

Kausalanalyse und Machinelles Lernen mit R

Ein Leitfaden für reproduzierbare Forschung

Martin C. Arnold, Christoph Hanck

2023-11-01

Inhaltsverzeichnis

1	Start	4
2	Statistische Programmierung mit R	5
2.1	Lange, weite und “tidy” Datenformate	6
2.2	Pinguine und Pipes	9
2.2.1	dplyr::mutate()	10
2.2.2	dplyr::select()	11
2.2.3	dplyr::filter()	12
2.2.4	dplyr::summarise()	14
2.2.5	dplyr::arrange()	15
2.2.6	Operationen mit gruppierten Datensätzen	16
2.3	Eine explorative Analyse mit ggplot2	18
3	Matching	22
3.1	<i>Balance</i> : Vergleichbarkeit von Behandlungs- und Kontrollgruppe	24
3.1.1	Matching durch Gewichtung	28
3.1.2	Entropy Balancing	34
3.1.3	Mehrere Matching-Variablen und der Propensity Score .	36
3.2	Selektierende Matching-Verfahren	43
3.2.1	Exaktes Matching	44
3.2.2	Coarsened Exact Matching	45
3.2.3	Matching mit der Mahalanobis-Distanz	46
3.2.4	Propensity Score Matching	50
3.3	Schätzung und Inferenz für den Behandlungseffekt nach Matching	55
3.3.1	Cluster-robuste Standardfehler	58
3.4	Bootstrap-Schätzung kausaler Effekte bei Matching	62
3.5	Regression, Matching und Doubly-Robust-Schätzung	69
4	Regression Discontinuity Designs	74
4.1	Sharp Regression Discontinuity Design	75
4.2	Manipulation am Schwellenwert	80
4.2.1	Case Study: Amtsinhaber-Vorteil (Lee 2008)	83
4.3	Fuzzy Regression Discontinuity Design	92
4.4	Case Study: Protestantische Arbeitsethik	95
4.4.1	Aufbereitung der Daten	97
4.4.2	Deskriptive Statistiken	99
4.4.3	Modellspezifikation und First-Stage-Ergebnisse	101
4.4.4	Second-Stage-Ergebnisse	105
4.4.5	Addendum: FRDD-Schätzung mit rdrobust()	109

5	Regularisierte Regression	112
5.1	Ridge Regression	113
5.1.1	Eigenschaften des Schätzers	115
5.1.2	Ridge Regression mit <code>glmnet</code>	118
5.1.3	Beispiel: Vorhersage von Abschlussnoten in Mathe . . .	125
5.2	Lasso Regression	132
5.2.1	Lasso ist Soft Thresholding	133
5.2.2	Berechnung der Lasso-Lösung mit dem LARS-Algorithmus	136
5.2.3	Wahl des Regularisierungsparameters λ für den Lasso-Schätzer	139
5.3	Vergleich von Lasso- und Ridge-Regression mit Simulation . . .	144
5.3.1	Prognosegüte in diversen Szenarien	144
5.3.2	Visualisierung des Bias-Variance-Tradeoffs bei Prognosen	150
5.4	Inferenz für Treatment-Effekt-Schätzung mit vielen Variablen .	156
5.4.1	Case Study: Makroökonomisches Wachstum	163
6	Synthetic Control	171
6.1	Schätzung von Interventionseffekten mit SCM	172
6.2	Case Study: Ökonomische Kosten des Brexit	174
6.2.1	Placebo-Tests: Grafische Inferenz	187
6.2.2	Statistische Inferenz	197
	Literatur	200

1 Start

2 Statistische Programmierung mit R

Dieses Kapitel ist *nicht* als umfassende Einführung in R gedacht, sondern behandelt Kernfunktionen aus der Paketsammlung **tidyverse**. Wenngleich die Inhalte deutlich über ein Hallo-Welt-Beispiel¹ hinausgehen, betrachten wir hier grundlegende Funktionen für Datenmanipulation und Visualisierung. Diese sind Voraussetzung für das Verständnis fortgeschrittener Code-Bausteine in späteren Kapiteln. Falls Sie bereits über Grundkenntnisse im Umgang mit **tidyverse** verfügen, können Sie dieses Kapitel überspringen. Sollten Sie nicht oder nur teilweise mit den hier gezeigten Befehlen vertraut sein oder keinerlei Erfahrung mit R haben, empfiehlt sich vorab eine Erarbeitung bzw. Wiederholung der Inhalte. Nachstehende Ressourcen finden wir hilfreich:

- Feedbackgestützte interaktive Übungsaufgaben bei DataCamp², bspw.
 - [Einführung in R](#)
 - [Introduction to Data Visualization with ggplot2](#)
 - [Data Manipulation with dplyr](#)
- Open-source-Literatur wie
 - der umfangreiche Leitfaden von [Ellis und Mayer \(2023\)](#)
 - [R for Data Science](#)
 - [Hands-On Programming with R](#)

Wir laden zunächst die Paketsammlung **tidyverse**. Für die Reproduktion mit dem [R GUI](#) oder mit [RStudio](#) muss das Paket vorab mit `install.packages()` installiert werden. In den interaktiven R-Konsolen in diesem Kapitel (und im Rest des Buchs) sind die benötigten R-Pakete bereits installiert *und* geladen, sofern nicht anders beschrieben.

```
# Paket tidyverse installieren
# install.packages("tidyverse")
```

¹<https://de.wikipedia.org/wiki/Hallo-Welt-Programm>

²Ein Teil des hier angebotenen Katalogs (exklusive *Einführung in R*) ist kostenpflichtig.

```
# Paket 'tidyverse' laden
library(tidyverse)
```

Für das Verständnis von Code-Chunks ist es hilfreich, Zwischenergebnisse explizit zu evaluieren und in der Konsole auszugeben. Hierfür umschließen wir häufig Code-Zeilen mit runden Klammern. Der nächste Chunk illustriert dies für die Variable `x`.

```
# Variable definieren...
x <- pi
# ... und evaluieren
x

# Äquivalent:
(
  x <- pi
)
```

2.1 Lange, weite und “tidy” Datenformate

Wir betrachten den in Tabelle 2.1 dargestellten Datensatz *Klausurergebnisse*.

Tabelle 2.1: Datensatz *Klausurergebnisse*

Name	Mikro	Makro	Mathe
Tim	NA	1.3	3
Lena	1	3	NA
Ricarda	2	1.7	1.3
Simon	2.3	3.3	NA

Der Datensatz ist noch nicht in der R-Arbeitsumgebung verfügbar. Mit der Funktion `tribble()` können wir Tabelle 2.1 händisch als R-Objekt der Klasse `tibble` definieren

```
# 'klausurergebnisse' als tibble definieren
(
  klausurergebnisse <- tribble(
```

```

~Name,    ~Mikro, ~Makro, ~Mathe,
"Tim",      NA,    1.3,    3.0,
"Lena",     1.0,    3.0,    NA,
"Ricarda",  2.0,    1.7,    1.3,
"Simon",    2.3,    3.3,    NA
)
)

```

`klausurergebnisse` enthält die Klausurnoten der vier Studierenden (Beobachtungen) spaltenweise *pro Modul*, d.h. die Spaltennamen `Mikro`, `Makro` und `Mathe` sind Ausprägungen der Variable `Modul`. Der Datensatz liegt also *nicht* im s.g. *Tidy-Format* vor.

💡 Tidy-Format

Tidy-Format: Jede Spalte ist **eine** Variable, jede Reihe ist **eine** Beobachtung und jede Zelle enthält einen **einen** Wert. Datensätze im Tidy-Format sind häufig lang: Die Zeilendimension ist größer als die Spaltendimension.

Das Tidy-Format ist hilfreich für statistische Analysen mit `tidyverse`-Funktionen wie bspw. `ggplot()`. Wir nutzen die Funktion `tidyr::pivot_longer()`, um `klausurergebnisse` ein (langes) Tidy-Format zu transformieren.

```

# 'klausurergebnisse' in Tidy-Format überführen
(
  long <- pivot_longer(
    data = klausurergebnisse,
    cols = Mikro:Mathe,
    names_to = "Modul",
    values_to = "Note"
  )
)

```

Beachte, dass die Spalte `Name` die Zugehörigkeit der Ausprägungen (`Note`) jeder Variable (`Modul`) zu einer Beobachtung identifiziert. Mit dieser Information können wir den langen Datensatz wieder in das ursprüngliche (weite) Format zurückführen. Wir nutzen hierzu `tidyr::pivot_wider()`.

```
# langes Format in das Ausgangsformat transformieren
(
  wide <- pivot_wider(
    data = long,
    id_cols = "Name",
    names_from = "Modul",
    values_from = "Note"
  )
)
```

Wenn die Zuweisung von Zwischenergebnissen in Variablen nicht benötigt wird, kann eine Verkettung von Funktionsaufrufen die Verständlichkeit des Codes verbessern. Hierzu wird der [Pipe-Operator %>%](#) genutzt. Wir wiederholen die Transformationen mit den `tidyr::pivot_*`-Funktion bei Verwendung von `%>%`.

```
# langes Format mit %>%
(
  long <- klausurergebnisse %>%
    pivot_longer(
      cols = Mikro:Mathe,
      names_to = "Modul",
      values_to = "Note"
    )
)

# weites Format mit %>%
(
  wide <- long %>%
    pivot_wider(
      id_cols = "Name",
      names_from = "Modul",
      values_from = "Note"
    )
)
```

Ein Beispiel für den Nachteil des weiten Formats im Umgang mit `tidyverse`-Paketen ist die Funktion `tidyr::drop_na()`. Diese entfernt sämtliche *Zeilen* eines Datensatzes, die `NA`-Einträge (d.h. fehlende Werte) aufweisen. Beachte,

dass diese Operation im ursprünglichen weiten Format zum Entfernen ganzer Beobachtungen aus `wide` führt.

```
# NA-Einträge aus dem "weiten" Format entfernen
wide %>%
  drop_na()
```

Im Tidy-Format `long` hingegen bleiben die übrigen Informationen betroffener Beobachtungen erhalten.

```
# NA-Einträge aus dem "langen" Format entfernen
long %>%
  drop_na()
```

2.2 Pinguine und Pipes

In diesem Abschnitt zeigen wir die Verwendung häufig verwendeter `dplyr`-Funktionen (s.g. *Verben*) für die Transformation von Datensätzen: `mutate()`, `select()`, `filter()`, `summarise()` und `arrange()`.

Für die Illustration verwenden wir den Datensatz `penguins` aus dem R-Paket `palmerpenguins`. Dieser Datensatz wurde im Zeitraum 2007 bis 2009 von Dr. Kristen Gorman im Rahmen des *Palmer Station Long Term Ecological Research Program* zusammengetragen und enthält Größenmessungen für drei Pinguinarten, die auf den Inseln des [Palmer-Archipels](#) in der Antarktis beobachtet wurden.

```
# Paket 'palmerpenguins' installieren
# install.packages("palmerpenguins")

# Paket 'palmerpenguins' laden
library(palmerpenguins)
```

Mit `data()` wird der Datensatz in der Arbeitsumgebung verfügbar gemacht. Wir nutzen `glimpse()`, um einen Überblick zu erhalten.

```
# Datensatz in der Arbeitsumgebung verfügbar machen
data(penguins)

# Übersicht anzeigen lassen
glimpse(penguins)
```

2.2.1 dplyr::mutate()

Mit `mutate()` können bestehende Variablen überschrieben oder neue Variablen als Funktion bestehender Variablen definiert werden. `mutate()` operiert in der Spaltendimension des Datensatz.

Wir definieren eine neue Variable `body_mass_kg` als Transformation `body_mass_g/1000`.

```
# Neue Variable mit Gewicht in Kg definieren
penguins %>%
  mutate(
    body_mass_kg = body_mass_g/1000
  ) %>%
  glimpse()
```

Mit `across()` kann die dieselbe Operation auf mehrere Variablen angewendet werden.

Im nachstehenden Beispiel ändern wir den typ (`type`) der Variablen `species`, `island`, `sex` und `year` zu `character`.

```
# species, island, sex und year in Typ 'character' umwandeln
penguins %>%
  mutate(
    across(
      c(species, island, sex, year),
      .fns = as.character
    )
  ) %>%
  glimpse()
```

`transmute()` ist eine Variante von `mutate()`, die lediglich die transformierten Variablen beibehält.

```
# Nur transformierte Variablen behalten
penguins %>%
  transmute(
    body_mass_kg = body_mass_g/1000
  )
```

2.2.2 `dplyr::select()`

Mit `select()` werden Variablen aus dem Datensatz ausgewählt. Dies geschieht entweder über den Variablennamen...

```
# 'species' auswählen
penguins %>%
  select(species)
```

... oder über eine Indexmenge.³

```
# Teilmenge von Variablen per Index auswählen
penguins %>%
  select(
    c(1, 2, 3)
  )
```

Variablen können anhand eines Muster im Namen selektiert werden. Die Selektion von `ends_with("mm")` bezieht nur Variablen mit Endung `mm` im Namen ein:

```
# Nur in mm gemessene Variablen auslesen
penguins %>%
  select(
    ends_with("mm")
  )
```

³Hilfreich: `dplyr::pull()` selektiert eine Variable und wandelt diese in einen Vektor um.

Mit `where()` können wir Variablen aufgrund bestimmter Eigenschaften ihrer Ausprägungen selektieren.

```
# Nur numerische Variablen auswählen
penguins %>%
  select(
    where(is.numeric)
  )
```

2.2.3 `dplyr::filter()`

Das Verb `filter()` filtert den Datensatz in der Zeilendimension. So können Beobachtungen ausgewählt werden, deren Merkmalsausprägungen bestimmte Kriterien erfüllen. Hierzu muss `filter()` ein logischer (`logical`) Ausdruck übergeben werden. Häufig erfolgt dies über Vergleichsoperatoren.

```
# Nur Pinguine mit bill_length_mm >= 39
penguins %>%
  filter(
    bill_length_mm >= 39
  )
```

```
# Nur Pinguine mit bill_length_mm <= 40
penguins %>%
  filter(
    bill_length_mm <= 40
  )
```

Oft ist es praktisch, mehrere Kriterien zu kombinieren.

```
# Kombiniertes Filter -- Variante 1
penguins %>%
  filter(
    bill_length_mm >= 39 & bill_length_mm <= 40
  )
```

Analog: komma-getrennte Kriterien werden intern über den Und-Operator (`&`) verknüpft.

```
# Kombiniertes Filter -- Variante 2
penguins %>%
  filter(
    bill_length_mm >= 39,
    bill_length_mm <= 40
  )
```

Ähnlich wie bei `select()` verwenden wir häufig nützliche Funktionen, welche die Interpretation des Codes erleichtern. `dplyr::between()` erlaubt filtern innerhalb eines Intervalls.

```
# Filtern mit Hilfsfunktion
penguins %>%
  filter(
    between(
      bill_length_mm, left = 39, right = 40
    )
  )
```

Mit diesen Verben sind wir bereits in der Lage, den Datensatz gemäß folgender Vorschrift zu bereinigen:

1. Entfernen der Maßeinheiten aus den Variablennamen
2. Entfernen von Pinguinen mit fehlenden Werten (NA)
3. Entfernen von Pinguinen mit einem Gewicht *oberhalb* des 95%-Stichprobenquantils

```
# Schritt 1
(
  penguins_cleaned <- penguins %>%
    rename(
      bill_length = bill_length_mm,
      bill_depth = bill_depth_mm,
      flipper_length = flipper_length_mm,
      body_mass = body_mass_g
    )
)
```

```
# Schritt 2
(
  penguins_cleaned <- penguins_cleaned %>%
    drop_na()
)
```

```
# Schritt 3
penguins_cleaned %>%
  filter(
    body_mass < quantile(body_mass, probs = .95)
  ) %>%
  glimpse()
```

Durch die Verkettung mit `%>%` können wir sämtliche Schritte für die Bereinigung ohne das Abspeichern von Zwischenergebnissen durchführen.

```
# Verketteter Funktionsaufruf für Datensatzbereinigung
penguins_cleaned <- penguins %>%
  rename(
    bill_length = bill_length_mm,
    bill_depth = bill_depth_mm,
    flipper_length = flipper_length_mm,
    body_mass = body_mass_g
  ) %>%
  drop_na() %>%
  filter(
    body_mass < quantile(body_mass, .95)
  )

penguins_cleaned %>%
  glimpse()
```

2.2.4 `dplyr::summarise()`

Das Verb `summarise()` fasst Variablen über Beobachtungen hinweg zusammen. Der nachstehende Code-Chunk erzeugt eine Tabelle mit Stichprobenmittelwert und -standardabweichung von `flipper_length_mm`.⁴ Um zu vermeiden, dass

⁴`dplyr::summarise()` darf nicht mit `base::summary()` verwechselt werden!

die Auswertung aufgrund fehlender Werte (NA) in `flipper_length_mm` scheitert, lassen wir NAs mit `na.rm = TRUE` bei der Berechnung unberücksichtigt (wir verwenden weiterhin den unbereinigten Datensatz `penguins`).

```
# statistische Zusammenfassung mit 'summarise()'
penguins %>%
  select(flipper_length_mm) %>%
  summarise(
    mean = mean(flipper_length_mm, na.rm = TRUE),
    sd = sd(flipper_length_mm, na.rm = TRUE)
  )
```

Varianten von `summarise()` können über mehrere Variablen angewendet werden. Wir verwenden `across()` und `where()`, um lediglich numerische Variablen mit den in der liste definierten Funktionen zusammenzufassen. Beachte, dass `\(x) mean(x)` eine anonyme Funktion definiert.

```
penguins %>%
  summarise(
    across(
      where(is.numeric),
      .fns = list(
        mean = \(x) mean(x, na.rm = TRUE),
        sd = \(x) sd(x, na.rm = TRUE)
      )
    )
  ) %>%
  glimpse()
```

2.2.5 `dplyr::arrange()`

Mit `arrange()` können Datensätze in Abhängigkeit der beobachteten Ausprägungen von Variablen sortiert werden.

```
# Datensatz aufsteigend nach 'body_mass_g' sortieren
penguins %>%
  arrange(body_mass_g)
```

Die Funktion `dplyr::desc()` kehrt die Reihenfolge zu einer absteigenden Sortierung um.

```
# Absteigende Sortierung nach 'body_mass_g'
penguins %>%
  arrange(
    desc(body_mass_g)
  )
```

Komplexe Sortier-Muster werden durch Übergabe von Variablennamen in der gewünschten Reihenfolge erreicht.

```
# Erst Sortierung nach 'sex', dann gruppenweise absteigend
# nach 'body_mass_g' sortieren
penguins %>%
  arrange(
    sex, desc(body_mass_g)
  )
```

2.2.6 Operationen mit gruppierten Datensätzen

Für manche Transformationen ist eine Gruppierung der Daten hilfreich. Wir illustrieren nachfolgend die unterschiedlichen Verhaltensweisen ausgewählter Verben durch Vergleiche von gruppierten und nicht-gruppierten Anwendungen.

```
# Datensatz gruppieren
penguins_grouped <- penguins %>%
  group_by(species)

# Datensatz hat nun die Eigenschaft 'Groups'
glimpse(penguins_grouped)
```

`species` hat drei Ausprägungen. Entsprechend ist `penguins_grouped` nun in drei Gruppen eingeteilt.

Bei gruppierten Datensätzen fasst `summarise()` die Variablen pro Gruppe zusammen.


```
# summarise -- ungruppiert:
penguins %>%
  summarise(
    across(
      where(is.numeric), \(x) mean(x, na.rm = T)
    )
  )
```

```
# summarise -- gruppiert:
penguins_grouped %>%
  summarise(
    across(
      where(is.numeric),
      ~ mean(., na.rm = T)
    )
  )
```

`mutate()` definiert bzw. transformiert für jede Gruppe separat. Im dies zu veranschaulichen, ziehen wir eine Zufallsstichprobe von 10 Pinguinen aus der Datensatz.

```
# Zufallsstichprobe generieren
set.seed(123)
(
  penguins_sample <- penguins %>%
    slice_sample(n = 10)
)
```

```
# mutate() -- ungruppiert:
penguins_sample %>%
  transmute(
    mean = mean(bill_length_mm)
  )
```

Für den ungruppierten Datensatz berechnet `mutate()` das Stichprobenmittel von `bill_length_mm` über *alle* zehn Datenpunkte und weist diesen Wert jeweils in der Variable `mean` zu.

```
# mutate() -- gruppiert
penguins_sample %>%
  group_by(species) %>%
  transmute(
    mean = mean(bill_length_mm)
  )
```

Bei gruppierten Daten berechnet `mutate()` das Stichprobenmittel *pro Gruppe* und weist die Mittelwerte entsprechend zu.

2.3 Eine explorative Analyse mit ggplot2

Der bereinigte Datensatz `penguins_cleaned` eignet sich gut für eine graphische Auswertung mit dem R-Paket `ggplot2`, welches Bestandteil des `tidyverse` ist. Nachfolgend untersuchen wir Zusammenhänge zwischen den Körpermaßen der Pinguine.

Wir erstellen zunächst einen einfachen Punkteplot des Gewichts (`body_mass`) und der Schnabeltiefe (`bill_depth`).

```
# Punkteplot: body_mass vs. bill_depth
penguins_cleaned %>%
  ggplot(
    mapping = aes(
      x = body_mass,
      y = bill_depth
    )
  ) +
  geom_point()
```

Die Grafik zeigt einen positiven Zusammenhang zwischen dem Gewicht und der Schnabeltiefe. Als nächstes passen wir den Code so an, dass die Datenpunkte entsprechend der Art (`species`) eingefärbt sind.

```
# Punkteplot: Farbliche Darstellung verschiedener Arten
penguins_cleaned %>%
  ggplot(
    mapping = aes(
```

```

    x = body_mass,
    y = bill_depth,
    color = species
  )
) +
geom_point()

```

Offenbar gibt es deutliche Unterschiede in der (gemeinsamen) Verteilung von Gewicht und Schnabeltiefe zwischen den verschiedenen Arten.

Um den Zusammenhang zwischen Gewicht und Schnabeltiefe zu untersuchen, schätzen wir lineare Regressionen

$$body_mass = \beta_0 + \beta_1 bill_depth + u$$

separat für jede der drei Pinguinarten mit der KQ-Methode. Anschließend zeichnen wir die geschätzten Regressionsgeraden ein.

```

# Lineare Regression per Art
penguins_cleaned %>%
  ggplot(
    aes(
      x = body_mass,
      y = bill_depth,
      color = species
    )
  ) +
  geom_point() +
  geom_smooth(method = "lm", se = F)

```

Die Schätzungen bekräftigen die Vermutung, dass der lineare Zusammenhang zwischen Gewicht und Schnabeltiefe sich nicht zwischen den verschiedenen Pinguinarten unterscheidet: Pinguine der Art *Gentoo* sind im Mittel schwerer als Pinguine der übrigen Arten, haben jedoch eine geringere Schnabeltiefe.

Der nachfolgende Code fügt der Grafik eine Regressionsline *über alle* Arten hinzu. Wir setzen hierbei das Argument `inherit_aes = FALSE` und legen damit fest, dass die Regression für `body_mass` und `bill_depth` ohne Differenzierung per `species` durchgeführt wird.

```

# Zusatz: Globale Regression
penguins_cleaned %>%
  ggplot(
    mapping = aes(
      x = body_mass,
      y = bill_depth,
      color = species
    )
  ) +
  geom_point() +
  geom_smooth(method = "lm", se = F) +
  # Regression für alle Datenpunkte
  geom_smooth(
    mapping = aes(
      x = body_mass,
      y = bill_depth
    ),
    method = "lm",
    se = F,
    inherit.aes = F
  )

```

Offenbar ist die vorherige Analyse per Spezies sinnvoller: Die Regression über alle Arten suggeriert einen negativen Zusammenhang zwischen Gewicht und Schnabeltiefe.

Facetting mit `facet_wrap()` erlaubt eine Untersuchung des Zusammenhangs je Insel (`island`), auf der die Messung erfolgt ist.

```

# Facettierung des per In
penguins_cleaned %>%
  ggplot(
    mapping = aes(
      x = body_mass,
      y = bill_depth,
      color = species
    )
  ) +
  geom_point() +
  geom_smooth(method = "lm", se = F) +
  facet_wrap(~ island)

```

Wir sehen, dass es hinsichtlich des Zusammenhangs von Gewicht und Schnabeltiefe keine wesentlichen Diskrepanzen zwischen den drei Inseln gibt. Darüber hinaus lässt sich anhand der Facetten leicht erkennen, wie die drei Arten über die Inseln verteilt sind.

3 Matching

```
#| context: setup
# create dataset directory
dir.create("datasets")
# Download the dataset
download.file(
  ↪ "https://raw.githubusercontent.com/mca91/kausal_data/main/darkmode.csv",
  'datasets/darkmode.csv',
)
```

```
#| context: setup
library(dplyr)
# make darkmode available using read.csv for now
# because there's some issue with readr::read_csv
# I can't fix right now
darkmode <- read.csv(
  file = "datasets/darkmode.csv",
  colClasses = c("numeric", "logical", "numeric", "numeric",
  ↪ "numeric")
)

options(pillar.bold = TRUE, pillar.subtle = FALSE)
```

In randomisierten kontrollierten Studien stellt eine randomisierte Behandlung sicher, dass die Individuen beider Gruppen im Mittel vergleichbar sind, dass heißt es gibt keine systematischen Unterschiede der Studiensubjekte hinsichtlich der Verteilung von Charakteristika zwischen den Gruppen. Dann ist es plausibel eine beobachtete mittlere Differenz in der Outcome-Variable alleine auf die Behandlung zurückzuführen.

In der Praxis, insbesondere in ökonomischen Studien, sind randomisierte kontrollierte Studien aus ethischen und/oder finanziellen Gründen nicht durchführbar. Stattdessen werden nicht-experimentelle Daten genutzt, die jedoch nur sehr selten eine Vergleichbarkeit von Behandlungs- und Kontrollgruppe gewährleisten.

In diesem Kapitel betrachten wir Methoden, die in solchen Forschungsdesigns – bei hinreichenden Informationen über die Studiensubjekte – eine Schätzung kausaler Effekte ermöglichen, indem eine statistische Vergleichbarkeit von Behandlungs- und Kontrollgruppe hergestellt wird. Dies kann durch eine gezielte Gewichtung von Beobachtungen anhand individueller Merkmale bei der Schätzung des Behandlungseffekts erfolgen. Andere etablierte Methoden schätzen den kausalen Effekt nach Selektion von vergleichbaren Teilmengen von Subjekten beider Gruppen aus der ursprünglichen Stichprobe, sogenanntes *selektierendes Matching*.

