

# Programmieren und Statistik mit R

## Campus 2019 - Forum von und für Stipendiat\*innen

Kerstin Pierick

Abteilung Pflanzenökologie und Ökosystemforschung, Universität Göttingen

13.08.2019



1

## Einführung in R

- Was ist R?
- Vorteile von R
- Benutzung
- Was kann R?
- Good practice
- Weiterführendes

2

## Praktischer Teil

# Was ist R?

# Was ist R?

- Statistik-orientierte Programmiersprache



# Was ist R?

- Statistik-orientierte Programmiersprache
- Funktional, dynamisch, objektorientiert



# Was ist R?

- Statistik-orientierte Programmiersprache
- Funktional, dynamisch, objektorientiert
- Freie Software (GNU General Public Licence)



# Was ist R?

- Statistik-orientierte Programmiersprache
- Funktional, dynamisch, objektorientiert
- Freie Software (GNU General Public Licence)
- Entwickelt und aktualisiert vom R Development Core Team



## Vorteile von R

# Vorteile von R

- Reproduzierbarkeit

# Vorteile von R

- Reproduzierbarkeit
- Frei, Open Source und kostenlos

# Vorteile von R

- Reproduzierbarkeit
- Frei, Open Source und kostenlos
- Gute Dokumentation

# Vorteile von R

- Reproduzierbarkeit
- Frei, Open Source und kostenlos
- Gute Dokumentation
- Aktive Community

# Vorteile von R

- Reproduzierbarkeit
- Frei, Open Source und kostenlos
- Gute Dokumentation
- Aktive Community
- Fortlaufende Aktualisierung und Weiterentwicklung

# Vorteile von R

- Reproduzierbarkeit
- Frei, Open Source und kostenlos
- Gute Dokumentation
- Aktive Community
- Fortlaufende Aktualisierung und Weiterentwicklung
- Weit verbreitet in Wissenschaft und Wirtschaft

# Vorteile von R

- Reproduzierbarkeit
- Frei, Open Source und kostenlos
- Gute Dokumentation
- Aktive Community
- Fortlaufende Aktualisierung und Weiterentwicklung
- Weit verbreitet in Wissenschaft und Wirtschaft
- Schnittstellen zu anderen Programmen und Programmiersprachen

# Vorteile von R

- Reproduzierbarkeit
- Frei, Open Source und kostenlos
- Gute Dokumentation
- Aktive Community
- Fortlaufende Aktualisierung und Weiterentwicklung
- Weit verbreitet in Wissenschaft und Wirtschaft
- Schnittstellen zu anderen Programmen und Programmiersprachen
- Für alle Betriebssysteme

# Vorteile von R

- Reproduzierbarkeit
- Frei, Open Source und kostenlos
- Gute Dokumentation
- Aktive Community
- Fortlaufende Aktualisierung und Weiterentwicklung
- Weit verbreitet in Wissenschaft und Wirtschaft
- Schnittstellen zu anderen Programmen und Programmiersprachen
- Für alle Betriebssysteme
- Kann alles

# Benutzung

# Editoren

R wird meistens innerhalb eines *Editors* (Synonym: *integrated development environment, IDE*) benutzt

Empfehlung: **RStudio**



# RStudio

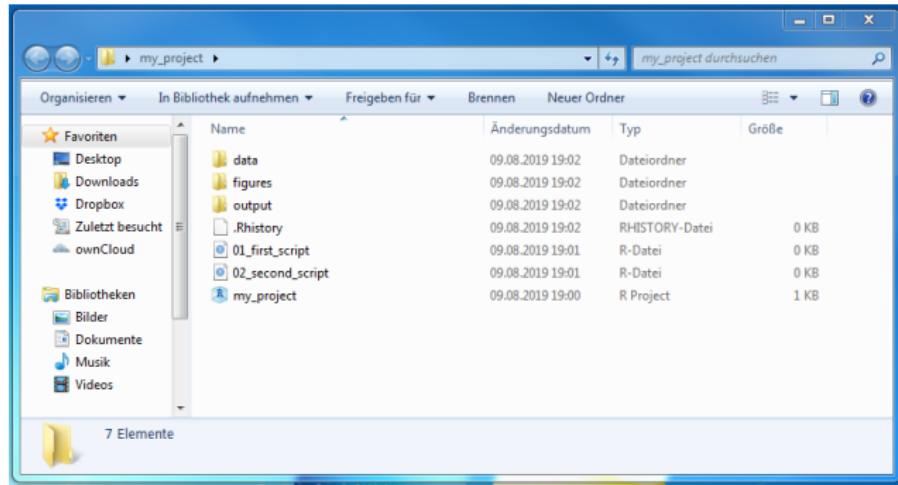
The screenshot shows the RStudio interface with the following components:

- Script Editor:** Displays the script `Skript_zur_VL.R` containing R code for basic operations and logical operators, and a session history with the `iris` dataset.
- Environment:** Shows the `iris` dataset with 150 observations and 5 variables: `sepal.length`, `sepal.width`, `petal.length`, `petal.width`, and `Species`. It also lists the function `se_of_mean`.
- Plots:** A scatter plot of `petal.length` (y-axis, 15 to 30) versus `petal.width` (x-axis, 3.0 to 6.0). The data points are open circles.

<https://www.rstudio.com/>

# Skripte und Projekte

- Code wird in **Skripten** gespeichert
- **Projekte** sind Verzeichnisse, in denen Skripte und alle verwendeten und erzeugten Daten gespeichert werden



# Grundlagen der Syntax

"To understand computations in R, two slogans are helpful: Everything that exists is an object. Everything that happens is a function call."

"Um Berechnungen in R zu verstehen sind zwei Sätze hilfreich: Alles, was existiert, ist ein Objekt. Alles, was passiert, ist ein Funktionsaufruf."

John M. Chambers: Object-Oriented Programming, Functional Programming and R. In: Statistical Science. Band 29, Nr. 2, 2014, S. 167–180, <https://arxiv.org/abs/1409.3531>.

# Grundlagen der Syntax

"To understand computations in R, two slogans are helpful: Everything that exists is an object. Everything that happens is a function call."

"Um Berechnungen in R zu verstehen sind zwei Sätze hilfreich: Alles, was existiert, ist ein Objekt. Alles, was passiert, ist ein Funktionsaufruf."

John M. Chambers: Object-Oriented Programming, Functional Programming and R. In: Statistical Science. Band 29, Nr. 2, 2014, S. 167–180, <https://arxiv.org/abs/1409.3531>.

