

# Evaluation Randomisiert-Kontrollierter Studien und Experimente mit R

## Programm

---

Prof. Dr. David Ebert & Mathias Harrer

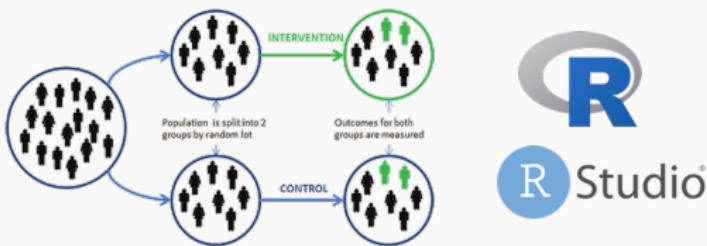
Graduiertenseminar TUM-FGZ

Psychology & Digital Mental Health Care, Technische Universität München

# **Herzlich Willkommen!**

---

## Evaluation Randomisiert-Kontrollierter Studien und Experimente mit R



"Zwei Workshops in Einem"

- grundlegende **statistische Methoden** zur **Analyse randomisiert-kontrollierter Studien** ("randomized controlled trials"; RCTs) ...
- ...und deren praktische **Umsetzung** in der **Programmiersprache R**.

**Fokus: zweiarmige, randomisiert-kontrollierte Studien** zur Evaluation medizinischer oder gesundheitsfördernder Interventionen.

Methoden sind aber auch übertragbar auf Bereiche wie Pädagogik, Soziologie, Ökologie, Wirtschaftswissenschaften etc. → überall, wo der Effekt einer “Behandlung” untersucht werden soll!



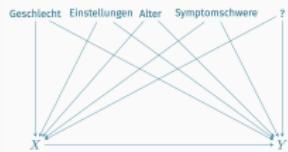
## Block I | R Entdecken

- ✓ Einführung in R & RStudio
- ✓ R Basics: Objektklassen, Funktionen, Operatoren, ...



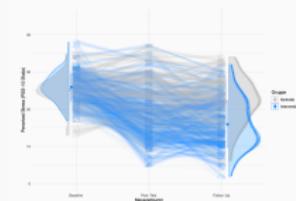
## Block II | Einführung & Hintergrund

- ✓ Methodischer Hintergrund von randomisierten Studien
- ✓ Import und Manipulation von Studiendaten mit R
- ✓ Schätzung fehlender Werte



## Block III | Analysieren & Verstehen

- ✓ Statistische Wirksamkeitsanalyse
- ✓ Alphakorrektur, Effektstärkenberechnung, Visualisierung, ...



# Vorwissen



# Voraussetzungen

- ✓ Installation von **R** (4.0.0+) und **RStudio** auf dem Computer.
- ✓ Vertrautheit mit basalen **Konzepten (frequentistischer Inferenzstatistik)** (Lage-/Streuungsmaße,  $p$ -Werte, (Ko-)varianz, lineare Regression, ...).
- ✓ Grundkenntnisse **experimenteller Forschungsdesigns**, sowie Gefährdungen von deren Validität.
- ✗ Vorkenntnisse in R (sind aber hilfreich!)
- ✗ Erfahrungen bei der Analyse randomisiert-kontrollierter Studien

...und das Wichtigste:

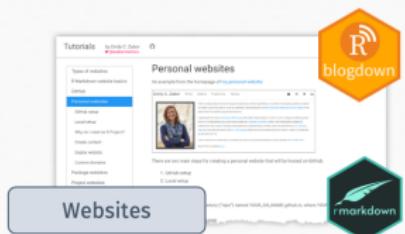
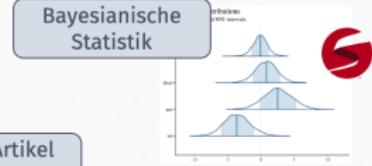
**DON'T PANIC**



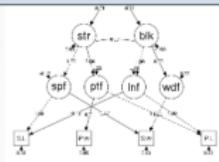
## **R & RCT-Analyse Lernen: Warum Es Den Aufwand Wert Ist**

---

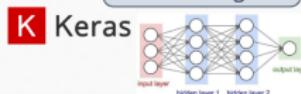
# R kann (fast) alles!



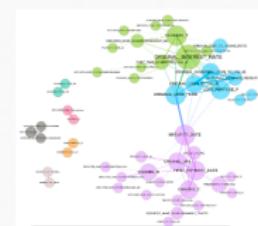
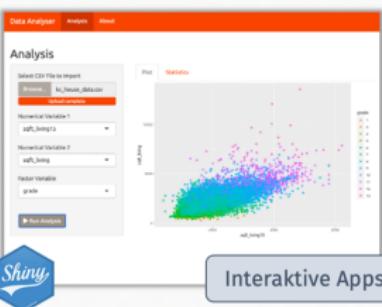
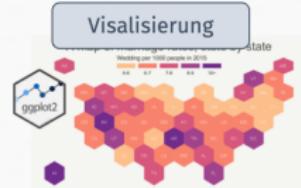
## Strukturgleichungsmodelle



## Machine/Deep Learning



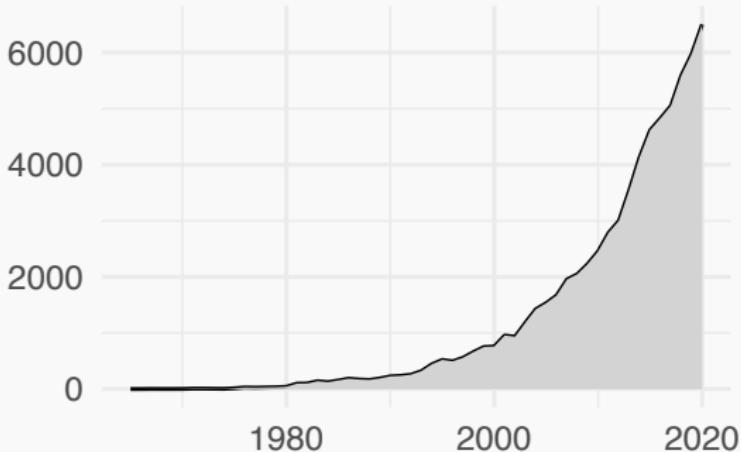
## Visualisierung



## Python Interface



...und sind ein über Disziplinen hinweg weit verbreitetes Forschungsdesign.



**Figure 1:** Hits für "randomized controlled trial" in PubMed, 1965-2020.

## Begleit-Website

---

[protectlab.org/workshop/rct-evaluation-in-r/](http://protectlab.org/workshop/rct-evaluation-in-r/) (auch nach Kursende verfügbar)

- **Foliensätze** für alle behandelten Themen
- **Zusätzliche Inhalte**, die nicht/nur knapp im Workshop behandelt werden
- **Kursmaterialien** & weiterführende Literatur
- **Code** für alle Praxis-Beispiele

The screenshot shows a website layout for a workshop. On the left, there's a sidebar with navigation links for various topics like 'Evaluation of RCTs mit R', 'Workshops', 'Programm', etc. The main content area has a title 'Evaluation Randomisiert-Kontrollierter Studien mit R' and a sub-section 'Workshop'. Below the title is a photograph of a person's hands typing on a laptop keyboard. To the right of the image are two code snippets in R:

**Imputation**

```
set.seed(123)
imp<-na.omit(data,
  predictorMatrix = pred,
  method = imp.method,
  imputationFunction = imp.function,
  group = imp.group.variable,
  m = 25, maxit = 25) → imp

save(imp, file = "data/imp.rda")
```

**Copy**

**Diagnostik**

```
# Trace plots
plot(imp,
  layout = c(4, ceiling(sum(!no.missing)/2)))

# Kernel densities
densityplot(imp, ~ pss.1 + pss.2)
densityplot(imp, ~ pss.1 + pss.2 | as.factor(group))
```

**Copy**

# Evaluation Randomisiert-Kontrollierter Studien und Experimente mit R

Einführung in R und RStudio

---

Prof. Dr. David Ebert & Mathias Harrer

Graduiertenseminar TUM-FGZ

Psychology & Digital Mental Health Care, Technische Universität München

## Block I | R Entdecken

- ✓ Einführung in R & RStudio
- ✓ R Basics: Objektklassen, Funktionen, Operatoren, ...



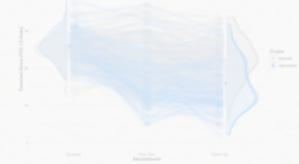
## Block II | Einführung & Hintergrund

- ✓ Methodischer Hintergrund von randomisierten Studien
- ✓ Import und Manipulation von Studiendaten mit R
- ✓ Schätzung fehlender Werte



## Block III | Analysieren & Verstehen

- ✓ Statistische Wirksamkeitsanalyse
- ✓ Alphakorrektur, Effektstärkenberechnung, Visualisierung, ...



## **Was ist R?**

---

R ist eine kostenfreie Programmiersprache und Softwareumgebung, mit besonderen Schwerpunkt bei der statistischen Datenanalyse.

- R unterscheidet sich von **Graphical User Interface** (GUI)-basierten Statistikprogrammen (wie IBM SPSS oder STATA). GUI-Software kann Analysen auf “Knopfdruck” durchführen, ist aber stets auf die Voreinstellungen der Hersteller begrenzt.
- R fordert im Gegenzug **mehr Hintergrundwissen**, bietet als vollständige Programmiersprache **maximale Flexibilität** - in R ist alles implementierbar!
- Überall auf der Welt können Experten zur Weiterentwicklung von R beitragen, vor allem durch sogenannte **Packages**.

# Über R (II)

**SPSS Data Editor**

The SPSS Data Editor window shows a dataset named "cable\_survey.sav". The "Analyze" menu is open, with "Discriminant" highlighted. The dataset contains 29 observations and 10 variables. The variables are:

- customer\_id
- age
- income
- marital
- homelife
- addressat
- service\_type
- service\_charge
- condemnation
- internet

**R Console**

The R Console window displays the following R session output:

```

R version 3.4.4 (2018-03-15) -- "Someone to Lean On"
Copyright (C) 2018 The R Foundation for Statistical Computing
Platform: i386-w64-mingw32/i386 (32-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

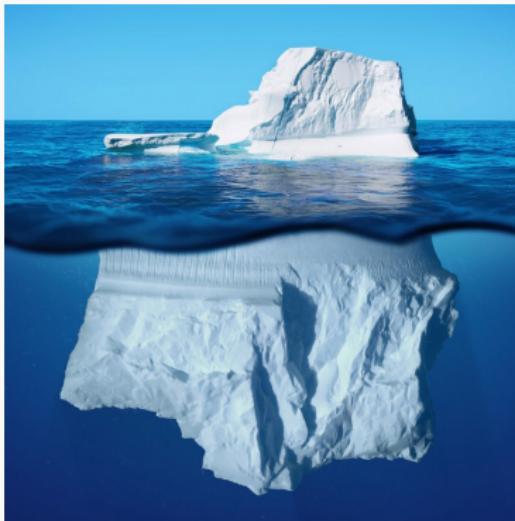
Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> |

```

## Die Programmiersprache R muss wie jede Sprache erlernt & geübt werden

- Frustration ist ein natürlicher Teil des Lernprozesses. Aber die anfängliche Anstrengung wird sich lohnen:



Mit etwas Zeit und Mühe kann jeder R erlernen!

## Was ist RStudio?

---

**RStudio ist ein Computerprogramm, das es ermöglicht, R auf “einfachere” Weise zu nutzen** (*integrated development environment; IDE*).

RStudio erleichtert u.a.

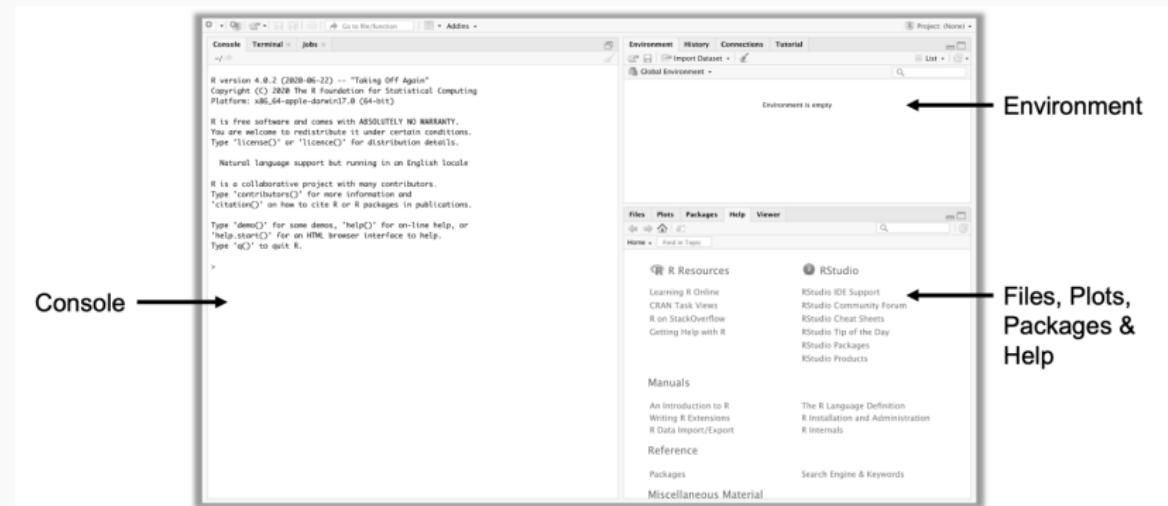
- das Schreiben und Ausführen von R Code.
- den Umgang mit Daten, Packages und Output.

## Cave: R versus RStudio

RStudio ist nicht identisch zu R: Zur Nutzung von RStudio muss R (die "Sprache") bereits installiert sein.



RStudio wird beim ersten Öffnen wahrscheinlich so aussehen:



(Harrer et al., 2021, Kap. 2.1)

## Fenster in RStudio

### Console

Die Konsole ist das Herzstück von R: Hinter dem Pfeil unten links kann R Code eingegeben und durch das Drücken der Entertaste ausgeführt werden.

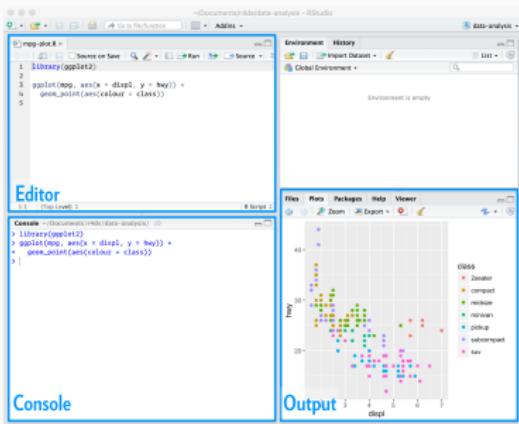
### Environment

Hier werden alle Objekte, die wir in R definieren, angezeigt.

### Files, Plots, Packages and Help

Dieses Fenster erfüllt mehrere Funktionen: Hier werden z.B. Ordner auf dem Computer, Graphen und installierte **packages** angezeigt. Auch Hilfeseiten sind hier zu finden.

**Ein viertes Fenster, der Editor, öffnet sich über “File > New File > R Script”**



## Editor

In R Skripten im Editor kann wichtiger Code gespeichert werden. R Skripte können später über *File > Save* in einer Datei mit der Erweiterung ".R" auf dem Computer gespeichert werden.

### Cave: Unterscheidung Editor und Konsole

In der Konsole wird R Code ausgeführt und im Editor lediglich gespeichert.

Wenn im Editor geschriebener Code ausgeführt werden soll, muss er dementsprechend der **Konsole übergeben werden**.

Dazu wird der relevante Code mit der Maus ausgewählt und entweder über den Button "**Run**" oben rechts im Editor oder die Tastenkombination Ctrl / Cmd + Enter ausgeführt.

# Evaluation Randomisiert-Kontrollierter Studien und Experimente mit R

## Packages

---

Prof. Dr. David Ebert & Mathias Harrer

Graduiertenseminar TUM-FGZ

Psychology & Digital Mental Health Care, Technische Universität München

## Packages in R

---

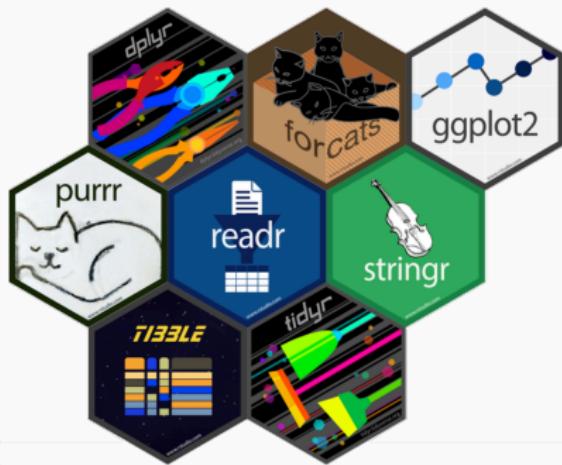
## Packages sind Sets von Funktionen:



- Packages stellen eine **Sammlung von R-Funktionen** dar, die für ein bestimmtes Problem oder eine bestimmte Anwendung nützlich sein können. Das package `{ggplot2}` stellt z.B. Funktionen zum Erstellen von Grafiken bereit.
- Packages können von **Experten** auf der ganzen Welt **entwickelt werden** und dann von **jedermann heruntergeladen** und in R genutzt werden.
- Allein das *Comprehensive R Archive Network* listet derzeit mehr als **16.000 installierbare Packages**.

- Manche Funktionen, Operatoren, Datensets und Plotting Tools sind bereits in R “vorinstalliert.” Viele dieser Funktionen existieren bereits seit der Entwicklung von R Anfang der 90er Jahre.
- Man spricht hierbei von **“Base R”**, das verwendet werden kann, ohne weitere Packages installieren zu müssen.
- Manche Packages, wie `{stats}` oder `{MASS}`, werden bei der Erstinstallation von R automatisch mitinstalliert, und gelten daher auch als “Base R.”

Das `{tidyverse}` ist kein einzelnes Package, sondern **bündelt eine ganze Reihe von Packages**. Ziel dieser Packages ist das **Erleichtern der Manipulation und Visualisierung** von Daten in R.



[www.tidyverse.org](http://www.tidyverse.org) (Wickham et al., 2019)

tidyverse stellt u.A. folgende Packages zur Verfügung:

## `{ggplot2}`

- Nutzung zur Erstellung von Grafiken

## `{dplyr}`

- Nutzung zur Manipulation von Daten

## `{purrr}`

- erleichtert "Functional Programming" in R
- erleichtert Arbeit mit Funktionen und Vektoren

Folgende Funktion muss mit dem Namen des package als Argument in die Konsole eingetragen und mit der Taste Enter ausgeführt werden:

```
install.packages("package")
```

## Info

Installierte Packages werden der *system library* hinzugefügt und unter *Packages* (in RStudio im rechten unteren Fenster zu finden) aufgelistet.

Um ein installiertes Package zu nutzen, muss es aus der *system library* geladen werden:

```
library(package)
```

## Cave

Packages müssen nur einmal installiert, aber jedes Mal neu geladen werden, wenn R neu gestartet wird.

# Praxis-Teil

## Installation des {tidyverse}



- ✓ **Der Befehl `install.packages()` wird normalerweise nicht in R Skripte mit aufgenommen:** wenn Skripte mit anderen geteilt werden, gilt es als "unziemlich", Settings auf Computern von anderen zu ändern.
- ✓ Zur besseren Übersicht wird empfohlen, **alle Packages**, die zur Ausführung des R Skripts geladen werden müssen, **an den Anfang des Skripts** zu schreiben (sogenannte "Dependencies").
- ✓ Manche Funktionen werden in verschiedenen Packages unterschiedlich genutzt. Mit folgender Schreibweise kann **genau spezifiziert werden, aus welchem Package die Funktion genommen werden soll:**  
`package::function()`.  
→ Beispiel: `ggplot2::ggplot()` bedeutet, dass die `ggplot` Funktion aus dem `{ggplot2}` Package genutzt wird.

(Wickham & Gromlund, 2016, Kap. 3)

## Working Directories

---

Ein Ordner auf dem Computer, auf den R zugreifen kann, um dort gespeicherte Daten zu nutzen und später Ergebnisse zu speichern

## Festlegen eines Working Directory

1. Erstellen eines Ordners auf dem Computer, in dem alle Analysedaten gespeichert sind (in unserem Fall `rct-workshop`<sup>1</sup>).
2. Öffnen des erstellten Ordners in RStudio über *Files* im rechten unteren Fenster. Der Analysedatensatz sollte hier aufgeführt sein.
3. Klicken auf das Zahnrad im rechten unteren Fenster und anschließend auf *Set as working directory* im sich öffnenden Dropdown-Menu.
4. Der aktuell geöffnete Ordner ist nun als Working Directory festgelegt.

<sup>1</sup>[protectlab.org/workshop/rct-evaluation-in-r/vorbereitung/rct-workshop.zip](http://protectlab.org/workshop/rct-evaluation-in-r/vorbereitung/rct-workshop.zip)

# Praxis-Teil

Das Working Directory festlegen



## Datenimport

---

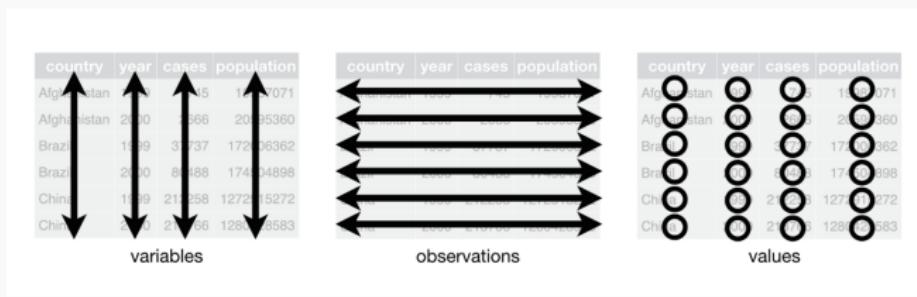
## Randomisiert-Kontrollierte Evaluation von "Get.On Fit im Stress" (adaptiert)

- ✓ **Stichprobe:** 264 erwachsene ArbeitnehmerInnen mit erhöhtem Stresslevel
- ✓ **Randomisierung:** Interventions- und Kontrollgruppe (Warteliste).
- ✓ **Intervention:** Internetbasierte Stressmanagement-Intervention.
- ✓ **Primärer Endpunkt:** Perceived Stress Scale (PSS-10), Post.
- ✓ **Messzeitpunkte:** Baseline, Post (7 Wochen), Follow-up (6 Monate).
- ✓ **Kodierung der Messzeitpunkte:** Variablenendung .0 für Baseline, .1 für Post, .2 für Follow-up)

angelehnt an Heber et al. (2016).