Da Matching Beobachtungen basierend auf beobachtbaren Merkmalen vergleicht, kann die Wahrscheinlichkeit einer verzerrten Schätzung des kausalen Effekts durch falsche Modellspezifikationen geringer sein als für eine Schätzung des Effekts anhand multipler Regression. Weiterhin basieren Matching-Methoden nicht auf der Annahme eines linearen Zusammenhangs zwischen Kovariablen und der erklärenden Variable und können für die Schätzung unterschiedlicher Spezifikationen von Behandlungseffekten herangezogen werden.

Für die Gültigkeit eines Schätzers basierend auf Matching sind zwei Annahmen erforderlich.

1. **Bedingte Unabhängigkeit.** Seien $Y_i^{(0)}$ und $Y_i^{(1)}$ potentielle Ergebnisse der Outcome-Variable Y für ein Subjekt i mit $B_i = 0$ (keine Zuweisung zur Behandlung) beziehungsweise $B_i = 1$ (Behandlung) und X_i die beobachteten Kovariablen. Wir nehmen an, dass

$$\{Y_i^{(0)}, Y_i^{(1)}\} \perp B_i | X_i, \quad (3.1)$$

d.h. die Behandlungszuweisung/-selektion ist unanabhängig von den potentiellen Ergebnissen $Y_i^{(0)}$ und $Y_i^{(1)}$, wenn wir für die Kovariablen X kontrollieren.

2. **Überlappung.** Für jede mögliche Kombination von Kovariablen X_i gibt es eine positive Wahrscheinlichkeit < 1 , sowohl zur Behandlungsgruppe ($B_i = 1$) als auch zur Kontrollgruppe ($B_i = 0$) zugewiesen zu werden,

$$0 < P(B_i = 1 | X_i) < 1, \quad (3.2)$$

d.h. keine Beobachtung hat eine Behandlungswahrscheinlichkeit von exakt 0 oder 1.

Annahme 1 stellt sicher, dass die Zuweisung zur Behandlungsgruppe nach Kontrolle für die Kovariablen X als zufällig betrachtet werden kann. Somit ist es möglich den kausalen Effekt der Behandlung zu identifizieren, indem wir hinsichtlich der Kovariablen X ähnliche Subjekte (vgl. Annahme 2) aus Kontroll- und Behandlungsgruppe vergleichen.

Annahme 2 setzt voraus, dass es eine ausreichende Überlappung in den Verteilungen der Kovariablen zwischen Behandlungs- und Kontrollgruppe gibt. Dann ist sichergestellt, dass für jedes Subjekt in einer Gruppe ein hinsichtlich seiner Charakteristika vergleichbares Subjekt in der anderen Gruppe geben kann, sodass ein Vergleich möglich ist.

3.1 **Balance: Vergleichbarkeit von Behandlungs- und Kontrollgruppe**

Der Lehrstuhl für Ökonometrie an der Universität Duisburg-Essen betreibt einen Ökonometrie-Blog und interessiert sich für den kausalen Effekt der Einführung eines [darkmode](#) auf die Verweildauer der User auf der Webseite. Die Webseite ist zwar nicht-kommerziell, hat sich allerdings insb. für die Akquise internationaler Studierender für den Studiengang MSc. Econometrics als wichtiges Marketing-Instrument erwiesen. Ein anprechendes Design wird daher als hoch-relevant erachtet.

Idealerweise sollte der Effekt des Design-Relaunches auf die Nutzungsintensität in einem kontrollierten randomisierten Experiment untersucht werden. Hierbei würden wir Nutzern zufällig das neue oder das alte Design zuweisen und den Effekt als Differenz des durchschnittlichen Verweildauer für die Gruppen bestimmen. Eine solche Studie ist jedoch aus technischen und finanziellen Gründen nicht realisierbar, sodass die Auswirkungen des darkmode mit vorliegenden nicht-experimentellen Nutzungsstatistiken für die Webseite geschätzt werden sollen.

Die Nutzungsstatistiken sind im Datensatz [darkmode.csv](#) enthalten und sollen der Analyse des Effekts des darkmode (`dark_mode`) auf die Verweildauer der Leser auf der Webseite (`read_time`) dienen.

Tabelle [3.1](#) zeigt die Definitionen der Variablen in *darkmode.csv*.

Tabelle 3.1: Variablen im Datensatz *darkmode*

Variable	Beschreibung
read_time	Lesezeit (Minuten/Woche)
dark_mode	Indikator: Beobachtung nach Einführung darkmode
male	Indikator: Individuum männlich
age	Alter (in Jahren)
hours	Bisherige Verweildauer auf der Seite

Für die Analyse lesen wir zunächst den Datensatz *darkmode.csv* mit `readr::read_csv()` ein und verschaffen uns einen Überblick über die verfügbaren Variablen.

```
# Paket `tidyverse` laden
library(tidyverse)

# Datensatz 'darkmode' einlesen
darkmode <- read_csv(
  file = "datasets/darkmode.csv"
)
```

`dark_mode` hat den Typ `logical`. Mit `dplyr::mutate_all()` können wir komfortabel alle Spalten in den Typ `numeric` transformieren.

```
# Alle Variablen zu typ 'numeric' formatieren...
darkmode <- darkmode %>%
  mutate_all(.funs = as.numeric)

# ... und überprüfen
glimpse(darkmode)
```

```
Rows: 300
Columns: 5
$ read_time <dbl> 14.4, 15.4, 20.9, 20.0, 21.5, 19.5, 22.0,
17.4, 23.6, 15.7, ~
$ dark_mode <dbl> 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0, 1, 1,
1, 1, 0, 0, 0, ~
$ male <dbl> 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0,
1, 0, 1, 0, 0, ~
```

```
$ age      <dbl> 43, 55, 23, 41, 29, 64, 18, 53, 59, 53, 43,
38, 42, 23, 39, ~
$ hours    <dbl> 65.6, 125.4, 642.6, 129.1, 190.2, 185.3,
333.5, 279.3, 1302.~
```

Eine naive Schätzung des durchschnittlichen Behandlungseffekts (ATE) $\hat{\tau}^{\text{naiv}}$ erhalten wir als Mittelwertdifferenz von `read_time` für die Behandlungsgruppe (`dark_mode == 1`) und die Kontrollgruppe (`dark_mode == 0`)

$$\hat{\tau}^{\text{naiv}} = \overline{\text{read_time}}_{\text{Behandlung}} - \overline{\text{read_time}}_{\text{Kontrolle}}. \quad (3.3)$$

Diese Berechnung ist schnell mit R durchgeführt.

```
# Naiver Schätzer für ATE:
# Differenz der Gruppen-Durchschnitte

# Outcome in Behandlungsgruppe
read_time_mTG <- darkmode %>%
  filter(dark_mode == 1) %>%
  pull("read_time")

# Outcome in Kontrollgruppe
read_time_mKG <- darkmode %>%
  filter(dark_mode == 0) %>%
  pull("read_time")

# Mittelwert-Differenz
mean(read_time_mTG) - mean(read_time_mKG)
```

```
[1] -0.4446331
```

Die Schätzung ergibt einen negativen Behandlungseffekt, mit der Interpretation, dass das neue Design zu einer Reduktion der Lesezeit um etwa 0.44 Minuten pro Woche führt. Dieses Ergebnis ist allerdings zweifelhaft, weil eine Isolierung des Behandlungseffekts aufgrund von Backdoor-Pfaden im DGP vermutlich nicht gewährleistet ist. Ein Indikator hierfür sind systematische Unterschiede hinsichtlich von (möglicherweise unbeobachtbaren) Charakteristika von Kontrollgruppe und Behandlungsgruppe.

Da die User sich beim Aufrufen der Seite aktiv für oder gegen den das neue Design entscheiden müssen (und somit selektieren, ob Sie in der Behandlungs-

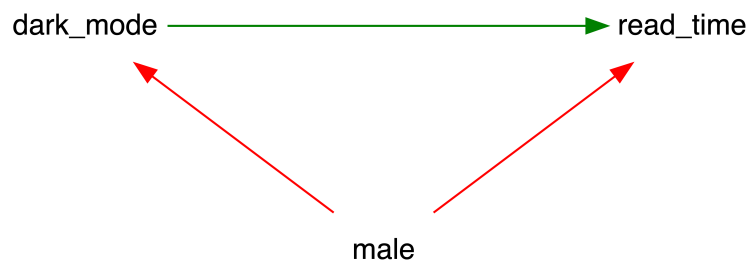


Abbildung 3.1: Backdoor durch ‘male’ im Website-Design-Bespiel

oder Kontrollgruppe landen), liegt wahrscheinlich *Confounding* vor: Unsere Hypothese ist zunächst, dass männliche User eine durchschnittlich längere Lesezeit aufweisen *und* mit größerer Wahrscheinlichkeit auf das neue Design wechseln als nicht-männliche Leser. Dann ist `male` eine Backdoor-Variable. Diese Situation ist unter der Annahme, dass nur diese Faktoren den DGP bestimmen, in Abbildung 3.1 dargestellt.

Der DGP in Abbildung 3.1 führt zu einer verzerrten Schätzung des kausalen Effekts von `dark_mode` auf `read_time` mit (3.3), wenn das Verhältnis von männlichen und nicht-männlichen Usern in Behandlungs- und Kontrollgruppe nicht ausgeglichen ist. Wir überprüfen dies mit R.

```

# Anteile männlicher und nicht-männlicher User
(
  anteile <- darkmode %>%
  group_by(dark_mode) %>%
  summarise(
    gesamt = n(),
    ant_m = mean(male),
    ant_nm = 1 - ant_m,
    anz_m = sum(male),
    anz_nm = gesamt - anz_m
  )
)

```

```

# A tibble: 2 x 6
  dark_mode gesamt ant_m ant_nm anz_m anz_nm
  <dbl>    <int> <dbl>  <dbl> <dbl>  <dbl>
1         0    151 0.338  0.662   51    100
2         1    149 0.658  0.342   98    51

```

Die Zusammenfassung `anteile_m` zeigt, dass der Anteil männlicher User in der Behandlungsgruppe deutlich höher ist als in der Kontrollgruppe.

3.1.1 Matching durch Gewichtung

Matching eliminiert die Variation von `male` zwischen den Gruppen. Eine Möglichkeit hierfür ist die Gewichtung der Beobachtungen in der Kontrollgruppe entsprechend der Anteile von Männern und Nicht-Männern in der Behandlungsgruppe, sodass die Vergleichbarkeit mit der Behandlungsgruppe hinsichtlich des Geschlechts gewährleistet ist. Dies wird in der Literatur als *Balance* bezeichnet. Der Behandlungseffekt wird dann analog zu (3.3) geschätzt.

Die Gewichte für Beobachtungen in der Kontrollgruppe w_i werden berechnet als

$$w_i = \begin{cases} \text{ant_m}_B / \text{anz_m}_K, & \text{falls } \text{male}_i = 1 \\ \text{ant_nm}_B / \text{anz_nm}_K, & \text{sonst.} \end{cases} \quad (3.4)$$

Anhand der Formel für einen gewichteten Durchschnitt,

$$\bar{X}_w = \frac{\sum_i w_i \cdot X_i}{\sum_i w_i}, \quad (3.5)$$

berechnen wir die gewichteten Mittelwerte für `male` und `read_time` in der Kontrollgruppe.

```
# Anteile und Anzahlen aus `anteile` auslesen
anz_m_K <- anteile %>%
  filter(dark_mode == 0) %>% pull(anz_m)

anz_nm_K <- anteile %>%
  filter(dark_mode == 0) %>% pull(anz_nm)

ant_m_B <- anteile %>%
  filter(dark_mode == 1) %>% pull(ant_m)

ant_nm_B <- anteile %>%
  filter(dark_mode == 1) %>% pull(ant_nm)
```

```
# Gewichtete Mittel für Kontrollgruppe berechnen
(
```

```

gew_K <- darkmode %>%
  filter(dark_mode == 0) %>%
  select(read_time, male) %>%
  mutate(w = ifelse(
    male == 1,
    ant_m_B/anz_m_K,
    ant_nm_B/anz_nm_K)
  ) %>%
  summarise(
    male_k = sum(male * w) / sum(w),
    mean_read_time_wK = sum(read_time * w) / sum(w)
  )
)

```

```

# A tibble: 1 x 2
  male_k mean_read_time_wK
  <dbl>         <dbl>
1  0.658         18.1

```

Ein Vergleich des gewichteten Mittelwertes von `male` in der Kontrollgruppe mit dem Mittelwert in der Behandlungsgruppe (`male_k`) zeigt, dass die Gewichte die Variation in `male` zwischen beiden Gruppen eliminieren, sodass die Backdoor durch `male` geschlossen ist. Mit `wmean_read_time_K` haben wir einen entsprechend gewichteten Mittelwert der Verweildauer für die Kontrollgruppe berechnet. Wir schätzen den Behandlungseffekt nun als

$$\hat{\tau}^w = \overline{\text{read_time}}_B - \overline{\text{read_time}}_{w,K}. \quad (3.6)$$

```
mean(read_time_mTG) - gew_K$mean_read_time_wK
```

```
[1] 0.6383579
```

Entgegen der naiven Schätzung anhand von (3.3) erhalten wir nach Matching für `male` eine positive Schätzung des Behandlungseffekts von etwa 0.64.

Die Schätzung des Behandlungseffekts anhand von (3.6) entspricht dem geschätzten Koeffizienten $\hat{\beta}_1$ aus einer gewichteten KQ-Regression im Modell

$$\text{read_time} = \beta_0 + \beta_1 \text{dark_mode} + u,$$

wobei die Beobachtungen der Kontrollgruppe wie in (3.4) gewichtet werden und $w_i = 1$ für Beobachtungen der Behandlungsgruppe ist. Wir überprüfen dies mit R.

```
darkmode_w <- darkmode %>%
  mutate(
    w = case_when(
      male == 1 & dark_mode == 0 ~ ant_m_B/anz_m_K,
      male == 0 & dark_mode == 0 ~ ant_nm_B/anz_nm_K,
      T ~ 1
    )
  )

lm(read_time ~ dark_mode, weights = w, data = darkmode_w) %>%
  summary()
```

Call:

```
lm(formula = read_time ~ dark_mode, data = darkmode_w, weights =
w)
```

Weighted Residuals:

Min	1Q	Median	3Q	Max
-11.4302	-0.6929	0.0814	0.7230	12.8698

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	18.0918	3.4796	5.199	3.72e-07 ***
dark_mode	0.6384	3.4912	0.183	0.855

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 3.48 on 298 degrees of freedom

Multiple R-squared: 0.0001122, Adjusted R-squared: -0.003243

F-statistic: 0.03343 on 1 and 298 DF, p-value: 0.855

Der geschätzte Koeffizient von `dark_mode` entspricht $\hat{\tau}^w$.

Da `male` eine binäre Variable ist, reduziert sich eine Beurteilung der Vergleichbarkeit der Verteilungen von `male` in Behandlungs- und Kontrollgruppe auf einen simplen Vergleich des Männeranteils beider Gruppen. In der Praxis gibt

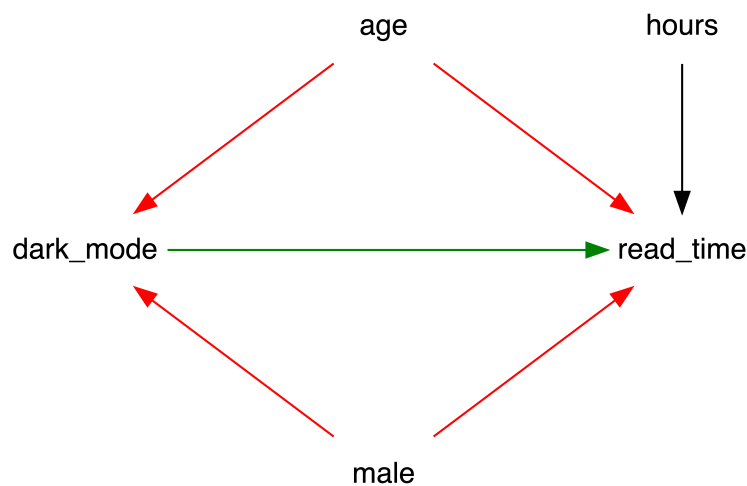


Abbildung 3.2: Erweiterter DGP im Website-Design-Beispiel

es meist eine Vielzahl potentieller Backdoor-Variablen, die zudem kontinuierlich verteilt sind. Es scheint plausibel, dass das Alter der Nutzer sowohl die Akzeptanz des Design-Updates als auch die Lesezeit beeinflusst. Die bisherige Verweildauer ist mindestens eine plausible Determinante der Lesezeit.

Der erweiterte DGP ist in [Abbildung 3.2](#) dargestellt, wobei der zusätzliche Backdoor-Pfad durch **age** ebenfalls mit roten Pfeilen gekennzeichnet sind.

Die Beurteilung der *Balance* von Kontrollgruppe und Behandlungsgruppe kann durch eine grafische Gegenüberstellung der empirischen Verteilungen der Kovariablen beider Gruppen erfolgen. Wir visualisieren die empirischen Verteilungen mit `ggplot2`. Hierzu standardisieren wir **age** und **hours** zunächst mit `scale()`.

```

# Datensatz für graphische Darstellung formatieren
darkmode_p <- darkmode %>%
  # Standardisierung mit 'scale()'
  mutate(
    dark_mode = as_factor(dark_mode),
    age = scale(age),
    hours = scale(hours)
  )

head(darkmode_p)

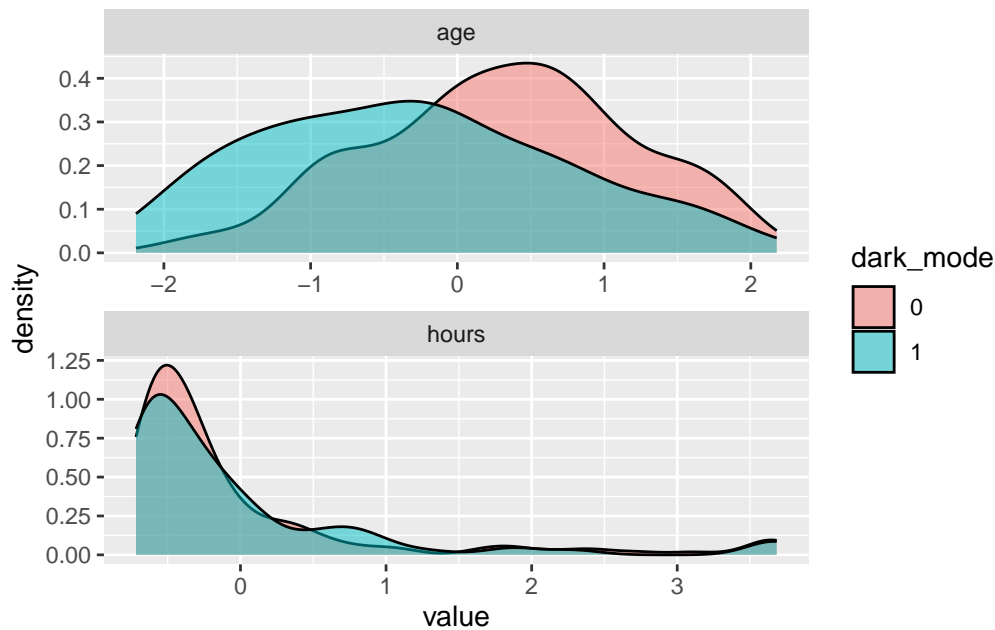
```

```
# A tibble: 6 x 5
  read_time dark_mode male age[,1] hours[,1]
    <dbl> <fct>      <dbl>  <dbl>    <dbl>
1    14.4 0          0  0.0377 -0.591
2    15.4 0          1  1.11    -0.459
3    20.9 1          0 -1.74    0.684
4    20 0          0 -0.141 -0.451
5    21.5 1          0 -1.21    -0.316
6    19.5 0          0  1.91    -0.327
```

Für `age` und `hours` eignen sich die geschätzten Dichtefunktionen für einen Vergleich der Verteilungen in Behandlungs- und Kontrollgruppe.

```
# Vergleich mit Dichteschätzungen
darkmode_p %>%
  select(dark_mode, hours, age) %>%
  # in langes Format überführen
  pivot_longer(cols = c(-dark_mode)) %>%

  ggplot(
    aes(x = value, fill = dark_mode)
  ) +
  geom_density(alpha = .5) +
  facet_wrap(
    facets = ~ name,
    scales = "free",
    nrow = 2
  )
```

Die graphische Analyse zeigt deutliche Unterschiede in den Verteilungen von **age** zwischen Kontroll- und Behandlungsgruppe. Für eine Beurteilung mit deskriptiven Statistiken wird häufig eine sogenannte *Balance Table* herangezogen. Wir berechnen diese für **age**, **hours** und **male** mit `cobalt::bal.tab()`

```
library(cobalt)

# Balance table mit 'cobalt::bal.tab()'
bal.tab(
  x = darkmode %>%
    select(age, hours, male),
  treat = darkmode$dark_mode,
  # berechne SMD für KG und TG:
  disp = "m",
  s.d.denom = "pooled"
)
```

Balance Measures

	Type	M.0.Un	M.1.Un	Diff.Un
age	Contin.	46.0132	39.0940	-0.6469
hours	Contin.	337.7775	328.5738	-0.0203
male	Binary	0.3377	0.6577	0.3200

Sample sizes

	Control	Treated
All	151	149

Die Einträge `M.0.Un` und `M.1.Un` zeigen die jeweiligen Stichprobenmittelwerte der Variablen für Kontroll- und Behandlungsgruppe. `Diff.Un` gibt eine standardisierte Mittelwertdifferenz *SMD* an, wobei

$$SMD_j := (\bar{X}_{j,B} - \bar{X}_{j,K}) / \sqrt{\frac{1}{2} (\widehat{\text{Var}}(X_{j,B}) + \widehat{\text{Var}}(X_{j,K}))},$$

mit Stichprobenmitteln $\bar{X}_{j,B}$ und $\bar{X}_{j,K}$ und Stichprobenvarianzen $\widehat{\text{Var}}(X_{j,B})$ und $\widehat{\text{Var}}(X_{j,K})$ für eine kontinuierliche Kovariable j .¹ Obwohl es keinen einheitlichen Schwellenwert für die standardisierte Differenz gibt, der ein erhebliches Ungleichgewicht anzeigt, gilt für kontinuierliche Variablen eine standardisierte (absolute) Differenz von weniger als 0.1 als Hinweis auf einen vernachlässigbaren Unterschied zwischen den Gruppen.

Die Balance Table weist also auf einen vernachlässigbaren Unterschied für `hours` hin und bestätigt den aus den Grafiken abgeleiteten Eindruck einer relevanten Differenzen für `age`.

3.1.2 Entropy Balancing

Entropy Balancing (Hainmueller 2012) ist eine weitere Methode zur Gewährleistung der Vergleichbarkeit von Behandlungs- und Kontrollgruppe anhand von Gewichten. Das Verfahren nutzt Konzepte aus der [Informationstheorie](#) um die Gewichte für Subjekte in der Kontrollgruppe so anzupassen, dass die Verteilung der Matchingvariablen in der Kontrollgruppe die Verteilung in der Behandlungsgruppe möglichst gut approximiert. Dies geschieht unter der Restriktion, dass bestimmte empirische Momente (meist Mittelwerte und Varianzen) der Matchingvariablen exakt übereinstimmen. Mathematisch werden die Gewichte für Kontrollgruppenbeobachtungen durch Minimierung der [Kullback-Leibler-Divergenz](#) zwischen den Verteilungen gefunden, wobei die Divergenz ein Maß für den Unterschied von Wahrscheinlichkeitsverteilungen ist.

Entropy Balancing ist im R-Paket `WeightIt` implementiert. Wir zeigen, wie die benötigten Gewichte für eine Schätzungen des ATT im Website-Beispiel mit `WeightIt::wightit()` bestimmt werden können. Über das Argument `moments` legen wir fest, dass die Gewichte unter der Restriktion übereinstimmender Mittelwerte aller Matching-Variablen zwischen Behandlungs- und Kontrollgruppe erfolgen soll.

¹Siehe P. Austin (2011) für einen Überblick zu Balance-Statistiken.

```
library(WeightIt)

# Gewichte für Entropy Balancing
(
  W1 <- weightit(
    dark_mode ~ age + male + hours,
    data = darkmode,
    method = "ebal",
    estimand = "ATT",
    moments = 1
  )
)
```

A weightit object

- method: "ebal" (entropy balancing)
- number of obs.: 300
- sampling weights: none
- treatment: 2-category
- estimand: ATT (focal: 1)
- covariates: age, male, hours

Wir schätzen den Behandlungseffekt nach Entropy Balancing mit gewichteter Regression.

```
# Mittelwert-Vergleich mit lm()
fit <- lm(
  formula = read_time ~ dark_mode,
  data = darkmode,
  weights = W1$weights
)
summary(fit)
```

Call:

```
lm(formula = read_time ~ dark_mode, data = darkmode, weights =
W1$weights)
```

Weighted Residuals:

	Min	1Q	Median	3Q	Max
	-16.6662	-2.3552	0.8786	2.9583	27.1840

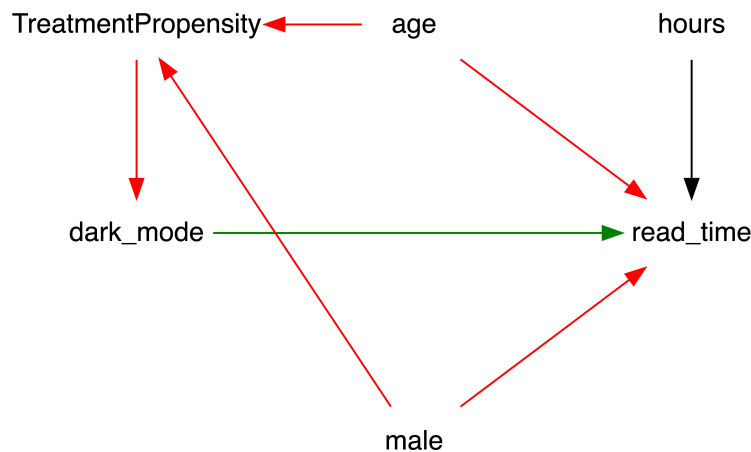


Abbildung 3.3: Propensity im Website-Design-Beispiel

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	17.7598	0.4093	43.394	<2e-16 ***
dark_mode	0.9704	0.5807	1.671	0.0958 .

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 5.029 on 298 degrees of freedom

Multiple R-squared: 0.009283, Adjusted R-squared: 0.005958

F-statistic: 2.792 on 1 and 298 DF, p-value: 0.09578

Beachte, dass die von `summary()` berechneten Standardfehler bei Entropy Balancing ungültig sind. In Abschnitt Kapitel 3.4 erläutern wir die Berechnung von Standardfehlern für Matching-Schätzer mit dem Bootstrap.

3.1.3 Mehrere Matching-Variablen und der Propensity Score

Bei mehreren Backdoor-Variablen kann eine Gewichtung anhand der Behandlungswahrscheinlichkeit (*Treatment Propensity*) erfolgen. Die Idee hierbei ist, dass der DGP wie in Abbildung 3.3 dargestellt werden kann.