- Objekte und Zuordnung
- Vektoren
- Funktionen

# Pakete

- Standardinstallation: *Base R*

# Pakete

- Standardinstallation: *Base R*
- Wird ergänzt durch z.Z. >15000 Pakete



# Pakete

- Standardinstallation: *Base R*
- Wird ergänzt durch z.Z. >15000 Pakete
- Pakete enthalten eine Sammlung an Funktionen für bestimmten Zweck (z.B. statistische Methode)



# Was kann R?

# Datensätze

- Zusammenfügen
- Umsortieren
- Neue Variablen
- Filtern
- Zusammenfassen
- ...

country	year	key	value	country	year	cases	population
Afghanistan	1999	cases	745	Afghanistan	1999	19987071	
Afghanistan	1999	population	19987071	Afghanistan	2000	2666	20595360
Afghanistan	2000	cases	2666	Brazil	1999	37737	172006362
Afghanistan	2000	population	20595360	Brazil	2000	80488	174504898
Brazil	1999	cases	37737	China	1999	212258	1272915272
Brazil	1999	population	172006362	China	2000	213766	1280428583
Brazil	2000	cases	80488				
Brazil	2000	population	174504898				
China	1999	cases	212258				
China	1999	population	1272915272				
China	2000	cases	213766				
China	2000	population	1280428583				

table2

Grolmund und Wickham (2016)

# Nützliches Paket: tidyverse

- Eine moderne Paketsammlung für Datenanalyse
- Enthält viele beliebte Pakete wie z.B. ggplot2 und dplyr
- Vereinfacht Datenanalyse in einem konsistenten Framework



# Statistik

- Beinahe alle statistischen Methoden sind in R möglich

# Statistik

- Beinahe alle statistischen Methoden sind in R möglich
- Standardmethoden in Base R, für alles andere: Pakete

# Statistik

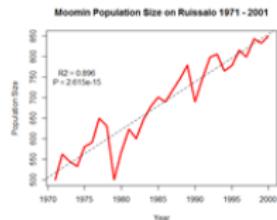
- Beinahe alle statistischen Methoden sind in R möglich
- Standardmethoden in Base R, für alles andere: Pakete
- Beispiele: `lme4` für Mixed Modelling, `lavaan` for Structural Equation Modelling...

# Datenvisualisierung

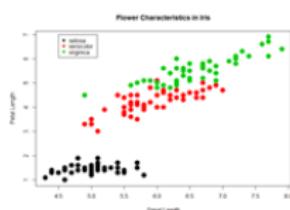
## Schnell & einfach mit Base R



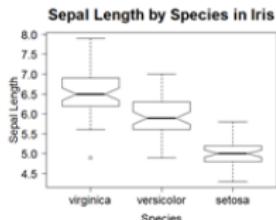
1. Basic Histogram



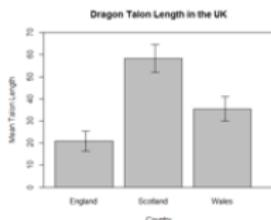
2. Line Graph with Regression



3. Scatterplot with Legend



4. Boxplot with reordered/ formatted axes

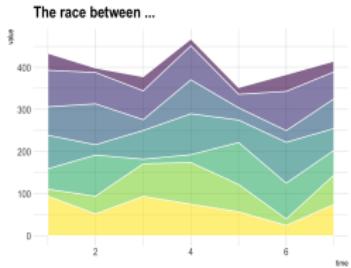
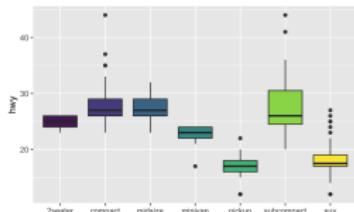
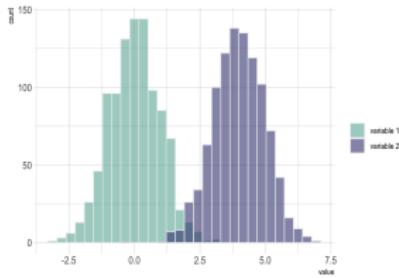
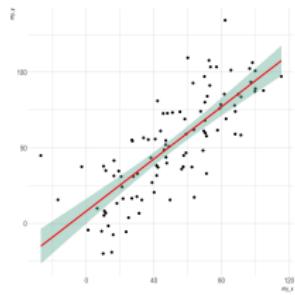


5. Boxplot with Error Bars

[https://rstudio-pubs-static.s3.amazonaws.com/7953\\_4e3efd5b9415444ca065b1167862c349.html](https://rstudio-pubs-static.s3.amazonaws.com/7953_4e3efd5b9415444ca065b1167862c349.html)