## Der Übungsdatensatz folgt den allgemeinen Regeln:

1. Spalten sind die gemessenen Variablen.
2. Beobachtungen / ProbandInnen sind in Zeilen.
3. Individuelle Werte sind in den Zellen.



(Wickham & Gromelund, 2016, Kap. 12.1)

## Cave: Working Directory beachten

Um Daten in R zu importieren, muss zunächst ein Working Directory festgelegt werden.

## Möglichkeit A | Import per "Klick":

1. Anklicken der Datei mit den Analysedaten im rechten unteren Fenster von RStudio unter *Files*.
2. *Import Dataset* auswählen.
3. In der sich öffnenden Datenvorschau *Import* auswählen.

## Möglichkeit B | Import per Code:

1. Gegenfalls Installation und Laden des package `{openxlsx}` für Daten im `.xlsx`-Format.
2. Da die Datei mit den Analysedaten im working directory gespeichert ist, kann der Dateiname direkt als Argument in folgender Funktion genutzt werden. Der vorangehende Pfeil zeigt an, dass die Analysedaten im Objekt `data` in R gespeichert werden sollen.

```
data <- read.xlsx("Dateiname")
# bzw.
data <- read.csv("Dateiname")
```

# Praxis-Teil:

Datensatz in R Importieren



Fragen?

Anmerkungen?

Kommentare?



# Evaluation Randomisiert-Kontrollierter Studien und Experimente mit R

## R Basics: Funktionen, Objekte, Operatoren und Fehlermeldungen

---

Prof. Dr. David Ebert & Mathias Harrer

Graduiertenseminar TUM-FGZ

Psychology & Digital Mental Health Care, Technische Universität München

## Funktionen

---

**Unabhängig von spezifischen Funktionen, können mithilfe von R klassische Rechenaufgaben gelöst werden:**

Probieren wir es selbst:

$$(49*6)/7$$

$$1^3 + 5^3 + 3^3$$

Funktionen sind **Kernelemente von R**: Sie erlauben es, vordefinierte Operationen auszuführen. Es besteht eine Parallele zur **mathematischen Formulierung** einer Funktion  $f(x)$ ; z.B. für die Quadratwurzel:

$$f(x) = \sqrt{x}$$

In R wird eine Funktion definiert, indem erst der **Name der Funktion** und dahinter in Klammern ihre **Inputs** (sog. **Argumente**) aufgeschrieben werden.

Funktionsname(**Argument1** = Wert1, **Argument2** = Wert2, ...)

In R wird so aus obiger Formel für die Quadratwurzel:

```
sqrt(x = 4)
```

## Position Matching

Der Argumentname kann auch **weggelassen** werden, solange die **Reihenfolge** der Argumente eingehalten wird.

Beispiel: “`sqrt(x = 4)`” und “`sqrt(4)`” führen zum gleichen Ergebnis, da beides mal 4 als erstes Argument auftaucht.

1. Was ist die Quadratwurzel von 9? Dazu können die Funktionen `sqrt()` nutzen:

```
sqrt(9)
```

```
## [1] 3
```

2. Logarithmus `log()` aller Werte der Variable age im Datensatz:

```
log(data$age)
```

```
## [1] 3.850 3.737 3.951 3.135 3.828 ...
```

→ Statt eines konkreten Wertes wird die gesamte Variable in die Funktion eingespeist, indem die Variable age über das Dollarzeichen aus unserem Datensatz `data` ausgewählt wird.

### 3. Mittelwert mean( ) der Variable pss zum Post-Zeitpunkt:

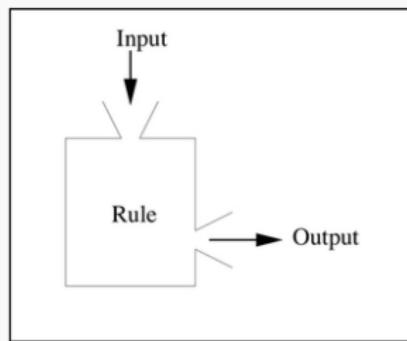
```
mean(data$pss.1)
## [1] NA
mean(data$pss.1, na.rm = TRUE)
[1] 20.423
```

In R kodiert NA, dass ein Wert fehlt.

Das Ergebnis der ersten Zeile Code ist “**not available**” (NA), da zum Post-Zeitpunkt **Beobachtungen fehlen** (d.h. NA sind) und der Mittelwert so nicht berechnet werden kann. Durch die Spezifikation des Arguments `na.rm` als TRUE, wird der Mittelwert nur über die **beobachteten Werte** gebildet und kann somit ausgegeben werden.

## Funktionen als "Herzstück" von R

Auch deutlich komplexere Funktionen in R funktionieren nach dem gleichen Prinzip: Man gibt die Parameterinformationen ein, die eine Funktion benötigt, und die Funktion nutzt diese Information, um ihre Berechnungen durchzuführen und schließlich das Ergebnis anzuzeigen.



## Die R Documentation

Viele Funktionen in R verlangen mehrere Argumente, und **niemand** kann die korrekte Nutzung aller Funktionen **auswendig lernen!**

- Die Lösung: Detaillierte Beschreibungen der Funktionen in der **R Documentation**.
- Die R Documentation kann entweder über **Help** im rechten unteren Fenster in Rstudio aufgerufen werden; oder direkt via Ausführen von `?funktionsname` in der Konsole; z.B. `?mean`.
- **Cave:** Die Dokumentation von Funktionen wird von den jeweiligen Package-Entwicklern selbst geschrieben. Sie ist daher nicht immer gleich informativ oder anfängerfreundlich.

Die R Documentation kann im Browser via [rdocumentation.org](http://rdocumentation.org) oder [rdrr.io](http://rdrr.io) eingesehen werden.

## Default Arguments

- Unter "Default Arguments" werden Argumente einer Funktion verstanden, deren Wert **vordefiniert ist und automatisch genutzt wird.**
- Default Arguments müssen beim Schreiben der Funktion also nur hinzugefügt werden, wenn sie **explizit von den Voreinstellungen abweichen.**
- Default-Werte einer Funktion können im Abschnitt "Usage" in **R Documentation** eingesehen werden

Siehe z.B. den Documentation-Eintrag für ?mean:

```
mean(x, trim = 0, na.rm = FALSE, ...)
```

Im Gegensatz zu “Object-Oriented Programming Languages” (z.B. Python, JS) konzentrieren sich **“Functional Programming Languages”** bei der Problemlösung auf **Funktionen**:

## Hauptmerkmale von Functional Programming Languages

- **First-Class Functions:** Vielseitige Einsetzbarkeit von Funktionen.
- **Pure Functions:** Output der Funktion hängt vom Input ab (d.h. Output reproduzierbar) und keine Nebeneffekte der Funktion (wie z.B. Veränderung des Wertes einer globalen Variable).

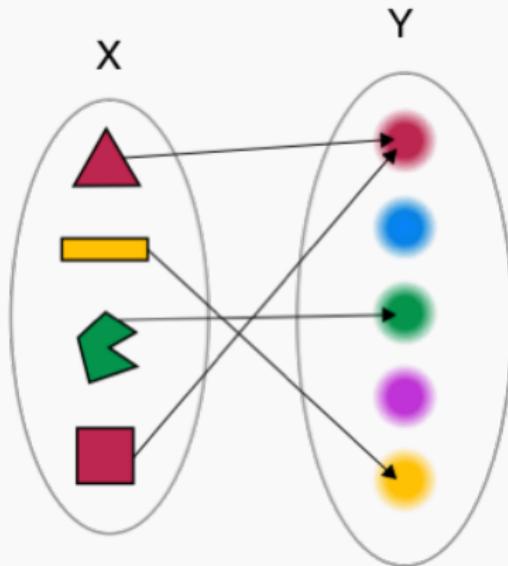
Auch wenn R diesen Kriterien nicht vollkommen entspricht, kann **R im Kern als Functional Programming Language definiert** werden.

Wickham (2019)

## Objekte

---

**Objekte** können als **Gegenspieler** von Funktionen verstanden werden: wir verwenden Funktionen, um Operationen an Objekten durchzuführen!



Um Objekte in R nutzen zu können, müssen wir diesen einen **Variablenamen** zuweisen. Dies ist möglich durch den **Zuweisungsoperator** `<-` (*assignment operator*).

```
geschlecht <- "Weiblich"
```

Eine Variablenname kann auch so zugewiesen werden:

```
"Weiblich" -> geschlecht  
geschlecht = "Weiblich"  
assign("geschlecht", "Weiblich")
```

Zur Inspektion des Objekts kann der Name des Objekts eingegeben werden:

```
geschlecht  
## [1] "Weiblich"
```

- Sobald Objekte einem Variablenamen zugewiesen worden sind, werden diese in RStudio im **Environment** rechts oben angezeigt.
- Dies bedeutet, dass das Objekt (temporär) in unserer Programmierumgebung **gespeichert** ist, und für weitere Operationen zur Verfügung steht.
- Existierende Objekte werden **überschrieben**, nicht vorhandene neu erzeugt.
- Mit der `rm` Funktion lassen sich Objekte aus dem Environment löschen, z.B. `rm(geschlecht)`.

## Cave: Benennung von Objekten

- Objektnamen müssen mit einem Buchstaben beginnen und können nur Buchstaben, Zahlen, Unterstriche und Punkte beinhalten.
- Konsistenz ist immer von Vorteil: z.B. `immer.namen.mit.punkten.trennen` oder `camelCaseVerwenden`.

(Wickham & Grolemund, 2016, Kap. 4.2)

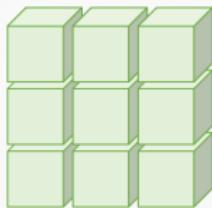
**vector** (numeric, character, factor, logical, ...)

$$\mathbf{a} = (a_1, a_2, a_3, \dots)$$

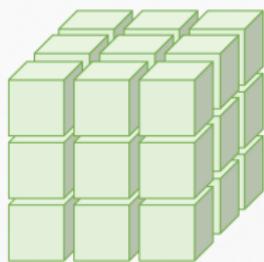


**matrix** (*m* rows, *n* columns)

$$A = \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} \end{bmatrix}$$



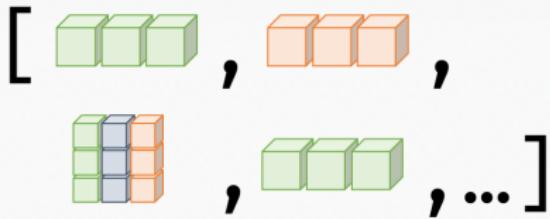
**array** (*m* columns, *n* layers, *l* rows)



**data.frame** (*m* rows, *n* columns)

7	"b"	TRUE
3	"f"	FALSE
9	"n"	TRUE

**list** (*n* elements)



## Vektoren

Vektoren (*vectors*) sind eine Sammlung von Werten (z.B. Zahlen, Wörter, Faktorstufen). Besteht ein Vektor nur aus einer Zahl, spricht man von einem **Skalar**.

Ein Vektor kann über die **concatenate**-Funktion `c` gebildet werden:

```
vector <- c(6, 9, 12, 18)
```

```
vector
```

```
#> [1] 6 9 12 18
```

Vektoren kommen in unterschiedlichen “Geschmacksrichtungen”:

- `numeric` oder `double`: in Zahlen gespeicherte Daten (z.B. Alter).
- `character`: in “Worten”/Buchstaben gespeicherte Daten.
- `logical`: binäre Variablen, die anzeigen, ob eine Bedingung TRUE oder FALSE ist.
- `factor`: in Zahlen gespeicherte Daten, wobei jede Zahl ein anderes Level einer Variable anzeigt (z.B. 1 = “wenig,” 2 = “mittel,” 3 = “hoch”).

Die Klasse eines Vektors kann mit der `class`-Funktion überprüft werden.

## Cave: Vektorklassen und weiterführende Analysen

Die Klasse eines Vektors hat Implikationen auf weitere Analyseschritte. Für `characters` kann z.B. kein Mittelwert berechnet werden.

**Alle Variablen des Datensatzes:** Funktion `glimpse` aus dem package `{tidyverse}`.

```
library(tidyverse)
glimpse(data)
## #> #> Rows: 264
## #> #> Columns: 34
## #> #> $ id              <chr> "stress_gui_002", "stress_gui_140", ~
## #> #> $ group           <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, ~
## #> #> $ sex              <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## #> #> ...
```

**Einzelne Variablen:** Anwendung der `class`-Funktion.

```
class(data$id)
## [1] "character"
```

## Dataframes

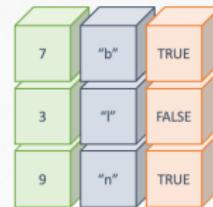
Dataframes (`data.frame`) sind die geläufigste **Struktur zur Sammlung von Daten** in R. Sie funktionieren wie einfache **Tabellen**: für jeden Zeileneintrag  $m$  gibt es Werte für  $n$  verschiedene Variablen.

Dataframes können aus **Vektoren zusammengestellt** werden. Im Gegensatz zur `matrix` können dabei unterschiedliche Vektorklassen (`numeric`, `logical`, `character`, ...) gebündelt werden.

```
name <- c("Lea", "Antonia", "Paula")
alter <- c(27, 22, NA)
weiblich <- c(TRUE, TRUE, TRUE)
data.frame(name, alter, weiblich)

##      name alter weiblich
## 1     Lea    27   TRUE
## 2 Antonia   22   TRUE
## 3    Paula   NA   TRUE
```

**data.frame** ( $m$  rows,  $n$  columns)



## Listen

Listen (lists) sind die **flexibelste** Datenstruktur in R. Sie erlauben es, jegliche Art von Objekt in einem “übergeordneten” Objekt zu sammeln (z.B. Dataframes, Vektoren, Arrays, Matrizen, einfache Werte, ...).

```
df <- data.frame(name, alter, weiblich) # siehe Folie zu data.frames  
df.beschreibung <- "Tabelle SHKs 2021"  
universitäten <- c("TUM", "FAU")  
list(df, df.beschreibung, universitäten)
```

```
## [[1]]  
##      name   alter  weiblich  
## 1     Lea     27    TRUE  
## 2 Antonia    22    TRUE  
## 3   Paula     NA    TRUE  
## [[2]]  
## [1] "Tabelle SHKs 2021"  
## [[3]]  
## [1] "TUM" "FAU"
```

list (n elements)  
[ , ,  
 , , ... ]

## Operatoren

---

✓ Einige Operatoren haben wir bereits kennengelernt:

- **Grundrechenarten:** +, -, \*, /
- **Potenz:** ^2, ^3, ^4, ...
- **Zuweisungsoperator:** <-, ->, =
- **“Pull”-Operator:** \$

→ Weitere Operatoren:

- **Vergleichsoperatoren:** >, >=, <, <=, != (nicht gleich), == (gleich)
- **Boole'sche Operatoren:** & (und), | (oder), ! (nicht)
- **Pipe-Operator:** %>%

(Wickham & Grolemund, 2016, Kap. 5.2)

Vergleichs- und Boole'sche Operatoren sind nützlich, um zu bestimmten, ob bestimmte Vektorelemente eine **Bedingung** erfüllen oder nicht.

```
"Variable" == "variable"  
# [1] FALSE  
  
x <- 10  
y <- 20  
x > 5 & y != 10  
## [1] TRUE  
  
data$cesd.0 > 16  
## [1] TRUE TRUE TRUE TRUE TRUE TRUE ...
```

Der **Pipe-Operator %>%** ist als einziger Operator nicht Teil von Base R<sup>1</sup>. Er ist erst verfügbar, sobald das `{tidyverse}` Package geladen wurde. Pipes haben **zwei große Vorteile**:

- Funktionen können auf ein Objekt angewandt werden, ohne dass das Objekt in der Funktion jeweils nochmal benannt werden muss.
- Mit Pipes können mehrere Funktionen **aneinandergekettet** werden.

```
library(tidyverse)
data %>% pull(pss.0) %>% mean() %>% sqrt()
## [1] 5.051627
```

## Die **pull**-Funktion

Die **pull**-Funktion ist das Äquivalent zum `$`-Operator innerhalb von Pipes. Die Funktion "zieht" eine Variable aus dem Datensatz und gibt sie weiter an die nächste Funktion.

---

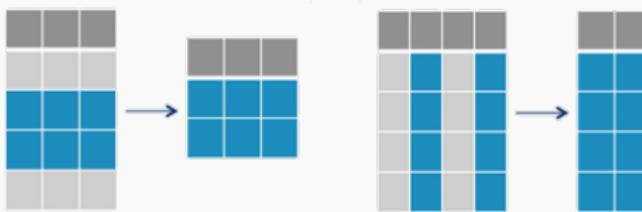
<sup>1</sup>Mit R Version 4.0.0 wurde nun auch ein Base-R Pipe-Operator eingeführt. Dieser benutzt jedoch "`|>`" als Symbol.

## **Indexing & Slicing**

---

Es gibt mehrere Wege, um in R Daten aus einem Dataframe zu extrahieren:

1. Mithilfe des **\$-Operators** oder `pull` (*schon besprochen*).
2. Über **eckige Klammern** `[ , ]`.
3. Über die Funktion `filter` bzw. `select` aus dem `{tidyverse}`.



### Slicing von Dataframes mit Eckigen Klammern

Subsetting von Dataframes mit eckigen Klammern ist etwas komplexer, erlaubt aber auch größere Flexibilität. Die generelle Form folgt der mathematischen Notation von Matrizen:

$$A[2,1] = A_{2,1} = \underbrace{\begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{bmatrix}}_{\text{Variablen}}$$

Die allgemeine Form zum Slicing ist also `data.frame[row, column]`.

## Slicing von Dataframes mit Eckigen Klammern

Um ein Subset der Daten auszuwählen, brauchen wir einen **Index**. Typischerweise ist dies eine **Zahl**, die die Zeilen- bzw. Spaltennummer(n) angibt.

```
data[3,15]  
## [1] 26
```

Es können auch mehrere Zeilen/Spalten ausgewählt werden:

```
data[1:3,c(15,17)]  
##   cesd.0 cesd.2  
## 1     18    16  
## 2     22    23  
## 3     26    27
```

## Slicing von Dataframes mit Eckigen Klammern

Wird ein Slot frei gelassen, wird die gesamte Zeile/Spalte ausgewählt:

```
data[,2]  
##      group  
## 1      0  
## 2      0  
## 3      0
```

Eine Indizierung ist auch mit dem **Variablennamen** möglich:

```
data[1,"pss.0"]  
## [1] 25
```

## Slicing von Dataframes mit Eckigen Klammern

Besonders hilfreich ist der Einsatz von `logicals` durch Vergleichsoperatoren. So kann z.B. der PSS-Wert aller Personen gefiltert werden, die älter als 40 sind:

```
data[data$age > 40, "pss.0"]  
##      pss.0  
## 1    25  
## 2    22  
## 3    25  
## [...]
```

Dies funktioniert, da der Boole'sche Ausdruck als Index fungiert:

```
data$age > 40  
## [1] TRUE TRUE TRUE FALSE TRUE ...
```

### Slicing von Dataframes mit filter und select

Die filter und select-Funktionen sind Teil des `{tidyverse}`. Sie erleichtern das Filtern und Selektieren von Dataframes, und sind besonders "Pipe-freundlich."

```
data %>%  
  filter(age > 40, sex == 0) %>%  
  select(pss.0, pss.1, pss.2) %>%  
  head(3)
```

```
#>     pss.0 pss.1 pss.2  
#> 1     25     15    21  
#> 2     22     18    24  
#> 3     25     NA    22
```

Die head-Funktion wird genutzt, um nur die ersten 3 Zeilen auszugeben.

## Fehlermeldungen

---

## Fehlerarten

- **Errors:** Eine Funktion kann nicht ausgeführt und muss gestoppt werden.
- **Warnings:** Es ist ein Fehler aufgetreten, aber die Funktion kann trotzdem (teilweise) ausgeführt werden.
- **Messages:** Information, dass eine Aktion für den Benutzer/ die Benutzerin ausgeführt wurde.

## Kein Grund zur Panik

- Fehlermeldungen in sind am Anfang meist sehr verwirrend, aber "normal".
- Im Laufe der Zeit werden Fehlermeldungen immer informativer und leichter zu entziffern.
- Googeln der Fehlermeldung ist hilfreich. Dazu sollte die Ausagesprache aber zuvor auf Englisch gestellt werden: `sys.setenv(LANG = "en")`.