Hierbei beeinflussen die Backdoor-Variablen *age* und *male* die Behandlungsvariable *dark_mode* lediglich durch die Behandlungswahrscheinlichkeit *Treatment Propensity*. Diese Darstellung zeigt, dass die mehrdimensionale Information bzgl. der Ähnlichkeit von Subjekten hinsichtlich der beobachteten Kovariablen in

einer einzigen Variable zusammengefasst werden kann. Die Backdoor-Pfade können daher geschlossen werden, indem wir Subjekte anhand von *Treatment Propensity* derart gewichten, dass beide Gruppen hinsichtlich der Verteilung der Behandlungswahrscheinlichkeit vergleichbar sind. Betrachte erneut (3.1) und beachte, dass

$$Y_i = Y_i^{(1)} D_i + Y_i^{(0)} (1 - D_i). \quad (3.7)$$

Rosenbaum und Rubin (1983) zeigen, dass es hinsichtlich (3.1) äquivalent ist für die *Treatment Propensity* $P_i(X_i) := P(B_i = 1 | X_i = x)$ zu kontrollieren, d.h.

$$\{Y_i^{(1)}, Y_i^{(0)}\} \perp B_i | X_i \Leftrightarrow \{Y_i^{(1)}, Y_i^{(0)}\} \perp B_i | P_i(X_i). \quad (3.8)$$

Der Behandlungseffekt kann so als Differenz von gewichteten Gruppenmittelwerten berechnet werden, mit inversem Wahrscheinlichkeitsgewicht (IPW) $w_{i,B} = 1/P_i(X_i)$ für Beobachtungen in der Behandlungsgruppe und $w_{i,K} = 1/(1 - P_i(X_i))$ für Beobachtungen in der Kontrollgruppe,

$$\tau^{\text{IPW}} = \frac{1}{n} \sum_{i=1}^n \left[\frac{B_i Y_i}{P_i(X_i)} - \frac{(1 - B_i) Y_i}{1 - P_i(X_i)} \right]. \quad (3.9)$$

Grundsätzlich ist *Treatment Propensity* eine nicht beobachtbare Variable und muss daher aus den Daten geschätzt werden. Eine geschätzte Behandlungswahrscheinlichkeiten $\hat{P}_i(X_i)$ wird als *Propensity Score* bezeichnet. In der Praxis erfolgt die Schätzung von *Propensity Scores* meist mit logistischer Regression. Ein erwartungstreuer Schätzer des ATE ist

$$\hat{\tau}^{\text{IPW}} = \frac{1}{n} \sum_{i=1}^n \left[\frac{B_i Y_i}{\hat{P}_i(X_i)} - \frac{(1 - B_i) Y_i}{1 - \hat{P}_i(X_i)} \right]. \quad (3.10)$$

Hirano, Imbens, und Ridder (2003) diskutieren Alternativen zu (3.10) für die Schätzung anderer Typen von Behandlungseffekten.

Wir schätzen nachfolgend die *Propensity Scores* für unser Anwendungsbeispiel, erläutern die Berechnung der Gewichte sowie die Schätzung von Behandlungseffekten mit gewichteter Regression. Hierbei betrachten wir eine Variante von (3.10) mit normalisierten Gewichten $\tilde{w}_{i,B} = w_{i,B} / \sum_i w_{i,B}$ und $\tilde{w}_{i,K} = w_{i,K} / \sum_i w_{i,K}$ die sich jeweils zu 1 summieren.² Dies ergibt den Hájek-

²Eine Normalisierung der Gewichte reduziert die Varianz des Schätzers, vgl. Hirano, Imbens, und Ridder (2003)

Schätzer³

$$\hat{\tau}_N^{\text{IPW}} = \frac{\sum_i \tilde{w}_{i,B} Y_i}{\sum_i \tilde{w}_{i,B}} - \frac{\sum_i \tilde{w}_{i,K} Y_i}{\sum_i \tilde{w}_{i,K}}. \quad (3.11)$$

Zunächst Schätzen wir ein logistisches Regressionsmodell mit `age`, `male` und `hours` als erklärende Variablen für `dark_mode`.

```
# Logit-Modell mit 'glm()' schätzen
(  
  darkmode_ps_logit <- glm(  
    formula = dark_mode ~ age + male + hours,  
    data = darkmode,  
    family = binomial  
  )  
)
```

```
Call:  glm(formula = dark_mode ~ age + male + hours, family =  
binomial,  
        data = darkmode)
```

Coefficients:

(Intercept)	age	male	hours
2.330e+00	-7.330e-02	1.623e+00	-9.293e-05

Degrees of Freedom: 299 Total (i.e. Null); 296 Residual

Null Deviance: 415.9

Residual Deviance: 346.4 AIC: 354.4

Die *Propensity Scores* erhalten wir als angepasste Werte aus der Regression `darkmode_ps_logit` mit `fitted()`. Wir erweitern den Datensatz mit den Ergebnissen.

```
# Datensatz um Propensity Scores erweitern
(  
  darkmode_probs <-  
    darkmode %>%  
    mutate(  
      # ...  
    )  
)
```

³Siehe Hájek (1971).

```

    PS = fitted(darkmode_ps_logit)
  )
)

```

A tibble: 300 x 6

	read_time	dark_mode	male	age	hours	PS
	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
1	14.4	0	0	43	65.6	0.304
2	15.4	0	1	55	125.	0.478
3	20.9	1	0	23	643.	0.642
4	20	0	0	41	129.	0.335
5	21.5	1	0	29	190.	0.547
6	19.5	0	0	64	185.	0.0849
7	22	1	0	18	334.	0.727
8	17.4	0	0	53	279.	0.171
9	23.6	0	0	59	1303.	0.108
10	15.7	0	0	53	16.1	0.174

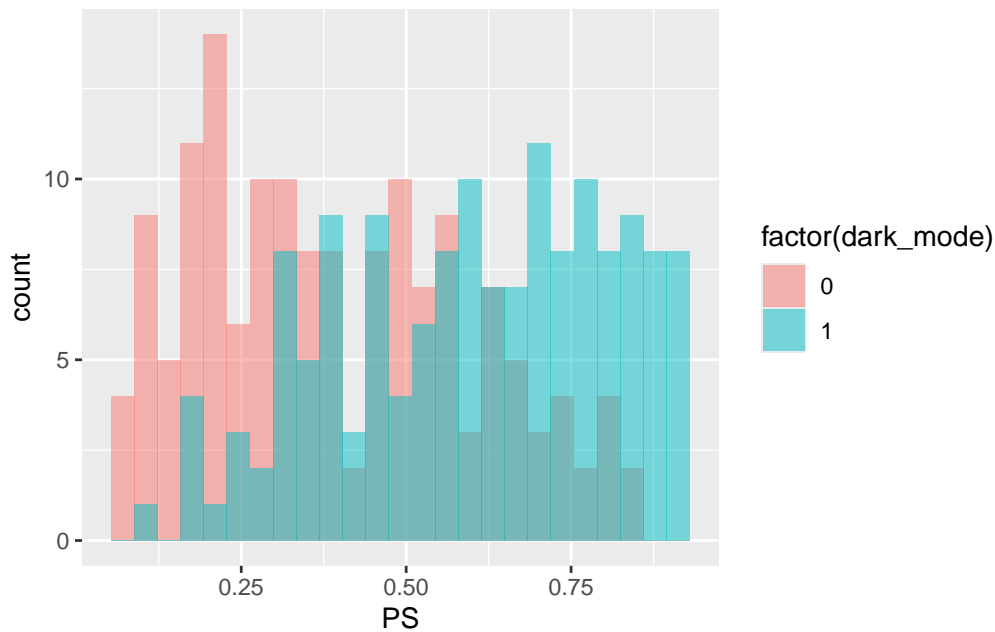
i 290 more rows

Zur Beurteilung der Überlappung (vgl. Annahme (3.2)) können wir die Verteilung der *Propensity Scores* nach Behandlungs-Indikator mit Histogrammen visualisieren.

```

# Überlappung prüfen:
# Histogramme der PS nach Treatment-Indikator
darkmode_probs %>%
  ggplot(
    mapping = aes(
      x = PS,
      fill = factor(dark_mode)
    )
  ) +
  geom_histogram(
    alpha = .5,
    bins = 25,
    position = "identity"
  )

```



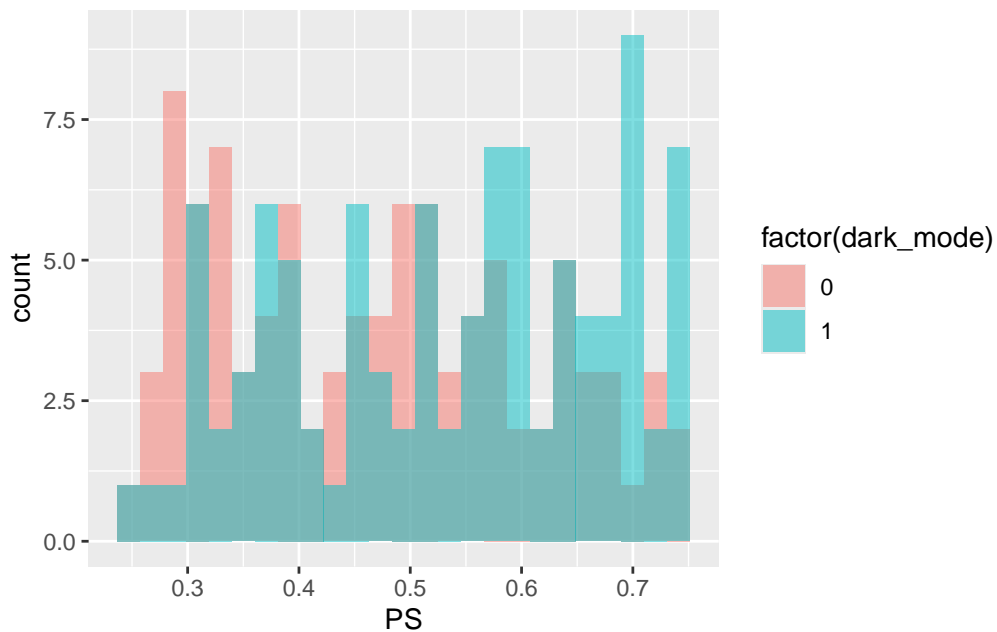
Ein Vergleich der Histogramme zeigt, dass die Überlappung der *Propensity Scores* in der linken Flanken der Verteilungen der Kontrollgruppe und in der rechten Flanke der Behandlungsgruppe schlechter wird. Wir entfernen zunächst Beobachtungen aus der Stichprobe deren *Propensity Scores* wenig bzw. keine Überlappung aufweisen.

```
# Datensatz nach PS trimmen
darkmode_probs <- darkmode_probs %>%
  filter(
    between(
      x = PS,
      left = .25,
      right = .75
    )
  )
```

```
# Überlappung nach trimming prüfen:
# Dichteschätzung der PS nach Treatment-Indikator
darkmode_probs %>%
  ggplot(
    mapping = aes(
      x = PS,
      fill = factor(dark_mode))
  ) +
```



```
geom_histogram(  
  alpha = .5,  
  bins = 25,  
  position = "identity"  
)
```



IPWs anhand der *Propensity Scores* können schnell mit der Vorschrift

$$\text{IPW} = \frac{\text{dark_mode}}{\text{PS}} + \frac{1 - \text{dark_mode}}{1 - \text{PS}}, \quad (3.12)$$

berechnet werden.

```
# Datensatz um IPWs erweitern  
darkmode_IPW <- darkmode_probs %>%  
  mutate(  
    IPW = dark_mode / PS + (1 - dark_mode) / (1 - PS)  
  )  
  
darkmode_IPW %>%  
  select(IPW)
```

```
# A tibble: 194 x 1  
  IPW  
  <dbl>
```

```

1  1.44
2  1.91
3  1.56
4  1.50
5  1.83
6  1.38
7  1.43
8  1.47
9  2.71
10 1.42
# i 184 more rows

```

Eine Schätzung des durchschnittlichen Behandlungseffekts gemäß (3.11) implementieren wir mit `dplyr`.

```

darkmode_IPW %>%
  group_by(dark_mode) %>%
  mutate(w = IPW / sum(IPW)) %>%
  summarise(weighted_mean = sum(read_time * w)) %>%
  summarise(diff = diff(weighted_mean))

```

```

# A tibble: 1 x 1
  diff
  <dbl>
1  1.90

```

Diese Schätzung des Behandlungseffekts ist äquivalent zur gewichteten KQ-Schätzung anhand eines einfachen linearen Regressionsmodells.

```

# Mit IPWs gewichteter KQ-Schaetzer berechnet den ATE
model_ipw <- lm(
  formula = read_time ~ dark_mode,
  data = darkmode_IPW,
  weights = IPW
)

summary(model_ipw)

```

Call:

```
lm(formula = read_time ~ dark_mode, data = darkmode_IPW, weights
= IPW)
```

Weighted Residuals:

	Min	1Q	Median	3Q	Max
	-18.5086	-4.4579	0.6096	4.1345	20.4566

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	17.9709	0.4942	36.362	< 2e-16 ***
dark_mode	1.9011	0.6952	2.735	0.00683 **

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 6.913 on 192 degrees of freedom

Multiple R-squared: 0.03749, Adjusted R-squared: 0.03248

F-statistic: 7.478 on 1 and 192 DF, p-value: 0.006829

Unsere Schätzung des ATE ist der geschätzte Koeffizient von `dark_mode`. Die ausgegebenen Standardfehler und Inferenzstatistiken sind jedoch *ungültig* aufgrund der Gewichtung mit IPWs, den inversen *geschätzten* Wahrscheinlichkeiten für eine Behandlung. Der Grund hierfür ist, dass die Berechnung der Standardfehler in `summary()` die zusätzliche Unsicherheit durch die geschätzten *Propensity Scores* nicht berücksichtigt! Später im Kapitel erläutern wir die Berechnung gültiger Standardfehler für IPW-Schätzer basierend auf *Propensity Scores* mit dem Bootstrap.

3.2 Selektierende Matching-Verfahren

Das grundsätzliche Konzept von selektierendem Matching wird in der nachstehenden interaktiven Grafik veranschaulicht. Hier betrachten wir beobachtete Ausprägungen von zwei (unabhängig und identisch verteilten) Matching-Variablen für Subjekte in der Behandlungsgruppe (blau) sowie Kontrollgruppe (rot). Als Matches qualifizieren sich sämtliche Beobachtungen der anderen Gruppe, deren [Euklidische Distanz](#) zu dem ausgewählten Punkt das über den Slider eingestellte Maximum *Caliper* nicht überschreitet.⁴ Diese Region wird durch den gestrichelten Kreis gekennzeichnet und kann über den zugehörigen Slider angepasst werden. Per Klick auf eine Beobachtung werden Matches aus

⁴Es handelt sich hierbei um einen Spezialfall von Matching anhand der Mahalanobis-Distanz.

der anderen Gruppe durch eine verbindende Linie und farbliches Hervorheben kenntlich gemacht. Mit dem Slider für k wird festgelegt, dass nur die nahesten k qualifizierten Beobachtungen als Matches behandelt werden. Die Grafik illustriert insbesondere, dass Beobachtungen in Abhängigkeit von k und Caliper falls gewünscht mehrfach (s.g. Matching mit Zurücklegen) oder gar nicht gematcht werden können.

Für die nachfolgenden Code-Beispiele verwenden wir das R-Paket `MatchIt`. `MatchIt::matchit()` nutzt standardmäßig Eins-zu-Eins-Matching (ohne Zurücklegen) von Beobachtungen der Treatment-Gruppe mit Beobachtungen der Kontrollgruppe.⁵ Die für das Matching zu verwendenden Variablen werden über das Argument `formula` als Funktion des Behandlungsindikators definiert. `matchit()` bereitet das Objekt für eine Schätzung des ATT mit einer geeigneten Funktionen, s. `?matchit` und hier insb. die Erläuterungen der Argumente `replace = F`, `ratio = 1` und `estimand = "ATT"` für Details. Mit `cobalt::balt.tab()` erhalten wir eine *balance table* für den gematchten Datensatz.

Wir zeigen als nächstes, wie `MatchIt::matchit()` für Matching anhand der Regressoren `age`, `hours`, und `male` in unserem Website-Beispiel für unterschiedliche Varianten durchgeführt werden kann.

3.2.1 Exaktes Matching

Exaktes Matching ordnet einem Subjekt aus der Behandlungsgruppe ein oder mehrere Subjekte aus der Kontrollgruppe zu, wenn die beobachteten Ausprägung der Matching-Variablen *exakt* übereinstimmen. Hierbei muss die ‘Distanz’ zwischen den Ausprägung der Matching-Variablen folglich 0 sein. Dieses Verfahren findet meist bei ausschließlich diskret verteilten Merkmalen Anwendung. Bei kontinuierlich verteilten Merkmalen (vgl. die obige interaktive Grafik) sind exakte Matches zwar theoretisch unmöglich, ergeben sich jedoch in der Praxis aus der Datenerfassung, bspw. durch Rundungsfehler. In `matchit()` erhalten wir exaktes Ein-zu-eins-Matching mit `method = "exact"`.

```
library(MatchIt)

# Exaktes Eins-zu-Eins-Matching durchführen
res_em <- matchit(
  formula = dark_mode ~ age + male + hours,
```

⁵Dieses Schema zielt auf eine Schätzung des ATT ab.

```
data = darkmode,
estimand = "ATT",
method = "exact"
)
```

Error in `matchit()`:
! No exact matches were found.

```
res_em
```

Error in eval(expr, envir, enclos): object 'res_em' not found

Aufgrund der kontinuierliche Verteilten Variable `hours` gibt es in unserem Website-Beispiel keine exakten Matches. Dieses Verfahren ist hier folglich ungeeignet.

3.2.2 Coarsened Exact Matching

Bei dieser Methode werden kontinuierliche Matching-Variablen grob (Engl. *coarse*) klassiert, ähnlich wie bei einem Histogramm. Diese Diskretisierung ermöglicht es exakte Übereinstimmungen zwischen Behandlungs- und Kontrollgruppenbeobachtungen hinsichtlich ihrer klassierten Ausprägungen zu finden. Sowohl Behandlungs- als auch Kontrollbeobachtungen die mindestens einen exakten Match haben, werden Teil des gematchten Datensatzes. In `matchit()` wird Coarsened Exact Matching mit `method = "cem"` durchgeführt. Über das Argument `cutpoints` geben wir an, dass `hours` in 6 Klassen und `age` in 4 Klassen eingeteilt werden soll.⁶ Mit `k1k = TRUE` erfolgt Eins-zu-eins-Matching: Bei mehreren exakten Matches wird die Beobachtung mit der geringsten Mahalanobis-Distanz (für die unklassierten Matching-Variablen) gewählt.

```
# Coarsened Exact Matching
res_CEM <- matchit(
  formula = dark_mode ~ age + male + hours,
  data = darkmode,
  estimand = "ATT",
  method = "cem",
  k2k = TRUE,
  cutpoints = list(
```

⁶Diese Werte wurden ad-hoc gewählt da sie zu einem guten Ergebnis führen.

```

    "hours" = 6,
    "age" = 4
  )
)
res_CEM

```

A `matchit` object

- method: Coarsened exact matching
- number of obs.: 300 (original), 164 (matched)
- target estimand: ATT
- covariates: age, male, hours

```

# Balance-Table Coarsened Exact Matching
bal.tab(res_CEM)

```

Balance Measures

	Type	Diff.Adj
age	Contin.	0.0106
male	Binary	0.0000
hours	Contin.	-0.0135

Sample sizes

	Control	Treated
All	151	149
Matched	82	82
Unmatched	69	67

Mit Coarsened Exact Matching erhalten wir einen Datensatz mit 82 Beobachtungen und guter Balance.

3.2.3 Matching mit der Mahalanobis-Distanz

Die Euklidische Distanz misst den direkten Abstand zwischen zwei Punkten und ist nicht invariant gegenüber Transformationen, insbesondere bei unterschiedlichen Skalierungen und bei Korrelation der Matching-Variablen. Die Mahalanobis-Distanz hingegen ist ein standardisiertes Distanzmaß, das unter Berücksichtigung der Varianz-Kovarianz-Struktur der Daten angibt, wie viele Standardabweichungen zwei Datenpunkte voneinander entfernt sind. Die

Mahalanobis-Distanz ist invariant gegenüber linearen Transformationen (Skalierung, Translation und Rotation) der Daten und bietet ein genaueres Maß für die Unähnlichkeit zweier Beobachtungen hinsichtlich ihrer Ausprägungen der Matching-Variablen.

Betrachte die Datenpunkte $P_1 = (X_1, Y_1)'$ und $P_2 = (X_2, Y_2)'$ für die Matching-Variablen X und Y . Die Mahalanobis-Distanz zwischen P_1 und P_2 ist definiert als

$$d_M(P_1, P_2) = \sqrt{(P_1 - P_2)' S^{-1} (P_1 - P_2)},$$

wobei S die Varianz-Kovarianz-Matrix von X und Y ist. Die Mahalanobis-Distanz $d_M(P_1, P_2)$ ist also die Euklidische Distanz zwischen den standardisierten Datenpunkten.

In empirischen Anwendungen ersetzen wir die (unbekannten) Komponenten der Varianz-Kovarianz-Matrix durch Stichprobenvarianten. Dies ergibt die Formel

$$\hat{d}_M(P_1, P_2) = \sqrt{\begin{pmatrix} X_1 - X_2 \\ Y_1 - Y_2 \end{pmatrix}' \begin{pmatrix} \widehat{\text{Var}}(X^2) & \widehat{\text{Cov}}(X, Y) \\ \widehat{\text{Cov}}(X, Y) & \widehat{\text{Var}}(Y^2) \end{pmatrix}^{-1} \begin{pmatrix} X_1 - X_2 \\ Y_1 - Y_2 \end{pmatrix}}.$$

Die nachstehende interaktive Grafik zeigt Beobachtungen zweier Matching-Variablen, die aus einer bivariaten Normalverteilung mit positiver Korrelation generiert wurden. Diese bivariate Verteilung ist identisch für Beobachtungen aus der Kontrollgruppe (rot) und Beobachtungen aus der Behandlungsgruppe (blau). Für die ausgewählte Beobachtung aus der Behandlungsgruppe (schwarzer Rand) werden potentielle Matches in der Kontrollgruppe innerhalb der vorgegebenen Mahalanobis-Distanz in Cyan kenntlich gemacht. Beachte, dass die Mahalanobis-Distanz Varianzen und Kovarianzen der Daten berücksichtigt, sodass die gematchten Beobachtungen in einem elliptischen Bereich um die betrachtete behandelte Beobachtung liegen. Eine Euklidische Distanz hingegen (gestrichelte Linie) ignoriert die Skalierung der Daten.

Für Eins-zu-Eins-Matching im Website-Beispiel anhand der Mahalanobis-Distanz mit `matchit()` setzen wir `distance = "mahalanobis"` und wählen `method = "nearest"`. Mit diesen Parametern wird jeder Behandlung aus der Behandlungsgruppe die gemäß d_M am ehesten vergleichbarste Beobachtung aus der Kontrollgruppe zugewiesen, wobei keine mehrfachen Matches zulässig sind.



```
# 1:1 Mahalanobis-Distanz-Matching
res_maha <- matchit(
  formula = dark_mode ~ age + male + hours,
  data = darkmode,
  estimand = "ATT",
  distance = "mahalanobis",
  method = "nearest"
)
res_maha
```

A matchit object

- method: 1:1 nearest neighbor matching without replacement
- distance: Mahalanobis
- number of obs.: 300 (original), 298 (matched)
- target estimand: ATT
- covariates: age, male, hours

```
# Balance-Table für 1:1 Mahalanobis-Matching
bal.tab(res_maha)
```

Balance Measures

	Type	Diff.Adj
age	Contin.	-0.5826
male	Binary	0.3154
hours	Contin.	0.0106

Sample sizes

	Control	Treated
All	151	149
Matched	149	149
Unmatched	2	0

Die Ergebnisse zeigen, dass für sämtliche 149 Beobachtungen aus der Behandlungsgruppe ein individueller Match in der Kontrollgruppe gefunden werden konnte. Es werden lediglich 2 Beobachtungen der 151 Beobachtungen in der Kontrollgruppe nicht gematcht.

Entsprechend zeigt die Balance-Table eine ähnliche Diskrepanz beider Gruppen hinsichtlich der Matching-Variablen an.

Mahalanobis-Distanz mit Caliper .25 für Propensity Scores basierend auf logistischer Regression

Für ein strengeres Matching-Kriterium kann ein *Caliper*, d.h. eine maximal zulässige Distanz, herangezogen werden. Die Mahalanobis-Distanz hat jedoch keine einheitliche Skala: Ob eine Distanz als groß oder klein betrachtet werden kann, hängt von der Anzahl der Matching-Variablen und dem Überlappungsgrad zwischen den Gruppen ab. Daher wird die Beschränkung durch einen Caliper nicht auf \hat{d}_M sondern auf Propensity Scores angewendet.

Im nächsten Code-Beispiel spezifizieren wir mit `distance = "glm"`, dass Propensity Scores gemäß der Vorschrift in `formula` geschätzt werden. Mit `mahvars = ~ age + male + hours` legen wir die Matching-Variablen für die Berechnung von \hat{d}_M fest. `caliper = .25` legt fest, dass lediglich Beobachtungen der Kontrollgruppe bei einer absoluten Differenz der Propensity Scores von höchstens 0.25 Standardabweichungen als Match für eine Beobachtung in der Behandlungsgruppe qualifiziert sind.

```
#| context: setup
# Logit-Modell mit 'glm()' schätzen
(
  darkmode_ps_logit <- glm(
    formula = dark_mode ~ age + male + hours,
    data = darkmode,
    family = binomial
  )
)
```

```
# Mahalanobis-Matching mit PS-Caliper
res_mahaC <- matchit(
  formula = dark_mode ~ age + male + hours,
  data = darkmode,
  distance = "glm",
  estimand = "ATT",
  method = "nearest",
  mahvars = ~ age + male + hours,
  caliper = .25
)
res_mahaC
```

A `matchit` object

- method: 1:1 nearest neighbor matching without replacement
- distance: Mahalanobis [matching]
Propensity score [caliper]
 - estimated with logistic regression
- caliper: <distance> (0.058)
- number of obs.: 300 (original), 208 (matched)
- target estimand: ATT
- covariates: age, male, hours

```
# Balance Table  
bal.tab(res_mahaC)
```

Balance Measures

	Type	Diff.Adj
distance	Distance	0.1812
age	Contin.	-0.1176
male	Binary	0.0481
hours	Contin.	-0.0001

Sample sizes

	Control	Treated
All	151	149
Matched	104	104
Unmatched	47	45

Die Balance-Table zeigt einen deutlichen Effekt der Beschränkung qualifizierter Beobachtungen durch `caliper = .25`: Aufgrund der oberen Grenze für die Propensity-Score-Differenz von 0.058 wird für lediglich 104 Beobachtungen aus der Behandlungsgruppe ein individueller Match in der Kontrollgruppe gefunden.⁷ Weiterhin finden wir eine verbesserte Balance für den gematchten Datensatz.

