



Analyse und Dokumentation

BSc Psychologie SoSe 2025

Belinda Fleischmann und Dirk Ostwald

	Gruppe 1/2	Gruppe 3	Format	Thema
1	Do, 10.04	Fr, 11.04	Seminar	(1) Quarto, Zotero, Tidyverse
2	Do, 17.04	Fr, 25.04	Seminar	(2) Ethik und Ethische Formalitäten
3	Do, 24.04	Fr, 02.05	Seminar	(3) Wissenschaftliche Berichte
4	Mi, 30.04	Fr, 09.05	Seminar	(4) Offenheit und Transparenz
5	Do, 08.05	Fr, 16.05	Praxisseminar	Offene Übung
6	Do, 15.05	Fr, 23.05	Präsentationen	Einfache Lineare Regression
7	Mi, 21.05	Fr, 30.05	Präsentationen	Korrelation
8	Do, 05.06	Fr, 06.06	Präsentationen	Einstichproben-T-Test
9	Do, 12.06	Fr, 13.06	Präsentationen	Zweistichproben-T-Test
10	Do, 19.06	Fr, 20.06	Präsentationen	Einfaktorielle Varianzanalyse
11	Do, 26.06	Fr, 27.06	Präsentationen	Zweifaktorielle Varianzanalyse
12	Do, 03.07	Fr, 04.07	Präsentationen	Multiple Regression
13	Do, 10.07	Fr, 11.07	Präsentationen	Kovarianzanalyse
	Juli		Klausurtermin	

(1) Quarto, Zotero, Tidyverse

Quarto

Zotero

Tidyverse

Quarto

Zotero

Tidyverse

Welcome to Quarto

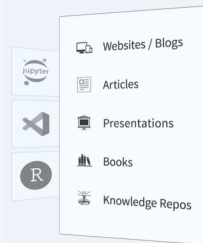
An open-source scientific and technical publishing system

- Author using [Jupyter](#) notebooks or with plain text markdown in your favorite editor.
- Create dynamic content with [Python](#), [R](#), [Julia](#), and [Observable](#).
- Publish reproducible, production quality articles, presentations, websites, blogs, and books in HTML, PDF, MS Word, ePub, and more.
- Share knowledge and insights organization-wide by publishing to [Posit Connect](#), [Confluence](#), or other publishing systems.
- Write using [Pandoc](#) markdown, including equations, citations, crossrefs, figure panels, callouts, advanced layout, and more.

Analyze. Share. Reproduce. You have a story to tell with data—tell it with Quarto.

Get Started

Guide



[Quarto Website](#)

Was ist Quarto?

- Ein seit 2022 verfügbares freies wissenschaftlich-technisches Publikationssystem
- Eine Weiterentwicklung von [RMarkdown](#) und [RBookdown](#) durch [Posit](#)
- RMarkdown/RBookdown sind RStudio Adaptationen von [Markdown](#) und [Jupyter Notebooks](#)
- Allgemeines Ziel ist hier die einfache Integration von ausführbarem Programmiercode in ein ansprechendes Text-, Tabellen- und Abbildungslayout für Web- und Printdokumente.
- Quarto nutzt [Markdown](#) und [Latex](#) für Layoutprozesse.
- Quarto nutzt [Pandoc](#) für multiple Outputformate (.html, .docx, .pdf, etc.)
- Quarto läuft smoother und schneller als RMarkdown und RBookdown.

Installation von Quarto

Get Started

Tutorial: Hello, Quarto
Tutorial: Computations
Tutorial: Authoring

Get Started

Install Quarto, then check out the tutorials to learn the basics.

Step 1

Install Quarto

Find your operating system in the table below


Platform	Download	Size	SHA-256
Ubuntu 18+/Debian 10+	quarto-1.4.554-linux-amd64.deb	111.82 MB	7b57d62
Linux x86 Tarball	quarto-1.4.554-linux-amd64.tar.gz	113.04 MB	f0d1283f
Linux Arm64	quarto-1.4.554-linux-arm64.deb	112.52 MB	4201e1b
Linux Arm64 Tarball	quarto-1.4.554-linux-arm64.tar.gz	113.6 MB	43c788d
RHEL 7 Tarball	quarto-1.4.554-linux-rhel7-amd64.tar.gz	113.4 MB	7d5264b
Mac OS	quarto-1.4.554-macos.pkg	186.2 MB	ab6a44c
Windows	quarto-1.4.554-win.msi	108.89 MB	f6d281d
Release notes and more downloads...			