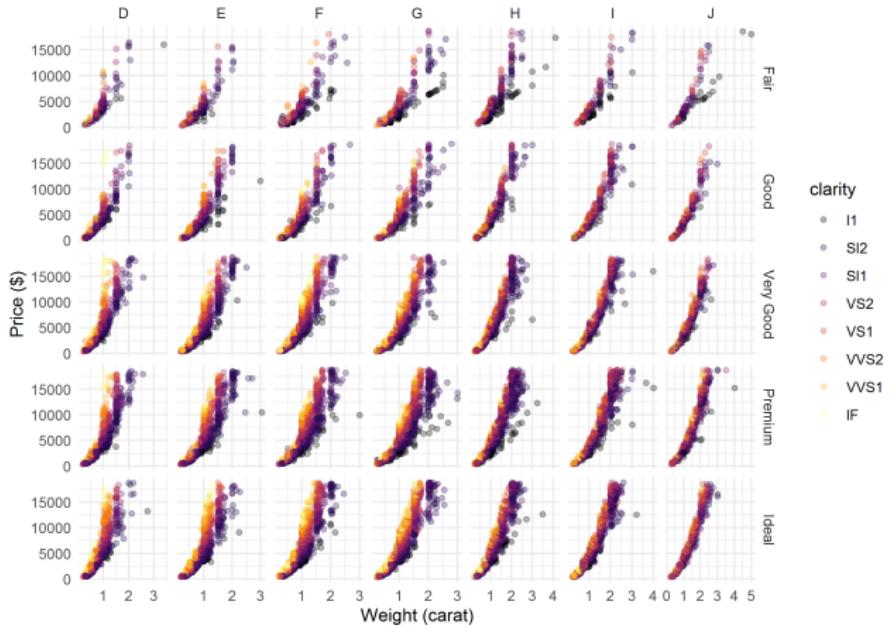
# Datenvisualisierung

Professionell (und auch nicht schwierig) mit `ggplot2`



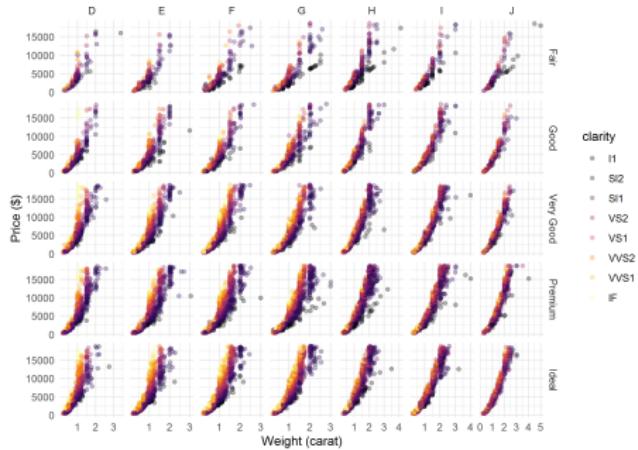
# Datenvisualisierung

Professionell (und auch nicht schwierig) mit `ggplot2`



# Datenvisualisierung

Professionell (und auch nicht schwierig) mit `ggplot2`

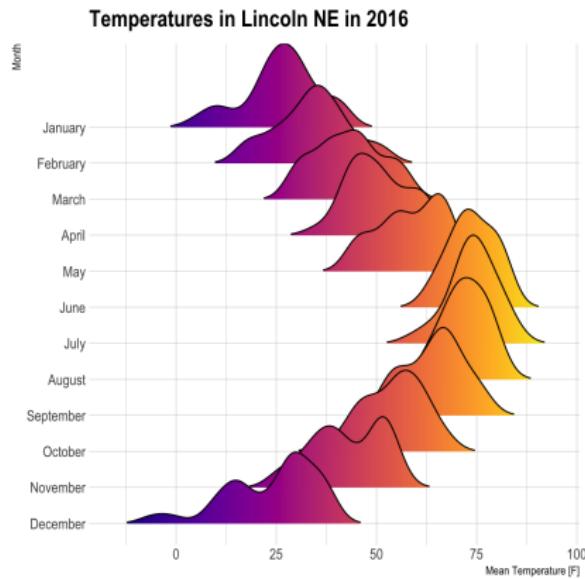


```
ggplot(diamonds, aes(x = carat, y = price, color = clarity)) +
  geom_point(alpha = 0.3) +
  scale_color_viridis_d(option = 3) +
  facet_grid(cut ~ color, scales = "free") +
  theme_minimal() +
  labs(x = "Weight (carat)", y = "Price ($)")

ggsave("figures/diamonds.png")
```

# Datenvisualisierung

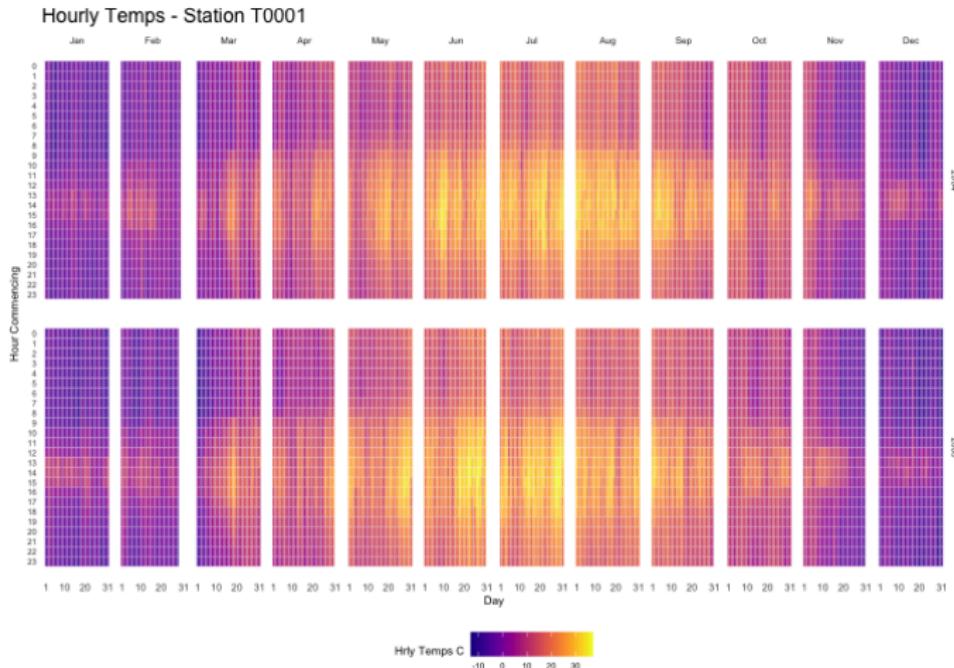
Professionell (und auch nicht schwierig) mit `ggplot2`



<https://www.r-graph-gallery.com/294-basic-ridgeline-plot.html>

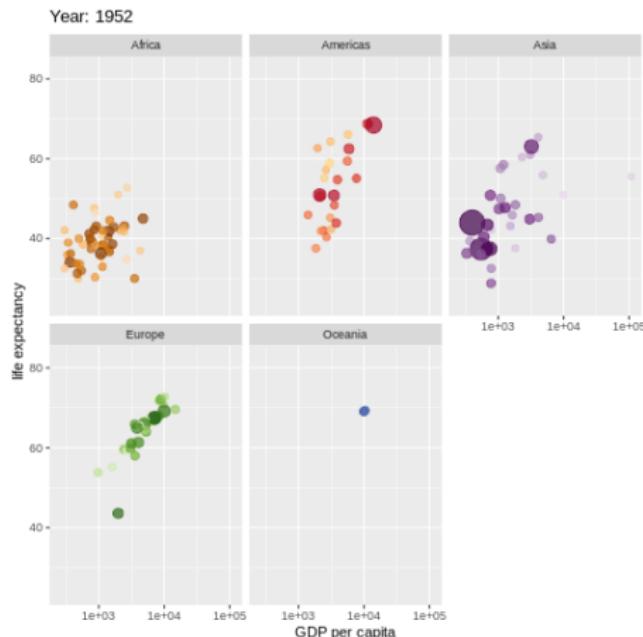
# Datenvisualisierung

Professionell (und auch nicht schwierig) mit `ggplot2`



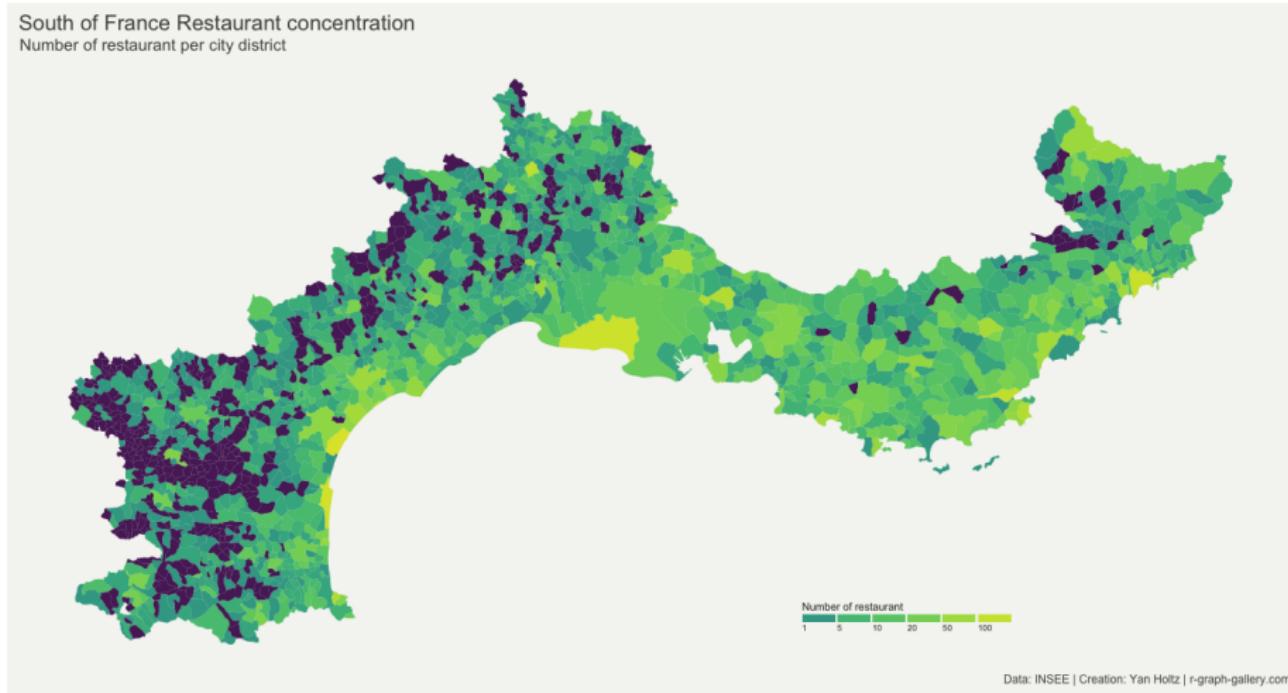
<https://www.r-graph-gallery.com/283-the-hourly-heatmap/>