3.2.4 Propensity Score Matching

Eine gängige Variante ist Matching ausschließlich anhand von Propensity Scores innerhalb eines Calipers.

⁷Die durch `caliper` implizierte Obergrenze ergibt sich als `.25 * sd(fitted(darkmode_ps_logit)))`.

```
# 1:1 Matching mit PS und Caliper
res_PSC <- matchit(
  formula = dark_mode ~ age + male + hours,
  data = darkmode,
  estimand = "ATT",
  distance = "glm",
  method = "nearest",
  caliper = .25
)
res_PSC
```

A matchit object

- method: 1:1 nearest neighbor matching without replacement
- distance: Propensity score [caliper]
 - estimated with logistic regression
- caliper: <distance> (0.058)
- number of obs.: 300 (original), 208 (matched)
- target estimand: ATT
- covariates: age, male, hours

```
# Balance Table
bal.tab(res_PSC)
```

Balance Measures

	Type	Diff.Adj
distance	Distance	0.1640
age	Contin.	-0.0976
male	Binary	0.0481
hours	Contin.	0.0134

Sample sizes

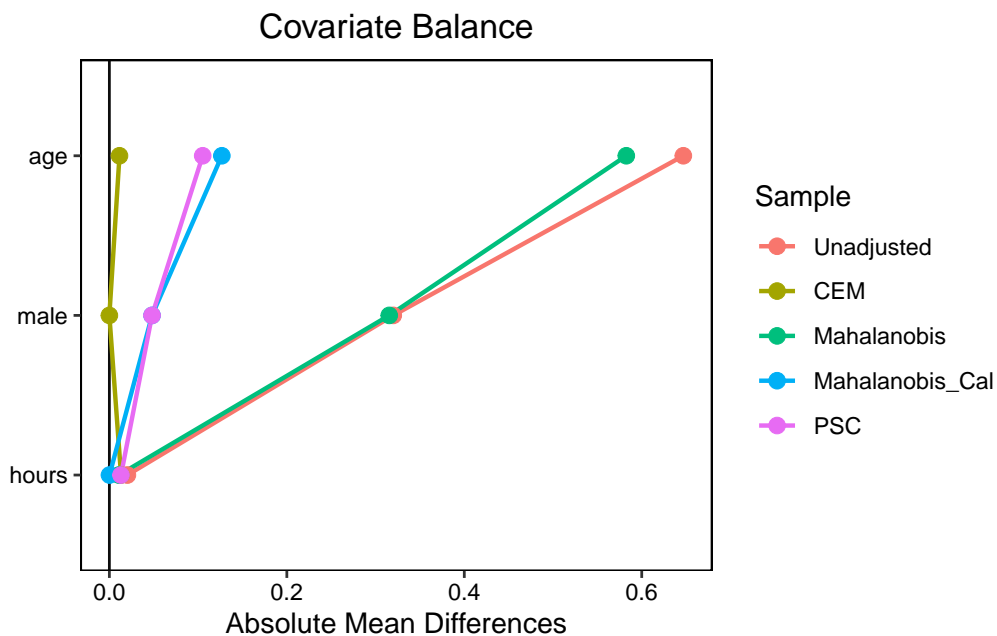
	Control	Treated
All	151	149
Matched	104	104
Unmatched	47	45

Laut Balance-Table führt Eins-zu-Eins-Matching basierend auf Propensity Scores zu einem Datensatz mit 104 gematchten Beobachtungen in der Behandlungsgruppe. Hinsichtlich der standardisierten Mittelwertdifferenz (Diff.Adj) erzielt diese Methode die beste Balance unter den betrachteten Ansätzen.

Vergleich der Balance verschiedener Verfahren mit Love-Plot

Standardisierte Mittelwertdifferenzen für verschiedene Matching-Verfahren können grafisch mit einem Love-Plot (Love 2004) veranschaulicht werden. Hierzu nutzen wir `cobalt::love.plot()` und übergeben die mit `matchit()` generierten Objekte im Argument `weights`.

```
# Love-Plot für
love.plot(
  x = dark_mode ~ age + male + hours,
  weights = list(
    CEM = res_CEM,
    Mahalanobis = res_maha,
    Mahalanobis_Cal = res_mahaC,
    PSC = res_PSC
  ),
  data = darkmode,
  line = T,
  # absolute Mittelwertdifferenz plotten
  abs = T
)
```



Die Grafik zeigt, dass Coarsened Exact Matching (CEM) unter allen betrachteten Verfahren die Stichprobe mit der besten Balance ergibt. Diesen gematchten Datensatz erhalten wir mit `MatchIt::match.data()`.

```
# gematchten Datensatz zuweisen
darkmode_matched_CEM <- match.data(res_CEM)
head(darkmode_matched_CEM)
```

```
# A tibble: 6 x 7
```

	read_time	dark_mode	male	age	hours	weights	subclass
	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<fct>
1	15.4	0	1	55	125.	1	26
2	20.9	1	0	23	643.	1	70
3	21.5	1	0	29	190.	1	79
4	22	1	0	18	334.	1	80
5	17.4	0	0	53	279.	1	11
6	20.4	0	0	43	138.	1	9

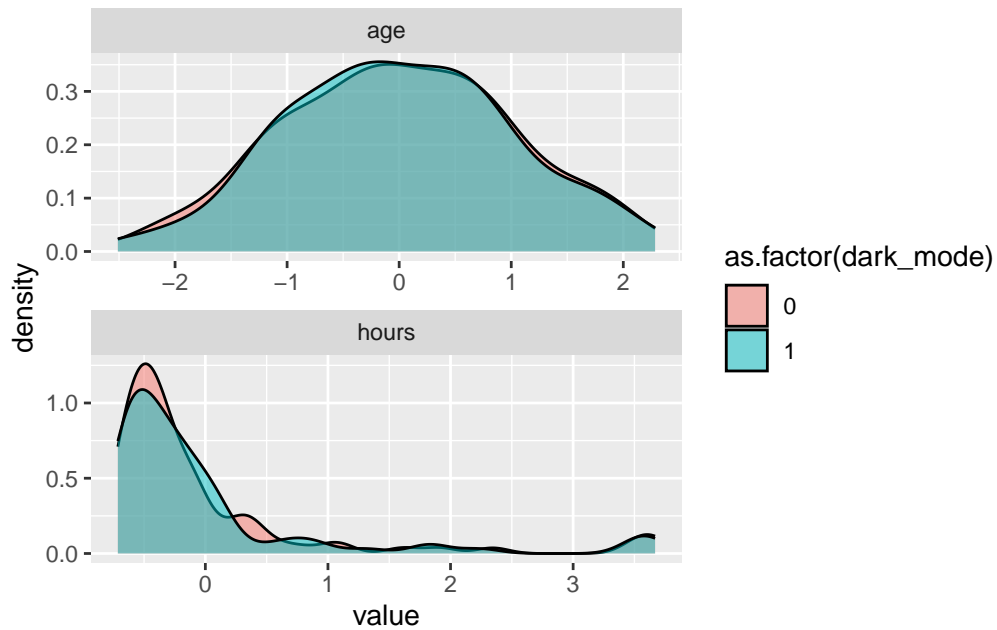
`darkmode_matched` enthält Gewichte (`weights`) für die jeweilige Gruppe zu denen gemachte Beobachtungen gehören (`subclass`). Dies ist relevant, falls Beobachtungen mehrfach gematcht werden. Wegen Eins-zu-eins-Matching *ohne* Zurücklegen gibt es in unserem Beispiel 82 Beobachtungspaare und sämtliche Gewichte sind 1. Die Berücksichtigung der Gewicht in den nachfolgenden Aufrufen von Schätzfunktionen (bspw. `lm()`) ist daher nicht nötig und erfolgt lediglich zur Illustration der grundsätzlichen Vorgehensweise.

Eine Wiederholung der grafischen Analyse in Kapitel 3.1 zeigt eine deutlich verbesserte Vergleichbarkeit hinsichtlich der Verteilung der Matching-Variablen in `darkmode_matched`.

```
darkmode_matched_CEM %>%
  group_by(dark_mode) %>%
  select(age, hours) %>%
  mutate_all(scale) %>%
  pivot_longer(cols = c(-dark_mode)) %>%

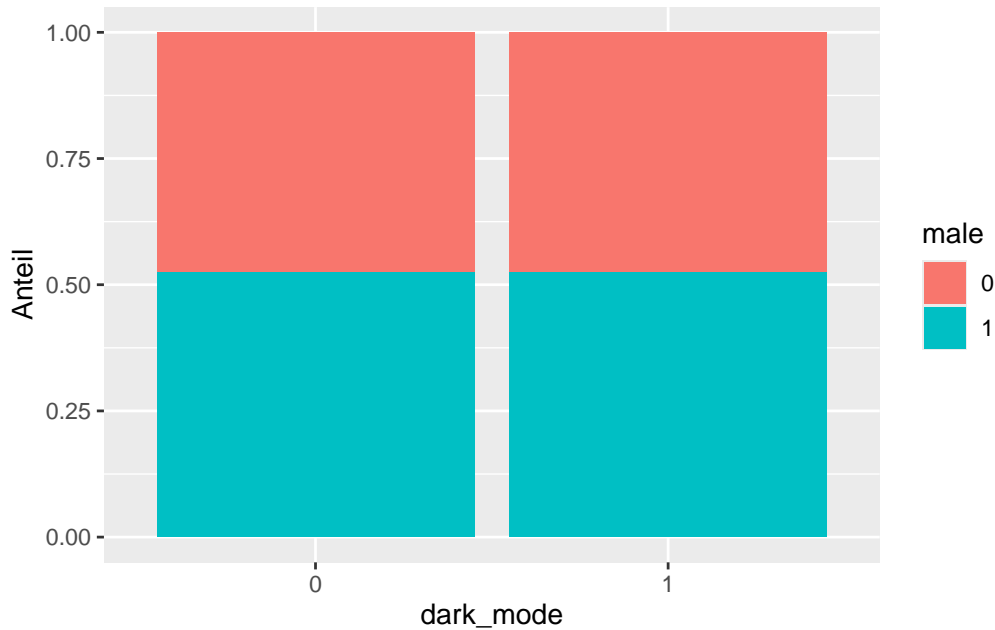
  ggplot(
    aes(x = value, fill = as.factor(dark_mode))
  ) +
  geom_density(alpha = .5) +
  facet_wrap(
    facets = ~ name,
    scales = "free",
```

```
nrow = 3
)
```



```
darkmode_matched_CEM %>%
  group_by(dark_mode) %>%
  mutate(
    male = as.factor(male),
    dark_mode = as.factor(dark_mode)
  ) %>%

  ggplot(
    aes(x = dark_mode, fill = male)
  ) +
  geom_bar(position = "fill") +
  ylab("Anteil")
```



Wir beobachten eine bessere Balance bei `age` und `hours`. Insbesondere ist `male` für Kontroll- und Behandlungsgruppe ausgeglichen.

3.3 Schätzung und Inferenz für den Behandlungseffekt nach Matching

Wir schätzen nun den Behandlungseffekt von `dark_mode` auf `read_time` für die mit CEM und Propensity Score Matching ermittelten Datensätzen und vergleichen die Ergebniss anschließend mit einer Regressionsschätzung ohne Matching.

Wir kombinieren die Matching-Verfahren mit linearer Regression, d.h. wir Schätzen den Behandlungseffekt anhand es gematchten Datensatzes als Mittelwertdifferenz nach zusätzlicher Kontrolle für die Matching-Variablen. Diese Kombination von Matching mit Regression wird in der Literatur als *Regression Adjustment* bezeichnet und ist insbesondere hilfreich, wenn Backdoors mit Matching geschlossen werden sollen, der kausale Effekt jedoch nur unter Verwendung einer nicht-trivialen Regressionsfunktion ermittelt werden kann. Zum Beispiel kann bei einer kontinuierlichen Behandlungsvariable und einem nicht-linearen Zusammenhang mit Y der kausale Effekt nicht durch einen bloßen Mittelwertvergleich erfasst werden, sondern erfordert eine adäquate Modellierung dieses Zusammenhangs in der Regressionsfunktion. Die zusätzliche Kontrolle für Matching-Variablen kann die Varianz der Schätzung verringern

und das Risiko einer verzerrten Schätzung abmildern, falls nach Matching noch Unterschiede in der Balance von Behandlungs- und Kontrollgruppe vorliegen.

```
# ATT mit linearem Modell schätzen: CEM Datensatz
ATT_mod_CEM <- lm(
  formula = read_time ~ age + male + hours + dark_mode,
  data = darkmode_matched_CEM,
  weights = weights
)

summary(ATT_mod_CEM)
```

Call:

```
lm(formula = read_time ~ age + male + hours + dark_mode, data =
darkmode_matched_CEM,
    weights = weights)
```

Residuals:

	Min	1Q	Median	3Q	Max
	-8.9310	-2.3856	-0.0194	2.5407	14.0020

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	16.6088953	1.5255529	10.887	< 2e-16 ***
age	0.0380982	0.0344699	1.105	0.27072
male	-4.0915725	0.6858131	-5.966	1.53e-08 ***
hours	0.0050129	0.0006977	7.185	2.45e-11 ***
dark_mode	1.6532234	0.6149439	2.688	0.00794 **

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 3.937 on 159 degrees of freedom

Multiple R-squared: 0.414, Adjusted R-squared: 0.3992

F-statistic: 28.08 on 4 and 159 DF, p-value: < 2.2e-16

```
# Datensatz für Propensity Score Matching zuweisen
darkmode_matched_PSC <- match.data(res_PSC)

# ATT mit linearem Modell schätzen: PSM Datensatz
```



```
ATT_mod_PSC <- lm(
  formula = read_time ~ age + male + hours + dark_mode,
  data = darkmode_matched_PSC,
  weights = weights
)

summary(ATT_mod_PSC)
```

Call:

```
lm(formula = read_time ~ age + male + hours + dark_mode, data =
darkmode_matched_PSC,
    weights = weights)
```

Residuals:

Min	1Q	Median	3Q	Max
-9.972	-2.605	-0.044	2.587	14.951

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	17.3654519	1.2762648	13.606	< 2e-16 ***
age	0.0283941	0.0301484	0.942	0.34741
male	-3.9858702	0.6480072	-6.151	4.03e-09 ***
hours	0.0046119	0.0006685	6.899	6.53e-11 ***
dark_mode	1.6346636	0.5769812	2.833	0.00507 **

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 4.141 on 203 degrees of freedom

Multiple R-squared: 0.3171, Adjusted R-squared: 0.3037

F-statistic: 23.57 on 4 and 203 DF, p-value: 5.098e-16

```
# ATT mit linearem Modell ohne Matching
ATT_mod_org <- lm(
  formula = read_time ~ age + male + hours + dark_mode,
  data = darkmode
)

summary(ATT_mod_org)
```

```

Call:
lm(formula = read_time ~ age + male + hours + dark_mode, data =
darkmode)

Residuals:
      Min       1Q   Median       3Q      Max
-10.2697  -2.6710   0.0164   2.5909  14.5739

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept) 16.859075   1.082303  15.577 < 2e-16 ***
age          0.051332   0.022215   2.311 0.02154 *
male        -4.485545   0.498957  -8.990 < 2e-16 ***
hours        0.004348   0.000516   8.427 1.58e-15 ***
dark_mode    1.385810   0.523793   2.646 0.00859 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 4.03 on 295 degrees of freedom
Multiple R-squared:  0.3434,    Adjusted R-squared:  0.3345
F-statistic: 38.57 on 4 and 295 DF,  p-value: < 2.2e-16

```

Beachte, dass für die gematchten Datensätze jeweils ein durchschnittlicher Behandlungseffekt für die Beobachtungen *mit* erfolgter Behandlung ermittelt wird: In sämtlichen oben gezeigten Matchibg-Verfahren werden mit `estimand = "ATT"` vergleichbarere Kontrollbeobachtungen für die behandelten Beobachtungen ermittelt. Wir schätzen den Effekt der Behandlung, indem wir die Ergebnisse von behandelten Personen mit denen von gematchten Personen vergleichen, die keine Behandlung erhalten haben. Diese Vergleichsgruppe dient als Ersatz für den hypothetischen Zustand der Behandlungsgruppe, wenn keine Behandlung erfolgt wäre. Dies entspricht der Definition eines ATT — ein average treatment effect *on the treated*.

3.3.1 Cluster-robuste Standardfehler

Für Matching-Verfahren sind die mit `summary()` berechneten Standardfehler (und damit auch Konfidenzintervalle, t-Statistiken und p-Werte) für den Behandlungseffekt *grundsätzlich ungültig*. Je nach Matching-Verfahren liegen unterschiedliche Quellen von Schätzunsicherheit vor, die bei der Berechnung

von Standardfehlern zusätzlich zu der “üblichen” Stichproben-Variabilität berücksichtigt werden müssen. Gründe hierfür sind der Matching-Prozess ansich oder weitere Unsicherheit durch die Schätzung zusätzlicher Parameter, etwa bei der Berechnung von Propensity Scores mit logistischer Regression. Eine weitere Ursache zusätzlicher Variation durch den Matching-Prozess, die wir bisher nicht näher betrachtet haben entsteht durch Zurücklegen, d.h. wenn Beobachtungen mehrfach gematcht werden können. Auch dieser Faktor wird in der von `summary()` verwendeten Formel für den Standardfehler des Effekt-Schätzers nicht berücksichtigt.

Die Standardfehlerberechnung für Matching-Schätzer von Behandlungseffekten ist ein Gegenstand aktueller methodischer Forschung. P. C. Austin und Small (2014) und A. Abadie und Spiess (2022) belegen die Gültigkeit von cluster-robusten Standardfehlern mit Clustering auf Ebene der Beobachtungsgruppen (`subclass` im output von `match.data()`) bei Matching ohne Zurücklegen. Für Matching anhand von Propensity Scores (auch mit Zurücklegen) zeigt Imbens (2016), dass ignorieren der zusätzlichen Unsicherheit durch die Schätzung der Propensity Scores zu konservativer Inferenz für den ATE anhand eines cluster-robusten Standardfehlerschätzers führt, jedoch ungültige Inferenz für die Schätzung des ATT bedeuten kann. Ähnlich zu P. C. Austin und Small (2014) deuten die Ergebnisse der Simulationsstudie von Bodory u. a. (2020) jedoch auf grundsätzlich bessere Eigenschaften der Schätzung hin, wenn die Standardfehler *nicht* für die Schätzung der Propensity Scores adjustiert werden.

Weiterhin ist die Kontrolle für Kovariablen mit Erklärungskraft für die Outcome-Variable und für die Matching-Variablen mit *Regression Adjustment* (vgl. Kapitel 3.3) für die Schätzung des Behandlungseffekts nach Matching eine etablierte Strategie, vgl. Hill und Reiter (2006) und A. Abadie und Spiess (2022). So können die Varianz der Schätzung und das Risiko einer Verzerrung der Standardfehler aufgrund verbleibender Imbalance von Behandlungs- und Kontrollgruppe nach Matching verringert werden.

Zur Demonstration von (cluster)-robuster Inferenz und für eine tabellarische Zusammenfassung der Ergebnisse nutzen wir die Pakete `marginaleffects` und `modelsummary`. Mit `marginaleffects::avg_comparisons()` können p-Werte und Kofindenzintervalle unter Berücksichtigung von robuster Standardfehlern und der Gewichte aus dem Matching-Verfahren berechnet werden.

```
library(marginaleffects)
```

```
# Inferenz: Multiple Regression, ungematchter Datensatz
```

```
(
  sum_orig <- avg_comparisons(
    model = ATT_mod_org,
    variables = "dark_mode",
    # Heteroskedastie-robuste SE:
    vcov = "HC3",
    # Identifizierung der Kontrollgruppe:
    newdata = subset(darkmode, dark_mode == 1)
  )
)
```

	Term Contrast	Estimate	Std. Error	z	Pr(> z)	S	2.5 %	97.5 %
dark_mode	1 - 0	1.39	0.537	2.58	0.00988	6.7	0.333	2.44

Columns: term, contrast, estimate, std.error, statistic,
p.value, s.value, conf.low, conf.high
Type: response

```
# Inferenz: Multiple Regression bei CEM
(
  sum_CEM <- avg_comparisons(
    model = ATT_mod_CEM ,
    variables = "dark_mode",
    # Cluster-robuste SE
    vcov = ~ subclass,
    newdata = subset(darkmode_matched_CEM, dark_mode == 1),
    wts = "weights"
  )
)
```

	Term Contrast	Estimate	Std. Error	z	Pr(> z)	S	2.5 %	97.5 %
dark_mode	1 - 0	1.65	0.549	3.01	0.00262	8.6	0.576	2.73

Columns: term, contrast, estimate, std.error, statistic,
p.value, s.value, conf.low, conf.high
Type: response

```
# Inferenz: Multiple Regression bei PSM
(
  sum_PSC <- avg_comparisons(
    model = ATT_mod_PSC ,
    variables = "dark_mode",
    vcov = ~ subclass,
    newdata = subset(darkmode_matched_PSC, dark_mode == 1),
    wts = "weights"
  )
)
```

	Term	Contrast	Estimate	Std. Error	z	Pr(> z)	S	2.5 %	97.5 %
dark_mode	1 - 0		1.63	0.565	2.89	0.00381	8.0	0.527	2.74

Columns: term, contrast, estimate, std.error, statistic,
p.value, s.value, conf.low, conf.high
Type: response

Wir fassen die Ergebnisse mit `modelsummary::modelsummary()` tabellarisch zusammen.

```
library(modelsummary)

# Tabellarische Zusammenfassung erzeugen
modelsummary(
  models = list(
    "Kein Matching" = sum_orig,
    "Coarsened Exact" = sum_CEM,
    "Propensity Scores" = sum_PSC
  ),
  stars = T,
  gof_map = "nobs",
  output = "gt"
```

```
) %>%
  tabopts()
```

	Kein Matching	Coarsened Exact	Propensity Scores
dark_mode	1.386** (0.537)	1.653** (0.549)	1.635** (0.565)
Num.Obs.	300	164	208

+ p < 0.1, * p < 0.05, ** p < 0.01, *** p < 0.001

3.4 Bootstrap-Schätzung kausaler Effekte bei Matching

Ein Bootstrap-Verfahren generiert mit Resampling (wiederholtes Ziehen mit Zurücklegen) aus dem Original-Datensatz (viele) künstliche Datensätze, für die der Schätzer (d.h. das gesamte Verfahren inkl. Matching!) jeweils berechnet wird. Die Verteilung der so gewonnenen Bootstrap-Schätzwerte approximiert die wahre, unbekannte Stichprobenverteilung des Schätzers des Behandlungseffekts. Mit dieser simulierten Verteilung können wir Inferenz betreiben: Wir können einen Bootstrap-Punktschätzer des Behandlungseffekts (Stichprobenmittel der Bootstrap-Schätzungen) sowie Standardfehler (Standardabweichung der der Bootstrap-Schätzungen) und p-Werte berechnen.

Der Bootstrap kann hilfreich sein, wenn unklar ist, wie Standardfehler für die Unsicherheit des Matching-Prozesses zu adjustieren sind, um gültige Inferenzstatistiken zu erhalten. A. Abadie und Imbens (2008) zeigen analytisch, dass der Standard-Bootstrap die Stichprobenverteilung für Schätzer kausaler Effekte anhand von gematchten Datensätzen (d.h. bei Zuordnung/Selektion von Beobachtungen mit Matching) nicht korrekt abbilden kann. Grundsätzlich problematisch hierbei ist, wenn der Bootstrap eine verzerrte Schätzung produziert und/oder zu kleine Standardfehler liefert. A. Abadie und Imbens (2008) belegen die Tendenz des Bootstraps zu *konservative* (d.h. zu große) Standardfehler zu produzieren. Simulationsnachweise (Bodory u. a. 2020; Hill und Reiter 2006; P. C. Austin und Small 2014; P. C. Austin und Stuart 2017) finden, dass Bootstrap-Standardfehler u.a. bei Propensity Score Matching mit Zurücklegen leicht konservativ sind somit das gewünschte nominale Signifikanzniveau eines Bootstrap-Hypothesentests nicht überschritten wird, weshalb der Standard-Bootstrap trotz seiner Schwächen in der empirischen Forschung oft angewendet wird.

Wir betrachten als nächstes einen Bootstrap-Algorithmus für Inferenz bezüglich eines kausalen Effekts nach Matching und demonstrieren die Schätzung anhand unseres Website-Beispiels für den ATT nach Propensity-Score-Matching.

Algorithmus: Bootstrap-Schätzer für Matching mit Regression Adjustment

1. Generiere eine Bootstrap-Stichprobe durch N Züge *mit Zurücklegen* aus der N -elementigen originalen Stichprobe.
2. Wende das Matching-Verfahren für die Bootstrap-Stichprobe an. Schätze den Behandlungseffekt β anhand der gematchten Stichprobe mit Regression. Speichere den Punktschätzer des Behandlungseffekts $\hat{\beta}_b^*$.
3. Führe die Schritte 1 und 2 für $b = 1, \dots, B$ aus, wobei B eine hinreichend große Anzahl von Bootstrap-Replikationen ist.
4. Berechne den Bootstrap-Schätzer des Behandlungseffekts $\bar{\beta}^* = \frac{1}{B} \sum_{b=1}^B \hat{\beta}_b^*$ und den Standardfehler $SE(\bar{\beta}^*) = \sqrt{\frac{1}{B-1} \sum_{b=1}^B (\hat{\beta}_b^* - \bar{\beta}^*)^2}$. Berechne Inferenz-Statistiken mit den üblichen Formeln.