Step 2

Choose your tool and get started



Quarto Website Installation


[Overview](#)[Get Started](#)[Guide](#)[Extensions](#)[Reference](#)[Gallery](#)[Blog](#)[Help](#)





[Get Started](#)
[Tutorial: Hello, Quarto](#)
[Tutorial: Computations](#)
[Tutorial: Authoring](#)


Tutorial: Hello, Quarto


Choose your tool


VS Code


Jupyter


RStudio


Neovim


Editor

Overview

In this tutorial we'll show you how to use Quarto with VS Code. Before getting started, you should install the [Quarto VS Code Extension](#), which includes many tools that enhance working with Quarto, including:

- Integrated render and preview for Quarto documents.
- Syntax highlighting for markdown and embedded languages
- Completion and diagnostics for YAML options
- Completion for embedded languages (e.g. Python, R, Julia, etc.)
- Commands and key-bindings for running cells and selected lines.


You can install the Quarto extension from within the **Extensions** tab in VS Code, from the [Extension Marketplace](#), the [Open VSX Registry](#) or directly from a [VSIX extension file](#).

Note

This tutorial focuses on editing plain text Quarto `.qmd` files in VS Code. Depending on your preferences and the task at hand there are two other editing modes available for Quarto documents: the [Visual Editor](#) and the [Notebook Editor](#). For the purposes of learning we recommend you work through this tutorial using the VS Code text editor, then after you've mastered the basics explore using the other editing modes.

Basic Workflow

Quarto `.qmd` files contain a combination of markdown and executable code cells. Here's what it might look like in VS Code to edit and preview a `.qmd` file:



On this page

- [Overview](#)
- [Basic Workflow](#)
- [Render and Preview](#)
- [YAML Options](#)
- [Markdown](#)
- [Code Cells](#)
- [External Preview](#)
- [Next Up](#)

[Edit this page](#)
[Report an issue](#)

Quarto Website Tutorial: VS Code

Markdown

- Eine Markup Language (Auszeichnungssprache) zur Erzeugung formatierten Texts
- Eine HTML Alternative zur Erstellung von Webseiten etc. mithilfe einfacher Texteditoren
- Von John Gruber und Aaron Swartz 2004 mit dem Ziel hoher Lesbarkeit entwickelt

Text using Markdown syntax	Corresponding HTML produced by a Markdown processor	Text viewed in a browser
<p>Heading *****</p> <p>Sub-heading -----</p> <p># Alternative heading</p> <p>## Alternative sub-heading</p> <p>Paragraphs are separated by a blank line.</p> <p>Two spaces at the end of a line produce a line break.</p>	<pre><h1>Heading</h1> <h2>Sub-heading</h2> <h1>Alternative heading</h1> <h2>Alternative sub-heading</h2> <p>Paragraphs are separated by a blank line.</p> <p>Two spaces at the end of a line
 produce a line break.</p></pre>	<p>Heading</p> <p>Sub-heading</p> <p>Alternative heading</p> <p>Alternative sub-heading</p> <p>Paragraphs are separated by a blank line.</p> <p>Two spaces at the end of a line produce a line break.</p>
<p>Text attributes <i>_italic_</i>, **bold**, <code>'monospace'</code>.</p> <p>Horizontal rule:</p> <p>---</p>	<pre><p>Text attributes italic, bold, <code>monospace</code>.</p> <p>Horizontal rule:</p> <hr /></pre>	<p>Text attributes <i>italic</i>, bold, <code>monospace</code>.</p> <p>Horizontal rule:</p> <hr/>