# Animierte Abbildungen mit gganimate



<https://github.com/thomasp85/gganimate/wiki/Gapminder>

# Räumliche Daten

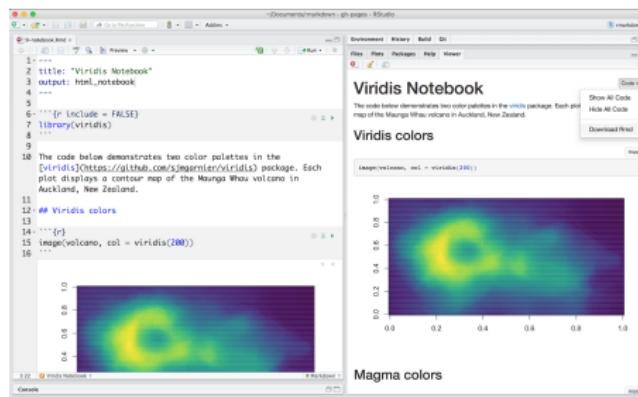


<https://www.r-graph-gallery.com/327-chloropleth-map-from-geojson-with-ggplot2/>

# Ergebnisse darstellen

## mit RMarkdown

- Textformatierung mit Einbindung von R Code, Abbildungen und Tabellen
- Praktisch für Dokumentation einer Datenanalyse
- Output als PDF, Word-Dokument oder HTML



<https://rmarkdown.rstudio.com/lesson-10.html>

# Weitere Möglichkeiten

- Web scraping

# Weitere Möglichkeiten

- Web scraping
- Interface zu öffentlichen Datenbanken (z.B. Deutscher Wetterdienst)

# Weitere Möglichkeiten

- Web scraping
- Interface zu öffentlichen Datenbanken (z.B. Deutscher Wetterdienst)
- Machine learning

# Weitere Möglichkeiten

- Web scraping
- Interface zu öffentlichen Datenbanken (z.B. Deutscher Wetterdienst)
- Machine learning
- Modellierung

# Weitere Möglichkeiten

- Web scraping
- Interface zu öffentlichen Datenbanken (z.B. Deutscher Wetterdienst)
- Machine learning
- Modellierung
- GIS

# Weitere Möglichkeiten

- Web scraping
- Interface zu öffentlichen Datenbanken (z.B. Deutscher Wetterdienst)
- Machine learning
- Modellierung
- GIS
- Text mining

# Weitere Möglichkeiten

- Web scraping
- Interface zu öffentlichen Datenbanken (z.B. Deutscher Wetterdienst)
- Machine learning
- Modellierung
- GIS
- Text mining
- ...

## Good practice

# Hilfe finden

- Funktionen ?, ??, help.search() und help()

# Hilfe finden

- Funktionen ?, ??, `help.search()` und `help()`
- Google

# Hilfe finden

- Funktionen ?, ??, `help.search()` und `help()`
- Google
- Cheatsheets

<https://www.rstudio.com/resources/cheatsheets/>

# Hilfe finden

- Funktionen ?, ??, `help.search()` und `help()`
- Google
- Cheatsheets  
<https://www.rstudio.com/resources/cheatsheets/>
- Stackoverflow <https://stackoverflow.com/questions/tagged/r>

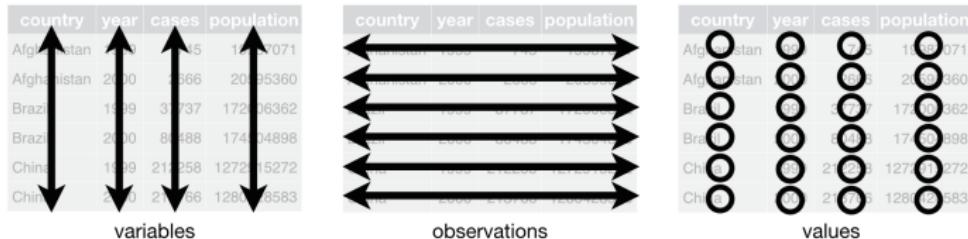
# Hilfe finden

- Funktionen ?, ??, `help.search()` und `help()`
- Google
- Cheatsheets  
<https://www.rstudio.com/resources/cheatsheets/>
- Stackoverflow <https://stackoverflow.com/questions/tagged/r>
- Vignetten

# "Ordentliche" Daten

"tidy data"

- Jede Zeile eine Beobachtung

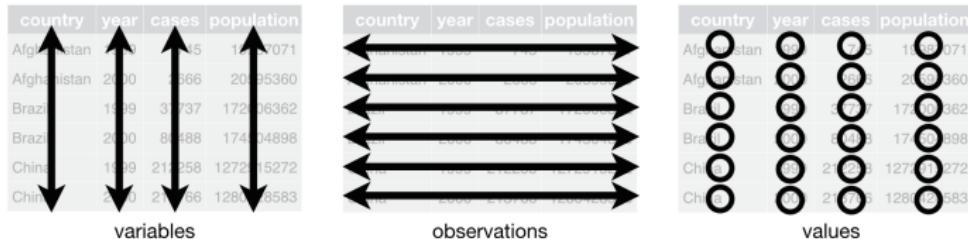


Grolemund und Wickham (2016)

# "Ordentliche" Daten

"tidy data"

