andere Objekttypen
  - z.B. Listen - unordentlich & unbeliebt bei allen R Lernenenden!
  - z.B. ‘logicals’ (TRUE/FALSE)
  - z.B. z.B. ganze Zahlen (integer) vs. Kommazahlen (double)
  - z.B. (un)sichtbare Attribute

→ Manche lernen wir später noch kennen

Einige nützliche Funktionen

Die vorhin gelernten Funktionen sind auch für Dataframes nützlich

```r
# head: erste n  Observationen
head(df) # Standard: n=6
```

```
  semester jahr                          kurs
1     SoSe 2023 Quantitative Sozialforschung
2     WiSe 2024            Ihr nächster Kurs
```

```r
# table(): Tabelle mit Häufigkeiten
table(df$semester)
```

```
SoSe WiSe
   1    1
```

Zusatzfunktion für Datensätze: `View()` öffnet ein neues Fenster

```r
# neues Fenster mit Übersicht der Tabelle
View(df)
```

Umgang mit dataframes

Erstellen neuer Variablen

In Dataframes können wir Variablen ähnlich erstellen, wie wir bisher Objekte erstellt haben

```
1  # neue Variable ects erstellen
2  df$ects <- c(9,9)
```

Dabei müssen wir darauf achten, dass wir die richtige Anzahl Observationen haben - sonst erhalten wir eine Fehlermeldung!

```
1  # neue Variable ects erstellen
2  df$ects <- c(9,9,9)
```

```
Error in `$<-.data.frame`(`*tmp*`, ects, value = c(9, 9, 9)): Ersetzung hat 3
Zeilen, Daten haben 2
```

`ifelse()`

Manchmal wollen wir Variablen abhängig von anderen Variablen erstellen - z.b. um etwas umzukodieren.

→ dabei hilft uns `ifelse()`

```r
jahreszahlen <- 1980:2010
generationen <- data.frame(jahr=jahreszahlen)

# ifelse(Bedingung,
#        Wert wenn-ja,
#        Wert wenn-nein)
generationen$generation <- ifelse(generationen$jahr<2000,
                                  "millenial",
                                  "generation z")
```

Datensätze gruppieren

Wir können data frames auch gruppieren → das wird später noch wichtig werden

```
1 # group_by(datensatz, variable)
2 gen_gruppiert <- group_by(generationen, generation)
```

z.B. können wir die Datensätze dann mit `tally()` nach Gruppen auszählen

```
1 tally(gen_gruppiert)
```

```
# A tibble: 2 × 2
  generation        n
  <chr>         <int>
1 generation z     11
2 millenial        20
```

Daten zusammenfassen

Gruppieren hilft auch dabei bestimmte Werte - z.B. Mittelwerte - nach Gruppen zusammenzufassen

→ `summarize()`

```
1 # summarize(Datensatz, Zusammenfassungsfunktion)
2 summarize(generationen, mean=mean(jahr))
```

```
  mean
1 1995
```

```
1 summarize(gen_gruppiert, mean=mean(jahr))
```

```
# A tibble: 2 × 2
  generation    mean
  <chr>        <dbl>
1 generation z 2005
2 millenial    1990.
```

→ mehr zu Mittelwerten in Sitzung 5!

Aufgabe: Umfrage

Bearbeiten Sie den zweiten Teil der Übungsaufgaben, inklusive der Aufgaben zu unseren Umfrage-Antworten

Ressourcen zu R

Die folgenden Bücher kann man gratis online lesen oder als Kopie kaufen:

- Grolemund and Wickham (2017): R for Data Science, Einführung in die **Datenanalyse** → Online Version
- Grolemund (2014): Hands-On Programming with R, **Fokus auf R Code** → Online Version
- Navarro (n.d.): Learning statistics with R: A tutorial for psychology students and other beginners, **Fokus auf Statistik** → Online Version
- Douglas et al. (2022): sehr detaillierte **Schritt für Schritt** Einführung in R → Online Version
- Gehrau, Maubach, and Fujarski (2022) - Nachschlagewerk zu Statistik in R auf deutsch
- Wickham (n.d.): Advanced R, **Fortgeschritteneres Buch**, insbesondere um Programmieren zu lernen oder zu verbessern → Online Version
- weitere Bücher im Syllabus

Nächste Woche: Sozialwissenschaftliche Daten

Thema: Was sind Umfragedaten und welche Besonderheiten haben Sie?

- Gehring and Weins Grundkurs Statistik für Politologen und Soziologen., 70-90 (Kapitel 4), 41-51, 61-67 (Kapitel 3)
- optional psychologische Hintergründe von Umfragen: Rainer Schnell SurveyInterviews: Methoden standardisierter Befragungen, Studienskripten zur Soziologie (Wiesbaden: Springer Fachmedien Wiesbaden, 2019), https: //doi.org/10.1007/978-3-531-19901-6., 21-52
- optional Arten von Fragen / Fragetechniken: Schnell, 65-94

→ Laden Sie den Datensatz auf der Moodle Seite möglichst bereits vor der Sitzung herunter

Literatur

Douglas, Alex, Deon Roos, Francesca Mancini, Ana Couto, and David Lusseau. 2022. *An Introduction to R*.

Gehrau, Volker, Katharina Maubach, and Sam Fujarski. 2022. *Einfache Datenauswertung mit R: Eine Einführung in uni- und bivariate Statistik sowie Datendarstellung mit RStudio und R Markdown*. Wiesbaden: Springer Fachmedien. https://doi.org/10.1007/978-3-658-34285-2.

Grolemund, Garrett. 2014. *Hands-On Programming with R*.

Grolemund, Garrett, and Hadley Wickham. 2017. *R for Data Science*.

Llaudet, Elena, and Kosuke Imai. 2023. *Data Analysis for Social Science: A Friendly and Practical Introduction*. Princeton: Princeton University Press.

Navarro, Danielle. n.d. “Learning Statistics with R.”

Wickham, Hadley. n.d. *Advanced R*. Accessed February 10, 2020.