- Ein Softwarepaket zur Vereinfachung von TeX
- TeX ist ein von Donald E. Knuth ab 1977 entwickeltes Textsatzsystem mit Makrosprache
- LaTeX wurde von Leslie Lamport Anfang 1984 entwickelt
- LaTeX ist insbesondere für mathematische Berichte und Präsentationen (Beamer) nützlich

```
\footnotesize
\begin{theorem}[Datenverteilung des Allgemeinen Linearen Modells]
\justifying
\normalfont
Es sei
\begin{equation}
\upsilon = X\beta + \varepsilon \text{ mit } \varepsilon \sim N(0_n, \sigma^2 I_n)
\end{equation}
das ALM. Dann gilt
\begin{equation}
\upsilon \sim N(\mu, \sigma^2 I_n) \text{ mit } \mu := X\beta \in \mathbb{R}^n.
\end{equation}
\end{theorem}
```




Theorem (Datenverteilung des Allgemeinen Linearen Modells)





Es sei

$$v = X\beta + \varepsilon \text{ mit } \varepsilon \sim N(0_n, \sigma^2 I_n) \quad (7)$$

das ALM. Dann gilt

$$v \sim N(\mu, \sigma^2 I_n) \text{ mit } \mu := X\beta \in \mathbb{R}^n. \quad (8)$$

 Overview Get Started Guide Extensions Reference Gallery Blog Help



Guide

Authoring

Computations

Tools

Documents

Presentations

Dashboards

Websites

Books

Manuscripts

Interactivity

Publishing

Projects

Advanced

Guide

Comprehensive guide to using Quarto. If you are just starting out, you may want to explore the tutorials to learn the basics.

Authoring

Create content with markdown

Markdown Basics

Figures

Tables

Diagrams

Citations & Footnotes

Cross References

Article Layout

Computations

Execute code and display its output

Using Python

Using R

Using Julia

Using Observable

Execution Options

Parameters

Tools

Use your favorite tools with Quarto

JupyterLab

RStudio IDE

V5 Code

Neovim

Text Editors

Visual Editor

Documents

Generate output in many formats

HTML

PDF

MS Word

Typst

Markdown

All Formats

Presentations

Present code and technical content

Presentation Basics

Reveal.js (HTML)

PowerPoint (Office)

Beamer (PDF)

Dashboards

Publish data with dashboards

Dashboard Basics

Layout

Data Display

Interactivity

Deployment

Websites

Create websites and blogs

Creating a Website

Website Navigation

Creating a Blog

Website Search

Website Listings

Books

Create books and manuscripts

Creating a Book

Book Structure

Book Crossrefs

Customizing Output

Manuscripts

Write and publish notebook-first scholarly articles

Getting Started

Authoring Manuscripts

Publishing Manuscripts

Using Manuscripts

Interactivity

Engage readers with interactivity

Overview

Observable JS

Shiny

Widgets

Component Layout

Publishing

Publishing documents and sites

Publishing Basics

Quarto Pub

GitHub Pages

Posit Connect

Posit Cloud

Netlify

Confluence

Other Services

Projects

Scale up your work with projects

Project Basics

Managing Execution

Project Profiles

Environment Variables

Project Scripts

Virtual Environments

Quarto Website Guide

Analyse und Dokumentation | © 2025 Belinda Fleischmann & Dirk Ostwald CC BY 4.0 | Folie 12

Quarto Beispiel

```
---
title: "Quarto Demonstration"
author: "Toni Demo"
date: today
format: pdf
---

# Überschrift zu Kapitel 1.

Hier steht der Text für Kapitel 1. Darin könnte auch eine Abbildung enthalten sein.

{width="10%"}

## Überschrift zum Unterkapitel 1.1

Hier steht der Text für Unterkapitel 1.1. Manche Worte möchte ich fett und manche Worte kursiv, und Befehle
in monospace schreiben. Mögliche Farben möchte ich mit Stichpunkten auflisten.

* \textcolor{blue}{blau}
* \textcolor{green}{grün}
* \textcolor{red}{rot}
* \textcolor{gray}{grau}

Wenn wir mathematische Ausdrücke mit Dollarzeichen umrahmen, werden sie mithilfe von LaTeX formatiert.
So können wir z.B. die Verteilung eines Zufallsvektors formal mit  $\epsilon \sim N(\mu, \sigma^2 I_n)$  mit
 $\mu := X\beta$  in  $\mathbb{R}^n$  aufschreiben.
```

Quarto Demonstration

Belinda Fleischmann

2024-05-02

Überschrift zu Kapitel 1.