- Jede Zeile eine Beobachtung
- Jede Spalte eine Variable

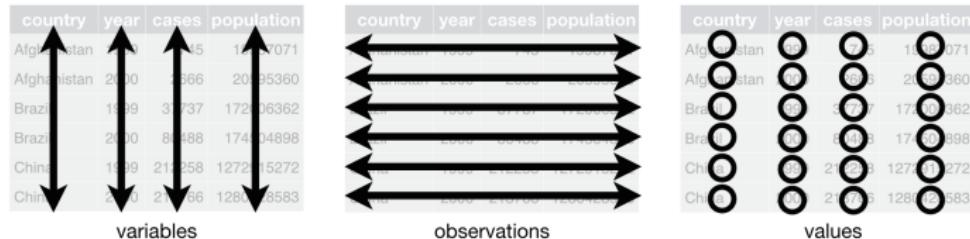


Grolemund und Wickham (2016)

# "Ordentliche" Daten

"tidy data"

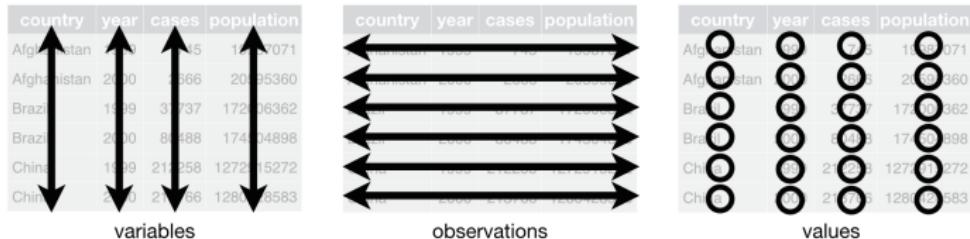
- Jede Zeile eine Beobachtung
- Jede Spalte eine Variable
- Jede Beobachtungseinheit eine Tabelle



# "Ordentliche" Daten

"tidy data"

- Jede Zeile eine Beobachtung
- Jede Spalte eine Variable
- Jede Beobachtungseinheit eine Tabelle
- Namen der Variablen ohne Sonderzeichen (außer \_ und .)

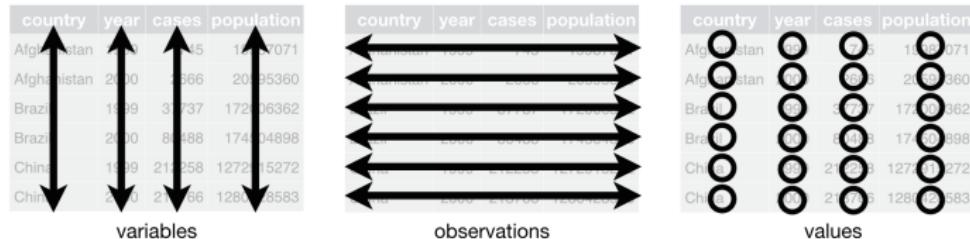


Grolemund und Wickham (2016)

# "Ordentliche" Daten

"tidy data"

- Jede Zeile eine Beobachtung
- Jede Spalte eine Variable
- Jede Beobachtungseinheit eine Tabelle
- Namen der Variablen ohne Sonderzeichen (außer \_ und .)
- Keine Kopfzeilen außer Namen der Variablen

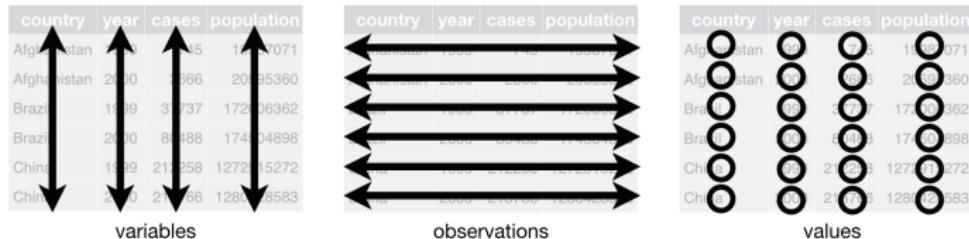


Grolemund und Wickham (2016)

# "Ordentliche" Daten

"tidy data"

- Jede Zeile eine Beobachtung
- Jede Spalte eine Variable
- Jede Beobachtungseinheit eine Tabelle
- Namen der Variablen ohne Sonderzeichen (außer \_ und .)
- Keine Kopfzeilen außer Namen der Variablen
- Metadaten in gesondertem Tabellenblatt



Grolemund und Wickham (2016)

# "Ordentliche" Projekte

- Jeder größere Arbeitsabschnitt ein Skript. Beispiel:

# "Ordentliche" Projekte

- Jeder größere Arbeitsabschnitt ein Skript. Beispiel:
  - ➊ Daten zusammenfügen und bereinigen, neue Variablen berechnen

# "Ordentliche" Projekte

- Jeder größere Arbeitsabschnitt ein Skript. Beispiel:
  - 1 Daten zusammenfügen und bereinigen, neue Variablen berechnen
  - 2 Ausführliche visuelle Exploration

# "Ordentliche" Projekte

- Jeder größere Arbeitsabschnitt ein Skript. Beispiel:
  - 1 Daten zusammenfügen und bereinigen, neue Variablen berechnen
  - 2 Ausführliche visuelle Exploration
  - 3 Statistische Analyse, Erzeugung von Output-Tabellen und schönen Abbildungen

# "Ordentliche" Projekte

- Jeder größere Arbeitsabschnitt ein Skript. Beispiel:
  - 1 Daten zusammenfügen und bereinigen, neue Variablen berechnen
  - 2 Ausführliche visuelle Exploration
  - 3 Statistische Analyse, Erzeugung von Output-Tabellen und schönen Abbildungen
- Durchnummerierte Skripte

# "Ordentliche" Projekte

- Jeder größere Arbeitsabschnitt ein Skript. Beispiel:
  - ① Daten zusammenfügen und bereinigen, neue Variablen berechnen
  - ② Ausführliche visuelle Exploration
  - ③ Statistische Analyse, Erzeugung von Output-Tabellen und schönen Abbildungen
- Durchnummerierte Skripte
- Aussagekräftige Dateinamen

# "Ordentliche" Projekte

- Jeder größere Arbeitsabschnitt ein Skript. Beispiel:
  - ① Daten zusammenfügen und bereinigen, neue Variablen berechnen
  - ② Ausführliche visuelle Exploration
  - ③ Statistische Analyse, Erzeugung von Output-Tabellen und schönen Abbildungen
- Durchnummerierte Skripte
- Aussagekräftige Dateinamen
- README-Datei mit Erläuterungen zum Projekt

# "Ordentliche" Projekte

- Jeder größere Arbeitsabschnitt ein Skript. Beispiel:
  - 1 Daten zusammenfügen und bereinigen, neue Variablen berechnen
  - 2 Ausführliche visuelle Exploration
  - 3 Statistische Analyse, Erzeugung von Output-Tabellen und schönen Abbildungen
- Durchnummerierte Skripte
- Aussagekräftige Dateinamen
- README-Datei mit Erläuterungen zum Projekt
- Daten und Abbildungen in Ordner