Wir Implementieren nun einen Bootstrap-Schätzer des ATT im Website-Beispiel für Propensity-Score-Matching. Hierzu definieren wir eine R-Funktion `boot_fun()` für die Schritte 1 und 2 im obigen Algorithmus.

```
# Bootstrap-Funktion für Schritte 1 und 2
boot_fun <- function(
  data, # originale Stichprobe
  i     # Indexmenge f. Bootstrap-Stichprobe
) {

  # Bootstrap-Stichprobe
  boot_data <- data[i, ]

  # 1:1 PS Matching
  match_res <- matchit(
    dark_mode ~ age + hours + male,
    estimand = "ATT",
    distance = "glm",
    method = "nearest",
```

```

    caliper = .25,
    data = boot_data
  )

  # Gematchten Datensatz zuweisen
  darkmode_matched <- match.data(
    object = match_res,
    data = boot_data
  )

  # Outcome-Modell schätzen
  ATT_mod <- lm(
    formula = read_time ~ age + male + hours + dark_mode,
    data = darkmode_matched,
    weights = weights
  )

  # ATT-Schätzung zurückgeben
  return(
    ATT_mod$coefficients["dark_mode"]
  )
}

```

Wir berechnen nun eine Bootstrap-Schätzung des ATT von `dark_mode` auf `readingtime` nach Propensity Score Matching mit einem caliper von 0.25 sowie den zugehörigen Standardfehler und ein 95%-KI mit der zuvor definierten Funktion `boot_fun`.

```

library("boot")
set.seed(4321)

# Anz. Bootstrap-Replikationen
B <- 999

# Bootstrap durchführen
(
  boot_out <- boot(darkmode, boot_fun, R = B)
)

```


ORDINARY NONPARAMETRIC BOOTSTRAP

Call:

```
boot(data = darkmode, statistic = boot_fun, R = B)
```

Bootstrap Statistics :

	original	bias	std. error
t1*	1.634664	-0.08144129	0.5935746

Den Bootstrap-Schätzer des ATT sowie den Bootstrap-Standardfehler berechnen wir mit `mean()` und `sd()` anhand der 999 Bootstrap-Replikationen in `boot_out$t`.

Beachte, dass der Bootstrap-Schätzer des Behandlungseffekts nicht unmittelbar von `boot()` ausgegeben wird. `original` ist die Schätzung anhand der gesamten Stichprobe (d.h. ohne Bootstrap) und `bias` ist die Differenz zwischen dieser Schätzung und dem Mittelwert der Bootstrap-Schätzungen.

```
# Bootstrap-Schätzer für den Treatment-Effekt
mean(boot_out$t)
```

```
[1] 1.553222
```

```
# bootstrap - original = bias
mean(boot_out$t) - boot_out$t0
```

```
dark_mode
-0.08144129
```

Wir können prüfen, dass die Berechnung des Standardfehlers dem Stichprobenstandardabweichung der Bootstrap-Schätzungen entspricht.

```
# Bootstrap-Standardfehler
sd(boot_out$t)
```

```
[1] 0.5935746
```

Ein 95%-Konfidenzintervall für den kausalen Effekt erhalten wir mit `boot::boot.ci()`.⁸

```
# 95% Bootstrap-KI für den Treatment-Effekt
boot.ci(
  boot.out = boot_out,
  type = "bca",
  conf = .95
)
```

BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS

Based on 999 bootstrap replicates

CALL :

```
boot.ci(boot.out = boot_out, conf = 0.95, type = "bca")
```

Intervals :

Level	BCa
-------	-----

95%	(0.527, 2.839)
-----	------------------

Calculations and Intervals on Original Scale

Beachte, dass der Bootstrap-Standardfehler sowie das Bootstrap-Konfidenzintervall nahe der mit `avg_comarisons` berechneten Werte für `sum_PSC` sind.

Bootstrap-Standardfehler für IPW-Schätzer des ATE berechnen

Die Bootstrap-Funktion `boot_fun` kann leicht für eine Schätzung des Standardfehlers für den IPW-Schätzer des ATE aus Kapitel 3.1.3 angepasst werden. Statt einer Matching-Prozedur berechnen wir hierzu für B Bootstrap-Stichproben den Schätzer $\hat{\tau}^{\text{IPW}}$ mit Trimming der Propensity Scores.

```
# IPW estimation with regression adjustment
IPW_boot <- function(
  data,
  i
) {

  # Bootstrap-Stichprobe erstellen
```

⁸`type = "bca"` (bias-corrected accelerated) ist eine gängige Implementierung für die Berechnung des Konfidenz-Intervalls.

```

data_boot <- data %>%
  slice(i)

# Logistische Regression
glm_fit <- glm(
  formula = dark_mode ~ age + hours + male,
  data = data_boot,
  family = binomial
)

# Propensity Scores berechnen
data_boot <- data_boot %>%
  mutate(
    ps = predict(glm_fit, type = 'response')
  )

# Beobachtungen anhand Propensity Scores trimmen
data_boot <- data_boot %>%
  filter(
    between(
      x = ps,
      left = .2,
      right = .7
    )
  )

# IPW berechnen
data_boot <- data_boot %>%
  mutate(
    ipw = case_when(
      dark_mode == 1 ~ 1 / ps,
      dark_mode == 0 ~ 1 / (1 - ps)
    )
  )

# Gewichtete Mittelwerte der Gruppen
w_means <- data_boot %>%
  group_by(dark_mode) %>%
  summarize(m = weighted.mean(read_time, w = ipw)) %>%
  arrange(dark_mode)

```

```
# ATT-Schätzwert
return(w_means$m[2] - w_means$m[1])
}
```

```
set.seed(1234)
# Bootstrap für IPW durchführen
(
  b_IPW <- boot(
    data = darkmode,
    statistic = IPW_boot,
    R = 999
  )
)
```

ORDINARY NONPARAMETRIC BOOTSTRAP

Call:

```
boot(data = darkmode, statistic = IPW_boot, R = 999)
```

Bootstrap Statistics :

	original	bias	std. error
t1*	1.962849	0.06297654	0.6599891

```
# Bootstrap-Schätzer und Standardfehler berechnen
mean(b_IPW$t)
```

```
[1] 2.025826
```

```
sd(b_IPW$t)
```

```
[1] 0.6599891
```

In diesem Fall ist der Bootstrap-Standardfehler von ca. 0.66 gut mit dem anhand von `summary(model_ipw)` berechneten Standardfehler vergleichbar. Ein 95%-Konfidenzintervall für den ATE erhalten wir wie zuvor mit `boot.ci()`.

```
# 95% Bootstrap-KI für den ATE
boot.ci(b_IPW, type = "bca")
```

BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS

Based on 999 bootstrap replicates

CALL :

```
boot.ci(boot.out = b_IPW, type = "bca")
```

Intervals :

Level BCa

95% (0.563, 3.134)

Calculations and Intervals on Original Scale

3.5 Regression, Matching und Doubly-Robust-Schätzung

Als *Doubly-Robust-Schätzer* bezeichnet man Methoden, die bei Fehlspezifikationen im Matching-Verfahrens *oder* der funktionalen Form der Regressionsfunktion für die Outcome-Variable eine zuverlässige Schätzungen des kausalen Effekts ermöglichen.⁹ Unter der Voraussetzung, dass die verwendeten Matching-Variablen sämtliche Backdoors schließen, ist so mit Doubly-Robust-Schätzung eine konsistente Schätzung des Behandlungseffekts unter abgeschwächten Annahmen gewährleistet. Dies macht Doubly-Robust-Schätzer besonders nützlich in Forschungskontexten, in denen präzise Modellspezifikationen herausfordernd sind.

Der von Wooldridge (2010) vorgeschlagene Doubly-Robust-Schätzer (IPWRA) für den ATE erreicht seine vorteilhaften Eigenschaften durch eine geschickte Kombination von IPW und Regression Adjustment.

Algorithmus: IPWRA-Schätzer des ATE

Vgl. Wooldridge (2010).

1. Berechne IPW anhand von Propensity Score mit logistischer Regression unter Verwendung der Matching-Variablen.
2. Regression Adjustment:

⁹Bspw. kann eine falsche funktionale Form bei logistischer Regression eine verzerrte Schätzung von Propensity Scores und damit eine unzureichende Balance bedeuten.

- (a) Schätze lediglich für die *Behandlungsgruppe* eine mit den IPW gewichtete Regressionspezifikation der Outcome-Variable mit Kontrolle für die Matching-Variablen.
 - (b) Wiederhole Schritt A für *Kontrollgruppe*.
3. Berechne Vorhersagen der Outcome-Variable für die angepassten Modelle aus 2 (a) und 2 (b) anhand des *gesamten* Datensatzes.
4. Schätze den ATE als Mittelwert-Differenz der Vorhersagen aus Schritt 3.

Wir implementieren den Doubly-Robust-Schätzer des ATE von Wooldridge (2010) für das Website-Beispiel in der Funktion `IPWRA()` unter Adaption des Schemas von `IPW_boot()`.

```
# IPW estimation with regression adjustment
IPWRA <- function(
  data,
  i = i) {

  # Bootstrap-Stichprobe zuweisen
  b_data <- data %>%
    slice(i)

  # Logistische Regression
  glm_fit <- glm(
    formula = dark_mode ~ age + hours + male,
    data = b_data,
    family = binomial(link = 'logit')
  )

  # Propensity Scores berechnen
  b_data <- b_data %>%
    mutate(ps = predict(glm_fit, type = 'response'))

  # Propensity Scores trimmen
  b_data <- b_data %>%
    filter(
      between(
        x = ps,
```

```

    left = .2,
    right = .7
  )
)

# IPW berechnen
b_data <- b_data %>%
  mutate(
    IPW = case_when(
      dark_mode == 1 ~ 1 / ps,
      dark_mode == 0 ~ 1 / (1 - ps))
  )

# Regression Adjustment:

# Schätzung des Behandlungseffekts für die ges. Stichprobe
# mit Modell für Behandlungsgruppe
mtreat <- b_data %>%
  filter(dark_mode == 1) %>%
  lm(read_time ~ age + hours + male, data = ., weights =
    ↪ . $IPW) %>%
  predict(newdata = b_data) %>%
  mean()

# Schätzung des Behandlungseffekts für die ges. Stichprobe
# mit Modell für Kontrollgruppe
mcont <- b_data %>%
  filter(dark_mode == 0) %>%
  lm(read_time ~ age + hours + male, data = ., weights =
    ↪ . $ipw) %>%
  predict(newdata = b_data) %>%
  mean()

# Regression adjusted ATE-Schätzer
return(mtreat - mcont)
}

```

Schätzung des ATE mit IPWRA():

```
IPWRA(data = darkmode, i = 1:nrow(darkmode))
```

```
[1] 1.96847
```

Wie bei $\hat{\tau}^{\text{IPW}}$ gibt es keine formale Darstellung des Standardfehlers für den IPWRA-Schätzer, sodass auch hier der Bootstrap genutzt werden sollte.

```
set.seed(1234)

# Bootstrap für IPWRA durchführen
(
  b_IPWRA <- boot(
    data = darkmode,
    statistic = IPWRA,
    R = B
  )
)
```

ORDINARY NONPARAMETRIC BOOTSTRAP

Call:

```
boot(data = darkmode, statistic = IPWRA, R = B)
```

Bootstrap Statistics :

	original	bias	std. error
t1*	1.96847	-0.001270721	0.5960465

Wir berechnen die interessierenden Statistiken analog zu der Vorgehensweise in Kapitel 3.4.

```
# Bootstrap-IPWRA-Schätzer des ATE
mean(b_IPWRA$t)
```

```
[1] 1.967199
```



```
# Bootstrap-Standardfehler  
sd(b_IPWRA$t)
```

```
[1] 0.5960465
```

```
# 95%-Bootstrap-KI für den ATE  
boot.ci(b_IPWRA, type = "bca")
```

BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS

Based on 999 bootstrap replicates

CALL :

```
boot.ci(boot.out = b_IPWRA, type = "bca")
```

Intervals :

Level	BCa
-------	-----

95%	(0.787, 3.098)
-----	------------------

Calculations and Intervals on Original Scale

Die Bootstrap-Schätzung des ATE mit IPWRA ist mit den Ergebnissen für die Bootstrap-Variante von $\hat{\tau}^{\text{IPW}}$ vergleichbar, hat allerdings einen etwas kleineren Standardfehler und ist somit genauer.

4 Regression Discontinuity Designs

Regression Discontinuity Design (RDD) ist ein Ansatz für die Schätzung von Behandlungseffekten mit Regression, wenn durch einen experimentell oder natürlich gegebenen Umstand die Behandlung an einem Schwellenwert (c) einer *Laufvariable* (X) sprunghaft beeinflusst wird. Ein RDD-Schätzer wird so implementiert, dass lediglich Beobachtungen mit Ausprägungen von X , die knapp ober- oder knapp unterhalb von c liegen, berücksichtigt werden. Die zentrale Idee hierbei ist, dass Individuen nahe bei c im Durchschnitt ähnliche Merkmale aufweisen. Beobachtungen nahe c sind dann insbesondere hinsichtlich potentieller Backdoor-Variablen vergleichbar, sodass deren problematische Pfade geschlossen sind. Das kausale Diagramm in Abbildung 4.1 zeigt den grundsätzlichen Zusammenhang.

RDD isoliert Variation auf dem Pfad *Oberhalb* $C \rightarrow \text{Behandlung } B \rightarrow Y$. Somit können Backdoor-Pfade über X oder weitere (möglichweise unbeobachtbare) Confounder (Z) vermieden werden, siehe Abbildung 4.1. Der kausale Effekt wird dabei als (lokaler) durchschnittlicher Behandlungseffekt der Diskontinuität auf die Outcome-Variable (Y) anhand von Beobachtungen *nahe bei* c ermittelt.

Hinsichtlich der Beeinflussung der Behandlung unterscheiden wir zwischen *Sharp* und *Fuzzy* Regression Discontinuity Designs (SRDD/FRDD). Bei einem SRDD ist die Zuweisung der Behandlung *deterministisch*, d.h. der Schwellenwert in der Laufvariable ist eine harte Grenze für die Gruppenzugehörigkeit: Die *Wahrscheinlichkeit* der Behandlung p springt bei $X = c$ von $p = 0$ um $\Delta p = 1$ auf $p = 1$.

Bei einem FRDD ist die Zuordnung in Behandlungs- und Kontrollgruppe nicht perfekt durch den Schwellenwert c bestimmt: Die Behandlungswahrscheinlichkeit p springt bei $X = c$ um $\Delta p < 1$. Im FRDD können grundsätzlich also sowohl behandelte Subjekte als auch Kontroll-Beobachtungen auf beiden Seiten der Diskontinuität vorliegen – die Trennung der Gruppen ist “unscharf”¹. Dieser Umstand ist oft in empirischen Studien mit nicht-experimentellen Daten gegeben, wenn es neben der Überschreitung von c weitere Determinanten der Behandlung gibt (für die wir nicht kontrollieren können). Die Wahl zwischen

¹Engl. *fuzzy*.

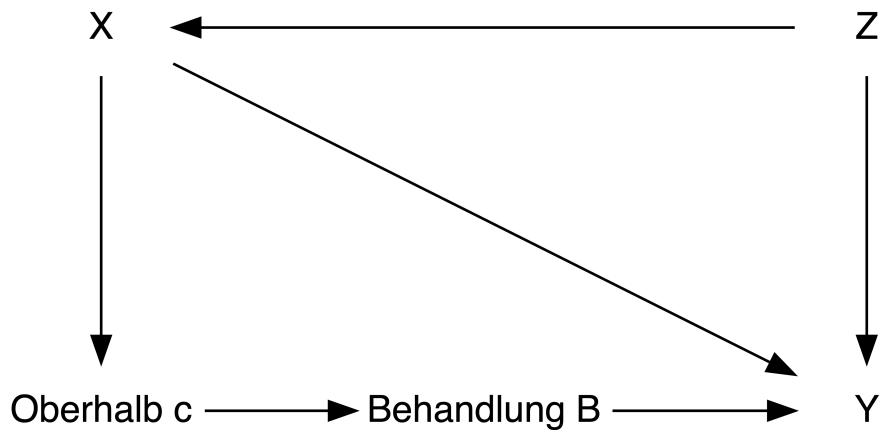


Abbildung 4.1: Kausales Diagramm für Sharp RDD

SRDD und FRDD hängt grundsätzlich vom datenerzeugenden Prozess und der Forschungsfrage ab.

4.1 Sharp Regression Discontinuity Design

Modell und funktionale Form

Die korrekte Spezifikation der funktionalen Form für ein RDD ist wichtig, um eine verzerrte Schätzung des Effekts zu vermeiden. Die einfachste Form eines SRDD kann anhand der linearen Regression

$$Y_i = \beta_0 + \beta_1 B_i + \beta_2 X_i + u_i \quad (4.1)$$

geschätzt werden, wobei B_i eine Dummy-Variable für das Überschreiten des Schwellenwertes c ist, d.h.

$$B_i = \begin{cases} 0 & X_i < c \\ 1 & X_i \geq c. \end{cases}$$

Damit ist B_i eine *deterministische* Funktion der Laufvariable X_i und zeigt die Zugehörigkeit zur Behandlungs- oder Treatmentgruppe an. Der Koeffizient β_1 misst den Behandlungseffekt.

Das Modell (4.1) unterstellt, dass X links- und rechtsseitig von c denselben Effekt auf Y hat. Diese Annahme ist restriktiv. Eine Alternative ist ein lineares

Interaktionsmodell

$$Y_i = \beta_0 + \beta_1 B_i + \beta_2(X_i - c) + \beta_3(X_i - c) \times B_i + u_i. \quad (4.2)$$

Das Modell (4.2) kann unterschiedliche lineare Effekte von X auf Y unterhalb (β_2) und oberhalb ($\beta_2 + \beta_3$) von c abbilden. Beachte, dass $(X_i - c)$ die um den Schwellenwert zentrierte Laufvariable ist, sodass β_1 wie in (4.1) den Unterschied des Effekts von X auf Y für Beobachtungen am Schwellenwert erfasst.

Um unterschiedliche nicht-lineare Zusammenhänge von X und Y unterhalb und oberhalb von c abzubilden, können (interargierte) Polynom-Terme in X verwendet werden. Häufig wird eine quadratische Regressionsfunktion genutzt,

$$Y_i = \beta_0 + \beta_1 B_i + \beta_2(X_i - c) + \beta_3(X_i - c)^2 \quad (4.3)$$

$$+ \beta_4(X_i - c) \times B_i + \beta_5(X_i - c) \times B_i + u_i. \quad (4.4)$$

Gelman und Imbens (2019) zeigen, dass Polynome höherer Ordnung zu verzerrten Schätzern und hoher Varianz führen können.² Die Autoren empfehlen stattdessen die Schätzung mit lokaler Regression.

Nicht-parametrische Schätzung und Bandweite

Aktuelle Studien nutzen nicht-parametrische Schätzer, die den Behandlungseffekt als Differenz der geschätzten Regressionsfunktionen am Schwellenwert c berechnen. Um auch nicht-lineare Regressionsfunktionen abzubilden zu können, wird häufig lokale Regression verwendet. Dieses Verfahren liefert eine "lokale" Schätzung der Regressionsfunktionen am Schwellenwert, bei der nur Beobachtungen nahe $X = c$ für die Schätzung berücksichtigt werden. Hinreichende Nähe wird hierbei durch eine sogenannte Bandweite h festgelegt, wobei

$$|(X_i - c)| \leq h \quad (4.5)$$

das Kriterium für eine Berücksichtigung von Beobachtung i bei der Schätzung ist.

Unter Verwendung einer Bandweite h wird der Regressionsansatz (4.2) als *lokale lineare Regression* mit Uniform-Kernelfunktion bezeichnet. Der Uniform-Kernel gibt allen Beobachtungen, innerhalb der Bandweite h dasselbe Gewicht. Ist h so groß, dass der gesamte Datensatz in die Schätzung einbezogen wird, entspricht der lokale lineare Regressions-Schätzer mit Uniform-Kernel dem (globalen) KQ-Schätzer in einem linearen Interaktionsmodell anhand aller Beobachtungen.

²Ursachen sind Überanpassung an die Daten sowie instabiles Verhalten der Schätzung nahe des Schwellenwertes.

Neben dem Uniform-Kernel ist der Triangular-Kernel eine in der Praxis häufig genutzte lineare Kernelfunktion. Der nachstehende Code plottet die Uniform- (grün) sowie die Triangular-Kernelfunktion (blau), siehe Abbildung 4.2.

```
library(ggplot2)
library(cowplot)

# Kernelfunktionen zeichnen
ggplot() +
  geom_function(
    fun = ~ ifelse(
      test = abs(.) <= 1,
      yes = 1/2,
      no = 0
    ),
    col = "green",
    n = 1000
  ) +
  geom_function(
    fun = ~ ifelse(
      test = abs(.) <= 1,
      yes = 1 - abs(.),
      no = 0
    ),
    col = "blue",
    n = 100
  ) +
  scale_x_continuous(
    name = "x",
    limits = c(-1.5, 1.5),
    breaks = c(-1, 0, 1)
  ) +
  scale_y_continuous(
    name = "K(x)",
    breaks = c(0, 1),
    limits = c(0, 1.25)
  ) +
  theme_cowplot()
```

In empirischen Studien wird als Basis-Spezifikation oft eine lokale lineare

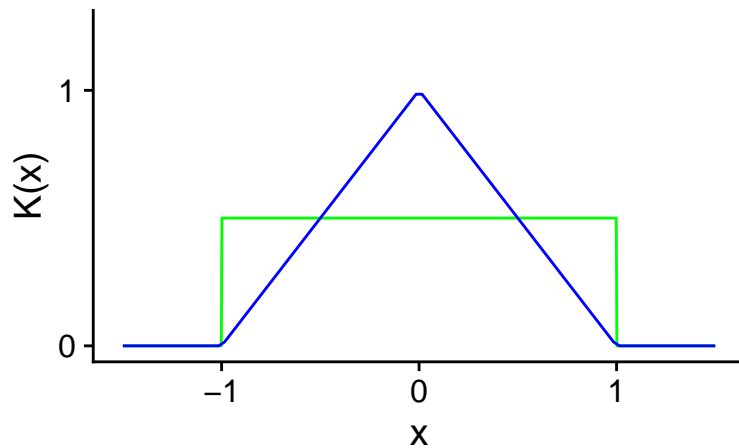


Abbildung 4.2: Kernelfunktionen auf $[-1, 1]$

Regression anhand von (4.2) mit einer linearen Kernelfunktionen und geringer bandweite h genutzt. Anschließend wird die Robustheit der Ergebnisse anhand flexiblerer Spezifikationen, die Nicht-Linearitäten in der Regressionsfunktion besser abbilden können, geprüft.

Die nachstehende Visualisierung zeigt die Schätzung des kausalen Effektes der Behandlung B_i anhand lokaler linearer Regression mit einem Uniform-Kernel für wiefolgt simulierte Daten:

$$Y_i = \beta_1 X_i + \beta_2 B + \beta_3 X_i^2 \times B_i + u_i,$$

$$u_i \sim N(0, 0.5), \quad X_i \sim U(0, 10), \quad B = \mathbb{I}(X_i \geq c = 5)$$

$$\beta_1 = .5, \quad \beta_2 = 1.5, \quad \beta_3 = -0.15$$

Diese Vorschrift ist schnell mit R umgesetzt:

```
set.seed(1234)
# Anz.Beobachtungen
n <- 750

# Parameter definieren
c <- 5
beta_1 <- .5
beta_2 <- 1.5
beta_3 <- -.15

# Regressionsfunktion definieren
```

```
f <- function(X) {
  beta_1 * (X - c) + beta_2 * B + beta_3 * B * (X - c)^2
}

# Daten erzeugen
X <- runif(n, 0, 11)
B <- ifelse(X - c >= 0, 1, 0)
Y <- f(X) + rnorm(n, sd = .5)

# Beobachtungen sammeln
dat <- data.frame(
  Y = Y, X = X - c, B = B
)
```

Diese interaktive Komponente des Buchs ist nur in der Online-Version verfügbar.

Der interessierende Effekt am Schwellenwert $c = 5$ beträgt $\beta_2 = 1.5$. Beachte, dass aufgrund des Terms $\beta_3 X_i^2 \times B_i$ ein quadratischer Zusammenhang von Y und X oberhalb von $X_i = c$ vorliegt. Es können folgende Eigenschaften der Schätzung in Abhängigkeit von der Bandweite h beobachtet werden:

- Für die voreingestellte Bandweite $h = 1.3$ liefert die lokale lineare Regression eine gute Approximation des Regressionszusammenhangs auf beiden Seiten des Schwellenwertes und die Schätzung des Behandlungseffekts liegt nahe beim wahren Wert $\beta_2 = 1.5$.
- Für kleinere Bandweiten verringert sich die Datenbasis der Schätzung. Die Varianz der Schätzung nimmt zu und die Approximation der Regressionsfunktion verschlechtert sich. Wir beobachten eine mit $h \rightarrow 0$ zunehmende Verzerrung bei der Schätzung des Behandlungseffekts.
- Größere Bandweiten h erhöhen die Datenbasis der Schätzung, führen aber zu einer Annäherung der lokalen Schätzung an die globale KQ-Schätzung. Linksseitig des Schwellenwertes erzielen wir damit eine Schätzung mit hoher Güte. Rechtsseitig von $X_i = c$ verschlechtert sich die lokale Anpassung

am Schwellenwert deutlich, weil die lineare Schätzung den tatsächlichen (nicht-linearen) Zusammenhang nicht adäquat abbilden kann. Die Schätzung des Behandlungseffekts ist hier deutlich verzerrt.

Die Wahl der Bandweite ist also eine wichtige Komponenten der RDD-Schätzung: Kleine Bandweiten erlauben eine Schätzung der Regressionsfunktion nahe des Schwellenwertes mit wenig Verzerrung. Allerdings kann diese Schätzung unpräzise sein, wenn nur wenige Beobachtungen (4.5) erfüllen. In der Praxis wird h daher mit einem analytischen Schätzer (vgl. G. Imbens und Kalyanaraman 2012) oder anhand von *Cross Validation* (bspw. G. W. Imbens und Lemieux 2008) bestimmt. Die später in diesem Kapitel betrachteten R-Pakete halten diese Methoden bereit.

4.2 Manipulation am Schwellenwert

Eine wichtige Annahmen für die Gültigkeit einer RDD-Schätzung ist, dass keine Manipulation der Gruppenzugehörigkeit am Schwellenwert vorliegt. Wenn sich Subjekte nahe des Schwellenwertes c — d.h. in Abhängigkeit der Laufvariable X — systematisch in den Confoundern Z unterscheiden, können wir den Backdoor-Pfad *Oberhalb* $C \rightarrow \text{Behandlung } B \rightarrow Y$ nicht isolieren. Wir erhalten dann eine verzerrte Schätzung des Behandlungseffekts.

In empirischen Studien mit Individuen kann Selbstselektion auftreten: Menschen mit $X < c$ aber nahe c (hier Kontrollgruppe) könnten aufgrund unbeobachtbarer Eigenschaften Z die Ausprägung ihrer Laufvariable zu $X > c$ (hier Behandlungsgruppe) manipulieren. Wenn Z die Outcome-Variable beeinflusst, bleibt der Backdoor-Pfad *Oberhalb* $C \rightarrow \text{Behandlung } B \rightarrow Y$ so bestehen.

Manipulation resultiert in Häufung von Beobachtungen am Schwellenwert. Die Verteilung der Laufvariable kann auf diese Unregelmäßigkeit hin untersucht werden. McCrary (2008) schlägt hierfür einen Verfahren vor, das die Kontinuität der Dichtefunktion von X am Schwellenwert testet.