Hier steht der Text für Kapitel 1. Darin könnte auch eine Abbildung enthalten sein.



Überschrift zum Unterkapitel 1.1

Hier steht der Text für Unterkapitel 1.1. Manche Worte möchte ich **fett** und manche Worte *kursiv*. und Befehle in `monospace` schreiben. Mögliche Farben möchte ich mit Stichpunkten auflisten.

- blau
- grün
- rot
- grau

Wenn wir mathematische Ausdrücke mit Dollarzeichen umrahmen, werden sie mithilfe von LaTeX formatiert. So können wir z.B. die Verteilung eines Zufallsvektors formal mit $v \sim N(\mu, \sigma^2 I_n)$ mit $\mu := X\beta \in \mathbb{R}^n$ aufschreiben.

Beispielbericht

Beispielpräsentation

Quarto

Zotero

tidyverse

Was ist ein Reference Manager?

- Reference Manager sind Literaturverwaltungsprogramme
- Reference Manager unterstützen Zitationen und das Erstellen von Literaturverzeichnissen
- Zitierstile können automatisch auf bestimmte Spezifikationen (z.B. APA) eingestellt werden
- Reference Manager dienen auch als digitale Bibliotheken
- Kommerzielle Reference Manager sind z.B. EndNote, Citavi, Mendeley und Papers
- Kostenlose/Freemium Reference Manager sind z.B. [JabRef](#) und [Zotero](#)
- Eine Integration in Quarto erlaubt z.B. der Export der eigenen Library in das [BibTeX](#) Format.

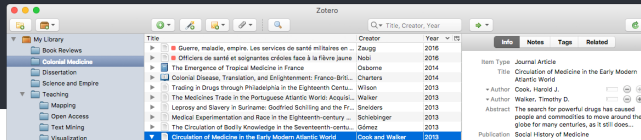
Your personal research assistant

Zotero is a free, easy-to-use tool to help you collect, organize, annotate, cite, and share research.

Download

Available for Mac, Windows, Linux, and iOS

Just need to create a quick bibliography? Try [ZoteroBib](#)



[Zotero Website](#)

[Zotero Documentation](#)

Quarto

Zotero

tidyverse

Tidyverse

[Packages](#) [Blog](#) [Learn](#) [Help](#) [Contribute](#)



R packages for data science

The tidyverse is an opinionated **collection of R packages** designed for data science. All packages share an underlying design philosophy, grammar, and data structures.

Install the complete tidyverse with:

```
install.packages("tidyverse")
```

Learn the tidyverse

[Tidyverse](#)

[Cheat Sheets](#)

Data transformation with dplyr : : CHEATSHEET



dplyr functions work with pipes and expect **tidy data**. In tidy data:

Each **variable** is in its own column

Each **observation**, or **case**, is in its own row

x > **f(y)** becomes **f(x, y)**

pipes

Summarize Cases

Apply **summary functions** to columns to create a new table of summary statistics. Summary functions take vectors as input and return one value (see back).

summary function

```
summarize(data, ...)
# Compute table of summaries.
mtcars >-> summarize(avg = mean(mpg))

count(data, ..., wt = NULL, sort = FALSE, name = NULL)
# Count number of rows in each group defined by the variables in ... Also tally(), add_count(), add_tally()
mtcars >-> count(cyl)
```

Group Cases

Use **group_by()** (data, ..., add = FALSE, drop = TRUE) to create a "grouped" copy of a table grouped by columns in ... dplyr functions will manipulate each "group" separately and combine the results.

```
mtcars >-> group_by(cyl) >-> summarize(avg = mean(mpg))
```

Use **rowwise()** (data, ...) to group data into individual rows. dplyr functions will compute results for each row. Also apply functions to list-columns. See tidy cheat sheet for list-column workflow.

```
starwars >-> rowwise() >-> mutate(film_count = length(film))
```

ungroup() (x, ...) Returns ungrouped copy of table.
g_mtcars <- mtcars %>% group_by(cyl)
ungroup(g_mtcars)