(Wickham & Gromelund, 2016, Kap. 6.2)

## Im Skript in RStudio werden typische Fehler automatisch markiert

Syntaxfehler werden mit einem roten Kreuz und potentielle Probleme mit gelbem Ausrufezeichen am linken Rand markiert:

The screenshot shows two code snippets in RStudio. The first snippet has a red error icon at the start of line 4, indicating a syntax error. The second snippet has a yellow warning icon at the start of line 17, indicating a potential problem.

```
4 x y <= 10
5
unexpected token 'y'
unexpected token '<='
```

```
-->
! 17 3 == NA
1 use 'is.na' to check whether expression evaluates to
1 NA
20
```

(Wickham & Grolemund, 2016, Kap. 6.2)

# Praxis-Teil



**Fragen & Antworten:** [protectlab.org/workshop/rct-evaluation-in-r/r-entdecken/slicing/#uebung](http://protectlab.org/workshop/rct-evaluation-in-r/r-entdecken/slicing/#uebung)

1. Log-transformiere die Variable age in data und speichere das Ergebnis unter dem Namen age.log.
2. Quadriere die Werte in pss.1 und speichere das Ergebnis unter dem Namen pss.1.squared.
3. Berechne den Mittelwert und die Standardabweichung (*SD*) der Variable cesd.2. Nutze bei Bedarf das Internet um herauszufinden, welche Funktion in R die Standardabweichung berechnet.
4. Packe den Mittelwert und die Standardabweichung von cesd.2 in eine Liste.
5. Hat die Variable mbi.0 die passende Objektklasse numeric? Überprüfe dies mit R Code.
6. Lege im Dataframe data zwei neue Variablen an: (1) age.50plus, eine logical-Variable die mit TRUE und FALSE angibt, ob das Alter age einer Person  $\geq 50$  ist; (2) pss.diff, eine Variable die den Unterschied zwischen pss.0 und pss.1 für jede Person angibt.
7. Ändere den Wert von ft.helps in der dritten und vierten Zeile zu NA.
8. Mit der order Funktion kann für Variablen ein Index gebildet werden. Dieser Index zeigt an, in welcher Reihenfolge die Elemente korrekt geordnet wären. Nutze die R Documentation (?order), um mehr über die Funktion zu erfahren. Nutze dann diese Funktion in einer eckigen Klammer, um data dem Alter age nach zu ordnen!

# Fragen?

# Anmerkungen?

# Kommentare?



# Evaluation Randomisiert-Kontrollierter Studien und Experimente mit R

Randomisiert-Kontrollierte Studien: Aufbau & Hintergrund

---

Prof. Dr. David Ebert & Mathias Harrer

Graduiertenseminar TUM-FGZ

Psychology & Digital Mental Health Care, Technische Universität München

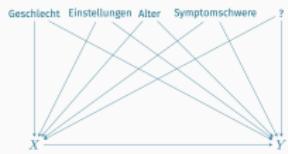
## Block I | R Entdecken

- ✓ Einführung in R & RStudio
- ✓ R Basics: Objektklassen, Funktionen, Operatoren, ...



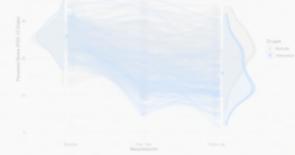
## Block II | Einführung & Hintergrund

- ✓ Methodischer Hintergrund von randomisierten Studien
- ✓ Import und Manipulation von Studiendaten mit R
- ✓ Schätzung fehlender Werte



## Block III | Analysieren & Verstehen

- ✓ Statistische Wirksamkeitsanalyse
- ✓ Alphakorrektur, Effektstärkenberechnung, Visualisierung, ...



## **Randomisiert-Kontrollierte Studien: Kernmerkmale**

---

*"Randomized controlled trials are the **most rigorous** way of determining whether a **cause-effect relation** exists between **treatment and outcome** and for assessing the **cost effectiveness** [sic] of a treatment."*

– Sibbald & Roland (1998)

Randomisiert-kontrollierte Studien werden häufig als "**Goldstandard**" zum Nachweis der Wirksamkeit einer Intervention angesehen.

(Akobeng, 2005; Backmann, 2017; Hariton & Locascio, 2018; Lilienfeld et al., 2018)

Dieser Status wird auch **teils hinterfragt und relativiert**.

(z.B. Cartwright, 2007, 2011; Grossman & Mackenzie, 2005; Kaptchuk, 2001; Pearl & Mackenzie, 2018, Kapitel 4).

### Kernmerkmale Randomisiert-Kontrollierter Studien

- **Zufällige Zuordnung** zu Experimental- und Kontrollbedingungen.
- Teilnehmende und Studienpersonal wissen nicht, wem welche Bedingung zugeordnet wurde (**"doppelte Verblindung"**). Dies ist oftmals aber nicht möglich!
- Alle Gruppen erhalten die gleiche Behandlung, bis auf die Experimentalgruppe
- Teilnehmende werden als Teil der Gruppe analysiert, zu der sie zugeordnet wurden; unabhängig davon, ob sie diese Behandlung tatsächlich (vollständig) erhalten haben ("once randomized, always analyzed"; **Intention-to-treat-Ansatz**)
- Die Analyse fokussiert auf die Schätzung der **Größe der Unterschiede zwischen den Gruppen** bezüglich eines vordefinierten Outcomes ab. Dieses Outcome ist zeitlich und hinsichtlich des genutzten Instruments genau definiert.

siehe Sibbald & Roland (1998).

## RCTs & Kausale Inferenz

---

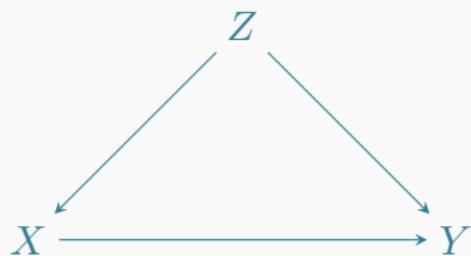
## Konfundierung

Ziel eines Experimentes ist es, **kausale Schussfolgerungen** ziehen zu können (z.B. “Behandlung  $X$  führt zu Outcome  $Y$ ”).

Das Experimentaldesign soll es erlauben, von beobachteten Zusammenhängen ausgehend ( $X \leftrightarrow Y$ ) Ursache-Wirkungs-Beziehungen abzuleiten ( $X \rightarrow Y$ ; “kausale Inferenz”).

Dieser Inferenzschluss ist gefährdet, wenn die Wirkungsbeziehung  $X \rightarrow Y$  durch Dritt faktoren  $Z$  **konfundiert** wird.

Kausalmechanismen können durch **Kausaldiagramme** (“directed acyclic graphs”; DAGs) formalisiert werden:



siehe Pearl (2009), Kapitel 1.2.1; Greenland et al. (1999).

## Potential Outcomes Model of Causation (Rubin, 1974)

Der **kausale Effekt einer Behandlung**  $\tau$  auf Person  $i$  ist die **Differenz** zum Messzeitpunkt  $t$  zwischen

- $Y$ , vorausgesetzt, dass  $i$  die Behandlung erhält ( $X_1$ ); und
- $Y$ , vorausgesetzt, dass  $i$  die Behandlung nicht erhält ( $X_0$ ):

$$\tau_{i,t} = Y_{i,t}(X_1) - Y_{i,t}(X_0)$$

**Problem:** Ein Individuum  $i$  kann nie gleichzeitig  $X_1$  und  $X_0$  erhalten;  $\tau_{i,t}$  basiert also gewissermaßen auf "**kontrafaktischen**" (counterfactual) Zuständen. Die Schätzung von  $\tau_{i,t}$  setzt daher die "**Austauschbarkeit**" (Exchangeability) von  $i$  voraus (Greenland & Robins, 1986, 2009).

→ Wie kann "Austauschbarkeit" erreicht werden, insbesondere wenn unbekannte Einflussfaktoren  $Z$  das Outcome  $Y$  ebenfalls beeinflussen?

## "The Skillful Interrogation of Nature": R. A. Fisher & die Erfindung des randomisierten Experiments (Lehmann, 2011, Kapitel 5)



aus Pearl & MacKenzie (2018), Kapitel 4.

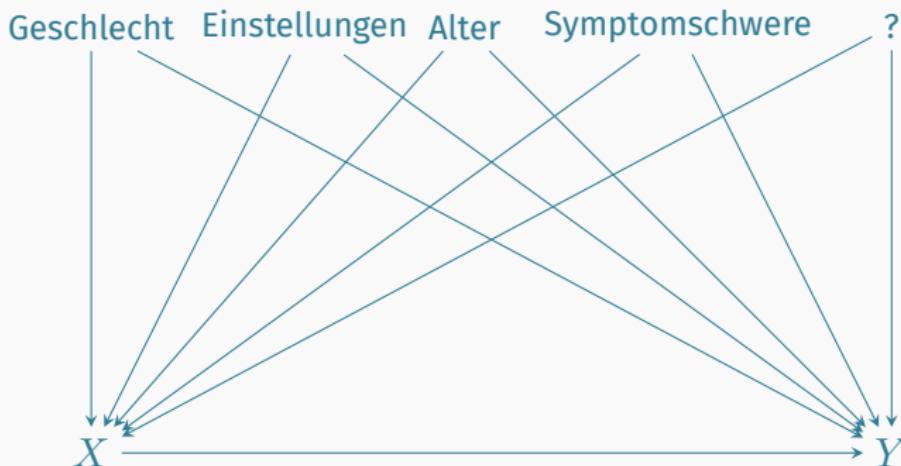
In den 1920er Jahren beginnt Fisher in der Rothamsted Forschungsstation mit randomisierten Experimenten zum Effekt von Düngemitteln.

Seine Einsicht: durch die Randomisierung wird der **systematische** Einfluss **sämtlicher** Konfundierungsvariablen ausgeschaltet; Unterschiede in den Gruppen kommen nur **zufällig** zustande.

## "The Skillful Interrogation of Nature": R. A. Fisher & die Erfindung des randomisierten Experiments (Lehmann, 2011, Kapitel 5)

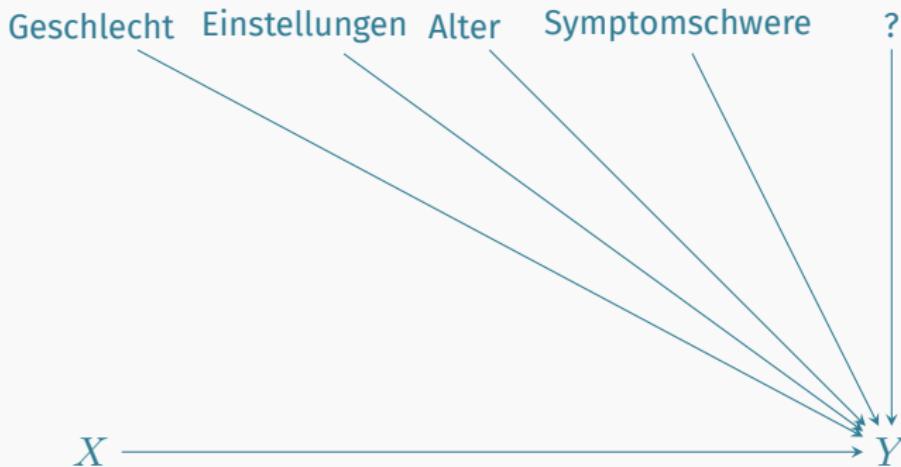
- Wird ein randomisiertes Experiment unendlich oft wiederholt, "**canceln**" sich **die Zufallseffekte gegenseitig aus**. Es entstehen dadurch zwei (hypothetische) Kollektive, die mit Blick auf alle (bekannten und unbekannten) Einflussfaktoren "**austauschbar**" sind.
- Die Randomisierung **ersetzt** zahllose **unbekannte und bekannte Einflussvariablen** durch eine **einige**: den unsystematischen Einfluss des **Zufalls**, der sich statistisch **quantifizieren** lässt (z.B. durch Konfidenzintervalle).

**"The Skillful Interrogation of Nature": R. A. Fisher & die Erfindung des randomisierten Experiments** (Lehmann, 2011, Kapitel 5)



Vor Randomisierung von  $X$ .

## "The Skillful Interrogation of Nature": R. A. Fisher & die Erfindung des randomisierten Experiments (Lehmann, 2011, Kapitel 5)



Nach Randomisierung von  $X$ : Confounders bleiben als **prognostische** Variablen erhalten.

# Praxis-Teil



## Anmerkung zu Randomisierungstests

- Zur Überprüfung, ob die Randomisierung "funktioniert" hat, werden immer noch häufig Randomisierungstests durchgeführt (z.B. ein *t*-Test zum Vergleich der Gruppen zu Baseline).
- Von solchen Tests wird in der methodischen Literatur weitgehend abgeraten (*"an unhealthy research behavior that is hard to eradicate"*).
- **Grund:** die Idee von Randomisierung ist nicht, jedes mal perfekt balancierte Gruppen zu erhalten, sondern über alle Randomisierungen hinweg → zufällige Gruppenunterschiede sind im Einzelfall selbst bei perfekter Randomisierung erwartbar!
- A priori für wichtig angesehene prognostische Variablen sollten ohnehin in der späteren Analyse kontrolliert werden.

(Austin et al., 2010; De Boer et al., 2015)

*“[S]ignificance testing of baseline differences in randomized controlled trials (RCTs) should not be performed, because it is **superfluous** and can **mislead investigators** and their readers.”*

– Moher et al. (2012), CONSORT 2010 statement

*“[T]he test actually assesses the probability (the ‘P value’) that the observed difference [...] could have occurred by chance when in reality there was no difference. [...] [P]erforming a significance test to compare baseline variables is to assess the probability of something having occurred by chance when we know that it did occur by chance. Such a procedure is **clearly absurd.**”*

– Altman (1985)

## **Studienhypthesen in RCTs**

---

## Konfirmatorische Hypothesen

**Superiority Trial (Einseitige Testung):** ist die Experimentalgruppe der Kontrollgruppe *überlegen*?

$$H_0 : \mu_1 \leq \mu_0 \Rightarrow \tau \leq 0$$

$$H_1 : \mu_1 > \mu_0 \Rightarrow \tau > 0$$

**Non-Inferiority Trial:** ist die Experimentalgruppe *nicht unakzeptabel schlechter* als die Kontrollgruppe?

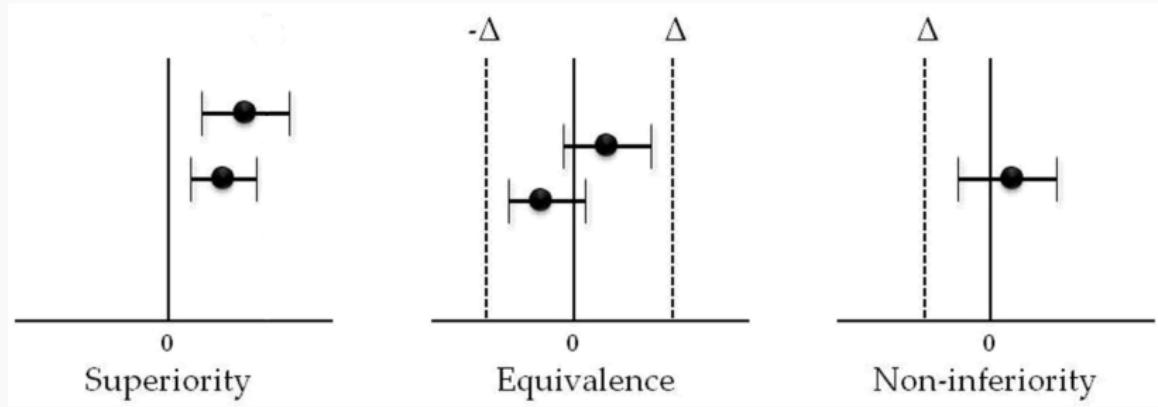
$$H_0 : \mu_1 < \mu_0 - \Delta \Rightarrow \tau < 0 - \Delta$$

$$H_1 : \mu_1 \geq \mu_0 - \Delta \Rightarrow \tau \geq 0 - \Delta$$

**Equivalence Trial:** ist die Experimentalgruppe mit der Kontrollgruppe *vergleichbar*?

$$H_0 : |\mu_1 - \mu_2| > \Delta \Rightarrow |\tau| > \Delta$$

$$H_1 : |\mu_1 - \mu_2| \leq \Delta \Rightarrow |\tau| \leq \Delta$$



adaptiert nach Schumi & Wittes (2011).

# Fragen?

# Anmerkungen?

# Kommentare?



# Evaluation Randomisiert-Kontrollierter Studien und Experimente mit R

Datenvorbereitung bei RCTs: Dos & Don'ts

---

Prof. Dr. David Ebert & Mathias Harrer

Graduiertenseminar TUM-FGZ

Psychology & Digital Mental Health Care, Technische Universität München

## **Vorbereitung der Daten: Dos & Don'ts**

---

Die Vorbereitung der Daten kann mühselig sein, ist aber das Grundgerüst für alle späteren Schritte.

→ Die Daten vor Import in R in das korrekte Format zu bringen, ist also lohnenswert.

## Dateiformat

Da der Import von *Microsoft Excel* in R sehr einfach ist, ist es empfehlenswert, Daten dort teilweise vorzubereiten. Excel Sheets können unkompliziert als *comma-separated values*-Datei (.csv) exportiert werden. Diese Dateiart kann besonders einfach und ohne großes Fehlerrisiko in R importiert werden (via `read.csv`).

## Dos and Dont's

- Eine **korrekte Benennung der Spalten** bereits in Excel spart oft viel Zeit, da keine Umbenennung mehr in R stattfinden muss.
- Die **Bezeichnung der Variablen** sollte **möglichst kurz sein** und einem **einheitlichen Schema** folgen.
- Für die Bezeichnung der **Zeitpunkte** empfiehlt sich eine **Zahlenkodierung**, die an das Ende des Variablenamens angehängt wird. (z.B. variable.0 für Baseline, variable.1 für Postmessung etc.)
- Spaltennamen sollten **keine Leerzeichen** beinhalten. Zur Trennung zweier Worte sollten Punkte oder camelCase genutzt werden.
- Auch **Sonderzeichen** in Variablennamen sollten vermieden werden, da sie beim Import ggf. verändert werden.

## Dos and Dont's

- Wenn der Name der Spalte in der **ersten Zeile der Spalte** eingetragen ist, wird R den Namen **automatisch als Spaltennamen** erkennen. Weitere Formatierung der Spalten ist dann nicht mehr notwendig.
- Die **Reihenfolge der Spalten** in der Excel-Datei ist **unerheblich**.
- Die Excel-Datei sollte nur **ein Tabellenblatt** haben.
- Sollte es mehrere **leere Zeilen und Spalten** geben, ist es ratsam, diese zu **löschen**. Beim Import könnten diese sonst als Missings interpretiert werden.
- Es sollte ein **Codebook** angelegt werden, in dem vermerkt ist, welcher Variable welches Kürzel zugeordnet ist und wie Antworten/Beobachtungen kodiert wurden.

siehe [Beispiel](#) für korrekt vorstrukturiertes Excel-Sheet.

# Evaluation Randomisiert-Kontrollierter Studien und Experimente mit R

Deskriptive Statistik und Analyse von Dropouts

---

Prof. Dr. David Ebert & Mathias Harrer

Graduiertenseminar TUM-FGZ

Psychology & Digital Mental Health Care, Technische Universität München

## **Deskriptive Statistik**

---

Es ist sinnvoll, vor der eigentlichen Analyse der Daten deskriptive Statistiken zu "eyeballen," um u.a. einen Eindruck von der Verteilung der Daten zu bekommen.

## Die `skim` Funktion aus dem package `{skimr}`

Die `skim`-Funktion bietet sich zur deskriptiven Statistik an, da sie einen guten Überblick über alle Variablen bietet:

- Zugehörigkeit zu Objektklassen (`character` oder `numeric`)
- Anzahl der Missings
- Bei `characters`: Anzahl uniquer Elemente
- Bei `numerics`: Mittelwert, Standardabweichung, Perzentile und Häufigkeitsverteilung

```
library(skimr)  
skim(data)
```

— Data Summary —————

	Values
Name	data
Number of rows	264
Number of columns	33

[...]

— Variable type: numeric —————

	skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
[...]											
12 child	0	1	0.530	0.500	0	0	1	1	1	1	
13 sess	132	0.5	5.70	2.32	0	5	7	7	7	7	
14 cesd.0	0	1	23.6	8.03	4	18	23	28	56	56	
15 cesd.1	22	0.917	18.5	9.72	1	12	17	24	49	49	
16 cesd.2	38	0.856	17.6	9.33	1	11	16	24	44	44	
17 hadsa.0	0	1	11.0	3.36	4	8	11	13	21	21	
[...]											

## Deskriptive Statistik separat für Interventions- und Kontrollgruppe:

```
library(skimr)
library(tidyverse)

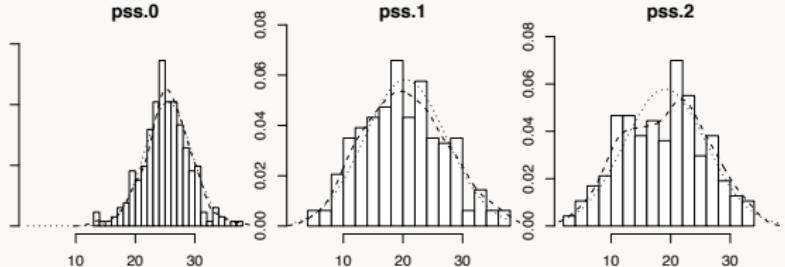
## Interventionsgruppe:
data %>%
  filter(group == 0) %>%
  skim()

## Kontrollgruppe:
data %>%
  filter(group == 1) %>%
  skim()
```

Es ist ratsam, sich die Häufigkeitsverteilung des primären Outcomes genauer anzusehen.

Mithilfe der Funktion `multi.hist` aus dem `{psych}` Package können mehrere Histogramme gleichzeitig erstellt werden:

```
library(psych)
library(tidyverse)
multi.hist(data %>% select(pss.0, pss.1, pss.2), ncol = 3)
```



## Analyse der Dropouts

---

Wie klassischerweise in RCTs zu erwarten, gibt es Missings (fehlende Werte) in unserem Datensatz zur Postmessung und zum Follow-up.