Der Test von McCrary (2008) ist in `rdd::DCdensity()` implementiert. Wir zeigen die Anwendung des Tests anhand der oben simulierten Daten. Beachte, dass $X_i \sim U(0, 10)$, d.h. die Laufvariable ist bei $X_i = c$ kontinuierlich verteilt. Die Nullhypothese (keine Manipulation) gilt für die simulierten Daten

```
# McCrary-Test durchführen
p_mccrary <- rdd::DCdensity(
  runvar = X,
```



```

    cutpoint = c,
    plot = F
)

# p-Wert
p_mccrary

```

```
[1] 0.5013939
```

Der p-Wert 0.5 ist größer als jedes übliche Signifikanzniveau. Damit liegt starke Evidenz für die Nullhypothese (keine Diskontinuität) und gegen Manipulation am Schwellenwert vor.

Cattaneo, Jansson, und Ma (2020) (CMJ) schlagen eine Weiterentwicklung des McCrary-Tests vor, die höhere statistische Macht gegenüber Diskontinuitäten hat am Schwellenwert hat. Der CJM-Test ist im Paket `rddensity` implementiert.

```

library(rddensity)

# CJM Schätzer berechnen
CJM <- rddensity(X, c = 5)

```

Mit der Funktion `rddensity::rdplotdensity()` erzeugen wir Abbildung 4.3.

```

# Plot für Dichtefunktion erstellen
plot <- rdplotdensity(
  rdd = CJM,
  X = X,
  # für Punkte- und Linienplots:
  type = "both"
)

```

Abbildung 4.3 zeigt die geschätzten Dichtefunktionen. Erwartungsgemäß finden wir eine große Überlappung der zugehörigen Konfidenzbänder (schattierte Flächen) am Schwellenwert $c = 5$.

Mit `summary()` erhalten wir eine detaillierte Zusammenfassung des Tests.

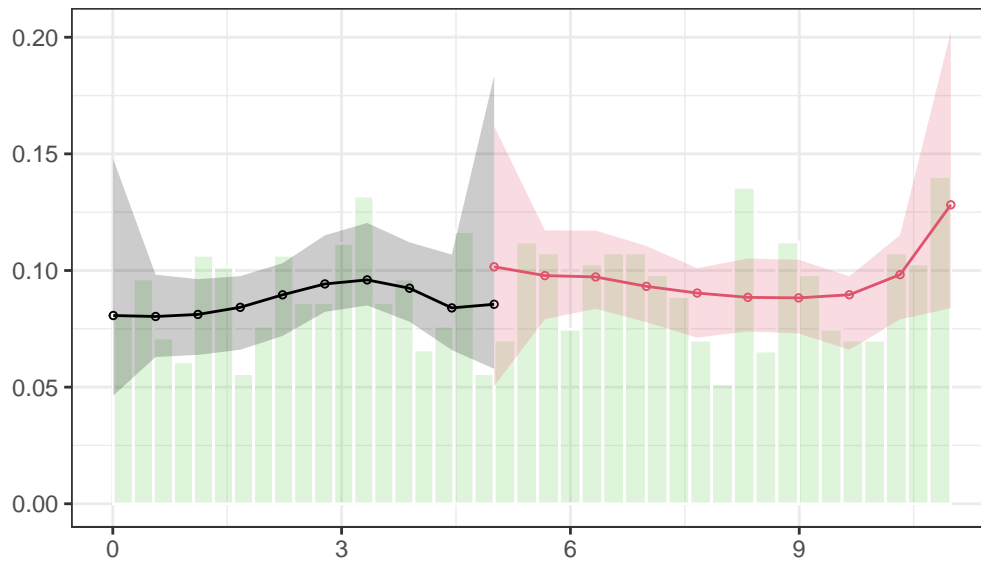


Abbildung 4.3: CJM-Test – geschätzte Dichtefunktionen der Laufvariable auf beiden Seiten des Schwellenwerts $c = 5$

```
# Statistische Zusammenfassung des CJM-Tests
summary(CJM)
```

Manipulation testing using local polynomial density estimation.

```
Number of obs =      750
Model =            unrestricted
Kernel =          triangular
BW method =       estimated
VCE method =      jackknife
```

$c = 5$	Left of c	Right of c
Number of obs	329	421
Eff. Number of obs	133	154
Order est. (p)	2	2
Order bias (q)	3	3
BW est. (h)	1.918	2.124

Method	T	$P > T $
Robust	-0.3338	0.7385

P-values of binomial tests ($H_0: p=0.5$).

Window Length		<c	>=c	$P > T $
0.346	+ 0.346	20	21	1.0000
0.521	+ 0.544	34	37	0.8126
0.696	+ 0.742	44	57	0.2323
0.870	+ 0.939	54	64	0.4075
1.045	+ 1.137	62	77	0.2349
1.220	+ 1.334	73	98	0.0661
1.394	+ 1.532	86	106	0.1701
1.569	+ 1.729	96	124	0.0685
1.743	+ 1.927	119	140	0.2139
1.918	+ 2.124	133	154	0.2377

Gemäß des p-Werts ($P > |T|$) von 0.74 spricht der CJM-Test noch deutlicher gegen eine Diskontinuität als der McCrary-Test.

4.2.1 Case Study: Amtsinhaber-Vorteil (Lee 2008)

Lee (2008) untersucht den Einfluss des Amtsinhaber-Vorteils auf die Wahl von Mitgliedern des US-Repräsentantenhaus. In den meisten Wahlkreisen entfallen große Anteile der Stimmen (oder gar ausschließlich) auf demokratische und republikanische Kandidat*innen, sodass sich die Studie auf diese Parteien beschränkt. Entfällt die Mehrheit der Stimmen auf eine*n Kandidat*in, gewinnt diese*r den Sitz für den Wahlkreis. Durch die Analyse der 6558 Wahlen im Zeitraum 1946-1998 mit einem SRDD kommt die Studie zu dem Ergebnis, dass Amtsinhabende im Durchschnitt einen Vorteil von etwa 8% bis 10% bei der Wahl haben. Dieses Ergebnis kann verschiedene Ursachen haben, bspw. dass die amtierende Partei höhere finanzielle Ressourcen besitzt und von einer besseren Organisation und durch Instrumentalisierung staatlicher Strukturen für die eigenen Zwecke profitiert.

Anhand der Datensätze `house` und `house_binned` illustrieren wir nachfolgend die Schätzung von SRDD-Modellen für den Wahlerfolg der demokratischen Partei, wenn diese Amtsinhaber ist. Wir lesen hierfür zunächst die Datensätze `house` und `house_binned` ein und verschaffen uns einen Überblick.

```
library(tidyverse)
library(modelsummary)
```

```
# Daten einlesen
house <- read_csv("datasets/house.csv")
# Gruppiertes Datensatz
house_binned <- read_csv("datasets/house_binned.csv")

# Überblick verschaffen
glimpse(house)
```

```
Rows: 6,558
Columns: 2
$ StimmenTm1 <dbl> 0.1049, 0.1393, -0.0736, 0.0868, 0.3994,
0.1681, 0.2516, 0.~
$ StimmenT <dbl> 0.5810, 0.4611, 0.5434, 0.5846, 0.5803,
0.6244, 0.4873, 0.5~
```

```
glimpse(house_binned)
```

```
Rows: 100
Columns: 2
$ StimmenT <dbl> 0.5995600, 0.5657000, 0.4272554, 0.5637456,
0.6868627, 0.60~
$ StimmenTm1 <dbl> 0.104764444, 0.135005263, -0.075690769,
0.084570886, 0.3951~
```

Der Datensatz `house` enthält die Stimmenanteile demokratischer Kandidat*innen bei der Wahl zum Zeitpunkt T (*StimmenT*) sowie die Differenz zwischen demokratischen und republikanischen Stimmenanteilen bei der vorherigen Wahl, d.h. zum Zeitpunkt $T-1$ (*StimmenTm1*). Der Schwellenwert für einen Wahlsieg liegt bei Stimmengleichheit, d.h. $StimmenTm1 = 0$.

`house_binned` ist eine aggregierte Version von `house` mit Mittelwerten von jeweils 50 gleichgroßen Intervallen oberhalb und unterhalb der Schwelle von $StimmenTm1 = 0$. Dieser Datensatz eignet sich, um einen ersten Eindruck des funktionalen Zusammenhangs auf beiden Seiten zu erhalten. Wir stellen zunächst diese klassierten Daten mit `ggplot2` graphisch dar.

```
# Klassierte Daten plotten
house_binned %>%
  ggplot(
    aes(x = StimmenTm1, y = StimmenT)
```

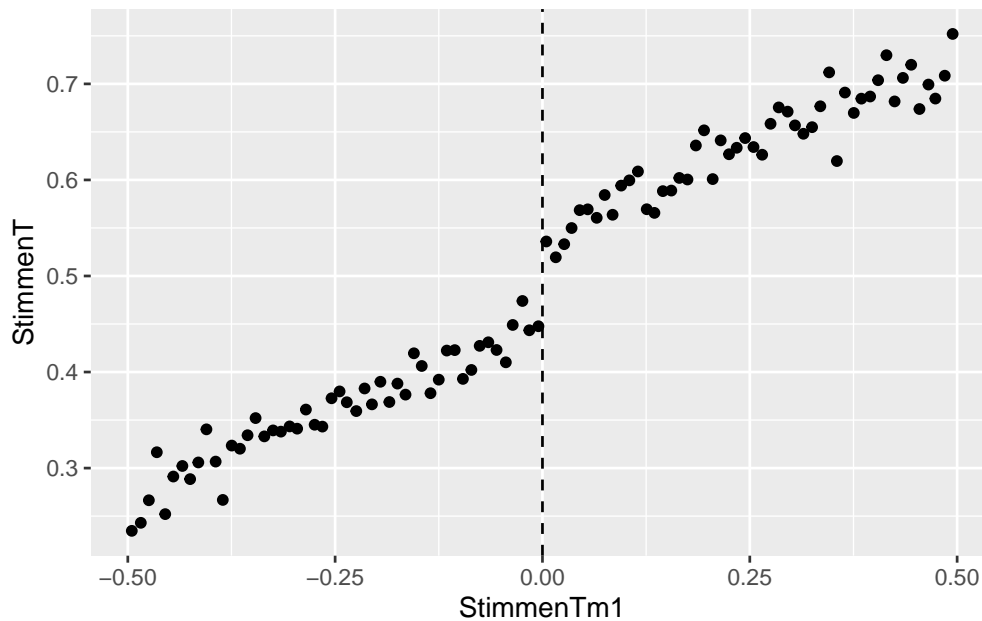


Abbildung 4.4: Klassierte Daten aus Lee (2008)

```
) +  
geom_point() +  
geom_vline(xintercept = 0, lty = 2)
```

Die Grafik zeigt eindeutig einen Sprung von *StimmenT* bei *StimmenTm1* = 0. Weiterhin erkennen wir, dass der Zusammenhang nahe 0 vermutlich jeweils gut durch eine lineare Funktion approximiert werden kann. Eine Modell-Spezifikation mit gleicher Steigung auf beiden Seiten des Schwellenwertes scheint hingegen weniger gut geeignet. Wir vergleichen diese Spezifikationen nachfolgend.

Zunächst fügen wir dem Datensatz eine Dummyvariable *B* hinzu. Diese dient als Indikator für den Wahlgewinn in der letzten Wahl und zeigt die Amtsinhaberschaft (Behandlung) an.

```
# Behandlungsindikator B hinzufügen  
house <- house %>%  
  mutate(B = StimmenTm1 > 0)  
  
glimpse(house)
```

Rows: 6,558

Columns: 3

```
$ StimmenTm1 <dbl> 0.1049, 0.1393, -0.0736, 0.0868, 0.3994,
0.1681, 0.2516, 0.~
$ StimmenT <dbl> 0.5810, 0.4611, 0.5434, 0.5846, 0.5803,
0.6244, 0.4873, 0.5~
$ B <lgl> TRUE, TRUE, FALSE, TRUE, TRUE, TRUE, TRUE,
TRUE, TRUE, TRUE~
```

Wir überprüfen die Laufvariable mit dem CJM-Test auf Manipulation am Schwellenwert $c = 0$.

```
# CJM-Test durchführen
CJM_Lee <- rddensity(X = house$StimmenTm1)

# Zusammenfassung anzeigen
summary(CJM_Lee)
```

Manipulation testing using local polynomial density estimation.

Number of obs =	6558	
Model =	unrestricted	
Kernel =	triangular	
BW method =	estimated	
VCE method =	jackknife	
 c = 0		
	Left of c	Right of c
Number of obs	2740	3818
Eff. Number of obs	1297	1360
Order est. (p)	2	2
Order bias (q)	3	3
BW est. (h)	0.236	0.243
 Method	T	P > T
Robust	1.4346	0.1514

P-values of binomial tests ($H_0: p=0.5$).

Window Length / 2	<c	>=c	P> T
0.004	21	24	0.7660

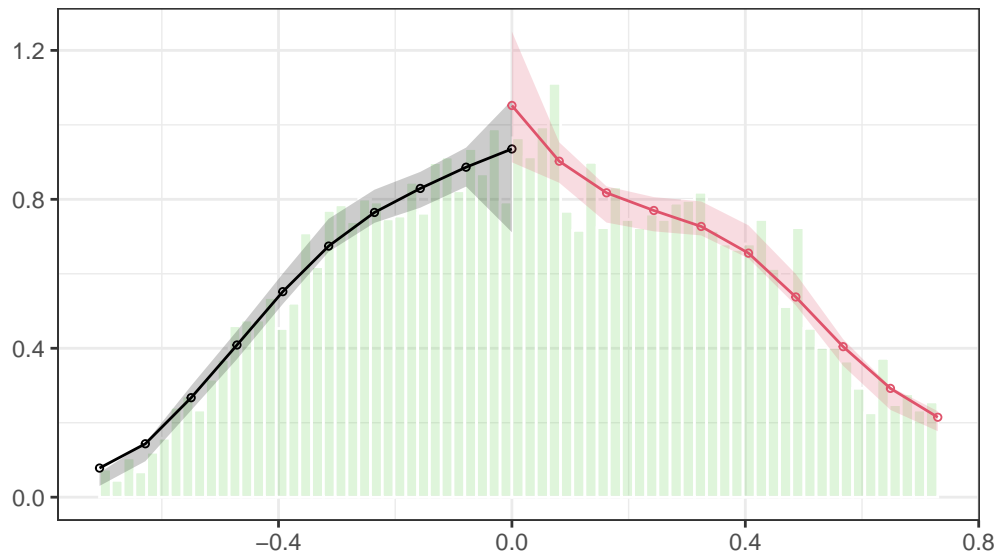


Abbildung 4.5: CJM-Test – geschätzte Dichtefunktionen der Laufvariable

0.007	38	46	0.4452
0.011	50	60	0.3909
0.014	73	77	0.8066
0.018	91	104	0.3902
0.022	124	132	0.6618
0.025	149	149	1.0000
0.029	163	174	0.5860
0.032	176	202	0.1984
0.036	197	223	0.2225

```
# CJM-Plot
plot <- rdplotdensity(
  rdd = CJM_Lee,
  X = house$StimmenTm1,
  type = "both",
)
```

Abbildung 4.5 und der p-Wert von 0.15 sind Evidenz gegen eine Manipulation am Schwellenwert.

Um den Behandlungseffekt anhand eines SRDDs zu ermitteln, schätzen wir das Interaktionsmodell

$$\text{StimmenT}_i = \beta_0 + \beta_1 B_i + \beta_2 (\text{StimmenTm1}_i - 50) + \beta_3 (\text{StimmenTm1}_i - 50) \times B_i + u_i$$

zunächst für eine Bandweite von $h = 0.5$. Aufgrund der Skalierung der Daten (Wahlergebnisse in %) bedeutet dies die Verwendung des *gesamten* Datensatzes für die Schätzung.

```
# Interaktionsmodell schätzen
house_llr1 <- lm(
  formula = StimmenT ~ B * StimmenTm1,
  data = house
)

# Zusammenfassung anzeigen
modelsummary(
  models = house_llr1,
  vcov = "HC1", # robuste Standardfehler
  stars = T,
  gof_map = "nobs",
  output = "gt"
) %>%
  tabopts
```

	(1)
(Intercept)	0.433*** (0.004)
BTRUE	0.118*** (0.006)
StimmenTm1	0.297*** (0.016)
BTRUE × StimmenTm1	0.046* (0.018)
Num.Obs.	6558

+ $p < 0.1$, * $p < 0.05$, ** $p < 0.01$, *** $p < 0.001$

Der geschätzte Koeffizient von B (BTRUE) beträgt etwa 0.12 und ist hochsignifikant. Übereinstimmend mit Abbildung 4.4 erhalten wir also eine positive Schätzung des Behandlungseffekts. Die Interpretation ist, dass die amtierenden Demokraten bei der Wahl von einem Amtsinhabervorteil profitieren. Dieser Effekt schlägt sich als Stimmenbonus von geschätzten 12% nieder. Diese Schätzung des Behandlungseffekts könnte jedoch verzerrt sein:

- Die (implizite) Wahl von $h = 0.5$ in unserer Schätzung macht die Isolation des relevanten Frontdoor-Paths ($c = 0 \rightarrow \text{Treatment} \rightarrow \text{StimmenT}$) wenig plausibel. h sollte mit einer datengetriebenen Methode gewählt werden.
- Weiterhin könnte die lineare funktionale Form der Regression inadäquat sein: Die lineare Approximation der wahren Regressionsfunktion nahe des Schwellenwerts 0 könnte unzureichend sein und in einer verzerrten Schätzung des Effekts resultieren. Zur Überprüfung der Robustheit der Ergebnisse sollte mit Schätzungen anhand nicht-linearer Spezifikationen verglichen werden.

Um diesen Gefahren für die Validität der Studie zu begegnen, schätzen wir nun weitere Spezifikationen. Im Folgenden verwenden wir eine Bandweitenschätzung gemäß G. Imbens und Kalyanaraman (2012).

```
# Bandweite mit Schätzer von IK (2012) berechnen
(
IK_BW <-
  rdd::IKbandwidth(
    X = house$StimmenTm1,
    Y = house$StimmenT
  )
)
```

```
[1] 0.2685123
```

Wir schätzen zunächst erneut das lineare Interaktionsmodell, diesmal jedoch mit der Bandweite IK_BW.

```
# Lineares Interaktionsmodelle mit IK-Bandweite
house_llin_IK <- lm(
  formula = StimmenT ~ B * StimmenTm1,
  data = house %>%
    filter(
      abs(StimmenTm1) <= IK_BW
    )
)
```

Für den Vergleich mit einer nicht-linearen Spezifikation schätzen wir auch ein quadratisches Interaktionsmodell.

```
# Quadratisches Interaktionsmodell mit IK-Bandweite
house_poly_IK <- update(
  object = house_llin_IK,
  formula = StimmenT ~ B * poly(StimmenTm1, degree = 2, raw = T)
)
```

Für eine Gegenüberstellung der Ergebnisse verwenden wir `modelsummary()`.

```
# Tabellarischer Modellvergleich
modelsummary(
  models = list(
    "Linear int." = house_llin_IK,
    "Quadratisch int." = house_poly_IK
  ),
  vcov = "HC1",
  stars = T,
  gof_map = "nobs",
  output = "gt"
) %>%
  tabopts
```

Tabelle 4.1: Vergleich von SRDD-Interaktionsmodellen für Lee (2008)

	Linear int.	Quadratisch int.
(Intercept)	0.450*** (0.005)	0.460*** (0.008)
BTRUE	0.085*** (0.008)	0.068*** (0.012)
StimmenTm1	0.360*** (0.036)	
BTRUE × StimmenTm1	0.055 (0.059)	
poly(StimmenTm1, degree = 2, raw = T)1		0.573*** (0.138)
poly(StimmenTm1, degree = 2, raw = T)2		0.798 (0.493)
BTRUE × poly(StimmenTm1, degree = 2, raw = T)1		0.036 (0.219)

BTRUE \times poly(StimmenTm1, degree = 2, raw = T) ²		-1.529+
		(0.834)
Num.Obs.	2956	2956

+ p < 0.1, * p < 0.05, ** p < 0.01, *** p < 0.001

Die Spalte (1) in Tabelle 4.1 zeigt die lokale Schätzung mit einem linearen Interaktionsmodell. Wir erhalten damit einen Behandlungseffekt von etwa 8.5%. Der Schätzwert fällt also etwas geringer aus als für die globale KQ-Schätzung des linearen Interaktionsmodells. Für das Modell (2) mit quadratischer Spezifikation liegt der Schätzwert mit 6.8% in der selben Größenordnung. Beide Schätzungen ergeben einen signifikant von 0 verschieden Effekt. Weiterhin fällt auf, dass in beiden Modellen keine Evidenz für unterschiedliche Formen der Regressionsfunktionen auf beiden Seiten des Schwellenwerts vorliegen: sämtliche Koeffizientenschätzwerte der Interaktionsterme haben hohe Standardfehler und sind nicht signifikant. Im quadratischen Modell hat auch der Term *StimmenTm1*² keinen signifikanten Effekt. Diese Ergebnisse deuten darauf hin, dass eine lineare Spezifikation ausreichend ist.

SRDD-Schätzung mit LOESS

Wir illustrieren nachfolgend die Schätzung des Behandlungseffekts mit einer flexiblen und in der Praxis häufig verwendeten Methode für lokale Regression. Die nachfolgende interaktive Grafik zeigt die klassierten Daten aus Lee (2008) auf dem Intervall $[-0.5, 0.5]$ gemeinsam mit einer nicht-parametrischen Schätzung des Zusammenhangs von *StimmenT* und *StimmenTm1* mittels LOESS.³ Diese Implementierung von lokaler Regression nutzt einen *tricube kernel*. Über den Input kann eine Bandweite $l \in (0, 1]$ für den LOESS-Schätzer auf beiden Seiten des Schwellenwerts 0 gewählt werden. Die Bandweite ist hier der *Anteil der Beobachtungen an der gesamten Anzahl an Beobachtungen*, die in die Schätzung einbezogen werden sollen.

Für die Schätzung am Schwellenwert berücksichtigte Daten sind in orange kenntlich gemacht. Die rote Linie zeigt die geschätzte Regressionsfunktion über gleichmäßig verteilte Werte von *StimmenTm1* auf $[-0.5, 0.5]$. Die Grafik verdeutlicht, dass die LOESS-Methode flexibel genug ist, um lineare und nicht-lineare Zusammenhänge abbilden zu können. Wie zuvor ist eine adäquate Wahl der Bandweite wichtig:

- Der mit LOESS geschätzte Zusammenhang auf beiden Seiten des Schwellenwerts ist etwa linear für den voreingestellten Parameter ($l = 0.28$).

³LOESS ist eine Variante von lokaler Polynom-Regression.

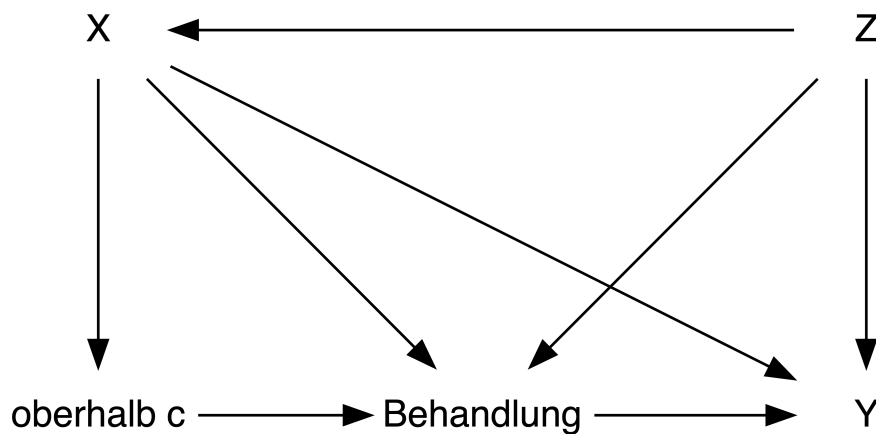


Abbildung 4.6: Kausales Diagramm für FRDD

- Für größere Werte von l nähert sich die Schätzung weiter einem linearen Verlauf an. Die Schätzung des Effekts bleibt vergleichbar mit den Ergebnissen des linearen Interaktionsmodell (s. oben).
- Für kleinere l erhalten wir eine stärkere Anpassung der Schätzung an die Daten. Zu kleine Werte führen zu einer Überanpassung (*overfitting*). Insbesondere tendiert die geschätzte Funktion zu extremer Steigung nahe des Schwellenwerts \rightarrow stark verzerrte Schätzung des Effekts!

Diese interaktive Komponente des Buchs ist nur in der Online-Version verfügbar.

4.3 Fuzzy Regression Discontinuity Design

Ein FRDD liegt vor, wenn die Zuweisung der Behandlung B durch die Laufvariable X (und möglicherweise weitere Variablen Z) beeinflusst wird. Im Vergleich zum SRDD ist die Behandlung dann also *nicht* ausschließlich durch Überschreiten des Schwellenwerts $X = c$ bestimmt.

Abbildung 4.6 zeigt den grundsätzlichen Zusammenhang. Hier genügt es weiterhin für X (und ggf. Z) zu kontrollieren, um den Pfad *oberhalb* $C \rightarrow \text{Behandlung}$ $B \rightarrow Y$ zu isolieren. Der so für *Behandlung* B ermittelte Effekt auf Y entspricht

jedoch *nicht* dem “vollständigen” Behandlungseffekt, da bei c die Zuweisung der Behandlung nicht von 0 auf 100% springt. Die Schätzung des FRDD berücksichtigt dies und skaliert den geschätzten Effekt entsprechend.