# Code kommentieren

- Kommentare werden durch # eingeleitet

```
# exclude Januaries (data from Januaries are in other files (see below))
new_files <- mutate(new_files,
  # parse date as date
  date = ymd(date),
  # create a variable where dates are rounded to month level
  year_month = floor_date(date, "month")
) %>%
# exclude Januaries
filter(year_month != "2016-01-01" & year_month != "2017-01-01")
```

# Code kommentieren

- Kommentare werden durch `#` eingeleitet
- Wichtig, um Code für Andere und dein zukünftiges Ich verständlich zu machen

```
# exclude Januaries (data from Januaries are in other files (see below))
new_files <- mutate(new_files,
  # parse date as date
  date = ymd(date),
  # create a variable where dates are rounded to month level
  year_month = floor_date(date, "month")
) %%
# exclude Januaries
filter(year_month != "2016-01-01" & year_month != "2017-01-01")
```

# Code kommentieren

- Kommentare werden durch `#` eingeleitet
- Wichtig, um Code für Andere und dein zukünftiges Ich verständlich zu machen
- Lieber zu viel als zu wenig!

```
# exclude Januaries (data from Januaries are in other files (see below))
new_files <- mutate(new_files,
  # parse date as date
  date = ymd(date),
  # create a variable where dates are rounded to month level
  year_month = floor_date(date, "month")
) %%
# exclude Januaries
filter(year_month != "2016-01-01" & year_month != "2017-01-01")
```

# Code kommentieren

- Kommentare werden durch # eingeleitet
- Wichtig, um Code für Andere und dein zukünftiges Ich verständlich zu machen
- Lieber zu viel als zu wenig!
- Präambel zu Beginn des Skripts

```
#####
#####                                     Unite all data
#####
#####                                     # Kerstin Pierick 2019
#####                                     # kerstin_pierick@riseup.net
#####
#####                                     # Aim: to create a data frame with all data used in the project
#####
#####
```

# Backup und Versionskontrolle

- Relevanz von Backups: Offensichtlich!

# Backup und Versionskontrolle

- Relevanz von Backups: Offensichtlich!
- Relevanz von Versionskontrolle: Es kann praktisch sein, sich frühere Versionen angucken zu können

# Backup und Versionskontrolle

- Relevanz von Backups: Offensichtlich!
- Relevanz von Versionskontrolle: Es kann praktisch sein, sich frühere Versionen angucken zu können
- Empfehlung: git (Programm für Versionskontrolle) + GitHub (Online-Plattform für git-basierte Projekte)



<https://github.com/>

# Backup und Versionskontrolle

- Relevanz von Backups: Offensichtlich!
- Relevanz von Versionskontrolle: Es kann praktisch sein, sich frühere Versionen angucken zu können
- Empfehlung: git (Programm für Versionskontrolle) + GitHub (Online-Plattform für git-basierte Projekte)
  - Einbindung in RStudio



<https://github.com/>

# Backup und Versionskontrolle

- Relevanz von Backups: Offensichtlich!
- Relevanz von Versionskontrolle: Es kann praktisch sein, sich frühere Versionen angucken zu können
- Empfehlung: git (Programm für Versionskontrolle) + GitHub (Online-Plattform für git-basierte Projekte)
  - Einbindung in RStudio
  - Von verschiedenen Computern aus am Projekt arbeiten können



<https://github.com/>

# Backup und Versionskontrolle

- Relevanz von Backups: Offensichtlich!
- Relevanz von Versionskontrolle: Es kann praktisch sein, sich frühere Versionen angucken zu können
- Empfehlung: git (Programm für Versionskontrolle) + GitHub (Online-Plattform für git-basierte Projekte)
  - Einbindung in RStudio
  - Von verschiedenen Computern aus am Projekt arbeiten können
  - Gemeinsames Arbeiten an einem Projekt möglich



<https://github.com/>

# Backup und Versionskontrolle

- Relevanz von Backups: Offensichtlich!
- Relevanz von Versionskontrolle: Es kann praktisch sein, sich frühere Versionen angucken zu können
- Empfehlung: git (Programm für Versionskontrolle) + GitHub (Online-Plattform für git-basierte Projekte)
  - Einbindung in RStudio
  - Von verschiedenen Computern aus am Projekt arbeiten können
  - Gemeinsames Arbeiten an einem Projekt möglich
  - Projekte öffentlich oder privat



<https://github.com/>

# Backup und Versionskontrolle

- Relevanz von Backups: Offensichtlich!
- Relevanz von Versionskontrolle: Es kann praktisch sein, sich frühere Versionen angucken zu können
- Empfehlung: git (Programm für Versionskontrolle) + GitHub (Online-Plattform für git-basierte Projekte)
  - Einbindung in RStudio
  - Von verschiedenen Computern aus am Projekt arbeiten können
  - Gemeinsames Arbeiten an einem Projekt möglich
  - Projekte öffentlich oder privat
  - Websites möglich (Beispiel: <https://kerstinpierick.github.io/Studentinnen-in-MINT-Faechern/>)



<https://github.com/>

## Weiterführendes

## Nützliches Paket: `swirl`

## Interaktive R-Kurse

Console Terminal x

~/Desktop/R\_Projekte/2019/RWorkshop\_Campus2019/ ↵

...

```
| ======  
| Now, create a new variable called my_div that gets the value of z divided by my_sqrt. | 63%  
> my_div <- z / my_sqrt  
  
| All that hard work is paying off!  
  
| ======  
| Which statement do you think is true? | 66%  
1: my_div is undefined  
2: The first element of my_div is equal to the first element of z divided by the first element of my_sqrt, and so on...  
3: my_div is a single number (i.e a vector of length 1)  
  
Selection: 2  
  
| Keep up the great work!  
  
| ======  
| Go ahead and print the contents of my_div. | 68%  
> my_div  
[1] 3.478505 3.181981 2.146460  
  
| All that hard work is paying off!  
  
| ======  
| When given two vectors of the same length, R simply performs the specified arithmetic operation ('+', '-' , '*', etc.) element-by-element. If the vectors are of different lengths, R 'recycles' the shorter | 71%
```

# Vernetzung

- Twitter: # rstats

# Vernetzung

- Twitter: # rstats
- -bloggers <https://www.r-bloggers.com/>

# Vernetzung

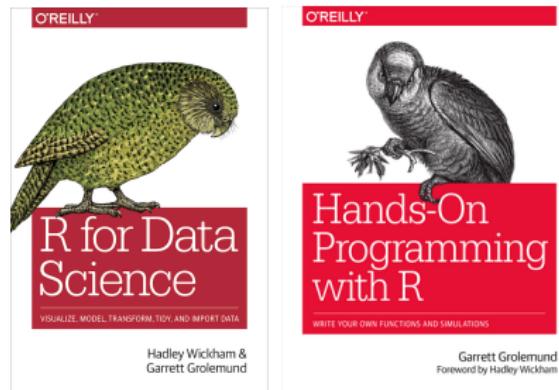
- Twitter: # rstats
- -bloggers <https://www.r-bloggers.com/>
- RLadies <https://rladies.org/>