→ Empfehlung: Quantifizierung der Missings, gesamt und nach Gruppe

## Grundlagen

Wird eine Variable in die Funktion `is.na` eingespeist, gibt sie einen Vektor der Klasse `logical` aus, in dem alle Missings als `TRUE` klassifiziert werden. Über die Funktion `sum` werden alle als `TRUE` klassifizierten Werte gezählt, wodurch man die **Anzahl der Missings** erhält.

```
x <- c(1, 2, 3, 4, 5, NA, NA, 200)
sum(is.na(x))
## [1] 2
```

## Cave

Missings in unserem Datensatz fallen patientenweise an, d.h. wenn eine Person einen Fragebogen zu einem Zeitpunkt nicht ausgefüllt wurde, hat sie keinen Fragebögen zu diesem Zeitpunkt ausgefüllt. Deshalb genügt es, wenn wir uns auf die Dropoutanalyse des primary outcome beschränken.

## Missings in Interventions- und Kontrollgruppe:

```
with(data, {  
  c(sum(is.na(pss.0)),  
    sum(is.na(pss.1)),  
    sum(is.na(pss.2)))  
) -> na.all  
  
## Werte in Prozent umrechnen  
na.all.p <- na.all/nrow(data)
```

## Missings in der Interventionsgruppe:

```
data %>%  
  filter(group == 1) %>%  
  with({  
    c(sum(is.na(pss.0)),  
      sum(is.na(pss.1)),  
      sum(is.na(pss.2)))  
  }) -> na.ig  
  
## Werte in Prozent umrechnen  
na.ig.p <- na.ig/nrow(data) %>% filter(group == 1))
```

→ Analog für Missings in der Kontrollgruppe!

Alle Werte in einen dataframe binden:

```
na <- data.frame(na.all, na.all.p = na.all.p*100,  
                  na.ig, na.ig.p = na.ig.p*100,  
                  na.cg, na.cg.p = na.cg.p*100)  
  
## Zeilennamen des Dataframe ändern  
rownames(na) = c("t0", "t1", "t2")  
  
na  
##   na.all na.all.p na.ig na.ig.p na.cg na.cg.p  
## t0     0     0.00    0     0.00    0     0.00  
## t1    21     7.95   16    12.12    5     3.78  
## t2    28    10.60   17    12.87   11     8.33
```

→ Die Dropout-Rate ist zum Follow-up mit 10.6% am höchsten und es haben mehr Teilnehmende aus der IG als aus der KG die Studie vorzeitig abgebrochen.

# Evaluation Randomisiert-Kontrollierter Studien und Experimente mit R

Missing Data & Multiple Imputation

---

Prof. Dr. David Ebert & Mathias Harrer

Graduiertenseminar TUM-FGZ

Psychology & Digital Mental Health Care, Technische Universität München

## **Missing Data-Mechanismen**

---

*“Obviously the best way to treat missing data is not to have them.”*

Orchard & Woodbury (1972)

Fehlende Werte sind aber, insbesondere in der medizinischen und psychologischen Forschung, häufig **unvermeidbar!**

## Taxonomie: Gründe für fehlende Werte in klinischen Studien

- **Instrumente:** Response Burden, zu langes Assessment, ...
- **Teilnehmende:** Überforderung, Privatsphäre, Motivationsprobleme, ...
- **Center:** Inadäquate Umsetzung des Studienmethodik, Personalmangel, ...
- **Personal:** Falsche Datenerfassung, Datenverlust, falsche Dateneingabe, ...
- **Studie:** Technische Fehler, Zeitverzögerungen bei Follow-Ups, ...

Palmer et al. (2018)

**Der Umgang mit fehlenden Werten in der (medizinisch-psychologischen) Forschung ist oftmals mangelhaft:** (Akl et al., 2015; Bell et al., 2014; Van Buuren, 2018, Kapitel 1.1.2; Wood et al., 2004)

- Verteilung fehlender Werte nicht transparent berichtet
- Unpassendes missing data handling (z.B. **listwise deletion**)
- Adäquate Imputationsmethoden (z.B. MI, FIML) häufig nicht benutzt, inadäquat angewendet, oder unzureichend berichtet

## R macht es nicht (automatisch) "richtig!"

```
y <- 1:10
x <- c(1, NA, NA, NA, 3, 5, 8, 10, -1, 10)
summary(lm(y ~ x))

## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 4.9403    1.7738   2.785   0.0387 *
## x           0.3172    0.2709   1.171   0.2945
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.904 on 5 degrees of freedom
## (3 observations deleted due to missingness)
```

## Was dagegen tun?

- Umsetzung von **Reporting-Standards** z.B. STROBE (Von Elm et al., 2007) & CONSORT (Schulz et al., 2010), s.a. Sterne et al. (2009).
- Adäquates **Missing Data Handling**
  - Basierend auf plausiblen Annahmen, warum fehlende Werte entstanden sind (“Missing Data Mechanism”)
  - ggf. Schätzung (Imputation) fehlender Werte unter Einbezug von deren Unsicherheit

## **MCAR, MAR & MNAR**

---



Donald B. Rubin

(Rubin, 1976)

**Grundannahme:** Das Fehlen oder Vorhandensein von Daten ist Resultat eines probabilistischen Prozesses.

Diesem Prozess versucht man sich durch ein Modell (***missing data model***) anzunähern.



(Harrer et al., 2021)

Nach Rubin (1976) können Missing Data-Mechanismen in 3 Untertypen klassifiziert werden:

### MCAR

Missing Completely At Random: rein zufällig fehlende Werte einer Variable.

### MAR

Missing At Random: das Fehlen von Werten einer Variable ist abhängig von anderen (beobachteten) Variablen.

### MNAR

Missing Not At Random / "Nonignorable Missing Data": das Fehlen von Werten einer Variable ist (u.A.) abhängig von den Werten der Variable selbst.

→ Für jede Annahme ergeben sich unterschiedliche Auswirkungen bei der Datenauswertung!

**Notation** (Van Buuren, 2018, Kapitel 2.2.3 & 2.2.4)

- $\mathbf{Y}$ :  $n \times p$  Matrix mit teils fehlenden Werten ( $n$  Personen,  $p$  Variablen).
- $\mathbf{X}$ : Matrix mit (vollständig observierten) Kovariaten.
- $\mathcal{D} = (\mathbf{Y}, \mathbf{X})$ : Gesamter Datensatz.
- $\mathbf{R}$ :  $n \times p$  Matrix mit 0 (Datenpunkt fehlt) und 1 (Datenpunkt observiert; "response indicator").
- $\mathbf{Y}_{\text{obs}}$ ,  $\mathbf{Y}_{\text{mis}}$ : Observierte Daten, fehlende Daten.
- $\psi$ : Parameter des Missing Data-Modells (typischerweise nicht für die wiss. Fragestellung selbst relevant).

→ Missing Data-Modelle treffen Aussagen darüber, **in welcher Beziehung**  $\mathbf{Y}_{\text{obs}}$ ,  $\mathbf{Y}_{\text{mis}}$  und  $\mathbf{R}$  miteinander stehen.

## MCAR

$$P(\mathbf{R} = 0 | \mathbf{Y}_{\text{obs}}, \mathbf{Y}_{\text{mis}}, \psi) \Rightarrow P(\mathbf{R} = 0 | \psi)$$

Werte in  $\mathbf{Y}$  fehlen "zufällig" (unabhängig von Werten von  $\mathbf{Y}$ ). Das Fehlen von Werten ist nur bestimmt durch die allgemeine Wahrscheinlichkeit, dass Werte fehlen (im Datensatz gab es eher viele oder wenige Missings).

### Beispiel

Das Fehlen der Variable "Alter" ist weder von der Variable "Neurotizismus", noch vom Alter der Person selbst abhängig.

**MAR**

$$P(\mathbf{R} = 0 | \mathbf{Y}_{\text{obs}}, \mathbf{Y}_{\text{mis}}, \psi) \Rightarrow P(\mathbf{R} = 0 | \mathbf{Y}_{\text{obs}}, \psi)$$

Werte in  $\mathbf{Y}$  fehlen abhängig von  $\psi$  **und observierter** Information  $\mathbf{Y}_{\text{obs}}$ .

**Beispiel**

Personen mit höheren Neurotizismus-Werten geben ihr Alter seltener an als Personen mit niedrigem Neurotizismus, unabhängig ihres Alters.

## MNAR

$$P(\mathbf{R} = 0 | \mathbf{Y}_{\text{obs}}, \mathbf{Y}_{\text{mis}}, \psi) \Rightarrow ?$$

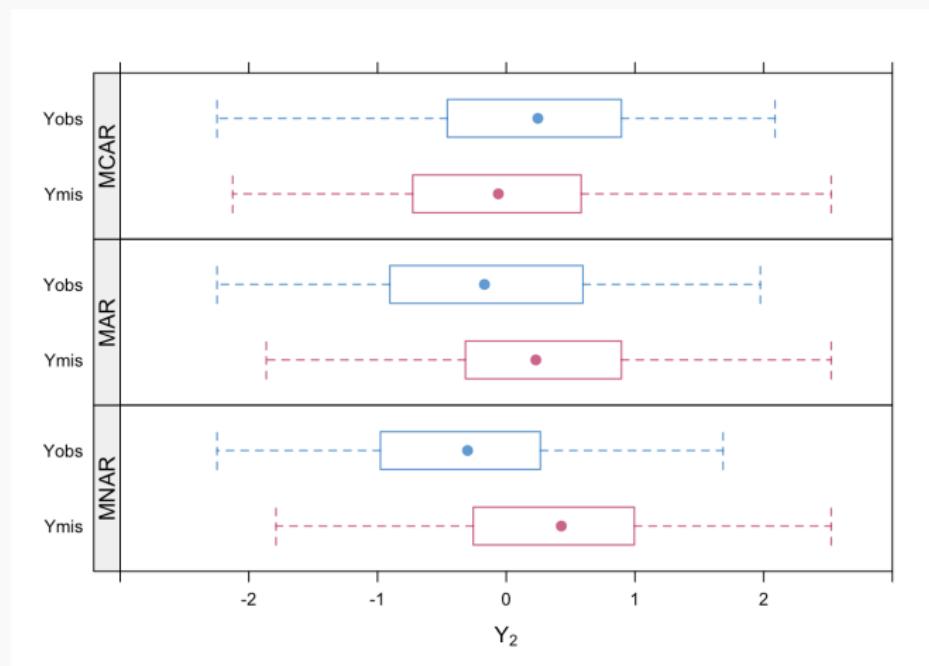
Die Formel lässt sich nicht "vereinfachen"!

Werte in  $\mathbf{Y}$  fehlen abhängig von  $\psi$ , **beobachteter Information**  $\mathbf{Y}_{\text{obs}}$ , **und** der **unbeobachteten Information**  $\mathbf{Y}_{\text{mis}}$  selbst.

### Beispiel

Das Fehlen der Alters-Variable hängt (u.A.) vom Alter der Person selbst ab. Eventuell geben ältere Personen mit hohem Neurotizismus ihr Alter seltener an. **Aber das wissen wir nicht**, da die Daten fehlen!

## Observierte und unobservierte Werte unter drei Missing Data-Annahmen



aus Van Buuren (2018), Kapitel 2.2.4

### Implikationen

- **MCAR:** Da die Werte zufällig fehlen, ist beim Einsatz passender statistischer Verfahren keine Verzerrung der Ergebnisse zu erwarten. Es besteht kein systematischer **Bias**; nur ein Verlust der statistischen Power durch den Datenverlust.
- **MAR:** Bei der Schätzung von Parametern ergeben sich evtl. Verzerrungen (Bias), wenn die abhängigen Variablen nicht im Modell berücksichtigt werden. Werden die abhängigen Variablen adäquat mit einbezogen, können Verzerrungen vermieden werden.

- **MNAR:** Bei der Schätzung von Parametern ergeben sich Verzerrungen.  
Da das Fehlen von Werten von den fehlenden Werten selbst abhängig ist, können nur “best guesses” hinsichtlich des **zugrundeliegenden Dropout-Mechanismus** getroffen werden. Ob diese korrekt sind, kann nicht empirisch überprüft werden.

Beim Vorliegen von MNAR spricht man von "**nonignorable missing data**" (Little & Rubin, 2019). Dies bedeutet, dass nicht von der gemeinsamen (Posterior-)Verteilung der **observierten Daten auf die der fehlenden Daten geschlossen werden kann**:

$$P(\mathbf{Y}|\mathbf{Y}_{\text{obs}}, \mathbf{R} = 1) \neq P(\mathbf{Y}|\mathbf{Y}_{\text{obs}}, \mathbf{R} = 0)$$

Das impliziert, dass eine Schätzung (Imputation) auf Basis der vorliegenden Werte nicht ohne weiteres möglich ist. Es müssen Annahmen getroffen werden, die "**über die Daten hinaus gehen**".

### Was bedeutet das für die Analyse von RCT-Daten?

#### MCAR

- Complete Case-Analysen und andere Ad Hoc-Verfahren führen nicht zu einer systematischen Verzerrung der Ergebnisse (aber durchaus zu einem Verlust statistischer Power/“Effizienz”).
- Die MCAR-Annahme ist typischerweise für RCT-Daten **nicht sehr plausibel**. (Bell et al., 2014; Mallinckrodt et al., 2004)
- Selbst wenn die MCAR-Annahme zutrifft, können Verfahren wie Multiple Imputation genutzt werden, z.B. um Konfidenzintervalle korrekter zu schätzen. (vgl. Pedersen et al., 2017)

### Was bedeutet das für die Analyse von RCT-Daten?

#### MAR

- Complete Case-Analysen führen zu einer Verzerrung der Ergebnisse.
- Verfahren wie Multiple Imputation, Full Information Maximum Likelihood (FIML) oder Mixed-Effect Models (eingeschränkt) können aber **genutzt werden, um die MAR-Annahme abzubilden**.
- Werden diese Modelle korrekt angewandt, vermeidet dies gebiaste Ergebnisse und führt zu einem korrekten Miteinbezug der Unsicherheit durch fehlende Werte (→ passende Konfidenzintervalle).

### Was bedeutet das für die Analyse von RCT-Daten?

#### MNAR

- Auch Verfahren wie Multiple Imputation (denen nur die observierten Daten zugrunde liegen) können zu **Verzerrungen in den Ergebnissen führen.**
- Methoden wie **Pattern-Mixture/Selektionmodelle** oder **referenzbasierte Imputation** können genutzt werden, um die Ergebnisse unter Annahme bestimmter Dropout-Mechanismen zu analysieren. (Little & Rubin, 2019, Kapitel 15; Carpenter et al., 2013; Heckman, 1976)
- Diese Annahmen sind jedoch nicht direkt empirisch nachweisbar.

*“MNAR models are [...] typically **highly dependent on untestable and often implicit assumptions** regarding the distribution of the unobserved measurements given the observed measurements.”*

—Molenberghs et al. (2004), S. 447

### Was soll ich für meinen Trial annehmen?

*“[W]e recommend that in trials [...], all data should be used in an analysis that makes a plausible assumption about missing data. Usually this will be a MAR assumption.”*

—Bell et al. (2014)

*“The assumption of ignorability is often sensible in practice, and generally provides a natural starting point.”*

—Van Buuren (2018), Kapitel 2.2.6

- Dropout & Fehlende Daten sind bei RCTs **kaum zu vermeiden**.
- Das Fehlen von Daten kann man sich als **“Produkt” eines wahrscheinlichkeitsbasierten Prozesses** vorstellen, der von einem (realen oder angenommenen) **Missing Data-Mechanismus** gesteuert wird.
- Derartige Mechanismen lassen sich in drei “Archetypen” zusammenfassen: **MCAR, MAR** und **MNAR**.
- Wird der zugrundeliegende Missing Data-Mechanismus nicht berücksichtigt, kann dies bei der Analyse zu **Verzerrungen** und **falschen Schlußfolgerungen** führen.



- Ob Daten MAR oder MNAR sind, kann **nicht anhand der Daten selbst bestimmt werden**; beide Annahmen sind immer nur mehr oder weniger plausibel.
- Bei RCTs ist die **Annahme von MAR häufig ein guter Startpunkt**; die Ergebnisse im Fall von nonignorable missing data (MNAR) können dann z.B. durch Sensitivitätsanalysen geprüft werden (Bell et al., 2014).
- Insbesondere **multiple Imputationsverfahren** sind ein gutes Mittel, Verzerrungen der Ergebnisse durch Dropout unter Annahme von MAR vorzubeugen.



## Kurze Anmerkung zu Little's MCAR Test

- Der  $\chi^2$  MCAR Test von Little findet immer noch häufig bei der Analyse von RCT Verwendung.
- Die Nullhypothese ist hierbei, dass Missings in den Daten zufällig auftreten ( $\rightarrow$  MCAR).
- Der **praktische Nutzen dieses Tests ist häufig begrenzt**: die Fähigkeit des Tests, die Nullhypothese (MCAR) zu verwerfen, hängt von der Größe des Datensets, und damit der statistischen Power ab. Bei großen Datensets können schon kleine Abweichungen zu einem Wert  $p < 0.05$  führen.
- Ein signifikanter MCAR-Test sagt nichts darüber, ob die Daten MAR oder MNAR sind!
- **Tipp:** stattdessen MAR als plausiblen "first start" annehmen, MNAR-Auswirkungen ggf. durch Sensitivitätsanalysen explorieren.

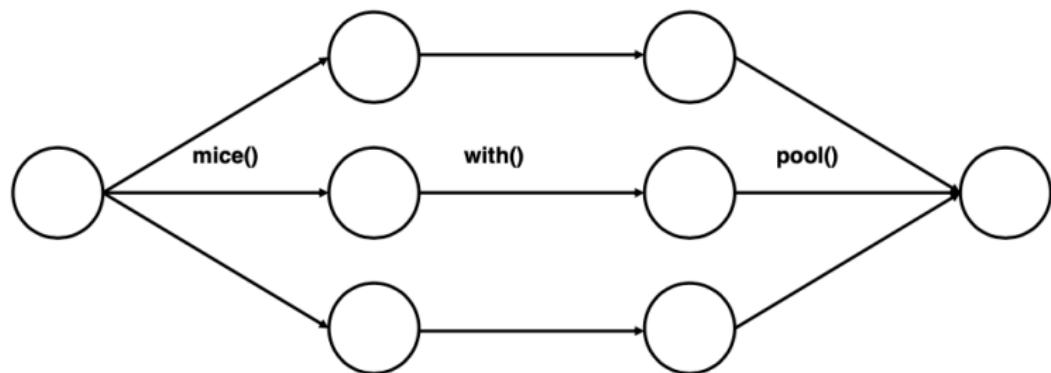
siehe Little (1988).

## **Multiple Imputation**

---

- Multiple Imputation (**MI**, Rubin, 1987) ist eine der flexibelsten und **gängigsten Methoden** zum Umgang mit fehlenden Werten.
- **Ziel von MI:** auf Basis der Verteilung der beobachteten Daten plausible Werte für die fehlenden Werte schätzen (“imputieren”).
- Um die Unsicherheit bei der Schätzung fehlender Werte abzubilden, werden **mehrere (“multiple”) Imputationen** für jeden fehlenden Datenpunkt erzeugt.
- Die so generierten vollständigen Datensätze werden dann **simultan analysiert** (z.B. Berechnung des Stichprobenmittelwerts) und Ergebnisse abschließend **gepools**t.

**incomplete data**      **imputed data**      **analysis results**      **pooled results**



aus Van Buuren & Groothuis-Oudshoorn (2011).

- MI kann unter **Annahme von MCAR und MAR** (sowie unter bestimmten Voraussetzungen auch MNAR) verwendet werden.
- Wenn korrekt angewandt, führt MI zur **verzerrungsfreien Schätzung von Populationsparametern** (Regressionsgewichte, Populationsmittelwerte, Korrelationen, etc.) sowie deren Varianz (“asymptotically unbiased”) – trotz des Vorliegens fehlender Werte.  
(White et al., 2011)

Es können 2 MI-Ansätze unterschieden werden:

- **Joint Modeling (JM):** Spezifikation einer multivariaten Verteilung (“joint model”) für die fehlenden Daten, auf deren Basis Imputationen mit Markov Chain Monte Carlo (MCMC) gesampelt werden (Schafer, 1997).
- **Fully Conditional Specification (FCS):** unvollständige Variablen werden in einem sequentiell-iterativen Prozess imputiert, eine Variable nach der Anderen.

Für FCS ist es im Gegensatz zu JM nicht notwendig, eine multivariate Verteilung der fehlenden Daten zu finden. FCS ist in dieser Hinsicht “atheoretisch” (vgl. White et al., 2011).

## Der MICE Algorithmus

Der '**MICE**' (*Multiple Imputation by Chained Equations*) Algorithmus ist eine der am häufigsten verwendeten und am besten erprobten FCS-Ansätze.

*"The MICE algorithm possesses a touch of magic."*

—Van Buuren & Groothuis-Oudshoorn (2011)

## Die Idee der “Chained Equations”

Annahme: unsere Daten  $\mathbf{Y} = Y_1, Y_2, \dots, Y_p$  (teils vorliegend, teils fehlend) sind das “Produkt” von  $\theta$ , einem (unbekannten) Vektor von Populationsparametern. Der multivariaten Verteilung von  $\theta$  wollen wir uns annähern, um korrekte Imputationen zu erzeugen.

Der MICE Algorithmus tut dies implizit, indem er iterativ für jede einzelne unvollständige Variable  $Y$  aus deren bedingter Verteilung sampelt, in folgender Form (Van Buuren & Groothuis-Oudshoorn, 2011):

$$P(Y_1 | \mathbf{Y}_{\setminus 1}, \theta_1)$$

⋮

$$P(Y_p | \mathbf{Y}_{\setminus p}, \theta_p)$$

Dies wird durch sogenanntes **Gibbs-Sampling** umgesetzt. Basierend auf Ausgangswerten wird dabei iterativ für jede Variable  $j$  jeweils zuerst  $\theta_j$  geschätzt, was wiederum direkt genutzt wird, um imputierte Werte  $Y_j^*$  zu erzeugen. Diese Werte bilden dann die Basis für das weitere Sampling.