Manipulate Cases

EXTRACT CASES

Row functions return a subset of rows as a new table.

```
filter(data, ..., preserve = FALSE)
# Extract rows that meet logical criteria.
mtcars >-> filter(mpg > 20)

distinct(data, ..., keep_all = FALSE)
# Remove rows with duplicate values.
mtcars >-> distinct(gear)

slice(data, ..., preserve = FALSE)
# Select rows by position.
mtcars >-> slice(10:15)

slice_sample(data, ..., n, prop, weight, by = NULL, replace = FALSE)
# Randomly select rows. Use n to select a number of rows and prop to select a fraction of rows.
mtcars >-> slice_sample(n = 5, replace = TRUE)

slice_min(data, order_by, ..., n, prop, with_ties = TRUE) and slice_max()
# Select rows with the lowest and highest values.
mtcars >-> slice_min(mpg, prop = 0.25)

slice_head(data, ..., n, prop) and slice_tail()
# Select the first or last rows.
mtcars >-> slice_head(n = 3)
```

Logical and boolean operators to use with filter()

==	<	<=	is.na()	%in%		xor()
!=	>	>=	!is.na()	!	&	

See [7bases:Logic and Comparison](#) for help.

ARRANGE CASES

```
arrange(data, ..., by_group = FALSE)
# Order rows by values of a column or columns (low to high), use with desc() to order from high to low.
mtcars >-> arrange(mpg)
mtcars >-> arrange(desc(mpg))
```

ADD CASES

```
add_row(data, ..., before = NULL, after = NULL)
# Add one or more rows to a table.
cars >-> add_row(speed = 1, dist = 1)
```

Manipulate Variables

EXTRACT VARIABLES

Column functions return a set of columns as a new vector or table.

```
pull(data, var = 1, name = NULL, ...)
# Extract column values as a vector, by name or index.
mtcars >-> pull(wt)

select(data, ...)
# Extract columns as a table.
mtcars >-> select(mpg, wt)

relocate(data, ..., before = NULL, after = NULL)
# Move columns to new position.
mtcars >-> relocate(mpg, cyl, after = last_col())
```

Use these helpers with select() and across()

e.g. mtcars >-> select(mpg:cyl)
contains() (match) **num_range()** (prefix, range) **l_e.g. mpg:cyl**
ends_with() (match) **all_of()** (any, of(x, ..., vars) **l_e.g. lpg**
starts_with() (match) **matches()** (match) **everything()**

MANIPULATE MULTIPLE VARIABLES AT ONCE

```
df <- tibble(x_1 = c(1, 2), x_2 = c(3, 4), y = c(4, 5))

across(cols, funs, ..., names = NULL)
# Summarize or mutate multiple columns in the same way.
df >-> summarize(across(everything(), mean))

c_across(cols)
# Compute across columns in row-wise data.
df >-> rowwise() >-> mutate(gm_total = sum(c_across(1:2)))
```

MAKE NEW VARIABLES

Apply **vectorized functions** to columns. Vectorized functions take vectors as input and return vectors of the same length as output (see back).

```
mutate(data, ..., keep = "all", before = NULL, after = NULL)
# Compute new column(s). Also add_column()
mtcars >-> mutate(gpm = 1 / mpg)
mtcars >-> mutate(gpm = 1 / mpg, keep = "none")

rename(data, ...)
# Rename columns. Use rename_with() to rename with a function.
mtcars >-> rename(miles_per_gallon = mpg)
```



Datenvorverarbeitung mit dplyr

```
D <- read.table("Daten_1.csv", sep = ",", header = TRUE) # Daten einlesen
```

Variable_1	Variable_2	Variable_3
34.87	34.61	33.56
32.16	22.89	15.75
33.95	31.82	28.83
28.78	25.91	20.04
30.13	26.83	22.00
30.50	26.50	24.42
32.48	26.92	22.96
31.66	31.84	28.83
32.76	33.00	33.28
31.60	26.77	21.21
32.44	28.55	28.63
29.48	25.33	24.19
31.24	28.97	25.18
34.33	31.31	28.22
31.56	27.11	22.92
31.87	30.95	30.30
27.07	21.94	17.60
29.36	25.41	19.32
36.07	33.56	33.41
33.03	28.81	26.58
33.12	32.20	29.44

Datenvorverarbeitung mit dplyr

Der Pipe operater %>% oder |> ermöglicht es, Funktionen in einer Reihe nacheinander auszuführen.