Wir betrachten zunächst den Zusammenhang

$$Y_i = \beta_0 + \beta_1 B_i + \beta_2 (X_i - c) + u_i. \quad (4.6)$$

In einem FRDD springt die Behandlungswahrscheinlichkeit am Schwellenwert c um $\Delta p < 1$. Wir können B also nicht als deterministische Funktion von X , welche die Zuweisung zu Behandlungs- bzw. Kontrollgruppe am Schwellenwert c anzeigt (wie im SRDD), definieren. Stattdessen betrachten wir

$$P(B_i = 1 | X_i) = \begin{cases} g_{X_i < c}(X_i), & X_i < c \\ g_{X_i \geq c}(X_i) & X_i \geq c \end{cases}. \quad (4.7)$$

Die Funktionen $g_{X_i < c}$ und $g_{X_i \geq c}$ können verschieden sein. Es muss jedoch

$$g_{X_i < c}(X_i = c) \neq g_{X_i \geq c}(X_i = c)$$

gelten. Die Behandlungsvariable B_i ist im FRDD also eine (binäre) Zufallsvariable, deren bedingte Wahrscheinlichkeitsfunktion $P(B_i = 1 | X_i)$ am Schwellenwert c eine Diskontinuität aufweist. Abbildung 4.7 zeigt heispielhafte Verläufe nicht-linearer bedingter Wahrscheinlichkeitsfunktion für die Behandlung mit einer Diskontinuität bei $X_i = c$.

```
library(ggplot2)
library(cowplot)

# Bedingte Behandlungswahrscheinlichkeit im FRDD illustrieren
ggplot() +
  geom_function(
    fun = ~ ifelse(
      . < 0,
      -.1 * .^2 + .25,
      -.1 * (.-1.5)^2 + 1
    ),
    n = 1000
  ) +
  geom_function(
    fun = ~ ifelse(
```

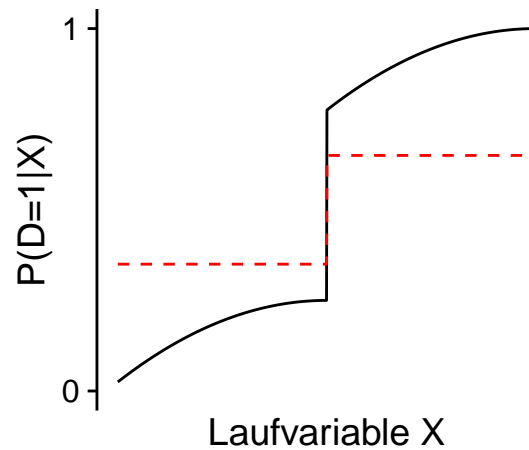


Abbildung 4.7: Bedingte Behandlungswahrscheinlichkeiten im FRDD

```

    . < 0,
    .35,
    .65
  ),
  n = 1000,
  lty = 2,
  col = "red"
) +
scale_x_continuous(
  name = "Laufvariable X",
  limits = c(-1.5, 1.5),
  labels = NULL,
  breaks = NULL
) +
scale_y_continuous(
  name = "P(D=1|X)",
  breaks = c(0, 1),
  limits = c(0, 1)
) +
theme_cowplot()

```

Definition (4.7) bedeutet, dass eine KQ-Schätzung von β_1 anhand (4.6) eine *verzerrte* Schätzung des Behandlungseffekts ist: Der in $\hat{\beta}_1$ erfasste Effekt auf Y ist auf einen Sprung der Behandlungswahrscheinlichkeit bei $X_i = c$ um *weniger* als 100% zurückzuführen. Der wahre Behandlungseffekt wird also *unterschätzt*. Daher muss $\hat{\beta}_1$ skaliert werden, sodass die Schätzung als Effekt einer Änderung

der Behandlungswahrscheinlichkeit um 100% interpretiert werden kann — der erwartete Effekt, wenn ausschließlich Subjekte mit $X_i \geq c$ behandelt würden. Diese skalierte Schätzung erhalten wir mit IV-Regression (vgl. Kapitel XYZ). Hierfür nutzen wir für B_i die Instrumentvariable

$$D_i = \begin{cases} 0, & X_i < c \\ 1, & X_i \geq c. \end{cases}$$

Angenommen $g_{X_i \geq c}(X_i) = \alpha_0$ und $g_{X_i < c}(X_i) = \alpha_0 + \alpha_1$ mit $\alpha_0 + \alpha_1 < 1$ (vgl. rote Funktion in Abbildung 4.6). Der FRDD-Schätzer des Behandlungseffekts ist dann $\hat{\gamma}_{\text{FRDD}}$ im 2SLS-Verfahren mit den Regressionen

$$\begin{aligned} \text{(I)} \quad B_i &= \alpha_0 + \alpha_1 D_i + \alpha_2 (X_i - c) + e_i, \\ \text{(II)} \quad Y_i &= \gamma_0 + \gamma_1 \hat{B}_i + \gamma_2 (X_i - c) + \epsilon_i, \end{aligned} \tag{4.8}$$

wobei \hat{B}_i die angepassten Werte aus Stufe (I) und e_i sowie ϵ_i Fehlerterme sind.

Analog zum SRDD müssen in empirischen Anwendungen geeignete Spezifikationen für die Regressionsfunktionen (4.6) und (4.7) gewählt und der 2SLS-Schätzer (4.8) entsprechend angepasst werden. Ein einfaches Interaktionsmodell wäre

$$\begin{aligned} \text{(I)} \quad B_i &= \alpha_0 + \alpha_1 D_i + \alpha_2 (X_i - c) \\ &\quad + \alpha_3 (X_i - c) \times D_i + e_i, \\ \text{(II)} \quad Y_i &= \gamma_0 + \gamma_1 \hat{B}_i \\ &\quad + \gamma_2 (X_i - c) + \gamma_3 (X_i - c) \times \hat{B}_i + \epsilon_i \end{aligned} \tag{4.9}$$

d.h. wir instrumentieren B_i mit D_i und dem Interaktionsterm $(X_i - c) \times D_i$.

Wie im SRDD werden die IV-Ansätze für das FRDD (4.8) und (4.9) in empirischen Studien unter Berücksichtigung einer Bandweite (i.d.R. dieselbe Bandweite für beide Stufen) angewendet.

4.4 Case Study: Protestantische Arbeitsethik

Die Studie *Beyond Work Ethic: Religion, Individual, and Political Preferences* (Basten und Betz 2013) untersucht den Zusammenhang zwischen Religion, individuellen Merkmalen und politischen Präferenzen. Das Hauptaugenmerk

ist die Rolle von Religiosität als Einflussfaktor auf politische Einstellungen. Die Hypothese der Autoren ist, dass Religiosität eines Individuums über den traditionellen Rahmen von Moralvorstellungen und sozialen Normen hinaus auch die politischen Präferenzen beeinflusst. Eine entsprechende Theorie wurde zu Beginn des 20. Jahrhunderts entwickelt und prominent von Max Weber (vgl. Weber 2004) vertreten. Weber argumentiert, dass die protestantische Arbeitsethik einen entscheidenden Einfluss auf die Entwicklung des Kapitalismus hatte. Laut Weber führte der protestantische Glaube an harte Arbeit, ein sparsames Leben und ethisches Verhalten zur einer in den damaligen Gesellschaften weit verbreiteten Geisteshaltung, die wirtschaftliches Wachstum förderte und den Aufstieg des Kapitalismus begünstigte.

Basten und Betz (2013) nutzen Wahlergebnisse sowie geo- und soziodemographische Datensätze für schweizer Gemeinden, um den Zusammenhang zwischen Religiosität und politischen Präferenzen wie links-rechts-Ausrichtung, Einstellungen zur Umverteilung und Einwanderung zu untersuchen. Hierfür verwenden die Autoren ein FRDD, dass eine historisch bedingte Diskontinuität der geographischen Verteilung von evangelischer bzw. katholischer Religionszugehörigkeit zwischen den Kantonen Freiburg (überwiegend dunkelrote Region, frz. *Fribourg*) und Waadt (kleinere hellrote Region, frz. *Vaud*) ausnutzt. Die historische Verteilung der Konfessionen in der betrachteten Region im 16. Jahrhundert durch Abspaltung des Kantons Freiburg ist in Abbildung 4.8 dargestellt.

Aufgrund von Bevölkerungsbewegungen ist die Verteilung der Konfessionen zwar nicht mehr eindeutig durch die Kantonsgrenze bestimmt, jedoch sind die Gemeinden der betrachteten Kantone auch heute noch mehrheitlich protestantisch bzw. katholisch. Es ist plausibel, dass eine Prägung gemäß Webers Theorie vorliegt, sich die Gemeinden nahe der Grenz aber hinsichtlich anderer Charakteristika (insb. der Bevölkerungsstruktur) nicht systematisch unterscheiden. Somit liegt ein quai

Die Ergebnisse der Studie zeigen einen signifikanten Einfluss von Protestantismus auf politische Präferenzen, die über traditionelle Moralvorstellungen hinausgehen: Die Autoren finden Hinweise, dass Einwohner evangelisch geprägter Gemeinden eher konservative soziale und politische Ansichten vertreten. Eine mögliche Erklärung für diesen Effekt ist, dass religiöse Institutionen auch eine soziale und politische Agenda verfolgen, die von den Gläubigen internalisiert wird.

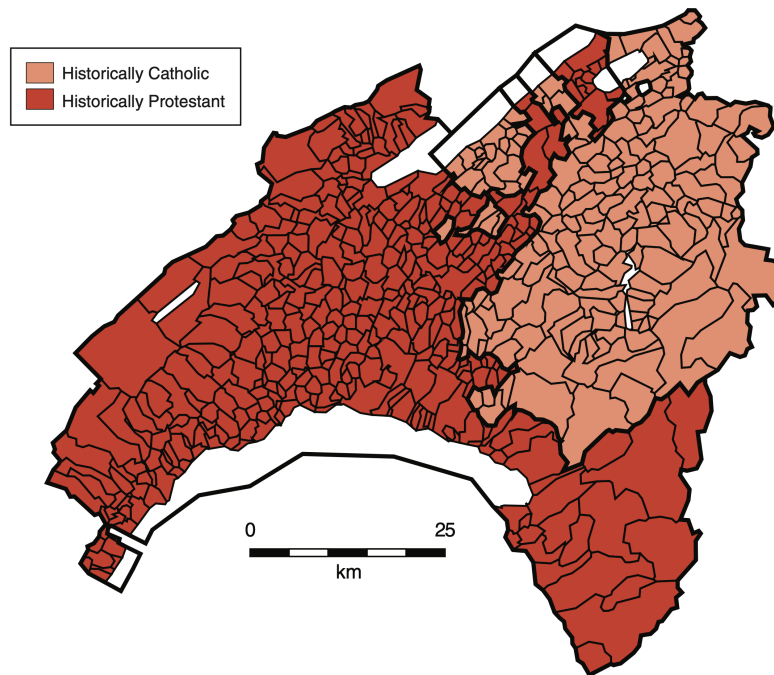


Abbildung 4.8:
Historische
Verteilung
von Religions-
zugehörigkeit
in Schweizer
Gemeinden
im 16. Jahr-
hundert.
Quelle: Bas-
ten und Betz
(2013).

4.4.1 Aufbereitung der Daten

In diesem Kapitel zeigen wir, wie die Kernergebnisse der Studie mit R reproduziert werden können. Hierfür werden folgende Pakete benötigt.

```
library(tidyverse)
library(haven)
library(vtable)
library(rdrobust)
```

Das Papier sowie der Datensatz **BastenBetz.dta** sind auf der [Übersichtsseite der AEA](#) verfügbar und liegt im STATA-Format **.dta** vor.⁴

```
# Datensatz einlesen
BastenBetz <- read_dta('BastenBetz.dta')
```

Der Datensatz **BastenBetz** enthält Beobachtungen zu 509 schweizer Gemeinden. Eine Vielzahl an Variablen ist lediglich für Robustheits-Checks relevant. Für die Reproduktion der Kernergebnisse erstellen wir zunächst einen reduzierten Datensatz und transformieren einige Variablen.

⁴Siehe alternativ das [working paper](#), falls kein Abbonement für AEA-Journals vorliegt.

```
# Reduzierten Datensatz erstellen
BastenBetz <- BastenBetz %>%
  transmute(
    gini = Ecoplan_gini,
    prot = prot1980s,
    bord = borderdis,
    vaud,
    pfl,
    pfr,
    pfi
  )
```

Die Definitionen der Variablen sind in Tabelle 4.2 gegeben. Die Präferenzen `pfl`, `pfr` und `pfi` basieren auf Wahlergebnissen auf Gemeindeebene zu Volksentscheiden.

Tabelle 4.2: `BastenBetz` – Variablen und Definitionen

Variable	Definition
<code>prot</code>	Anteil Protestanten im Jahr 1980 (%)
<code>gini</code>	Gini-Koeffizient
<code>bord</code>	Laufdistanz zur Kantonsgrenze (Km)
<code>vaud</code>	Dummyvariable: Gemeinde im Kanton Waadt
<code>pfl</code>	Präferenz für Freizeit (%)
<code>pfr</code>	Präferenz für Umverteilung (%)
<code>pfi</code>	Präferenz für wirtschaftliche Intervention des Staats (%)

Für die Berechnung der optimalen Bandweite des FRDD verwenden wir einen MSE-optimalen Schätzer, der in der Funktion `rdrobust::rdbwselect()` implementiert ist.⁵

```
# Bandweite schätzen (Bsp. für Freizeitpräferenz)
bw_selection <- rdbwselect(
  y = BastenBetz$pfl,
  x = BastenBetz$bord,
  fuzzy = BastenBetz$prot,
  bwselect = "mserd",
```

⁵Basten und Betz (2013) setzen $BW = 5.01$, den Durchschnitt von IK-Schätzungen über Modelle sämtlicher betrachteter Outcome-Variablen. Diese Bandweite liegt nahe des Ergebnisses von `rdbwselect`. Wir verwenden nachfolgend die Schätzung `0B`.

```

    kernel = "uniform"
  )

# Bandweite auslesen und zuweisen
(OB <- bw_selection$bws[1])

```

```
[1] 5.078001
```

4.4.2 Deskriptive Statistiken

Zur Reproduktion von Tabelle 1 aus Basten und Betz (2013) erzeugen wir eine nach Kantonen gruppierte Zusammenfassung der Daten und berechnen deskriptive Statistiken. Wie im Paper berücksichtigen wir hierbei nur Gemeinden innerhalb der geschätzten optimalen Bandweite OB.

```

# Datensatz für Reproduktion von Table 1 formatieren
T1 <- BastenBetz %>%
  filter(abs(bord) < OB) %>%
  mutate(
    vaud = ifelse(
      test = vaud == 1,
      yes = "Waadt",
      no = "Freiburg"
    ),
    prot = prot * 100
  ) %>%
  group_by(vaud) %>%
  summarise(
    across(
      everything(),
      list(
        Mean = mean,
        SD = sd,
        N = length
      )
    )
  ) %>%
  pivot_longer(
    cols = -vaud,

```

```
names_to = c("variable", "statistic"),
names_sep = "_"
)
```

Für die tabellarische Darstellung transformieren wir in ein weites Format, sodass die Tabelle die deskriptive Statistiken spaltenweise für die Kantone zeigt.

```
# Daten in weites Format überführen
T1_wider <- T1 %>%
  pivot_wider(
    names_from = c("vaud", "statistic")
  )
```

Die Tabelle erzeugen wir mit `gt::gt()`.

```
# Tabelle mit gt() erzeugen
T1_wider %>%
  gt(rowname_col = "Variable") %>%
  tab_spanner_delim(
    delim = "_",
  ) %>%
  tabopts
```

Tabelle 4.3: Datensatz **BastenBetz** – Zusammenfassende Statistiken

variable	Freiburg			Waadt		
	Mean	SD	N	Mean	SD	N
gini	0.302	0.029	49	0.367	0.052	84
prot	9.428	5.695	49	83.245	11.411	84
bord	−2.327	1.274	49	2.493	1.201	84
pfl	48.239	4.774	49	39.508	5.723	84
pfr	43.049	2.634	49	39.19	5.025	84
pfi	52.642	2.94	49	47.086	3.368	84

Die Statistiken in Tabelle 4.3 scheinen konsistent mit der (historischen) Verteilung der Religionszugehörigkeit und politischen Einstellung gemäß der Hypothese: Im überwiegend katholischen Freiburg finden wir eine größere Einkom-

mensungleichheit und höhere aus Wahlergebnissen abgeleitete Präferenzen für Freizeit, Umverteilung sowie staatliche Interventionen.

4.4.3 Modellspezifikation und First-Stage-Ergebnisse

Die Kantone Waadt und Freiburg haben bis heute mehrheitlich protestantische bzw. katholische Gemeinden. Die Verteilung von Protestantismus ist also, u.a. aufgrund von Bevölkerungsbewegungen, nicht mehr deterministisch. An der Kantonsgrenze besteht jedoch eine deutliche Diskontinuität im Anteil protestantischer Einwohner, die auf die historische Verteilung der Religionszugehörigkeit zurückzuführen ist. Damit kann ein FRDD implementiert werden, bei dem die Distanz zur Grenze (*bord*) die zentrierte Laufvariable ist und die Zugehörigkeit zum Kanton Waadt (*vaud*) ein Instrument für die Behandlungsvariable (*prot*) ist.

Wir nutzen die Funktion `rdrobust::rdplot` um diesen Zusammenhang für verschiedene Bandweiten anhand des linearen Interaktionsmodells

$$\begin{aligned} prot_i = & \alpha_0 + \alpha_1 vaud_i + \alpha_2 bord_i \\ & + \alpha_3 bord_i \times vaud_i + u_i \end{aligned} \quad (4.10)$$

grafisch darzustellen. Dies ist die First-Stage-Regression für die 2SLS-Schätzung der Behandlungseffekte.

```
# Reproduktion von Abbildung 3 in Basten und Betz (2013)
plots_BB <- list(
  # gesch. optimale Bandweite
  p_OB = rdplot(
    y = BastenBetz$prot,
    x = BastenBetz$bord,
    h = c(OB, OB),
    x.label = "Distanz zur Grenze (bord)",
    y.label = "Anteil Protestanten (prot)",
    title = "Gesch. Bandweite",
    p = 1,
    nbins = c(6, 14),
    masspoints = "off"
  ),

  # Bandweite 10
  p_BW10 = rdplot(
```

```

    y = BastenBetz$prot,
    x = BastenBetz$bord,
    h = c(10, 10),
    x.label = "Distanz zur Grenze (bord)",
    y.label = "Anteil Protestanten (prot)",
    title = "Bandweite = 10",
    p = 1,
    nbins = c(6, 14),
    masspoints = "off"
  ),

  # Bandweite 20
  p_BW20 = rdplot(
    y = BastenBetz$prot,
    x = BastenBetz$bord,
    h = c(20, 20),
    x.label = "Distanz zur Grenze (bord)",
    y.label = "Anteil Protestanten (prot)",
    title = "Bandweite = 20",
    p = 1,
    nbins = c(6, 14),
    masspoints = "off"
  ),

  # Gesamter Datensatz
  p_G = rdplot(
    y = BastenBetz$prot,
    x = BastenBetz$bord,
    x.label = "Distanz zur Grenze (bord)",
    y.label = "Anteil Protestanten",
    title = "Ges. Datensatz",
    p = 1,
    nbins = c(6, 14),
    masspoints = "off"
  )
)

```

Wir sammeln die Ergebnisse in einem Plot-Gitter mit `cowplot::plot_grid()`.

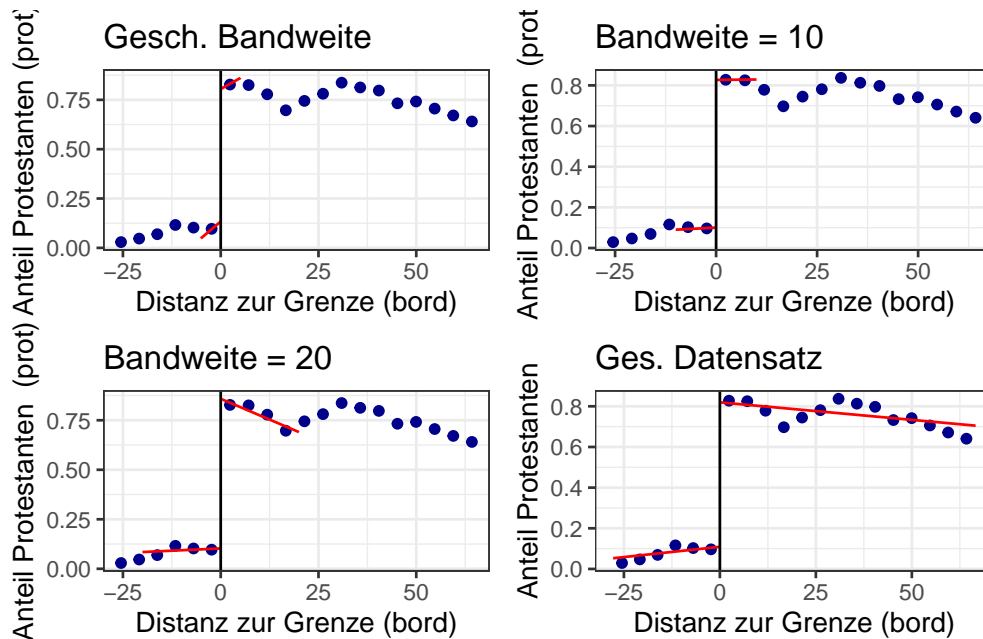


Abbildung 4.9:
First-Stage-
Regressionen

```
# Reproduktion von Abbildung 3 in Basten und Betz (2013)
plot_grid(
  plotlist = map(plots_BB, ~ .$rdplot), ncol = 2
)
```

Die Grafiken in Abbildung 4.9 zeigen deutliche Hinweise auf die Diskontinuität in `prot` nahe der Kantonsgrenze. Die Größe des geschätzten Sprungs scheint nur wenig sensitiv gegenüber der gewählten Bandweite zu sein. Die Signifikanz des Effekts können wir anhand der jeweiligen KQ-Regressionen beurteilen.⁶

```
# Reproduktion der First-Stage-Regressionen
# s. Tabelle 2 in Basten und Betz (2013)

# (1) BW = 0B
FS1 <- lm(
  formula = prot ~ vaud + bord + vaud * bord,
  data = BastenBetz %>%
    filter(
      abs(bord) <= 0B
    )
)
```

⁶Wir nutzen `update()` um die Regression mit weniger Code für verschiedene Bandweiten zu schätzen.

```

# (2) BW = 10
FS2 <- update(
  FS1,
  data = BastenBetz %>%
    filter(
      abs(bord) <= 10
    )
)

# (3) BW = 20
FS3 <- update(
  FS1,
  data = BastenBetz %>%
    filter(
      abs(bord) <= 20
    )
)

# (4) Ges. Datensatz
FS4 <- update(
  object = FS1,
  data = BastenBetz
)

```

```

# Tabellarische Darstellung
modelsummary(
  list(
    "BW = 0B" = FS1,
    "BW = 10" = FS2,
    "BW=20" = FS3,
    "Ges. Datensatz" = FS4
  ),
  vcov = "HC1",
  stars = T,
  gof_map = "nobs",
  output = "gt"
) %>%
  tabopts()

```


Tabelle 4.4: First-Stage-Regressionen

	BW = OB	BW = 10	BW=20	Ges. Datensatz
(Intercept)	0.134*** (0.017)	0.100*** (0.013)	0.103*** (0.010)	0.109*** (0.009)
vaud	0.671*** (0.034)	0.726*** (0.022)	0.756*** (0.018)	0.710*** (0.014)
bord	0.017** (0.006)	0.001 (0.003)	0.001 (0.001)	0.002* (0.001)
vaud × bord	-0.006 (0.012)	-0.001 (0.005)	-0.009*** (0.003)	-0.004*** (0.001)
Num.Obs.	133	207	312	509

+ p < 0.1, * p < 0.05, ** p < 0.01, *** p < 0.001

Für die geschätzte Bandweite schätzen wir einen hochsignifikanten Sprung in *prot* von etwa 67% an der Kantonsgrenze. Auch für größere Bandweiten von 10km und 20km sowie für den gesamten Datensatz finden wir vergleichbare signifikante Effekte, was eine bei zunehmender Distanz zur Grenze persistente Diskrepanz der Religionszugehörigkeit bestätigt.

4.4.4 Second-Stage-Ergebnisse

Wir schätzen nun den LATE von Protestantismus für die Outcome-Variablen *gini*, *pfl*, *pfi* und *pfr*, vgl. Tabelle 4.2. Die Spezifikation für die Second-Stage-Regression der FRDD-Schätzung ist

$$Y_i = \gamma_0 + \gamma_1 \widehat{prot}_i + \gamma_2 bord_i + \gamma_3 bord_i \times vaud_i + e_i, \quad (4.11)$$

wobei \widehat{prot}_i angepasste Werte aus der KQ-Schätzung von (4.10) mit Bandweite OB sind. Dazu erzeugen wir zunächst eine angepasste Version des Objekts *BastenBetz*, welche nur Gemeinden innerhalb der Bandweite enthält.

```
# Gemeinden innerhalb der Bandweite filtern
BastenBetz_OB <- BastenBetz %>%
  filter(
    abs(bord) <= OB
  )
```

Zur Illustration schätzen wir nun die Second-Stage-Regression für $Y = pfl$.

```
# Second-Stage-Regression für `pfl`
BastenBetz_OB %>%
  mutate(
    prot_fitted = fitted(FS1)
  ) %>%

lm(
  pfl ~ prot_fitted + bord + vaud:bord,
  data = .
) %>%
  summary()
```

Call:

```
lm(formula = pfl ~ prot_fitted + bord + vaud:bord, data = .)
```

Residuals:

Min	1Q	Median	3Q	Max
-12.8870	-3.8621	-0.0423	3.4993	12.1636

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	50.5275	1.9721	25.621	< 2e-16 ***
prot_fitted	-13.4600	3.1749	-4.240	4.24e-05 ***
bord	0.4380	0.6528	0.671	0.503
bord:vaud	-0.3636	0.7939	-0.458	0.648

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 5.433 on 129 degrees of freedom

Multiple R-squared: 0.383, Adjusted R-squared: 0.3686

F-statistic: 26.69 on 3 and 129 DF, p-value: 1.704e-13

Der Koeffizient `prot_fitted` ist der gesuchte Behandlungseffekt. Beachte, dass die von `summary()` berechneten Standardfehler ungültig sind, weil diese die zusätzliche Unsicherheit durch die Berechnung von \widehat{prot} über die First-Stage-Regression nicht berücksichtigen. Nachfolgend nutzen wir `AER::ivreg()`, um

komfortabel gültige (heteroskedastie-robuste) Inferenz betreiben zu können.⁷

```
# Schätzung mit 2SLS
# s. Tabelle 4 in Basten und Betz (2013)
#
# Wir instrumentieren Treatment (`prot1980s`) mit dem
#   ↳ Schwellenindikator (`vaud`)
# ivreg: exogene Variablen instrumentieren sich selbst, daher
# ' | vaud * borderdis '
library(AER)
# (1) Präferenz für Freizeit
SS_pfl <- ivreg(
  formula = pfl ~ prot + bord:vaud + bord | vaud * bord,
  data = BastenBetz_0B
)

# (2) Präferenz für Umverteilung
SS_pfr <- update(
  object = SS_pfl,
  formula = pfr ~ prot + bord:vaud + bord | vaud * bord,
)

# (3) Präferenz für Intervention
SS_pfi <- update(
  object = SS_pfl,
  formula = pfi ~ prot + bord:vaud + bord | vaud * bord,
)

# (4) Einkommensungleichheit
SS_gini <- update(
  object = SS_pfl,
  formula = pfi ~ prot + bord:vaud + bord | vaud * bord,
)

# Tabellarische Darstellung
modelsummary(
```

⁷Die Autoren geben an, robuste SEs zu nutzen. Das scheint nicht der Fall zu sein, denn `vcov = "HC0"` liefert die Ergebnisse im Paper. Die von Stata berechneten HC1-SEs weichen ab. Dies ändert allerdings nichts an der Signifikanz der Koeffizienten. Wir nutzen `vcov = "HC1"`.

```

list(
  "(1) Freizeit"= SS_pfl,
  "(2) Umverteilung" = SS_pfr,
  "(3) Intervention" = SS_pfi,
  "(4) Ungleichheit" = SS_gini
),
vcov = "HC1",
stars = T,
gof_map = "nobs",
output = "gt"
) %>%
tabopts()

```

Tabelle 4.5: Ergebnisse der Second-Stage-Regressionen

	(1) Freizeit	(2) Umverteilung	(3) Intervention	(4) Ungleichheit
(Intercept)	50.528*** (1.918)	44.560*** (0.950)	52.871*** (1.063)	52.871*** (1.063)
prot	-13.460*** (3.161)	-5.061* (2.161)	-6.487*** (1.738)	-6.487*** (1.738)
bord	0.438 (0.639)	0.444 (0.357)	-0.165 (0.332)	-0.165 (0.332)
bord × vaud	-0.364 (0.811)	-0.909 (0.561)	0.011 (0.432)	0.011 (0.432)
Num.Obs.	133	133	133	133

+ p < 0.1, * p < 0.05, ** p < 0.01, *** p < 0.001

Die Koeffizienten von **prot** in Tabelle 4.5 sind die mit 2SLS ermittelten erwarteten Behandlungseffekte einer 100%-Reformation (d.h. von 100% katholisch zu 100% protestantisch) für eine durchschnittliche Gemeinde nahe der Kantonsgrnze. Es handelt sich jeweils um einen lokalen durchschnittlichen Behandlungseffekt (LATE). Gem. der Definition der abhängigen Variablen, interpretieren wir die Koeffizienten von **prot** in de Regressionen (1), (2) und (3) als erwartete Prozentänderung durch Reformation. Der Koeffizient in Regression (4) gibt die erwartete Änderung des Gini-Index an. Sämtliche geschätzte Effekte sind signifikant und haben ein mit der Hypothese der Autoren konsistentes negatives Vorzeichen.