Für eine beliebige Iteration  $t$  ergibt sich so:

$$\theta_1^{*t} \sim P(\theta_1 | Y_1^{\text{obs}}, Y_2^{t-1}, \dots, Y_p^{t-1})$$

$$Y_1^{*t} \sim P(Y_1 | Y_1^{\text{obs}}, Y_2^{t-1}, \dots, Y_p^{t-1}, \theta_1^{*t})$$

⋮

$$\theta_p^{*t} \sim P(\theta_p | Y_p^{\text{obs}}, Y_1^t, \dots, Y_{p-1}^t)$$

$$Y_p^{*t} \sim P(Y_p | Y_p^{\text{obs}}, Y_1^t, \dots, Y_{p-1}^t, \theta_p^{*t})$$

Im Normalfall **konvergiert** dieser Prozess nach einer bestimmten Anzahl Iterationen und erreicht Stationarität (“pendelt sich ein”).

Da in MICE die vorherige Imputation  $Y_j^{t-1}$  zur Imputation von  $Y_j$  nicht direkt einfließt, wird dies **relativ schnell erreicht** (oft schon nach 5-10 Iterationen).

Dieser iterative Prozess wird **parallel mehrfach durchgeführt**, um die  $m$  Imputationssets zu erzeugen.

Die genaue Technik, mit der die Imputationen erzeugt werden, kann bei MICE **flexibel für jede Variable festgelegt werden:**

Technik	Implementierung	Skala
Predictive Mean Matching	pmm	sämtliche
Bayesian Linear Regression	norm	kontinuierlich
Uncodition Mean Imputation	mean	kontinuierlich
Bayesian Logistic Regression	logreg	binär
Bayesian Polytomous Regression	polyreg	faktoriell

Typische Techniken zur Imputation mit MICE, nach Skalentyp.

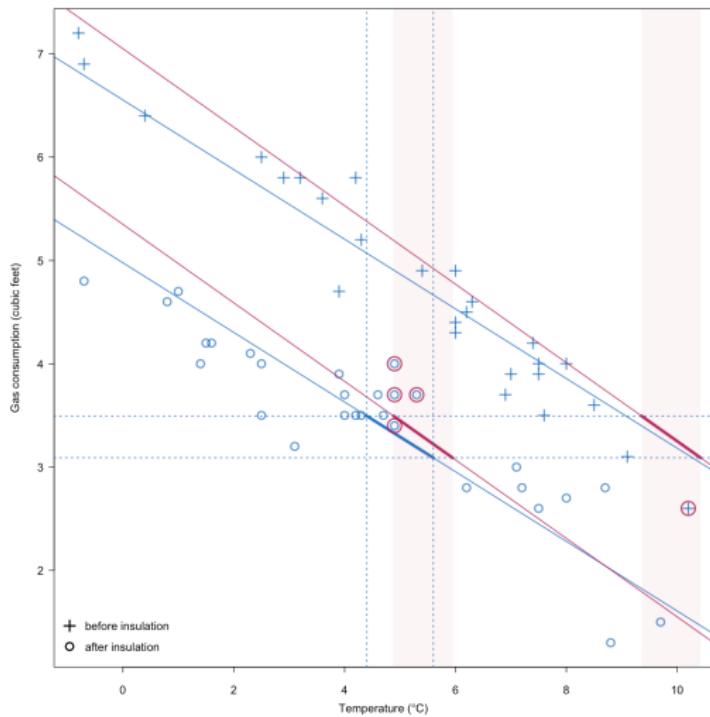
## Predictive Mean Matching (PMM)

- PMM (Morris et al., 2014) ist eine sogenannte “Hot Deck”-Methode.
- Dabei werden fehlende Werte anhand der eines **“Spenders”** imputiert.
- Der Spender wird dabei zufällig aus einer Anzahl von Kandidaten ausgewählt, die der Person mit fehlenden Werten **statistisch “ähnlich”** sind (d.h. ähnliche vorhergesagte Werte hat):

$$\arg \min_i \left| \hat{y}_i^{(\text{obs})} - \hat{y}_j^{(\text{mis})} \right|$$

- Eine Anzahl von  $d = 3\text{-}10$  Spenderkandidaten ist dabei häufig passend (Morris et al., 2014).

# Multiple Imputation (XI)



(Van Buuren, 2018, Kapitel 3.4.1)

### Predictive Mean Matching (PMM)

- PMM ist eine breit einsetzbare und typischerweise **robuste Technik** zur Imputation (Kleinke, 2017).
- Da alle Werte von beobachteten Spendern stammen, ist das Imputieren "**unmöglicher**" Werte (z.B. Alter von -2) **ausgeschlossen**.

## Multiple Imputation: Praktische Fragen

- **Anzahl der Imputationssets?**  $m = 20$  Sets sind häufig ausreichend bei moderater Menge an Missings. Mehr (z.B.  $m \approx 50-100$ ) sind notwendig, um schwer zu schätzende Parameter (z.B. Standardfehler, Varianzkomponenten) genau zu schätzen; oder wenn zahlreiche Missings vorliegen (van Buuren, 2018, Kapitel 2.8).
- **Anzahl der Iterationen?** 20-30 Iterationen häufig bereits ausreichend; dies kann erhöht werden, wenn Konvergenz in Frage steht.
- **Breite des Imputationsmodells?** "So breit wie möglich, so schmal wie nötig". Eine breite Auswahl von (sinnvollen) "Hilfsvariablen" macht die MAR-Annahme plausibler, kann aber zu Softwareproblemen führen ( $\rightarrow$  zu hohe Komplexität). Deshalb: "goldene Mitte" finden, Modellkomplexität reduzieren, ohne die Plausibilität des Modells zu sehr zu vermindern.

*"In general, one should try to **simplify the imputation structure without damaging it**; for example, omit variables that seem on exploratory investigation unlikely to be required in 'reasonable' analysis models, but avoid omitting variables that are in the analysis model or variables that clearly contribute towards satisfying the MAR assumption."*

— White et al. (2011)

# Praxis-Teil



Der **Influx- & Outflux-Koeffizient** kann genutzt werden, um potenziell (un-)relevante Prädiktoren für das Imputationsmodell zu identifizieren.

Bei gleicher Anzahl von Missings zeigen diese Koeffizienten an, welche Variablen besser mit dem Rest der (beobachteten) Daten “verbunden” sind.

Bei RCTs ist der heuristische Wert der Koeffizienten oft begrenzt, da Daten Zeitpunkt-abhängig fehlen (z.B. fehlen für Person  $i$  durch Dropout sämtliche Daten zum Postzeitpunkt).

(Van Buuren, 2018, Kapitel 4.1.3)

Der **Influx-Koeffizient**  $I_j$  einer Variable  $j$  gibt an, wie gut *fehlende* Werte in  $j$  durch *beobachtete* Werte in anderen Variablen abgedeckt werden.

Es sei  $R_{ij}$  der Responseindikator (beobachtet = 1; fehlend = 0) für Person  $i$  bei Variable  $j$ .  $R_{ik}$  sei der Indikator für selbige Person bei einer beliebigen anderen Variable  $k$ .

**Für den Influx-Koeffizienten von  $j$  ergibt sich so:**

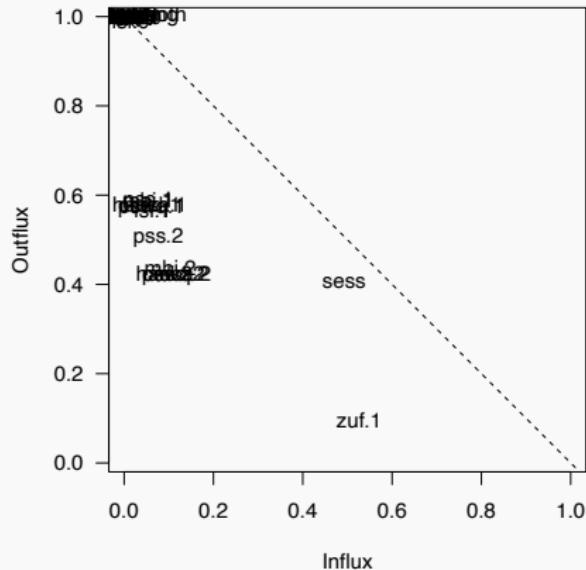
$$I_j = \frac{\sum_j^p \sum_k^p \sum_i^n (1 - R_{ij}) R_{ik}}{\sum_k^p \sum_i^n R_{ik}}$$

(Van Buuren, 2018, Kapitel 4.1.3)

Der **Outflux-Koeffizient**  $O_j$  einer Variable  $j$  gibt an, wie gut *beobachtete* Werte in  $j$  zur Abeckung *fehlender* Werte in anderen Variablen genutzt werden können.

Für Variable  $j$  ergibt sich so:

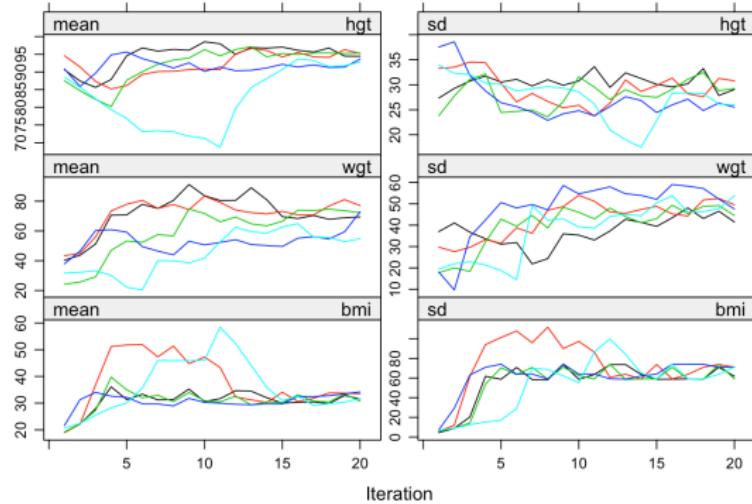
$$O_j = \frac{\sum_j^p \sum_k^p \sum_i^n R_{ij} (1 - R_{ik})}{\sum_k^p \sum_i^n 1 - R_{ij}}$$



Werte im **oberen Bereich** sowie nah an der Diagonalen sind eher hilfreich als Prädiktoren. Werte im **unteren linken Bereich** können eher entfernt werden (insofern inhaltlich nicht/wenig relevant).

**CAVE: Der Flux-Koeffizient sagt nichts über die tatsächliche Vorhersagekraft der Variable aus!**

**Beispiel für Nonkonvergenz:** Chains vermischen sich kaum & zeitlicher Trend ersichtlich!



(Van Buuren, 2018, Kapitel 6.5.2)

- Typischerweise sollte ein plausibles Imputationsmodell Werte  $\hat{Y}_{mis}$  erzeugen, die in ihrer Verteilung denen von  $Y_{obs}$  ähneln.
- Unter MAR sind jedoch auch **systematische Unterschiede** zwischen  $\hat{Y}_{mis}$  und  $Y_{obs}$  plausibel (z.B. Mittelwerts- oder Streuungsunterschiede)!
- Besonders **starke Divergenzen** sowie ausgesprochen unplausible/extreme Ergebnisse können jedoch auf Probleme des Imputationsmodells hinweisen.

# Evaluation Randomisiert-Kontrollierter Studien und Experimente mit R

Auswertung Multipel Imputierter Daten: Die Rubin-Regeln

---

Prof. Dr. David Ebert & Mathias Harrer

Graduiertenseminar TUM-FGZ

Psychology & Digital Mental Health Care, Technische Universität München

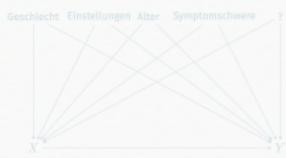
## Block I | R Entdecken

- ✓ Einführung in R & RStudio
- ✓ R Basics: Objektklassen, Funktionen, Operatoren, ...



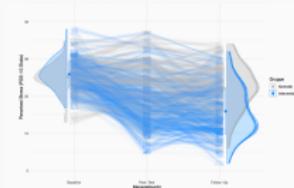
## Block II | Einführung & Hintergrund

- ✓ Methodischer Hintergrund von randomisierten Studien
- ✓ Import und Manipulation von Studiendaten mit R
- ✓ Schätzung fehlender Werte



## Block III | Analysieren & Verstehen

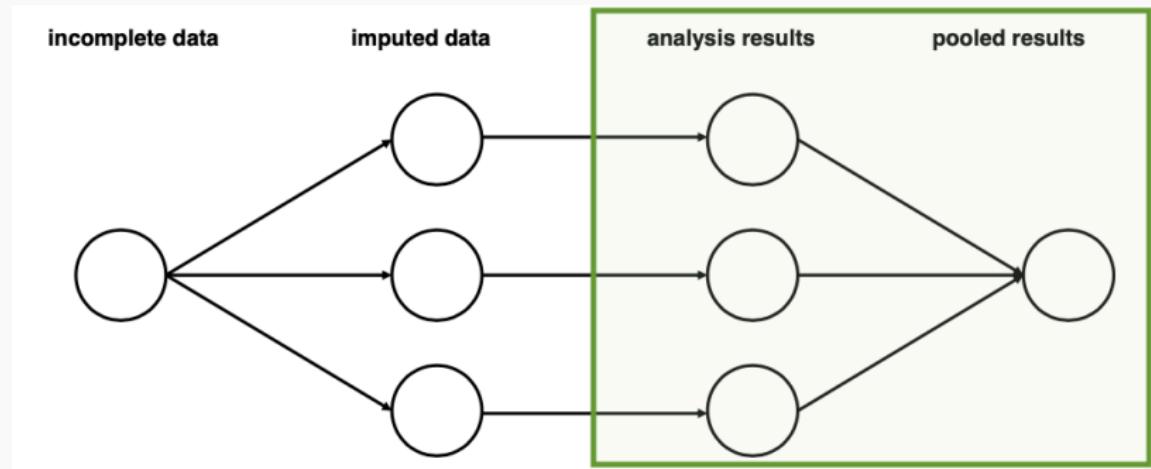
- ✓ Statistische Wirksamkeitsanalyse
- ✓ Alphakorrektur, Effektstärkenberechnung, Visualisierung, ...



## Die 3 Variationsquellen

---

Wie komme ich mit  $m$  Imputationssets zu einem gemeinsamen Analyseergebnis?



adaptiert von Van Buuren & Groothuis-Oudshoorn (2011).

→ **Lösung:** Durchführung der selben Analyse in allen  $m$  imputierten Datensets parallel, dann werden alle  $m$  relevanten Parameter/Schätzwerte zu einem Wert aggregiert (“**gepoolt**”).

Dabei müssen wir miteinbeziehen, dass unsere Schätzwerte **mit Unsicherheit behaftet** sind. Diese Unsicherheit hat bei MI mindestens zwei Gründe:

1. Die Teilnehmenden der Studie stellen nur eine Stichprobe der untersuchten Studienpopulation dar, sind also mit Stichprobenfehler (“**sampling error**”) behaftet.
2. Die Daten enthalten **fehlende Werte**, deren Schätzung **unsicher** ist. Diese Unsicherheit wird dadurch reflektiert, dass multipel imputierte Werte sich zwischen Imputationssets unterscheiden (können).

Es sei  $Q$  ein zu **schätzender wahrer Wert** (oder ein Vektor von Werten) der Population (z.B. Populationsmittelwert, Regressionskoeffizienten, ...).

Aufgrund der zuvor genannten Gründe ist  $Q$  **unbekannt** und muss durch einen Schätzer  $\hat{Q}$  **angenähert** werden. Dies ist nur durch die **beobachteten Werte**  $Y_{\text{obs}}$  möglich.

Der Erwartungswert (d.h. die **bestmögliche Annäherung**) von  $Q$  gegeben  $Y_{\text{obs}}$  ist (Van Buuren, 2018, Kapitel 2.3.2):

$$E(Q|Y_{\text{obs}}) = E\left(E(Q|Y_{\text{obs}}, Y_{\text{mis}}) \mid Y_{\text{obs}}\right)$$

d.h. der Durchschnittswert der (imputierten) Schätzungen des Mittelwerts von  $Q$  über alle multiplen Imputationen hinweg.

### Kombination von Punktschätzungen:

Es sei  $\hat{Q}_\ell$  die Schätzung von  $Q$  im  $\ell$ -ten von  $m$  Imputationssets. Der "gepoolte" Schätzwert von  $Q$  ist damit:

$$\bar{Q} = \frac{1}{m} \sum_{\ell=1}^m \hat{Q}_\ell$$

### Aggregation von Punktschätzungen bei MI

Um Punktschätzungen (z.B. Parameter der Wahrscheinlichkeitsverteilung wie Mittelwert & Standardabweichung; Regressionsgewichte, etc.) zu poolen, wird in jedem Imputationsset der Wert des Punktschätzers berechnet, und daraufhin der Mittelwert über alle Imputationsets gebildet.

Aber wie kann die **Unsicherheit** (Varianz) von  $Q$  in MI-Daten geschätzt werden? Die Varianz von  $Q$  gegeben  $Y_{\text{obs}}$  besteht aus **zwei Komponenten**:

$$V(Q|Y_{\text{obs}}) = \underbrace{E\left(V(Q|Y_{\text{obs}}, Y_{\text{mis}}) \mid Y_{\text{obs}}\right)}_{\substack{\text{Mittelwert d. Varianzen über alle MI-Sets} \\ \rightarrow \text{Within-Variance } (\bar{U})}} + \underbrace{V\left(E(Q|Y_{\text{obs}}, Y_{\text{mis}}) \mid Y_{\text{obs}}\right)}_{\substack{\text{Varianz d. Mittelwerte über alle MI-Sets} \\ \rightarrow \text{Between-Variance } (B)}}$$

## Bestimmung der gepoolten Varianz von Parametern bei MI

Um die Varianz eines Parameters  $Q$  zu bestimmen (z.B. für Konfidenzintervalle), muss bei MI sowohl die (gemittelte) Varianz durch den Stichprobenfehler  $\bar{U}$ , als auch die Imputationsunsicherheit  $B$  mit berücksichtigt werden. Die Berechnung der gepoolten Varianz erfolgt durch die sog. "**Rubin-Regeln**" (s. n. Folie).

### "Rubin's Rules" - Die Kombinationsregeln nach Rubin (Rubin, 1987)

Die Rubin-Regeln stellen eine allgemeine Formel dar, nach der die MI-Varianz eines Punktschätzers  $Q$  im konkreten Fall berechnet werden kann:

$$\begin{aligned}\hat{V} &= \overbrace{\left( \frac{1}{m} \sum_{\ell=1}^m \bar{U}_\ell \right)}^{\bar{U}} + \left( 1 + \frac{1}{m} \right) \overbrace{\left( \frac{1}{m-1} \sum_{\ell=1}^m (\hat{Q}_\ell - \bar{Q})(\hat{Q}_\ell - \bar{Q})' \right)}^B \\ &= \bar{U} + \left( 1 + \frac{1}{m} \right) B \\ &\Rightarrow \bar{U} + B \quad \text{as} \quad m \rightarrow \infty\end{aligned}$$

...aber warum *drei* Variationsquellen?



**YOU'RE MISSING THAT THIRD  
KIND OF HEAT.**

Die Rubin-Formeln beziehen auch mit ein, dass immer nur eine finite Anzahl an Imputationssets generiert werden ( $\rightarrow$  Einbezug der **Simulationsvarianz**).

$$\begin{aligned}\hat{V} &= \overbrace{\left( \frac{1}{m} \sum_{\ell=1}^m \bar{U}_\ell \right)}^{\bar{U}} + \left( 1 + \frac{1}{m} \right) \overbrace{\left( \frac{1}{m-1} \sum_{\ell=1}^m (\hat{Q}_\ell - \bar{Q})(\hat{Q}_\ell - \bar{Q})' \right)}^B \\ &= \bar{U} + \left( 1 + \frac{1}{m} \right) B \\ \Rightarrow \bar{U} + B &\quad \text{as } m \rightarrow \infty\end{aligned}$$

→ Je größer  $m$ , desto **geringer fällt diese Komponente ins Gewicht**.

Insbesondere, wenn besonders genaue Varianzschätzungen notwendig sind, empfiehlt sich daher eine **hohe Anzahl an Imputationssets**.

Ein niedriges  $m$  führt zu **Konfidenzintervallen**, die **etwas breiter** sind als wenn  $m \rightarrow \infty$  (niedrigere Effizienz). Dieser Unterschied ist in der Praxis jedoch typischerweise **überschaubar**.

## **Metriken & Freiheitsgrade bei MI-Analysen**

---

## Metriken für Parameterschätzungen in MI (Van Buuren, 2018, Kapitel 2.3.5)

**Relative Increase in Variance Due to Nonresponse** (RIV): Relativer Anstieg der Varianz aufgrund der Imputationsunsicherheit (wenn RIV > 1: Imputationsvarianz größer als “echte” Varianz in  $Y$ ):

$$r_Q = \frac{B_Q/m + B_Q}{\bar{U}_Q}$$

**Fraction of Missing Information Due to Nonresponse** (FMI): Anteil der Information über  $Q$ , die durch die Imputationsunsicherheit “verloren geht”:

$$\gamma_Q = \frac{(r_Q + 2)/(\nu_Q + 3)}{1 + r_Q}$$

Wobei  $\nu$  (“nu”) für die Freiheitsgrade bei der Schätzung von  $Q$  steht.

## Freiheitsgrade bei MI-Analysen (Van Buuren, 2018, Kapitel 2.3.6)

Die Freiheitsgrade eines Modells sind definiert als **Anzahl der Beobachtungen** nach Abzug der **Modellparameter**:  $\nu = n - k$  (für Mittelwert z.B.  $\nu = n - 1$ ).