Mit der R-Funktion mutate() können wir neue Spalten erzeugen (auch als Funktionen bestehender Spalten).

```
library(dplyr)
n <- nrow(D)
D_processed <- D %>%
  mutate(ID = seq(n)) %>%
  mutate(Summe = Variable_1 + Variable_2 + Variable_3)
```

Anzahl Beobachtungen
D wird an nächste Funktion übergeben
ID-Spalte hinzufügen
Summen-Spalte hinzufügen

Variable_1	Variable_2	Variable_3	ID	Summe
34.87	34.61	33.56	1	103.04
32.16	22.89	15.75	2	70.79
33.95	31.82	28.83	3	94.60
28.78	25.91	20.04	4	74.74
30.13	26.83	22.00	5	78.96
30.50	26.50	24.42	6	81.42
32.48	26.92	22.96	7	82.37
31.66	31.84	28.83	8	92.34
32.76	33.00	33.28	9	99.05
31.60	26.77	21.21	10	79.58
32.44	28.55	28.63	11	89.62
29.48	25.33	24.19	12	79.00
31.24	28.97	25.18	13	85.40
34.33	31.31	28.22	14	93.86
31.56	27.11	22.92	15	81.59
31.87	30.95	30.30	16	93.12
27.07	21.94	17.60	17	66.61
29.36	25.41	19.32	18	74.09
36.07	33.56	33.41	19	103.04
33.03	28.81	26.58	20	88.42
33.12	32.20	29.44	21	94.75

Datenvorverarbeitung mit dplyr

Mit der R-Funktion `filter()` können wir Zeilen gemäß bestimmten Bedingungen auswählen.

```
D_selected <- D_processed %>%  
  filter(ID %in% 1:10) %>%           # Auswahl der IDs 1-10  
  filter(Summe > 90)                 # Selektion der Beobachtungen mit Summe > 90
```

Variable_1	Variable_2	Variable_3	ID	Summe
34.87	34.61	33.56	1	103.04
33.95	31.82	28.83	3	94.60
31.66	31.84	28.83	8	92.34
32.76	33.00	33.28	9	99.05

Plotten mit ggplot2

Data visualization with ggplot2 : : CHEATSHEET



Basics

ggplot2 is based on the **grammar of graphics**, the idea that you can build every graph from the same components: a **data set**, a **coordinate system**, and **geoms**—visual marks that represent data points.



To display values, map variables in the data to visual properties of the geom (**aesthetics**) like **size**, **color**, and **x** and **y** locations.



Complete the template below to build a graph.

```
ggplot (data = DATA) +  
  GEOM FUNCTION (mapping = aes(MAPPINGS),  
  size = SIZE, position = POSITION),  
  COORDINATE FUNCTION )  
+  
  FACTOR FUNCTION +  
  SCALE FUNCTION +  
  THEME FUNCTION
```

ggplot(data = mpg, aes(x = cty, y = hwy)) Begins a plot that you finish by adding layers to. Add one geom function per layer.

last_plot() Returns the last plot.

ggsave("plot.png", width = 5, height = 5) Saves last plot as 5 x 5 file named "plot.png" in working directory. Matches file type to file extension.

Aes

Common aesthetic values.
color and fill - string ("red", "RRRRGGGB")

linetype - integer or string (0 = "blank", 1 = "solid", 2 = "dashed", 3 = "dotted", 4 = "longdash", 5 = "longdash", 6 = "twodash")

size - integer (line width in mm)

shape - integer/shape name or a single character ("a")



Geoms

Use a geom function to represent data points, use the geom's aesthetic properties to represent variables.

Each function returns a layer.