# Vernetzung

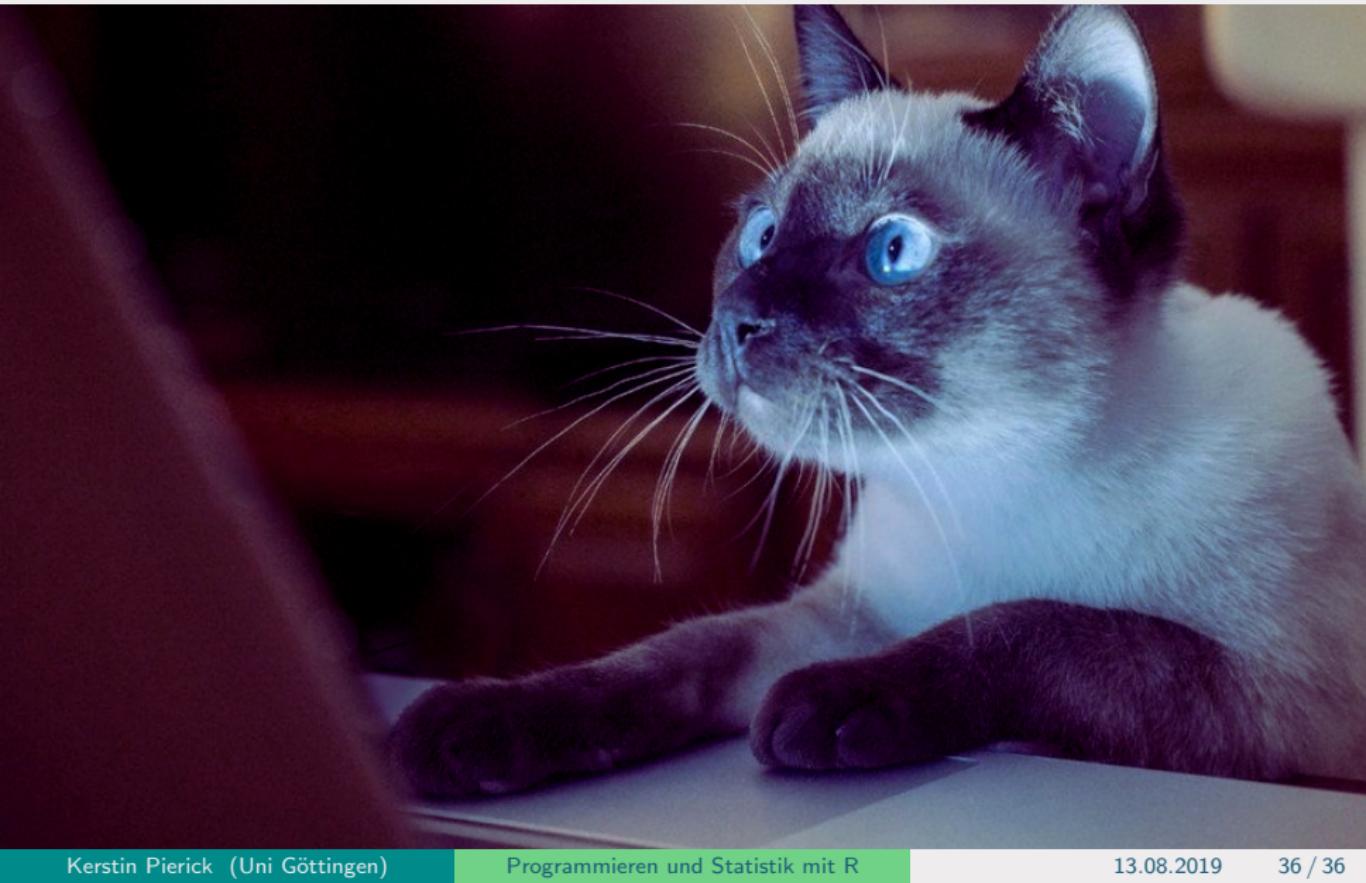
- Twitter: # rstats
- -bloggers <https://www.r-bloggers.com/>
- -Ladies <https://rladies.org/>
- Data Science Meetups in deiner Stadt

# Literatur

- Hadley Wickham und Garrett Grolemund: *R for Data Science*. (O'Reilly 2017) <https://r4ds.had.co.nz/>
- Garrett Grolemund: *Hands-on programming with R*. (O'Reilly 2014) <https://rstudio-education.github.io/hopr/>