Bei MI sind manche Elemente von  $n$  nur **Schätzungen von Beobachtungen**, daher muss  $\nu$  dafür **korrigiert** werden (Rubin, 1987):

$$\nu_{(MI)} = (m - 1) \left( 1 + \frac{1}{r^2} \right)$$

$$\lim_{r \rightarrow 0} \nu_{(MI)} = \infty \quad \text{sowie} \quad \lim_{r \rightarrow \infty} \nu_{(MI)} = m - 1$$

Diese Formel basiert auf der Annahme, dass die **Freiheitsgrade des vollständigen Datensatzes** (den MI zu schätzen versucht) **unendlich groß** sind! Diese Approximation ist aber erst sinnvoll, wenn ein **relativ großes Sample** vorliegt.

## Freiheitsgrade bei MI-Analysen (Van Buuren, 2018, Kapitel 2.3.6)

Für kleine Stichproben kann eine adaptierte Formel genutzt werden (Barnard & Rubin, 1999). Dafür müssen zuerst die **“hypothetischen” Freiheitsgrade** bestimmt werden, wenn die Daten komplett wären (“complete data degrees of freedom;  $\nu_{\text{com}}$ ). Es sei  $n$  die Anzahl der Beobachtungen und  $k$  die Anzahl der Parameter:

$$\nu_{\text{com}} = n - k$$

Daraus lassen sich die Freiheitsgrade der **beobachteten Werte** ( $\nu_{\text{obs}}$ ) bestimmen:

$$\nu_{\text{obs}} = \frac{\nu_{\text{com}} + 1}{\nu_{\text{com}} + 3} \nu_{\text{com}} \left(1 - \frac{r}{r + 1}\right)$$

Diese können wiederum zur **Korrektur** von  $\nu_{(\text{MI})}$  verwendet werden:

$$\nu_{(\text{MI})}^* = \frac{\nu_{(\text{MI})} \nu_{\text{obs}}}{\nu_{(\text{MI})} + \nu_{\text{obs}}}$$

## Freiheitsgrade bei MI-Analysen (Van Buuren, 2018, Kapitel 2.3.6)

### Praktische Anmerkungen

- Wird die unkorrigierte Formel verwendet (default bei `{mitml}`), können die **angezeigten Freiheitsgrade** eines Modells/Parameters **größer als  $n$  sein!**
- Da die Anzahl der Freiheitsgrade bei MI anhand der obigen Formeln angenähert wird, muss  $\nu$  im konkreten Fall **keine natürliche Zahl** sein (z.B.  $\nu = 226.6559$ )!

## Konfidenzintervalle (Van Buuren, 2018, Kapitel 2.4.2)

Die berechneten Freiheitsgrade können zur Berechnung von **Konfidenzintervallen** genutzt werden. Zur Inferenz von Skalaren (=  $\bar{Q}$  ist ein einziger Wert) wird dabei häufig eine  $t$ -Verteilung angenommen.

$\bar{Q}$  sei ein aggregierter Parameter (z.B. ein Regressionsgewicht  $b$ ), und  $Q_0$  der Referenzwert der Nullhypothese (typischerweise 0):

$$\frac{\bar{Q} - Q_0}{\sqrt{\hat{V}}} \sim t_{\nu}$$

Zusammen mit den berechneten MI-Freiheitsgraden lässt sich so ein passendes 95% Konfidenzintervall berechnen, dass die Imputationsunsicherheit berücksichtigt:

$$\bar{Q} \pm t_{\nu_{(MI)}^{(*)}, 0.975} \times \sqrt{\hat{V}}$$

## **Wichtige Anmerkungen zur Analyse von MI-Daten**

---

## Cave I: Aggregation von Teststatistiken

- In der Praxis ist es häufig notwendig, Teststatistiken (z.B.  $\chi^2$  oder  $F$ -Werte) in **jedem Imputationsset zu berechnen, und daraufhin zu poolen.**
- Teststatistiken können aber typischerweise nicht einfach wie bei  $\bar{Q}$  durch das **arithmetische Mittel** gepoolt werden!
- Hintergrund dafür ist, dass die **Größe der Teststatistiken von Varianz & Freiheitsgraden** abhängig ist. Diese ist/sind **bei MI größer bzw. kleiner** aufgrund der **Imputationsunsicherheit**.
- Um Teststatistiken nicht zu **überschätzen**, müssen daher **besondere Formeln** zur Aggregation eingesetzt werden.
- Implementationen in R sind beispielsweise die `micombine.chisquare` und `micombine.F` function im `{miceadds}` package.

→ Methoden zur Aggregation von Test(statistiken) sind ein **aktives Forschungsfeld**, und weitere Implementierungen in R sind zu erwarten (s. z.B. Grund, Lüdtke, et al., 2021).

## Cave II: Aggregation von Korrelationen

- Der Wert einer Korrelation  $\rho$  kann die **Maximalwerte**  $[-1; 1]$  **nicht überschreiten**; dies bedeutet, dass die **Varianz** von  $\rho$  **eingeschränkt** wird, je weiter  $|\rho|$  gegen 1 geht.
- Dies führt dazu, dass für  $\rho$  **keine asymptotische Normalverteilung** angenommen werden kann.
- Dadurch kann zur Aggregation von Korrelationen **nicht einfach der Mittelwert** berechnet werden. Korrelationen sollten vorher einer **varianzstabilisierenden Transformation** unterzogen werden, der **Fisher z-Transformation**.
- In R können Korrelationen komfortabel mit der `micombine.cor` function im `{miceadds}` package aggregiert werden.

vgl. Marshall et al. (2010).

## Cave III: Analyse in aggregierten Daten

- In der Praxis findet man häufig das Vorgehen, **alle MI-Sets zu einem vollständigen Datensatz** zu aggregieren, und dann Analysen in diesem (einen) Datensatz durchzuführen.
- Dies erleichtert zwar die Auswertung mit gängiger Statistiksoftware enorm, ist aber **aus statistischer Sicht unbedingt zu vermeiden!**
- Durch die Aggregation zu einem Dataset wird "**vorgegaukelt**", dass **keine Imputationsunsicherheit existiert**; dies führt zu inkorrekten, weil antikonservativen  $p$ -Werten, Konfidenzintervallen, Teststatistiken, etc.
- Eine Analyse in aggregierten Daten sollte daher höchstens dann durchgeführt werden, wenn keinerlei statistische Tests oder Quantifizierung der Parameterunsicherheit angestrebt wird; dies ist jedoch in der Praxis selten der Fall.

vgl. Van Buuren (2018), Kapitel 5.1.2.

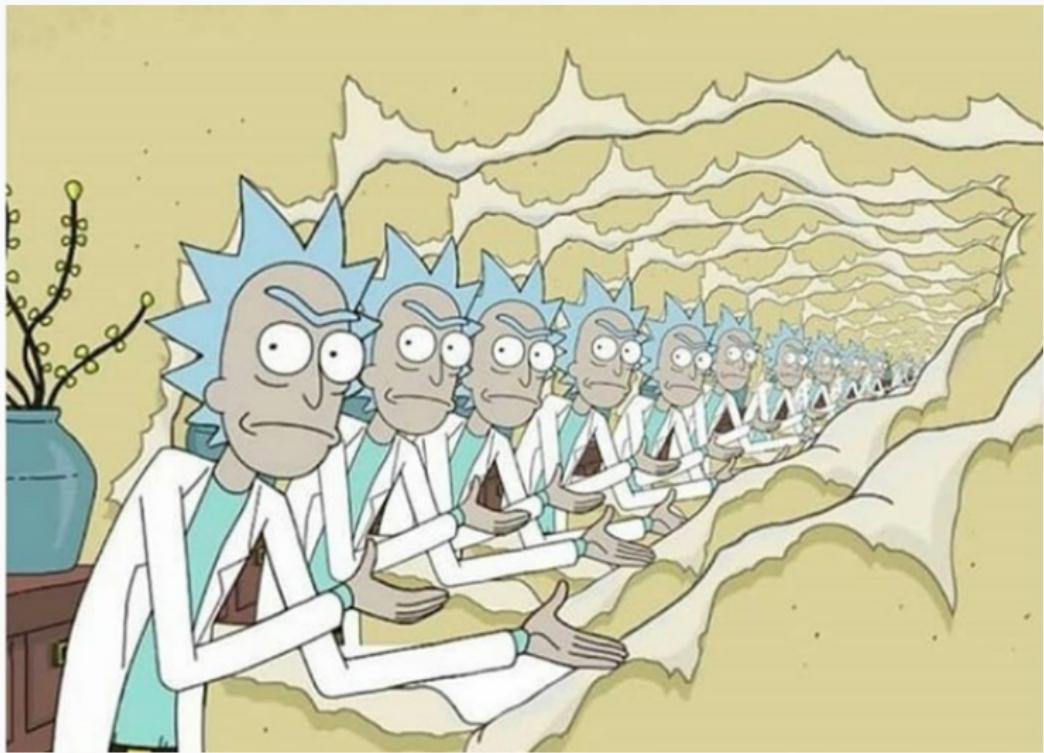
Fragen?

Anmerkungen?

Kommentare?



## “Rinse & Repeat”: Parallel Bearbeitung von Listenelementen



Bei der Auswertung von multipel imputierten Daten in R müssen häufig **Operationen in allen  $m$  Sets gleichzeitig durchgeführt** werden. Dies führt oft dazu, dass selbst einfache Analyseschritte **deutlich komplizierter** werden.

Eine mögliche Implementierung sind **for Loops**:

```
# Berechne aggregierten Mittelwert über alle Sets
means = vector()
for (i in 1:25){
  means[[i]] <- implist[[i]] %>% pull(pss.0) %>% mean()
}
mean <- mean(means)
```

→ **Nachteil:** trotz einfacher Operation komplexer Code, lange Rechenzeit.

## Functional Programming (Wickham, 2019, Kapitel 9)

Anstatt mit dem gleichen Befehl durch alle Sets zu loopen, kann stattdessen ein **“funktionaler” Programmierstil** gewählt werden. D.h. es wird eine **Funktion** genutzt, die **wiederum selbst Funktionen** auf alle Imputationssets **anwendet**.

In Base-R sind dies Funktionen wie `apply`, `mapply`, `vapply`, `Reduce`, etc.

Besonders benutzerfreundlich und konsistent sind aber die `map`-Funktionen im package `{purrr}` (Henry & Wickham, 2020):

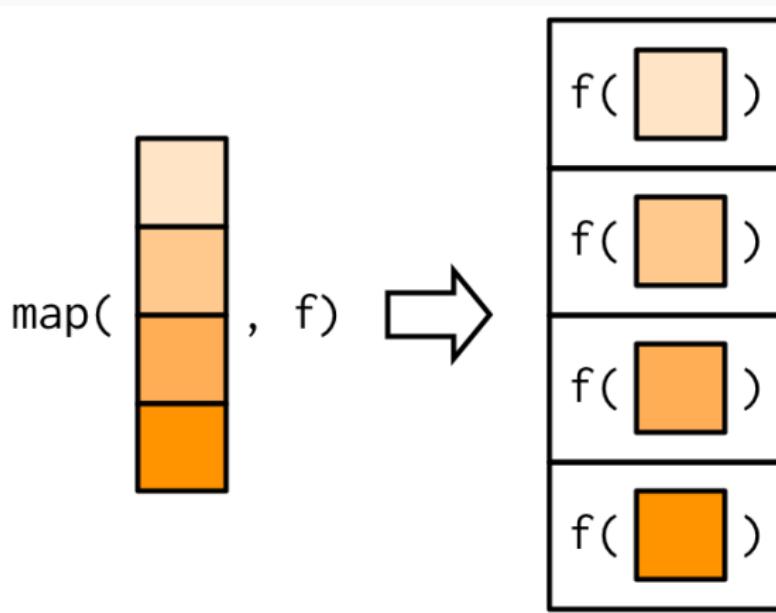
```
mean <- implist %>%  
  map_dbl(~mean(. $pss.0)) %>%  
  mean()
```

→ **Vorteil:** knapper, übersichtlicher Code; stark verkürzte Rechenzeit.



## Functional Programming (Wickham, 2019, Kapitel 9)

Die Funktionsweise von map:



## Functional Programming (Wickham, 2019, Kapitel 9)

### Die “Geschmacksrichtungen” von map:

- `map`: Output ist ein list-Objekt.
- `map_dbl`: Output ist ein numeric-Vektor.
- `map_chr`: Output ist ein character-Vektor.
- `map_lgl`: Output ist ein logical-Vektor.
- `map_dfr`: Output ist ein `data.frame`.
- `map2(_*)`: Iteration über zwei Listen gleichzeitig.



## Functional Programming (Wickham, 2019, Kapitel 9)

map akzeptiert Funktionen auf 2 Arten:

1. **“Klassisch”**: eine “voll funktionstüchtige” Funktion wird in map gesteckt.

```
# 'x' repräsentiert das individuelle Listenelement in 'list'  
list %>% map(function(x) mean(x$variable))
```

2. **“Verkürzt”**: mit “~” und “.” wird der Funktionscode abgekürzt.

```
# '.' repräsentiert das individuelle Listenelement in 'list'  
# Der Beginn einer Funktion wird durch '~' (tilde) angezeigt.  
list %>% map(~ mean(.variable))
```



Fragen?

Anmerkungen?

Kommentare?



# Praxis-Teil



## Primäre Wirksamkeitsanalyse

---

In der primären Wirksamkeitsanalyse wird die **Effektivität der Interventionsbedingung** evaluiert (“hatte die Interventionen einen Effekt?”).

Die Analyse fokussiert dabei auf den **primären Endpunkt** (mit *a priori* definiertem Messzeitpunkt und Messinstrument), und ob darin **Unterschiede zwischen den Gruppen** bestehen:

$$\begin{aligned}|Y_{i,t}(X_1) - Y_{i,t}(X_0)| &> 0 \\ \Rightarrow |\hat{\mu}_{1,t} - \hat{\mu}_{0,t}| &> 0\end{aligned}$$

Können wir dies statistisch Nachweisen, kann geschlossen werden, dass ein Effekt der Intervention vorliegt ( $|\tau| > 0$ ).

Das übliche Verfahren hierzu stellt die **Analysis of Covariance (ANCOVA)** dar.

## Analysis of Covariance (Montgomery, 1997, Kapitel 15.3; Dunn & Smyth, 2018, Kapitel 2.9)

ANCOVAs untersuchen, ob zwei oder mehrere Gruppen sich hinsichtlich einer gemessenen Variable  $y$  unterscheiden, wenn für den **Einfluss** von (einer oder mehreren) **Kovariaten** (z.B. Baseline-Messung des primären Endpunkt) kontrolliert wird.

Die Methode der AN(C)OVA geht auf R.A. Fisher zurück, und ist eng mit der experimentellen Methodik randomisierter Studien verbunden.

TABLE 38

	Degrees of Freedom.	Sum of Squares.	
Within classes . .	$n'(k - 1)$	$\sum_{i=1}^{n'} (x - \bar{x}_p)^2$	$n's^2(k - 1)(1 - r)$
Between classes . .	$n' - 1$	$k \sum_{i=1}^{n'} (\bar{x}_p - \bar{x})^2$	$n's^2[1 + (k - 1)r]$
Total . .	$n'k - 1$	$\sum_{i=1}^{n'} (x - \bar{x})^2$	$n's^2k$

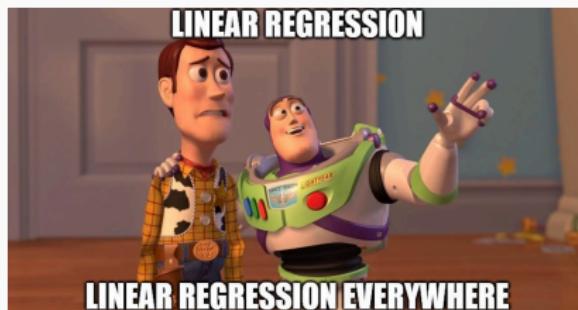
TABLE 39

	Degrees of Freedom.	Sum of Squares.	
Within classes	$n'(k - 1)$	$\sum_{i=1}^{n'} (x - \bar{x}_p)^2$	$n'(k - 1)B = n's^2(k - 1)(1 - r)$
Between classes	$n' - 1$	$k \sum_{i=1}^{n'} (\bar{x}_p - \bar{x})^2$	$(n' - 1)(k\Lambda + B) = (n' - 1)s^2[1 + (k - 1)r]$
Total	$n'k - 1$	$\sum_{i=1}^{n'} (x - \bar{x})^2$	$(n' - 1)k\Lambda + (n'k - 1)B = s^2[n'k - 1 - (k - 1)r]$

Statistical Methods for Research Workers (1925)

## Analysis of Covariance (Montgomery, 1997, Kapitel 15.3; Dunn & Smyth, 2018, Kapitel 2.9)

→ Die Durchführung von ANCOVAs in RCT-Analysen ist daher auch historisch zu erklären. Konzeptuell ist das Modell hinter ANCOVAs schlicht ein **Spezialfall linearer Regression!**



## Analysis of Covariance (Montgomery, 1997, Kapitel 15.3; Dunn & Smyth, 2018, Kapitel 2.9)

Es sei:

- $y_{ij}$  der Wert des (kontinuierlichen) primären Endpunkts von Person  $j$  in Gruppe  $i$ ;
- $\tau_i$  der Effekt der  $i$ -ten Behandlungsgruppe (z.B. Intervention oder Kontrolle);
- $i_j$  der Wert von  $ij$  auf einer Kovariate.

Das Modell der ANCOVA ist dann:

$$y_{ij} = \mu + \tau_i + \beta(x_{ij} - \bar{x}) + \epsilon_{ij} \quad \begin{cases} i = 1, 2, \dots, a \\ j = 1, 2, \dots, n \end{cases}$$

Die **Residuale** des Modells folgen einer Normalverteilung mit Mittelwert 0:  $\epsilon \stackrel{\text{i.i.d.}}{\sim} \mathcal{N}(0, \sigma^2)$ . ANCOVAs basieren auf einer **Effekt-Kodierung** der Treatment-Variable (z.B. -1 und 1), sodass sich die  $\tau_i$ -Werte zu null aufsummieren:

$$\sum_{i=1}^a \tau_i = 0$$

## Analysis of Covariance (Montgomery, 1997, Kapitel 15.3; Dunn & Smyth, 2018, Kapitel 2.9)

AN(C)OVAs basieren auf dem Prinzip der **Varianzzerlegung**:

$$SS_{\text{Total}} = SS_{\text{Prädiktor}} + SS_{\text{Residual}}$$

Daten = Modellfit + Unerklärte Varianz

Variationsquelle	Quadratsumme	Freiheitsgrade	Mittlere Quadratsumme
Systematisch	$SS_{\text{Prädiktor}}$	$\nu_{\text{num}} = p - 1$	$MSS_{\text{Prädiktor}} = \frac{SS_{\text{Prädiktor}}}{p-1}$
Zufällig	$SS_{\text{Residual}}$	$\nu_{\text{den}} = n - p$	$SS_{\text{Residual}} = \frac{SS_{\text{Residual}}}{n-p}$
Total	$SS_{\text{Total}}$	$n - 1$	-

→ Bei ANCOVAs wird der Einfluss der Kovariate auf die Within-Varianz ( $SS_{\text{Residual}}$ ) "herausgerechnet".

## Analysis of Covariance (Montgomery, 1997, Kapitel 15.3; Dunn & Smyth, 2018, Kapitel 2.9)

Die **Signifikanz des Treatment-Effekts** wird dann über den  $F$ -Test ermittelt. Dieser vergleicht die **Variation durch den Treatmentfaktor** mit der **unerklärten Variation** in den Daten:

$$F_{\nu_{\text{num}}, \nu_{\text{den}}} = \frac{\text{SS}_{\text{Prädiktor}}/(p - 1)}{\text{SS}_{\text{Residual}}/(n - p)}$$

### Warum adjustieren? Reduktion von $\text{SS}_{\text{Residual}}$ bei ANCOVAs

- Durch den Einschluss prognostischer Variablen wird die unerklärte Varianz innerhalb der Gruppen verringert. Dadurch verkleinert sich  $\text{SS}_{\text{Residual}}$  und der  $F$ -Wert wird größer  $\Rightarrow$  mehr Power zum Nachweis des Treatment-Effekts!
- **Cave:** Dies ist nur der Fall, wenn die Kovariate tatsächlich prognostisch relevant ist. Wenn nicht, erhöht sich nur die Anzahl der Parameter  $p$   $\Rightarrow$  **Power sinkt!**

## Analysis of Covariance (Montgomery, 1997, Kapitel 15.3; Dunn & Smyth, 2018, Kapitel 2.9)

### Vorteile der Adjustierung von Baseline-Variablen:

- **Interpretierbarkeit:** *"The central question is for two patients with the same pre measurement value of  $x$ , one given treatment A and the other treatment B, will the patients tend to have different post-treatment values? This is exactly what analysis of covariance assesses."* (Harrell, 2021)
- **Power:** Adjustierung von prognostischen Variablen führt zu höherer Effizienz der Analysen (engere Konfidenzintervalle); bei dichotomen Outcomes (Odds Ratio) erhöht sich der Effekt selbst (Hernández et al., 2004).
- **Baselineunterschiede:** Kovariaten adjustieren für systematische Baselineunterschiede, sollten diese tatsächlich vorliegen.

## Analysis of Covariance (Montgomery, 1997, Kapitel 15.3; Dunn & Smyth, 2018, Kapitel 2.9)

### Empfehlungen

- Generell sollten nur Kovariation adjustiert werden, für die ein **starker prognostischer Zusammenhang plausibel** ist → bei der Baselinemessung des primären Outcomes voraussetzbar!
- Bei stratifizierter Randomisierung sollten die **Stratifizierungsvariablen** kontrolliert werden (Kahan & Morris, 2012).
- Alle Kovariaten sollten ***a priori*** präspezifiziert werden (Assmann et al., 2000).