GRAPHICAL PRIMITIVES

a <- ggplot(economics, aes(date, unempLOY))
b <- ggplot(aes(x = long, y = lat))

a = geom_blank() and a = expand_limits() ensure limits include values across all plots.

b = geom_curve(aes(xend = lat + 1, xend = long + 1, curvature = 1) - x, yend, yend, alpha, angle, color, curvature, linetype, size)

a = geom_path(linewidth = "thin")
linetype = "round", linewidth = 2)

a = geom_polygon(aes(alpha = 50)) - x, y, alpha, color, fill, group, linewidth, linetype, size

b = geom_rect(aes(xmin = long, ymin = lat, xmax = long + 2, ymax = lat + 2) - xmin, ymin, ymax, ymin, alpha, color, fill, linetype, size)

a = geom_ribbon(aes(ymin = unempLOY - 900, ymax = unempLOY + 900)) - x, ymin, ymax, alpha, color, fill, group, linetype, size

LINE SEGMENTS

common aesthetics: x, y, alpha, color, linetype, size

b = geom_abline(aes(intercept = 0, slope = 1))
b = geom_hline(aes(yintercept = lat))

b = geom_vline(aes(xintercept = long))

b = geom_segment(aes(xend = lat + 1, xend = long + 1))
b = geom_spoke(aes(angle = 1:155, radius = 2))

ONE VARIABLE continuous

c <- ggplot(mpg, aes(hwy)); c2 <- ggplot(mpg)

a = geom_area(stat = "bin")
x, y, alpha, color, fill, linetype, size

c = geom_density(kernel = "gaussian")
x, y, alpha, color, fill, group, linetype, size, weight

c = geom_dotplot()
x, y, alpha, color, fill

c = geom_freqpoly()
x, y, alpha, color, group, linetype, size

c = geom_histogram(bins = 3)
x, y, alpha, color, fill, linetype, size, weight

c2 = geom_qq(aes(sample = hwy))
x, y, alpha, color, fill, linetype, size, weight

discrete
d <- ggplot(mpg, aes(v))

d = geom_bar()
x, alpha, color, fill, linetype, size, weight

TWO VARIABLES both continuous

e <- ggplot(mpg, aes(cty, hwy))

a = geom_label(aes(label = cty, nudje, x = 1, nudje, y = 2) - x, y, label, alpha, angle, color, family, fontsize, hjust, linewidth, size, vjust)

e = geom_point()
x, y, alpha, color, fill, shape, size, stroke

e = geom_quantile()
x, y, alpha, color, group, linetype, size, weight

e = geom_rug(aes(l = "lat")
x, y, alpha, color, linetype, size

e = geom_smooth(method = lm)
x, y, alpha, color, fill, group, linetype, size, weight

e = geom_text(aes(label = cty, nudje, x = 1, nudje, y = 2) - x, y, label, alpha, angle, color, family, fontsize, hjust, linewidth, size, vjust)

one discrete, one continuous

f <- ggplot(mpg, aes(class, hwy))

f = geom_col()
x, y, alpha, color, fill, group, linetype, size

f = geom_boxplot()
x, y, lower, middle, upper, ymin, ymax, alpha, color, fill, group, linetype, shape, size, weight

f = geom_dotplot(bins = "y", stackdir = "center")
x, y, alpha, color, fill, group

f = geom_violin(aes(scale = "area")
x, y, alpha, color, fill, group, linetype, size, weight

both discrete

g <- ggplot(diamonds, aes(carat, color))

g = geom_count()
x, y, alpha, color, fill, shape, size, stroke

g = geom_jitter(height = 2, width = 2)
x, y, alpha, color, fill, shape, size

THREE VARIABLES

sealsize <- ggplot(aes(sex, origin, delta, lat2)) { c <- ggplot(sealsize, aes(long, lat))

l = geom_contour(aes(z))
x, y, alpha, color, group, linetype, size, weight

l = geom_contour_filled(aes(fill = z))
x, y, alpha, color, fill, group, linetype, size, subgroup

continuous bivariate distribution

h <- ggplot(diamonds, aes(carat, price))