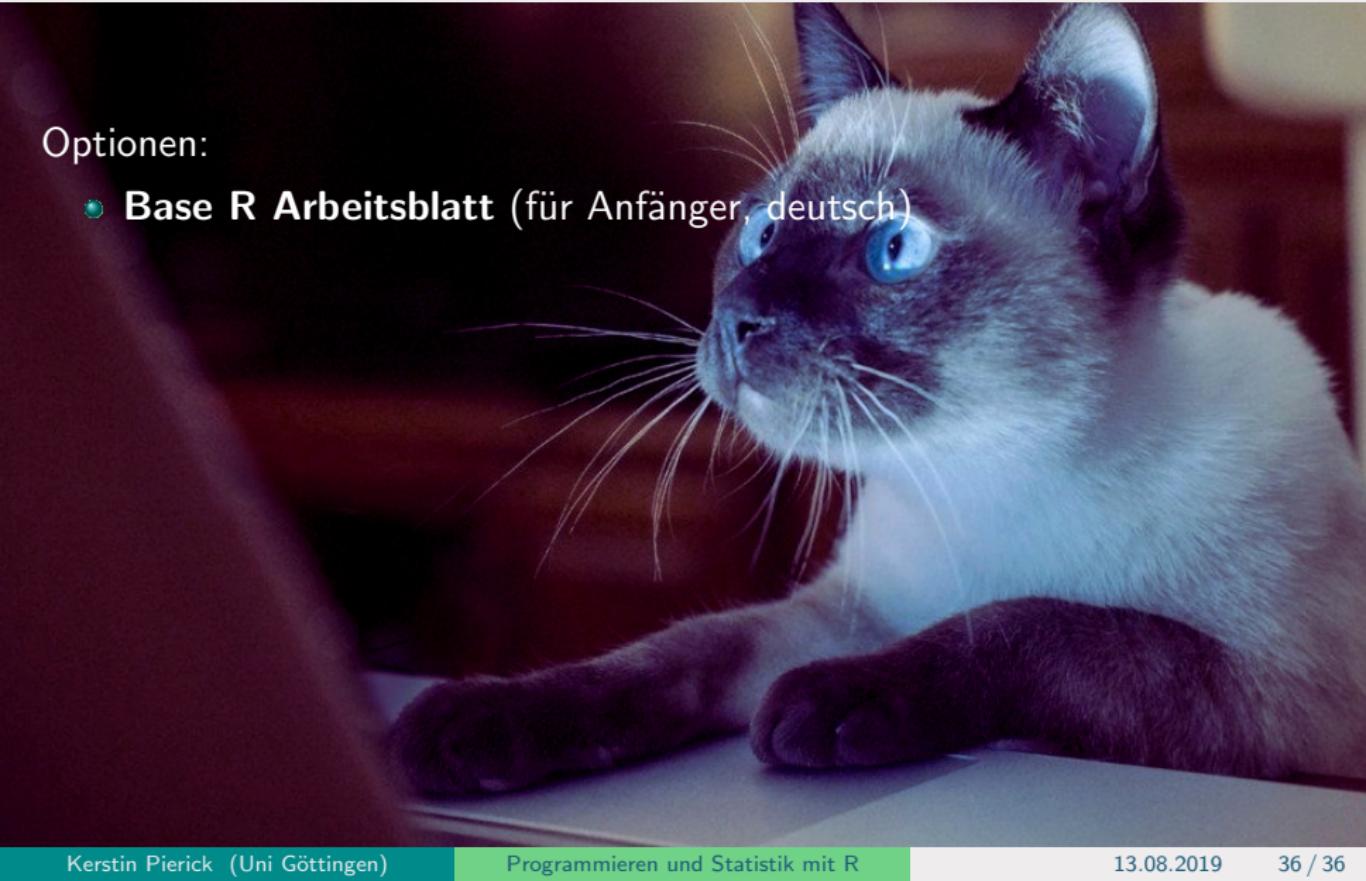
# Praktischer Teil



# Praktischer Teil

Optionen:

- **Base R Arbeitsblatt** (für Anfänger, deutsch)

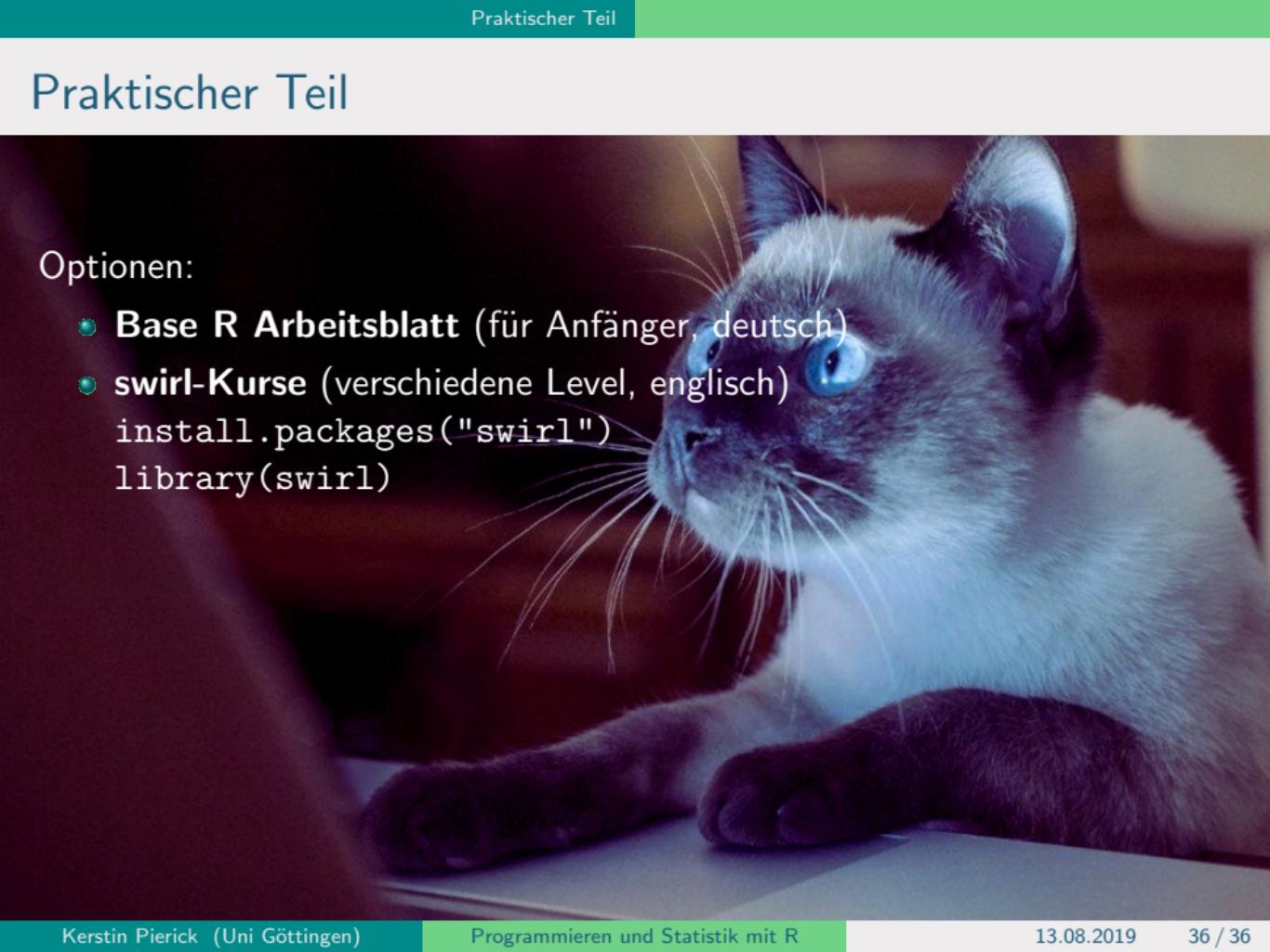


# Praktischer Teil

Optionen:

- **Base R Arbeitsblatt** (für Anfänger, deutsch)
- **swirl-Kurse** (verschiedene Level, englisch)

```
install.packages("swirl")  
library(swirl)
```



# Praktischer Teil

Optionen:

- **Base R Arbeitsblatt** (für Anfänger, deutsch)  
<https://r4ds.had.co.nz/data-visualisation.html>
- **swirl-Kurse** (verschiedene Level, englisch)  
`install.packages("swirl")`  
`library(swirl)`
- **ggplot2 lernen** (Vorkenntnisse nötig, englisch)  
<https://r4ds.had.co.nz/data-visualisation.html>

# Praktischer Teil

Optionen:

- **Base R Arbeitsblatt** (für Anfänger, deutsch)
- **swirl-Kurse** (verschiedene Level, englisch)  
`install.packages("swirl")`  
`library(swirl)`
- **ggplot2 lernen** (Vorkenntnisse nötig, englisch)  
<https://r4ds.had.co.nz/data-visualisation.html>
- **Eigenes Projekt**

# Praktischer Teil

Optionen:

- **Base R Arbeitsblatt** (für Anfänger, deutsch)
- **swirl-Kurse** (verschiedene Level, englisch)  
`install.packages("swirl")`  
`library(swirl)`
- **ggplot2 lernen** (Vorkenntnisse nötig, englisch)  
<https://r4ds.had.co.nz/data-visualisation.html>
- **Eigenes Projekt**

Alle Materialien auf <https://github.com/KerstinPierick/RWorkshop>