## Analysis of Covariance (Montgomery, 1997, Kapitel 15.3; Dunn & Smyth, 2018, Kapitel 2.9)

### Anzahl der Kovariaten: Richtlinien der European Medicines Agency

“No more than **a few covariates should be included** in the primary analysis. Even though methods of adjustment, such as analysis of covariance, can theoretically adjust for a large number of covariates it is safer to pre-specify a simple model.”

“Results based on such a model are more likely to be numerically stable, the assumptions underpinning the statistical model are easier to validate and generalisability of the results may be improved.”

EMA (2015), Absatz 6.2.



# Praxis-Teil



## ANCOVAs in R

---

ANCOVAs basieren auf einem **linearen Modell**. Daher kann zur Berechnung als erster Schritt die `lm`-Funktion genutzt werden:

Notwendige Argumente für `lm` sind:

- **formula:** Die Formel des linearen Modells, mit der Grundform " $y \sim 1 + x$ ." Dabei steht "1" für das Intercept (nicht zwingend notwendig).
  - Interaktionen werden mit "\*" gebildet, z.B. " $x1 * x2$ ".
  - Kontinuierliche Prädiktoren können mit `scale()` zentriert und skaliert werden, z.B. " $y \sim 1 + scale(x)$ ".
- **data:** Ein `data.frame` mit den zu verwendenden Daten.

→ Eine **Varianzanalyse** des Modells kann dann durch den `anova`-Befehl durchgeführt werden.

## Berechnung in multipel imputierten Daten

Mit Funktionen des **{mitml} package** (Grund, Robitzsch, et al., 2021) lassen sich (generalisierte) lineare Modelle unkompliziert in multipel imputierten Daten fitten.

Dazu müssen die Imputationen als eine Liste vom Typus “mitml.list” vorliegen. Diese Klasse kann durch die “as.mitml.list”-Funktion festgelegt werden.

Modelle können dann mit der `with`-Funktion **in allen MI-Sets gerechnet** werden, z.B.:

```
m <- with(implist, lm(y ~ 1 + trt + x))
```

Daraufhin kann die `testEstimates`-Funktion genutzt werden, um die **Modellparameter** mit den Rubin-Regeln zu **poolen**:

```
testEstimates(m)
```

Fragen?

Anmerkungen?

Kommentare?



## **Klinisch Signifikante & Reliable Veränderung**

---

- Methoden wie AN(C)OVAs evaluieren nur, ob das **Treatment einen signifikanten Einfluss** auf den primären Endpunkt hat.
- Dies sagt aber noch wenig darüber aus (1) wie **bedeutsam** dieser Einfluss ist, noch (2) was dieser Effekt auf **individueller Ebene bedeutet**.
- Aus diesem Grund wird bei RCTs häufig<sup>1</sup> auch die **klinische Signifikanz** der Ergebnisse beurteilt, sowie der Anteil der Teilnehmenden, bei denen eine **reliable Veränderung** stattgefunden hat.

---

<sup>1</sup>Derartige Analysen finden sich v.a. bei klinisch-psychologischen und biomedizinischen Trials.

**Klinisch Signifikante (CS) Veränderung** (Jacobson & Truax, 1992; Kroenke et al., 2001; Wise, 2004; Zahra & Hedge, 2010)

Eine klinisch signifikante Veränderung kann als "**Behandlungserfolg**" definiert werden. Dieser bemisst sich am Unterschreiten eines **klinisch oder praktisch relevanten Cut-offs**.

Dieser Cut-off kann je nach Instrument und Untersuchungsgegenstand **variieren** (z.B. Scores  $\leq 9$  und 50% Symptomreduktion beim PHQ-9; Kroenke, 2001).

Die zugrundeliegende Idee ist, dass Personen damit von einer "**dysfunktionalen**" in eine "**funktionale**" Population **übergewechselt** sind.

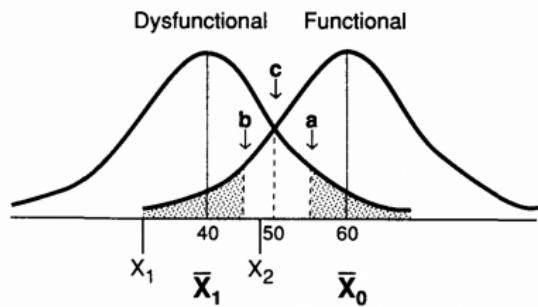


Figure 1. Pretest and posttest scores for a hypothetical subject ( $x$ ) with reference to three suggested cutoff points for clinically significant change (a, b, c).

aus Jacobson & Truax (1992)

**Reliable Veränderung (RC)** (Jacobson & Truax, 1992; Kroenke et al., 2001; Wise, 2004; Zahra & Hedge, 2010)

- Werden bei einer Person  $i$  zu Baseline und Post-Zeitpunkt **unterschiedliche Werte** für das primäre Outcome  $y$  gemessen (z.B.  $y_{1,i} > y_{0,i}$ ) ist unklar, ob dies eine **tatsächliche Verbesserung/Verschlechterung** darstellt!
- Die Veränderung könnte u.A. **zufälliges "Rauschen"** darstellen, oder durch den **Messfehler des Instruments** verursacht sein.
- Es muss daher ein **reliables** Veränderungsmaß berechnet werden. Dafür wird häufig der "**Reliable Change Index**" (Jacobson et al., 1984; Jacobson & Truax, 1992) herangezogen.

Es seien  $y_{1,i}$  und  $y_{0,i}$  die Werte des primären Outcomes einer Person  $i$  zum Post-test und zu Baseline. **Der Reliable Change Index (RCI) ist dann definiert als:**

$$\text{RCI} = \frac{y_{1,i} - y_{0,i}}{s_{\text{diff}}} \quad \text{mit} \quad s_{\text{diff}} = \sqrt{2(s_0 \sqrt{1 - r_{yy}})^2}$$

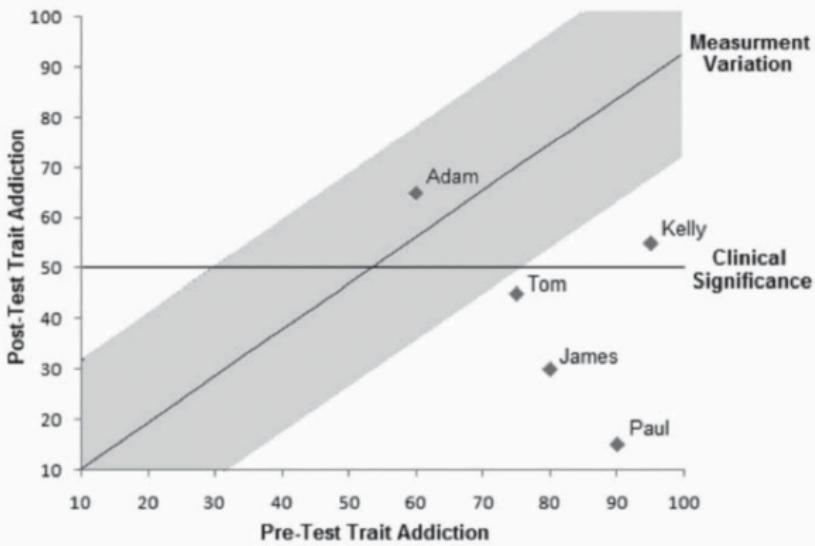
$s_0$  ist typischerweise die **Standardabweichung von  $y_0$** , und  $r_{yy}$  ist die **Test-Retest-Reliabilität** von  $y$ . Insgesamt repräsentiert  $s_{\text{diff}}$  die **erwartete Breite der Verteilung der Change Scores**, wenn **keine tatsächliche Veränderung** stattgefunden hat.

Für die berechneten RCIs wird eine Standardnormalverteilung angenommen:

$\text{RCI} \sim \mathcal{N}(0, 1)$ . Dadurch ergibt sich:

wenn 
$$\begin{cases} |\text{RCI}| \geq \Phi^{-1}(0.975) \Rightarrow |\text{RCI}| \geq 1.96 : \text{reliable Veränderung.} \\ |\text{RCI}| < \Phi^{-1}(0.975) \Rightarrow |\text{RCI}| < 1.96 : \text{keine reliable Veränderung.} \end{cases}$$

Liegt sowohl klinisch signifikante als auch reliable Verbesserung vor, spricht man von ***reliable and clinically significant improvement*** (RCSI).



aus Zahra & Hedge (2010).

## **Number Needed To Treat**

---

Die **Number Needed To Treat (NNT)** gibt an, wie viele **zusätzliche Personen** mit der untersuchten **Intervention behandelt** werden müssen, um ein gewünschtes Ereignis zu erreichen (Altman, 1998).

→ Ist z.B. NNT=3, dann müssen 3 Personen die Behandlung erhalten, um bei einer Person eine reliable Verbesserungen zu bewirken (oder um bei einer Person ein negatives Outcome zu verhindern; je nach Fragestellung).

Die NNT ist definiert als die **inverse absolute Risikoreduktion** (ARR, Harrer et al., 2021). Es sei  $e$  ein gewünschtes Ereignis, z.B. reliable Verbesserung:

$$\text{NNT} = (p_{e_{\text{treat}}} - p_{e_{\text{control}}})^{-1} = \text{ARR}^{-1}$$

$$p_{e_{\text{treat}}} = \frac{n_{e_{\text{treat}}}}{n_{\text{treat}}} \quad \text{und} \quad p_{e_{\text{control}}} = \frac{n_{e_{\text{control}}}}{n_{\text{control}}}.$$

Ist die NNT negativ, spricht man stattdessen von der **Number Needed to Harm (NNH)**.

# Praxis-Teil

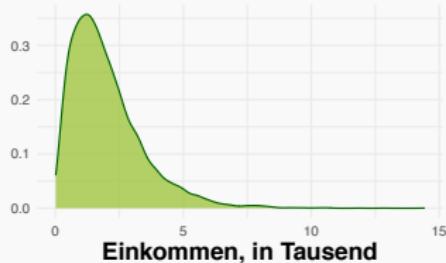


## **Generalisierte Lineare Modelle (GLM)**

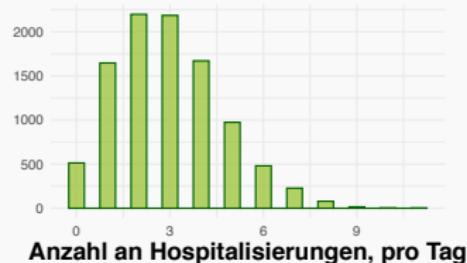
---

Was tun, wenn der primäre Endpunkt  $y$  nicht kontinuierlich & ungefähr normalverteilt ist?

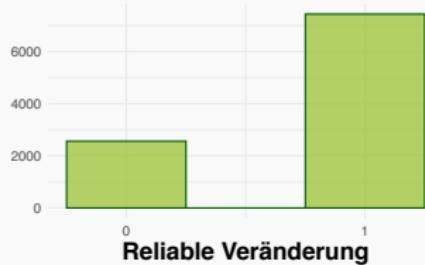
Gammaverteilung:  $y \sim \Gamma(\alpha = 2, \beta = 1)$



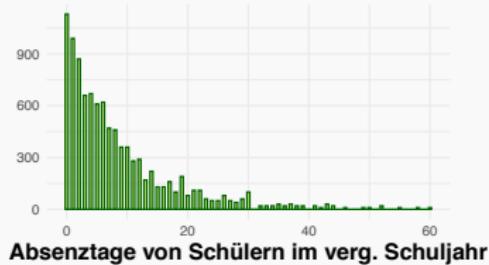
Poissonverteilung:  $y \sim \text{Pois}(\lambda = 3)$



Binomial-/Bernoulliverteilung:  $y \sim B(n = 1, p = 0.75)$



Negative Binomialverteilung:  $y \sim NB(r = 1, p = 0.1)$



## Die Struktur von GLMs (Dunn & Smyth, 2018, Kapitel 5)

- GLMs sind eine **Verallgemeinerung** linearer Modelle auf die gesamte Klasse der **Exponentialfamilie** von Verteilungen (“exponential dispersion model family”; EDM).
- Damit können auch Endpunkte wie **Zähl-/Eventdaten** oder **dichotome Outcomes** modelliert werden.
- Einfache lineare Modelle können direkt durch Matrixalgebra berechnet werden (**Ordinary Least Squares**; OLS). Bei (anderen) GLMs muss die **Optimierung iterativ** durch **Maximum-Likelihood-Schätzung** (MLE) vorgenommen werden.

## Die Struktur von GLMs (Dunn & Smyth, 2018, Kapitel 5)

GLMs bestehen aus zwei Komponenten: dem **linearen Prädiktor**  $\eta_i$  ...

$$\eta_i = (o_i) + \alpha + \mathbf{x}_i^\top \boldsymbol{\beta}$$

... und einer **Kopplungsfunktion** ("Linkfunktion"):

$$g(\mu_i) = \eta_i$$

Die inverse Kopplungsfunktion "**übersetzt**" dabei die Werte des **linearen Prädiktors** in das **Format der Responsevariable**:  $g^{-1}(\eta_i) = \mu_i$ ;  $\mu = E(y_i)$ .

In R können GLMs mit der `glm`-Funktion unter Angabe der Linkfunktion berechnet werden;  
z.B.

```
glm(y ~ trt + x, data = data, link = binomial("logit"))
```

## Die Struktur von GLMs (Dunn & Smyth, 2018, Kapitel 5)

$$\eta_i = (\textcolor{brown}{o}_i) + \alpha + \mathbf{x}_i^\top \beta$$

Der Term  $\textcolor{brown}{o}_i$  steht hier für einen **Offset**, der v.a. bei Poisson- und negativ binomialen Modellen verwendet wird. Offsets sind *a priori* bekannte Werte, für die **keine Parameterschätzung** notwendig ist; in der Biomedizin oft die Dauer der Exposition  $E$  bei der Analyse klinisch relevanter Events (z.B. Schlaganfälle während variierenden Zeitraums  $E$ ).

Für das Poisson-Modell ergibt sich so z.B.:

$$\begin{aligned} g(\mu) &= \overbrace{\log_e(\mu)}^{\text{Linkfunktion}} = \eta \\ g(\mu) &= \log_e\left(\frac{\mu}{E}\right) = \eta \quad (\text{mit Offset}) \\ &= \log_e(\mu) = \eta + \log_e(E) \end{aligned}$$

```
glm(y ~ trt + x + offset(E), data = data, link = poisson("log"))
```

## GLMs & Link-Funktionen: Übersicht

\* bekannt als "Logistische Regression."

Verteilung	Link-Typ	Linkfunktion	Support ( $y$ )	$\exp(\beta_{\text{treat}})$	family
Normal	Identity	$\mu$	$\mathbb{R}$	$\log_e(\text{MD})$	<code>gaussian(„identity“)</code>
Binomial	Logit*	$\log_e\left(\frac{\mu}{1-\mu}\right)$	$\frac{0,1,\dots,m}{m}$	OR	<code>binomial(„logit“)</code>
	Log	$\log_e(\mu)$	$\frac{0,1,\dots,m}{m}$	RR	<code>binomial(„log“)</code>
	Complem.	$\log_e[-\log_e(1-\mu)]$	$\frac{0,1,\dots,m}{m}$	HR	<code>binomial(„cloglog“)</code>
	Log-Log				
Negativ Binomial	Log	$\log_e\left(\frac{\mu}{\mu+k}\right)$	$\mathbb{N}_0$	(I)RR	<code>log (in MASS::glm.nb)</code>
Poisson	Log	$\log_e(\mu)$	$\mathbb{N}_0$	(I)RR	<code>poisson(„log“)</code>
Gamma	Inverse	$-\mu^{-1}$	$\mathbb{R}_+$	-	<code>gamma(„inverse“)</code>

Fragen?

Anmerkungen?

Kommentare?



# Praxis-Teil



# Evaluation Randomisiert-Kontrollierter Studien und Experimente mit R

Visualisierung der Studienergebnisse

---

Prof. Dr. David Ebert & Mathias Harrer

Graduiertenseminar TUM-FGZ

Psychology & Digital Mental Health Care, Technische Universität München

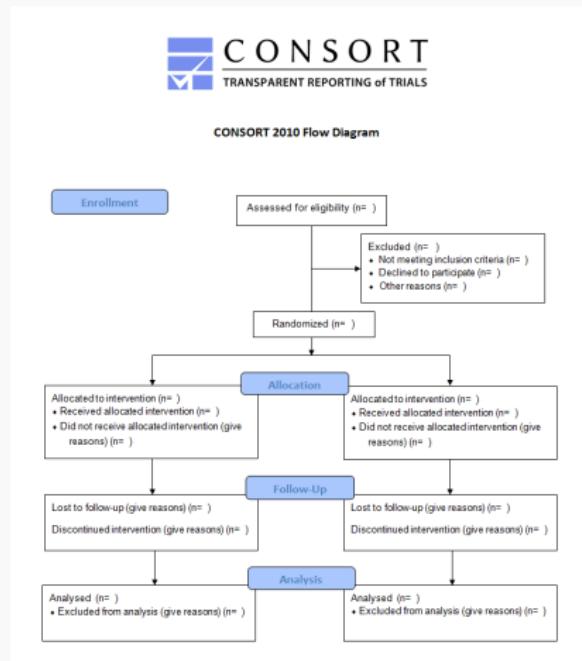
## **CONSORT Flow Chart**

---

Die Erstellung eines **Studien-Flowcharts**  
wird im CONSORT Statement ausdrücklich  
empfohlen (Item 13a).

Die maßgebliche Vorlage dazu kann als MS Word-Dokument heruntergeladen werden:  
[consort-statement.org/consort-statement/flow-diagram](http://consort-statement.org/consort-statement/flow-diagram).

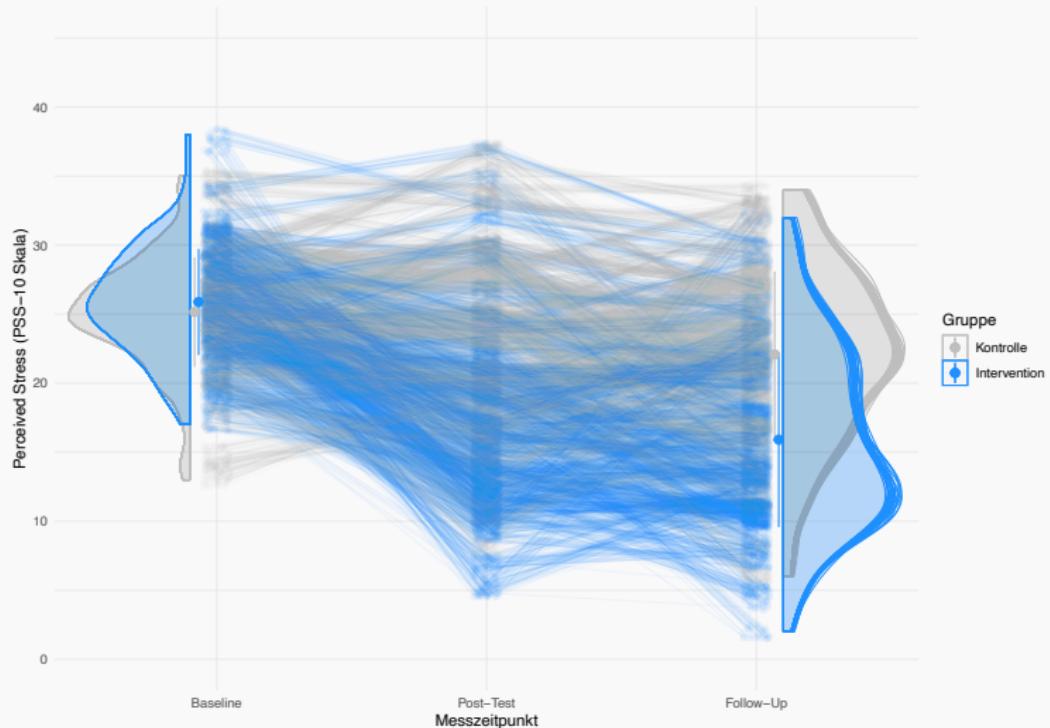
Eine Erstellung in R ist möglich (s. z.B. [hier](#),  
oder indirekt mit [TikZ](#)), aber oft vergleichsweise mühselig.



## “Raincloud” Plots

---

## “Raincloud” Plots (I)



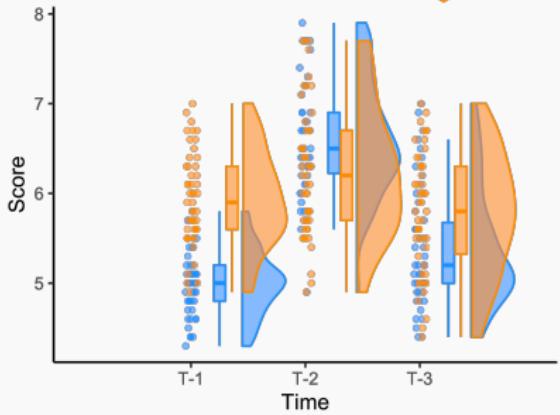
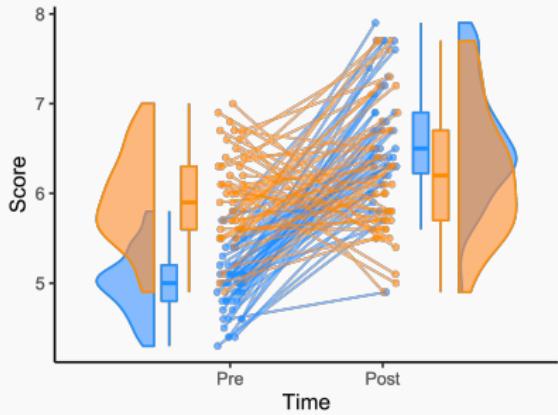
# Praxis-Teil



Beschränkt man sich auf ein Datenset (z.B. ein zufällig gewähltes Imputationsset) können auch die Funktionen des [{raincloudplots} Package](#) (Allen et al., 2019) genutzt werden (Dokumentation).