h = geom_bin2d(binswidth = c(0.25, 500))
x, y, alpha, color, fill, linetype, size, weight

h = geom_density_2d()
x, y, alpha, color, group, linetype, size

h = geom_hex()
x, y, alpha, color, fill, size

continuous function

i <- ggplot(economics, aes(date, unempLOY))

i = geom_area()
x, y, alpha, color, fill, linetype, size

i = geom_line()
x, y, alpha, color, group, linetype, size

i = geom_step(direction = "hv")
x, y, alpha, color, group, linetype, size

visualizing error

d <- data.frame(p = c("A", "B"), fit = 4.5, se = 1.2)
j <- ggplot(d, aes(p, fit, ymin = fit - se, ymax = fit + se))

j = geom_cransfordation - 2) - x, y, ymax, ymin, alpha, color, fill, group, linetype, size

j = geom_errorbar() - x, ymax, ymin, alpha, color, group, linetype, size, width

j = geom_errorbarh() - x, ymax, ymin, alpha, color, group, linetype, size, width

j = geom_linerange()
x, ymin, ymax, alpha, color, group, linetype, size

j = geom_pointrange() - x, y, ymin, ymax, alpha, color, fill, group, linetype, shape, size

maps

map <- data.frame(murder = USArrests\$Murder, state = tolower(rownames(USArrests)))

map <- map_data("state")
k <- ggplot(map, aes(l = murder))

k = geom_map(aes(map_id = state), map = map)

map_id = map\$long, y = map\$lat
map_id = alpha, color, fill, linetype, size

Plotten mit ggplot2

Beispieldatensatz

```
library(dplyr) # Für Pipe (%>%), mutate()

# Daten vorbereiten
D <- read.table("Daten_2.csv", sep = ",", header = TRUE) # Daten einlesen
n_pat <- nrow(D) # Anzahl Patientinnen
D_processed <- D %>% # PatientIn ID hinzufügen
  mutate(PatientIn = seq(n_pat))
```

Die ersten 12 Zeilen des Dataframes:

DUR	BDI	PatientIn
1.37	9	1
2.18	8	2
1.16	11	3
3.60	0	4
2.33	8	5
1.18	7	6
2.49	3	7
2.74	7	8
2.58	3	9
1.69	11	10
3.51	9	11
2.39	7	12

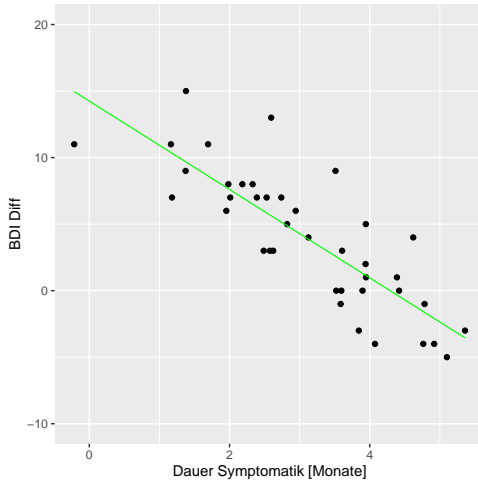
Plotten mit ggplot2

```
library(ggplot2)                                # Für ggplot()

# Visualisierung
ggplot(
  data = D_processed,                           # Daten
  mapping = aes(x = DUR, y = BDI)               # Daten-Axen-mapping
) +
  coord_cartesian(ylim = c(-10, 20)) +          # y-limits anpassen
  geom_point() +                                # Datenpunkte zeichnen
  geom_smooth(                                  # Ausgleichsgerade zeichnen
    method = "lm",
    color = "green", se = F, linewidth = 0.4
  ) +
  ylab("BDI Diff") + xlab("Dauer Symptomatik [Monate]") # Achsenbeschriftung
graphics.off()                                  # Schließt browser

ggsave(                                         # Abbildung speichern
  filename = "ggplot_beispiel.pdf",
  height = 5, width = 5
)
```

Plotten mit ggplot2



Visual Studio Code (VS Code) Website

VS Code-R Wiki

R for Data Science (2e)

ggplot2: Elegant Graphics for Data Analysis (3e)