## Installation:

```
remotes::install_github('jorvlan/raincloudplots')
```



# Evaluation Randomisiert-Kontrollierter Studien und Experimente mit R

Multiples Testen & Alphafehler-Korrektur

---

Prof. Dr. David Ebert & Mathias Harrer

Graduiertenseminar TUM-FGZ

Psychology & Digital Mental Health Care, Technische Universität München

## Multiples Testen

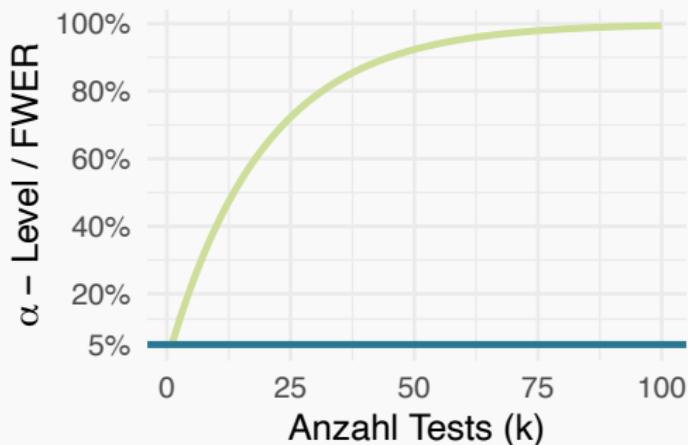
---

Zur Evaluation der Studienhypothesen ist in manchen Fällen **mehr als ein Test notwendig**. Zwei typische Szenarien sind:

- **Multiple primäre Endpunkte:** es wurden mehrere Outcomes gemessen (z.B. depressive Symptomschwere, Lebensqualität, Lungenfunktion, Gait Speed, ...), für die alle ein Interventionseffekt nachgewiesen werden soll.
- **Multiple Assessments des primären Endpunkts:** es gibt nur ein primäres Outcome, dieses wurde aber mehrfach gemessen, und der Interventionseffekt soll zu allen Zeitpunkten nachgewiesen werden.
- **Multiple Interventionsgruppen:** es gibt mehr als eine Interventionsgruppe, und es soll der Effekt aller untersuchten Interventionen im Vergleich zur Kontrollgruppe nachgewiesen werden.

Werden  $k$  multiple Tests mit einem nominalen Signifikanzniveau von (typischerweise)  $\alpha = 0.05$  getestet, führt dies zur **Inflation der family-wise error rate** (FWER, Li et al., 2017; Vickerstaff et al., 2019):

$$P(\text{mindestens ein signifikantes Ergebnis}) = 1 - (1 - \alpha)^k$$



## Die Bonferroni- und Holm-Bonferroni-Korrektur

Um eine Inflation falsch-positiver Befunde zu vermeiden und das nominale Signifikanzniveau trotz multipler Tests zu erhalten, ist daher eine  $\alpha$ -Fehlerkorrektur notwendig.

Zwei häufig gewählte Verfahren sind die (überkonservative) **Bonferroni**- und (weniger strenge) **Holm-Bonferroni**-Korrektur (Holm, 1979). Diese lassen sich in R mit der `p.adjust`-Funktion anwenden:

```
p <- c(0.001, 0.023, 0.025)          # Sammle p-Werte multipler Tests
p.adjust(p, method = "bonferroni")    # Bonferroni-Korrektur
p.adjust(p, method = "holm")         # Holm-Bonferroni-Korrektur

#> [1] 0.003 0.069 0.075
#> [1] 0.003 0.046 0.046
```

### Wann muss korrigiert werden? (Li et al., 2017; Vickerstaff et al., 2019)

- **Multiple Outcomes:** eine Korrektur ist nur notwendig, wenn alle Outcome-Tests als *konfirmatorisch* betrachtet werden. Gibt es nur einen primären und mehrere sekundäre Endpunkte, ist keine Korrektur notwendig. Ergebnisse der sekundären Endpunkte sind dann aber nur explorativ, kein konfirmatorischer Nachweis!
- **Multiple Messzeitpunkte:** ist ein Messzeitpunkt als konfirmatorisch-primär präspezifiziert, ist keine Korrektur notwendig; ebenso bei Tests, bei denen über alle MZP *hinweg* modelliert wird (z.B. repeated-measures ANOVA, gemischte Modelle).
- **Multiple Interventionsgruppen:** eine Korrektur ist insbesondere dann notwendig, wenn die Interventionsgruppen “*verwandt*” sind (z.B. gleiches Medikament in unterschiedlicher Dosis).

## **Einseitig oder Zweiseitig?**

---

Wir erinnern uns → Studienhypothesen bei Superiority Trials:

$$H_0 : \mu_1 \leq \mu_0 \Rightarrow \tau \leq 0$$

$$H_1 : \mu_1 > \mu_0 \Rightarrow \tau > 0$$

Dies impliziert eigentlich einen **einseitigen Test**, da wir nur testen wollen, ob  $\mu_1$  größer als  $\mu_0$  ist, nicht ob  $\mu_1 \neq \mu_0$ . "By default" ausgegebene Tests sind aber (häufig) zweiseitig (z.B. *t*-Tests)!

Unter **gewissen Voraussetzungen** kann eine einseitige Testung vorgenommen, bzw.  $\alpha$  auf 10% erhöht werden (Dunnett & Gent, 1996; Knottnerus & Bouter, 2001):

- Das Ziel der Studie ist es nicht, **negative Effekte konfirmatorisch bestätigen zu können** (z.B. weil diese auf Basis vorheriger Forschung als unwahrscheinlich gelten).
- Wird **kein positiver Effekt** der Intervention festgestellt, so wird dies als die **Kontrollgruppe favorisierende Evidenz** ausgelegt.

Fragen?

Anmerkungen?

Kommentare?



# Evaluation Randomisiert-Kontrollierter Studien und Experimente mit R

Effektstärken

---

Prof. Dr. David Ebert & Mathias Harrer

Graduiertenseminar TUM-FGZ

Psychology & Digital Mental Health Care, Technische Universität München

## **Effektstärken: Hintergrund**

---

- Der Begriff "**Effektstärke**" (*effect size*) ist **nicht klar definiert**; manche verstehen darunter nur **standardisierte Mittelwertsunterschiede** (Cohen's *d*); andere bevorzugen **breitere Definitionen**.
- Die "**engere**" **Definition** ist nur **schwer haltbar**, da auch Korrelationen, Odds Ratios, *z*-Werte etc. Richtung und Stärke eines Effekt ausdrücken können (und teils auch ineinander transformierbar sind).
- Im Kontext von RCTs werden Effektstärken genutzt, um die **Größe des Interventionseffekts** zu quantifizieren und **vergleichbar** zu machen.
- Eine praktische Schwierigkeit stellt dabei die **korrekte Berechnung von Konfidenzintervallen** dar. Für viele Effektstärken existieren **geschlossene Formeln zur Berechnung der sampling-Varianz**; diese beziehen aber nicht die **Imputationssicherheit** mit ein!

Eine elegante Form der Berechnung von Effekstärken und deren 95%-Konfidenzintervalle bei MI stellt die Nutzung der "**natürlichen Interpretation des  $\beta$ -Gewichts der Treatmentvariable**" in (G)LM dar.

Es sei  $f(\beta)$  eine je nach Linkfunktion variierende Transformationsfunktion (häufig die Exponentialfunktion), und  $\hat{\theta}$  die zu berechnende Effektstärke:

$$\hat{\theta} = f(\hat{\beta}_{\text{treat}})$$

$$\hat{\theta}_{\text{lower}} = f\left(\hat{\beta}_{\text{treat}} - t_{\nu_{(\text{MI})}, 0.975} \times \text{S.E.}_{\hat{\beta}_{\text{treat}}}\right)$$

$$\hat{\theta}_{\text{upper}} = f\left(\hat{\beta}_{\text{treat}} + t_{\nu_{(\text{MI})}, 0.975} \times \text{S.E.}_{\hat{\beta}_{\text{treat}}}\right)$$

## Viele Effektstärken lassen sich direkt aus GLM ableiten!

Verteilung	Link-Typ	Linkfunktion	Support ( $y$ )	$\exp(\beta_{\text{treat}})$	family
Normal	Identity	$\mu$	$\mathbb{R}$	$\log_e(\text{MD})$	<code>gaussian(„identity“)</code>
Binomial	Logit*	$\log_e\left(\frac{\mu}{1-\mu}\right)$	$\frac{0,1,\dots,m}{m}$	OR	<code>binomial(„logit“)</code>
	Log	$\log_e(\mu)$	$\frac{0,1,\dots,m}{m}$	RR	<code>binomial(„log“)</code>
	Complem. Log-Log	$\log_e[-\log_e(1-\mu)]$	$\frac{0,1,\dots,m}{m}$	HR	<code>binomial(„cloglog“)</code>
	Log	$\log_e\left(\frac{\mu}{\mu+k}\right)$	$\mathbb{N}_0$	(I)RR	<code>log (in MASS::glm.nb)</code>
Poisson	Log	$\log_e(\mu)$	$\mathbb{N}_0$	(I)RR	<code>poisson(„log“)</code>
Gamma	Inverse	$-\mu^{-1}$	$\mathbb{R}_+$	-	<code>gamma(„inverse“)</code>

### Beispiel: Standardisierte Mittelwertsunterschiede (Cohen's $d$ )

$$d = \frac{\mu_1 - \mu_0}{s_{\text{pooled}}} = \frac{\mu_1 - \mu_0}{\sqrt{\frac{(n_1-1)s_1^2 + (n_0-1)s_0^2}{(n_1-1) + (n_0-1)}}}$$

→ Hier kann die gepoolte Standardabweichung als “Offset” genutzt werden, um  $d$  direkt aus einem linearen Model zu berechnen.

```
# Der Wert von s_pooled wurde mit 6.32 berechnet
with(implist, lm(I(pss.1/6.32) ~ 1 + group + pss.0)) %>%
  testEstimates() -> m

m; confint(m)
```

**Ergebnis:**  $d = -0.92$ , 95%CI:  $-1.16 \text{---} -0.68$ .

```
#>           Estimate Std. Error   t.value      df P(>|t|)    RIV    FMI
#> (Intrcpt)  1.486     0.400    3.719  4878.106  0.000  0.075  0.071
#> group      -0.918     0.122   -7.548  4141.218  0.000  0.082  0.077
#> pss.0       0.085     0.015    5.517  6989.062  0.000  0.062  0.059
#>
#> Unadjusted hypothesis test as appropriate in larger samples.
#>
#>           2.5 %     97.5 %
#> (Intercept) 0.70280613  2.2697069
#> group        -1.15677072 -0.6797633
#> pss.0        0.05486783  0.1153496
```

## Beispiel: Odds Ratio (OR)

Für eine logistische Regression mit dummy-kodierter Treatmentvariable gilt:

$$\exp(\hat{\beta}_{\text{treat}}) = \widehat{\text{OR}}$$

Wir können also in R die **Exponentialfunktion** `exp` nutzen, um direkt die **OR und das Konfidenzintervall** zu berechnen:

```
with(implist, glm(ri ~ 1 + group + pss.0, binomial("logit"))) %>%
  testEstimates() -> mi.logreg

c(mi.logreg$estimates[2,1], confint(mi.logreg)[2,]) %>% exp()

#>      2.5 % 97.5 %
#> 5.958 3.366 10.546
```

## Cohen's $d$ von Within-Group-Vergleichen

---

- Cohen's  $d$  ist bei Zwischengruppenvergleichen (z.B.  $\hat{\mu}_{\text{treat}} - \hat{\mu}_{\text{control}}$ ) eindeutig definiert.
- Strittiger ist, wie Cohen's  $d$  bei **Within-Group-Vergleichen** (z.B.  $\hat{\mu}_{\text{pre}} - \hat{\mu}_{\text{post}}$ ) zu berechnen ist, und ob dann noch von "dem" Cohen's  $d$  die Rede ist.
- Das Kernproblem liegt dabei in der **passenden Berechnung von  $s_{\text{pooled}}$**  zur Standardisierung des Mittelwertsunterschieds.

Five different “Cohen’s d”  
statistics for within-  
subject designs

© March 25, 2016    effect size

siehe Westfall (2016).

### Die "agnostische" Methode

Eine Möglichkeit zur Berechnung von  $d_{\text{within}}$  ist:

$$d_{\text{within}} = \frac{\hat{\mu}_{\text{pre}} - \hat{\mu}_{\text{post}}}{s_{\text{pooled}}} = \frac{\hat{\mu}_{\text{pre}} - \hat{\mu}_{\text{pre}}}{\sqrt{(s_{\text{pre}}^2 + s_{\text{post}}^2)/2}}$$

Unter Annahme dass  $n_{\text{pre}} = n_{\text{post}}$ .

Hier wird schlicht "ignoriert," dass zu beiden Messzeitpunkten die gleichen Personen gemessen wurden (abhängige Stichprobe). Die Standardisierung erfolgt durch die **mittlere Standardabweichung der beiden Messzeitpunkte**.

## Die "modellbasierte" Methode

*"Effect size is how large the true condition effect is relative to the true amount of variability in this effect across the population. Measures of true effect and true amount of variability are only defined in statistical models. They don't really exist accept within the context of a model."* (Rouder, 2016)

Nach dieser Logik ist  $s_{\text{pooled}}$  die **Standardabweichung der Residuale** des dem Design **zugrundeliegenden Modells!** Bei within-group-Vergleichen ist dies typischerweise eine *repeated-measures ANOVA* oder ein gemischtes Modell (da mehrere Messzeitpunkte pro Person).

### **Cave**

Bei reinen prä-post-Vergleichen führt dies zu **nahezu identischen Ergebnissen** wie das "agnostische" Vorgehen. Der "modellbasierte" Ansatz ist aber **breiter einsetzbar** (z.B. repeated-measures ANOVA mit zwei Gruppen, AB-Designs, ...)

## Beispiel: Berechnung von $d_{\text{within}}$ aus rmANOVA

```
(m <- aov(pss ~ time + Error(id), data.ig)$Within)
cbind(coef(m), confint(m))
```

```
#>                               time Residuals
#> Sum of Squares   4811.095  7394.886
#> Deg. of Freedom       1        261
#>
#> Residual standard error: 5.322866
#> Estimated effects are balanced
#>                               2.5 %    97.5 %
#> time1 -8.537879 -9.828029 -7.247729
```

$$d_{\text{within}} = \frac{\text{time1}}{\text{Residual standard error}} = \frac{-8.538}{5.323} \approx -1.6$$

# Evaluation Randomisiert-Kontrollierter Studien und Experimente mit R

Reporting & Reproduzierbarkeit

---

Prof. Dr. David Ebert & Mathias Harrer

Graduiertenseminar TUM-FGZ

Psychology & Digital Mental Health Care, Technische Universität München

## **Reporting & Reproduzierbarkeit**

---

## Folgende Szenarien sind im Forschungsalltag keine Seltenheit:

- Wir finden einen **Fehler** in unserem Code, und müssen Teile unserer **Analyse** mit leichten Veränderungen **wiederholen**.
- Mitarbeiter oder **Reviewer** wünschen sich weitere oder andere Analysen unserer Daten.
- Wir müssen Teile der Analysen an Mitarbeiter **delegieren**.
- Wir wollen unsere Ergebnisse mit **Kollegen teilen**, aber diese kennen R nicht und haben RStudio nicht installiert.

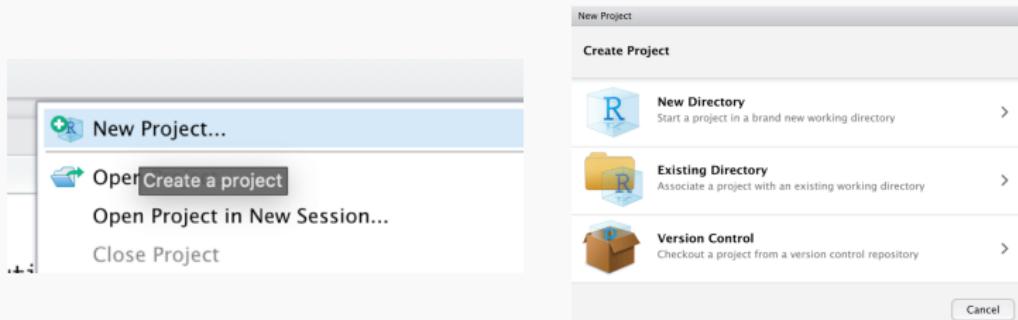
→ In all diesen Fällen ist ein **reproduzierbarer Workflow** extrem vorteilhaft!

## R Projekte

---

**R Projekte** erstellen eine neue Umgebung in einem Ordner auf unserem Computer. In diesem Ordner werden alle Daten und der R Code, den wir für unsere Analysen benötigen, gespeichert.

Die Durchführung von Analysen in einem R-Projekt bedeutet, dass alle von uns erstellten Objekte vorübergehend in der Projektumgebung gespeichert werden und beim nächsten Öffnen wieder zugänglich sind.

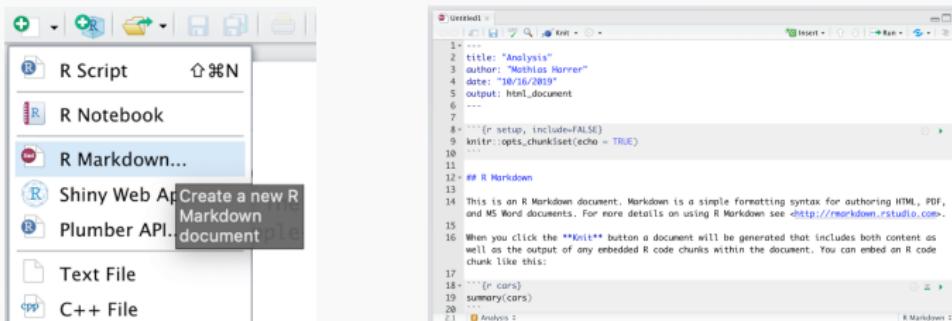


## R Markdown

---

**Markdown** ist eine einfache Auszeichnungssprache zur Textformatierung. **R Markdown** (Xie, Allaire und Grolemund 2018) ist eine Erweiterung von Markdown und macht es einfach, Text, R Code und Output in einem HTML- oder PDF-Dokument zu kombinieren.

Die Markdown-Syntax ist schnell erlernbar: siehe [Cheatsheet](#).



Der wichtigste Teil von R Markdown-Dokumenten ist der **YAML**-Header.

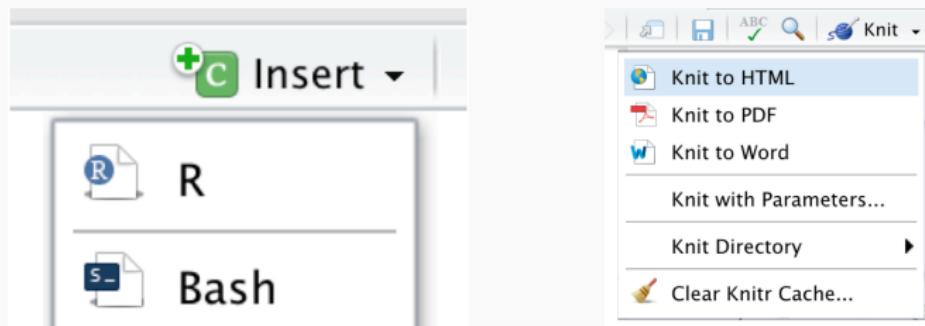
```
---
```

```
title: "Analysis"
author: "Author Name"
date: "10/16/2020"
output: html_document
```

```
---
```

Er steuert den Titel, den Autor, das Datum und das Exportformat des Dokuments. Das Ausgabeformat, das wir für unser Dokument gewählt haben, ist `html_document` (Export als HTML).

Es können **neue Code-Chunks** hinzugefügt werden. Der "**Knit**"-Button führt den R-Code in den Chunks aus und **rendert** das Dokument.



## **Open Science Framework (OSF)**

---

- Das Open Science Framework ([osf.io](https://osf.io)) erlaubt die versionierte Veröffentlichung von Daten, Code und anderen Materialien.
- OSF Registries erlaubt die **Präregistrierung** von Analysen.
- OSF Preprint ermöglicht die Veröffentlichung von **Preprints**.

The screenshot shows the OSF Home page for a project titled "Meta-Analysis Project". The top navigation bar includes links for "My Quick Files", "My Projects", "Search", "Support", "Donate", and "Mathias Harter". A message at the top states: "Notice: This site will undergo maintenance between Oct 17, 2019 3:08 AM and Oct 17, 2019 4:00 AM (H200 UTC). Thank you for your patience." Below the message, the project details are listed: "Date created: 2019-10-16 05:55 PM | Last Updated: 2019-10-16 05:55 PM", "Category: Project", "Description: Add a brief description to your project.", and "License: Add a license". The main interface includes sections for "Wiki" (with a note to add important information, links, or images), "Files" (with a note to click on a storage provider or drag and drop to upload), and "Tags" (with a note to add a tag to enhance discoverability). At the bottom, there are sections for "Name" (with "Meta-Analysis Project" selected) and "Modified" (with "OSF Storage Limited Status").

The screenshot shows the OSF Registries "New registration" page for the same "Meta-Analysis Project". The top navigation bar includes links for "Add New", "Help", "Donate", and "Mathias Harter". On the left, a sidebar lists registration components: "Metadata", "Study Information", "Design Plan", "Sampling Plan", "Variables", "Analysis Plan", "Other", and "Review". The main area is titled "Registration Metadata" with the sub-section "Title". The "Title" field contains "Meta-Analysis Project". The "Description" field is empty. At the bottom, there is a section for "Contributors" with the note "Edit contributors on your project" and a table for managing contributor names, permissions, and citations. A blue button on the right says "Next >". A status message at the bottom right indicates "Autosaved: a minute ago".