Die Ergebnisse sind Evidenz, dass Protestantismus zu verringerter Präferenz

für Freizeit, Umverteilung sowie wirtschaftspolitische Intervention seitens des Staats führt. Auch die ökonomische Ungleichheit ist signifikant geringer, als in einer durchschnittlichen vollständig katholischen Gemeinde.

4.4.5 Addendum: FRDD-Schätzung mit `rdrobust()`

Die Funktion `rdrobust::rdrobust()` erlaubt die Schätzung von SRDD und FRDD mit einer Vielzahl von Optionen, s. `?rdrobust`. Dies erleichtert die Schätzung mehrerer Modellspezifikationen und Bandweiten. Mit dem nachstehenden Befehl schätzen wir den LATE von Reformation auf die Präferenz für Umverteilung anhand lokaler quadratischer Regression. Der Output gibt einen Überblick der Bandweiteschätzung sowie der 2 Stufen des 2SLS-Schätzers, inkl. robuster Inferenzstatistiken.

```
pfr_rdr <- rdrobust(
  y = BastenBetz$pfr,
  x = BastenBetz$bord,
  fuzzy = BastenBetz$prot,
  p = 2,
  kernel = "uniform",
  vce = "HC1"
)

pfr_rdr %>%
  summary()
```

Fuzzy RD estimates using local polynomial regression.

Number of Obs.	509	
BW type	mserd	
Kernel	Uniform	
VCE method	HC1	
Number of Obs.	127	382
Eff. Number of Obs.	85	131
Order est. (p)	2	2
Order bias (q)	3	3
BW est. (h)	10.796	10.796
BW bias (b)	22.271	22.271

rho (h/b)	0.485	0.485
Unique Obs.	97	261

First-stage estimates.

Method	Coef.	Std. Err.	z	P> z	[
95% C.I.]					
Conventional	0.701	0.039	17.782	0.000	
[0.624 , 0.778]					
Robust	-	-	15.837	0.000	
[0.599 , 0.768]					

Treatment effect estimates.

Method	Coef.	Std. Err.	z	P> z	[
95% C.I.]					
Conventional	-5.047	2.254	-2.239	0.025	
[-9.464 , -0.629]					
Robust	-	-	-2.210	0.027	
[-10.114 , -0.607]					

Auch für die quadratische Spezifikation erhalten wir mit -5.047 ein vergleichbares signifikantes Ergebnis für den LATE von Protestantismus auf Umverteilung, vgl. Spalte (2) in Tabelle 4.5.

Mit der Option `bwselect = "msetwo"` kann die Bandweite jeweils für die lokale Regression links- und rechtsseitig des Schwellenwerts geschätzt werden.

```
pfr_rdr %>%
  update(bwselect = "msetwo") %>%
  summary()
```

Fuzzy RD estimates using local polynomial regression.

Number of Obs.	509
----------------	-----

BW type	msetwo	
Kernel	Uniform	
VCE method	HC1	
Number of Obs.	127	382
Eff. Number of Obs.	51	134
Order est. (p)	2	2
Order bias (q)	3	3
BW est. (h)	5.340	11.387
BW bias (b)	13.917	22.330
rho (h/b)	0.384	0.510
Unique Obs.	97	261

First-stage estimates.

Method	Coef.	Std. Err.	z	P> z	[
95% C.I.]					
Conventional	0.649	0.046	14.216	0.000	
	[0.560 , 0.739]				
Robust	-	-	11.970	0.000	
	[0.534 , 0.743]				

Treatment effect estimates.

Method	Coef.	Std. Err.	z	P> z	[
95% C.I.]					
Conventional	-7.487	3.378	-2.216	0.027	
	[-14.109 , -0.866]				
Robust	-	-	-2.156	0.031	
	[-14.750 , -0.704]				

Trotz Diskrepanz der geschätzten Bandweiten erhalten wir eine größere aber vergleichbare Schätzung für einen negativen Effekt.

5 Regularisierte Regression

In diesem Kapitel betrachten wir Varianten von Koeffizientenschätzern im linearen Modell

$$Y_i = \beta_1 X_{1,i} + \cdots + \beta_k X_{k,i} + u_i, \quad i = 1, \dots, n, \quad (5.1)$$

deren Motivation die Schätzung von $\beta := (\beta_1, \dots, \beta_k)'$ in Anwendungen ist, in denen der KQ-Schätzer

$$\begin{aligned} \hat{\beta} &= \arg \min_{\beta} \text{RSS}(\beta) \\ &= \arg \min_{\beta} \sum_{i=1}^n (Y_i - \beta_1 X_{1,i} + \cdots + \beta_k X_{k,i})^2 \end{aligned} \quad (5.2)$$

keine stabile Schätzung zulässt oder nicht eindeutig definiert ist, und damit gar nicht erst berechnet werden kann. Solche Szenarien ergeben sich in der empirischen Forschung, wenn die Regressoren stark korreliert sind und/oder das Modell viele Regressoren enthält ($k \lesssim n$), oder das Regressionsproblem hoch-dimensional ist ($k > n$).

Regularisierte Regressionsschätzer begegnen dieser Problematik mit einer Modifikation der Verlustfunktion RSS in (5.2),

$$\text{RSS}(\beta, p, \lambda) := \text{RSS}(\beta) + \lambda \|\beta\|_p. \quad (5.3)$$

Hierbei ist $\lambda > 0$ ein Tuning-Parameter und $p \geq 1$ definiert die p -Norm des Koeffizientenvektors,

$$\|\beta\|_p := \left(\sum_{j=1}^k |\beta_j|^p \right)^{1/p} > 0. \quad (5.4)$$

Wegen $\lambda \|\beta\|_p > 0$ kann die p -Norm des Koeffizientenvektors β das Optimierungsproblem

$$\min_{\beta} \text{RSS}(\beta, p, \lambda) \mid p, \lambda$$

derart restringieren, dass die geschätzten Koeffizienten

$$\hat{\beta}_{p,\lambda} := \arg \min_{\beta} \text{RSS}(\beta, p, \lambda)$$

im Erwartungswert absolut kleiner ausfallen als bei der KQ-Schätzung: Der Schätzer ist in Richtung 0 verzerrt.¹ Dieser Effekt der Regularisierung wird in der Literatur als *Shrinkage* bezeichnet.

Die grundlegenden Eigenschaften des Schätzers $\hat{\beta}_{p,\lambda}$ werden maßgeblich durch den Parameter p bestimmt, der hinsichtlich des zu lösenden Regressionsproblems *a priori* gewählt wird.²

Shrinkage ist eine Motivation für die Anwendung regularisierter Schätzer in Modellen, die auch mit KQ geschätzt werden könnten. Um dies zu verstehen, nehmen wir an, dass die Gauss-Markov-Annahmen in (5.1) gelten. Dann hat der KQ-Schätzer die kleinste Varianz unter allen *unverzerrten* Schätzern. Aufgrund der Shrinkage fallen regularisierte Schätzer zwar nicht unter das Gauss-Markov-Theorem, können dafür aber eine geringere Varianz haben als KQ. Schätzer mit solchen Eigenschaften sind nützlich, wenn eine unverzerrte Schätzung von β nicht unser primäres Ziel ist: Für Vorhersagen kann es hilfreich sein, etwas Verzerrung bei der Koeffizientenschätzung in Kauf zu nehmen, um eine hinreichend große Varianzreduktion zu erreichen, sodass ein geringerer erwarteter Vorhersagefehler als für KQ resultiert. Hierbei liegt, eine Abwägung zwischen Verzerrung und Varianz (*Bias Variance Tradeoff*) vor, der durch den Regularisierungsparameter λ beeinflusst wird.

Für die Berechnung des Schätzers in empirischen Anwendungen wird λ meist datengetrieben (mit [Cross Validation](#) oder einem Informationskriterium) geschätzt oder mit einer analytisch fundierten Faustregel gewählt.

Nachfolgend betrachten wir zwei häufig verwendete regularisierte Schätzer, die sich durch die Wahl $p = 1$ (Lasso Regression) bzw. $p = 2$ (Ridge Regression) ergeben und illustrieren ihre Anwendung mit R.

5.1 Ridge Regression

Ridge Regression wurde von Hoerl und Kennard (1970) als Alternative zur KQ-Schätzung bei hoch-korrelierten Regressoren eingeführt. Die Verlustfunktion

¹Beachte, dass für $\lambda = 0$ die Verlustfunktion des KQ-Schätzers folgt.

²D.h. wir wählen p , um einen Schätzer mit für die konkrete Anwendung hilfreichen Eigenschaften zu erhalten.

lautet

$$\text{RSS}(\beta, p = 2, \lambda) = \text{RSS}(\beta) + \lambda \|\beta\|_2, \quad (5.5)$$

d.h. der Parameter λ reguliert den Einfluss eines ℓ_2 -Strafterms

$$\|\beta\|_2 = \sqrt{\sum_{j=1}^k \beta_j^2}$$

auf die Verlustfunktion $\text{RSS}(\beta, p = 2, \lambda)$. Der Ridge-Schätzer ergibt sich als

$$\hat{\beta}_\lambda^R := \arg \min_{\beta} \text{RSS}(\beta) + \lambda \|\beta\|_2. \quad (5.6)$$

Für Das Optimierungsproblem (5.6) kann wir aus den Bedingungen 1. Ordnung

$$-2\mathbf{X}'(\mathbf{Y} - \mathbf{X}\beta) + 2\lambda\beta = \mathbf{0} \quad (5.7)$$

die analytische Lösung

$$\hat{\beta}_\lambda^R = (\mathbf{X}'\mathbf{X} + \lambda\mathbf{I}_p)^{-1}\mathbf{X}'\mathbf{Y}, \quad (5.8)$$

bestimmt werden, wobei \mathbf{I}_k die $k \times k$ Einheitsmatrix ist. Aus Gleichung (5.8) kann die Wirkungsweise des Strafterms $\lambda\|\beta\|_2$ abgeleitet werden: Ridge Regression modifiziert die Diagonale der zu invertierenden Matrix $\mathbf{X}'\mathbf{X}$ durch Addition von $\lambda > 0$. Dies ist hilfreich, wenn

- $k \geq n$ und damit $\mathbf{X}'\mathbf{X}$ nicht invertierbar (singulär) ist. Dann kann der KQ-Schätzer nicht berechnet werden.³ Die Inverse $(\mathbf{X}'\mathbf{X} + \lambda\mathbf{I}_p)^{-1}$ hingegen existiert unter milden Bedingungen.
- hohe Kollinearität vorliegt, sodass $(\mathbf{X}'\mathbf{X})^{-1}$ zwar existiert, aber zu einer instabilen KQ-Schätzung mit hoher Varianz führt.

Für eine grafische Betrachtung des Optimierungskalküls (5.6) betrachten wir die äquivalente Darstellung als Lagrange-Problem

$$\hat{\beta}_\lambda^R := \arg \min_{\|\beta\|_2 \leq t} \text{RSS}(\beta). \quad (5.9)$$

³Beispiel: `X <- matrix(rnorm(100), ncol = 10)`. Vergleiche `solve(t(X) %*% X)` und `solve(t(X) %*% X + diag(.01, nrow = 10))`

In der folgenden interaktiven Grafik illustrieren wir das Optimierungsproblem (5.9) sowie den resultierenden Schätzer der Koeffizienten (β_1, β_2) in einem multiplen Regressionsmodell mit den Regressoren X_1 und X_2 .

- Die blaue Ellipse ist die Menge aller Schätzwerte $(\hat{\beta}_1, \hat{\beta}_2)$ für den angegebenen Wert von RSS. Im Zentrum der Ellipse liegt der KQ-Schätzer, welcher RSS minimiert.
- Der blaue Kreis ist die Menge aller Koeffizienten-Paare (β_1, β_2) , welche die Restriktion $\beta_1^2 + \beta_2^2 \leq t$ erfüllen. Beachte, dass die Größe des Kreises nur durch den Parameter t bestimmt wird, welcher für einen vorgegebenen Wertebereich variiert werden kann.
- Der blaue Punkt ist der Ridge-Schätzer $(\hat{\beta}_{1,t}^R, \hat{\beta}_{2,t}^R)$. Dieser ergibt sich als Schnittpunkt zwischen der blauen RSS-Ellipse und der Restriktionsregion und variiert mit t . Die gestrichelte rote Kurve zeigt den Ridge-Lösungspfad.
- Für kleine Werte t drückt die Shrinkage die geschätzten Koeffizienten Richtung 0, wobei der Lösungspfad i.d.R. nicht-linear verläuft, d.h. die Shrinkage auf den Koeffizienten ist grundsätzlich unterschiedlich. Die Lösung $(\hat{\beta}_{1,t}^R, \hat{\beta}_{2,t}^R) = (0, 0)$ existiert nur als Grenzwert für $t \rightarrow 0$.
- Beachte, dass der Effekt von t auf die Schätzung umgekehrt für λ verläuft: Größere λ führen zu stärkerer Regularisierung.

Diese Interaktive Komponente des Buchs ist nur in der Online-Version verfügbar.

5.1.1 Eigenschaften des Schätzers

Der Ridge-Schätzer $\hat{\beta}_\lambda^R$ ist nicht invariant gegenüber der Skalierung der Regressoren. Für empirische Daten sollte daher vorab eine Standardisierung der erklärenden Variablen durchgeführt werden.⁴ Um die Eigenschaften des Ridge-Schätzers besser zu verstehen, betrachten wir hier den Fall orthonormaler

⁴Bspw. mit der Funktion `scale()`.

Regressoren \mathbf{X}_j .⁵ Dann ist

$$\widehat{\beta}_{\lambda,j}^R = (1 + \lambda)^{-1} \cdot \widehat{\beta}_j, \quad j = 1, \dots, k, \quad (5.10)$$

d.h. der Ridge-Schätzer skaliert die KQ-Lösung mit einem von λ abhängigen Faktor.⁶

Wir illustrieren dies, indem wir den Zusammenhang zwischen KQ- und Ridge-Schätzer im orthonormalen Fall als R-Funktion `ridge_ortho()` implementieren und für die Parameterwerte $\lambda \in \{0, 0.5, 2\}$ plotten.

```
library(tidyverse)

# Funktion für Rige Regression bei orthonormalen Regressoren
ridge_ortho <- function(KQ, lambda) {
  1/(1 + lambda) * KQ
}
```

```
# KQ-Schätzer gegen Ridge-Schätzer plotten
dat <- tibble(KQ = seq(-1, 1, .01))

ggplot(dat) +
  geom_function(fun = ridge_ortho,
               args = list(lambda = 0),
               lty = 2) +
  geom_function(fun = ridge_ortho,
               args = list(lambda = .5),
               col = "red") +
  geom_function(fun = ridge_ortho,
               args = list(lambda = 2),
               col = "blue") +
  xlim(-.4, .4) +
  xlab("KQ-Schätzer von beta_1") +
  ylab("Ridge-Schätzer von beta_1")
```

Abbildung 5.1 zeigt, dass der Ridge-Schätzer eine lineare Transformation des KQ-Schätzers (gestrichelte Linie) ist. Größere Werte des Regularisierungsparameters λ führen zu stärkerer Shrinkage des Koeffizientenschätzers in Richtung

⁵Orthonormalität heißt $\mathbf{X}'_i \mathbf{X}_j = 1$ für $i = j$ und 0 sonst. Dann ist $\mathbf{X}'\mathbf{X} = \mathbf{I}_k$.

⁶ $(1 + \lambda)^{-1}$ wird auch als *Shrinkage-Faktor* bezeichnet.

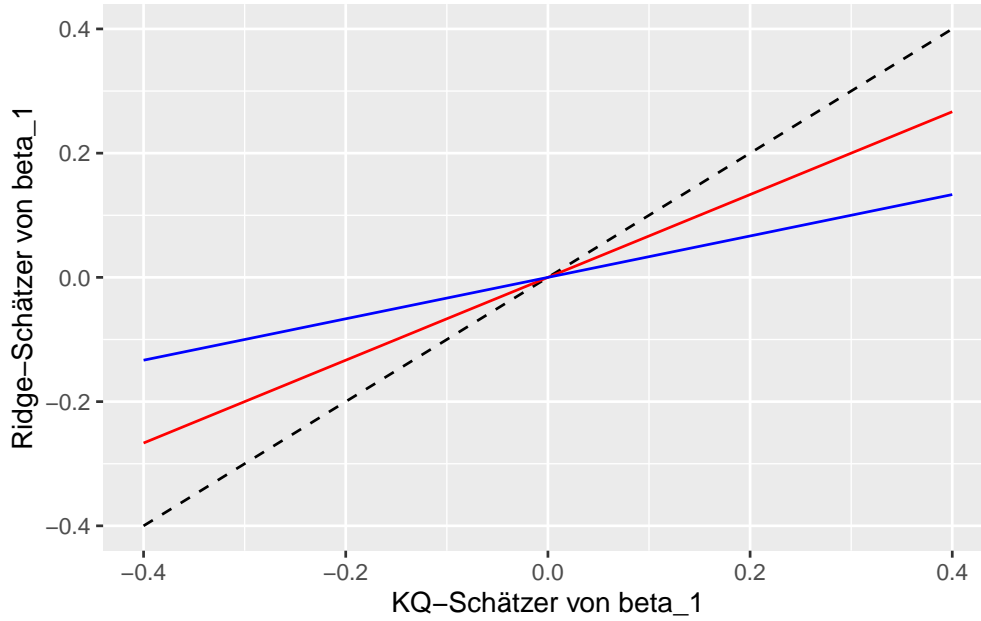


Abbildung 5.1: Shrinkage des OLS-Schätzers bei Ridge Regression

0. Die ℓ_2 -Norm führt zu proportional zum Absolutwert des KQ-Schätzers verlaufender Shrinkage: Größere Koeffizienten werden stärker bestraft als kleine Koeffizienten.

Die Eigenschaft

$$\mathbb{E}(\hat{\beta}_{\lambda,j}^R) = (1 + \lambda)^{-1} \cdot \beta_j$$

zeigt, dass $\hat{\beta}_{\lambda,j}^R$ (für fixes $\lambda > 0$) nicht erwartungstreu für β_j ist. Weiterhin ist

$$\begin{aligned} \text{Var}(\hat{\beta}_{\lambda,j}^R) &= \text{Var}(\hat{\beta}_j) \cdot \left(\frac{\lambda}{1 + \lambda^2} \right) \\ &= \sigma^2 \cdot \left(\frac{\lambda}{1 + \lambda^2} \right), \end{aligned}$$

wobei σ^2 die Varianz des Regressionsfehlers u ist. Wegen $\lambda < (1 + \lambda)^2$ für $\lambda > 0$ gilt

$$\text{Var}(\hat{\beta}_{\lambda,j}^R) < \text{Var}(\hat{\beta}_j).$$

Der Ridge-Schätzer hat also eine kleinere Varianz als der KQ-Schätzer. Diese Eigenschaften können auch für korrelierte Regressoren gezeigt werden.

5.1.2 Ridge Regression mit glmnet

Wir zeigen nun anhand simulierter Daten, wie der Ridge-Lösungspfad mit dem R-Paket `glmnet` berechnet werden kann. Wir erzeugen zunächst Daten gemäß der Vorschrift

$$\begin{aligned} Y_i &= \mathbf{X}_i' \boldsymbol{\beta} + u_i, \\ \beta_j &= \frac{5}{j^2}, \quad j = 1, \dots, 5, \\ \beta_j &= -\frac{5}{(j-5)^2}, \quad j = 6, \dots, 10, \end{aligned} \tag{5.11}$$

$$\mathbf{X}_i \sim N(\mathbf{0}, \boldsymbol{\Sigma}), \quad u_i \stackrel{u.i.v.}{\sim} N(0, 1), \quad i = 1, \dots, 25.$$

Hierbei wird $\boldsymbol{\Sigma}$ so definiert, dass jeder Regressor $N(0, 1)$ -verteilt ist und eine Korrelation von 0.8 mit allen anderen Regressoren aufweist. Mit der Vorschrift für die β_j stellen wir sicher, dass es wenige Variablen gibt, die Y stark beeinflussen, da der Absolutbetrag der Koeffizienten in j abnimmt.⁷

```
library(gendata)
set.seed(1234)

# Parameter definieren
N <- 80
k <- 10

coefs <- 5/(1:(k/2))^2
beta <- c(coefs, -coefs)

# Beobachtungen simulieren
X <- as.matrix(
  genmvnorm(
    k = k,
    cor = rep(.8, (k^2-k)/2),
    n = N)
)
Y <- X %*% beta + rnorm(N)
```

⁷Für bessere Interpretierbarkeit der Grafischen Auswertung, wählen wir positive und negative Koeffizienten mit gleichem Betrag.

Wir schätzen nun ein Modell mit allen 10 Regressoren mit `glmnet`. Beachte, dass für den Ridge-Strafterm `alpha = 0` gesetzt werden muss.⁸

```
library(glmnet)

# Ridge-Regression anpassen
ridge_fit <- glmnet(
  x = X,
  y = Y,
  alpha = 0 # für Ridge-Strafterm
)
```

Der Lösungspfad der Ridge-Schätzung kann nach Transformation der geschätzten Koeffizienten und der zugehörigen λ -Werte in ein langes Format überführt und komfortabel mit `ggplot2` dargestellt werden.

```
# Lambda-Sequenz auslesen
lambdas <- ridge_fit$lambda

# Ridge-Schätzung für Lambdas im langen Format
as.matrix(ridge_fit$beta) %>%
  as_tibble() %>%
  rownames_to_column("Variable") %>%
  pivot_longer(-Variable) %>%
  group_by(Variable) %>%
  mutate(lambda = lambdas) %>%

# Grafik mit ggplot erzeugen
ggplot(
  mapping = aes(
    x = lambda,
    y = value,
    col = Variable
  )
) +
geom_line() +
ylab("gesch. Koeffizienten") +
scale_x_log10("log_10(lambda)")
```

⁸`alpha` ist ein Mischparameter im Algorithmus für `elastic net`, siehe `?glmnet`.

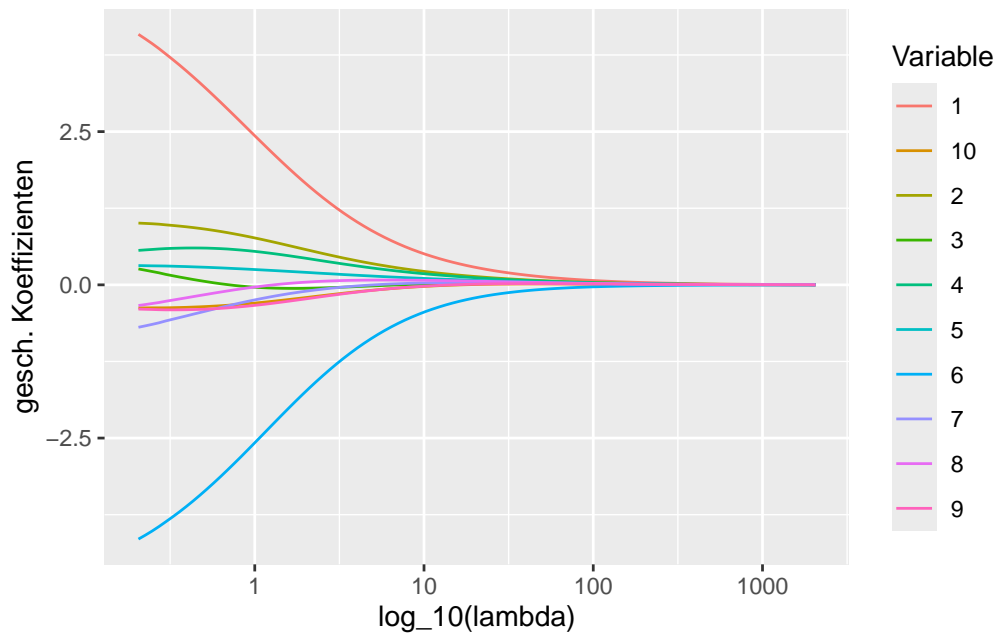


Abbildung 5.2: Lösungspfad für Ridge-Schätzung

Abbildung 5.2 zeigt den nicht-linearen Verlauf der Shrinkage auf den geschätzten Modellkoeffizienten. Die Koeffizienten werden mit zunehmendem λ von der KQ-Lösung ausgehend (linkes Ende der Skala) in Richtung 0 gezwungen.

Über die Funktion `cv.glmnet()` kann ein optimales λ mit Cross Validation (CV) ermittelt werden. Ähnlich wie bei `glmnet()` wird für die Validierung automatisch eine λ -Sequenz erzeugt. Wir nutzen `autoplot()` aus dem R-Paket `ggfortify` für die Visualisierung der Ergebnisse mit `ggplot2`.

```
library(ggfortify)

# Cross-validierte Bestimmung von lambda
ridge_cvfit <- cv.glmnet(
  y = Y,
  x = X,
  intercept = F,
  alpha = 0
)

# Ergebnisse plotten
ridge_cvfit %>%
  autoplot(label.n = 0)
```

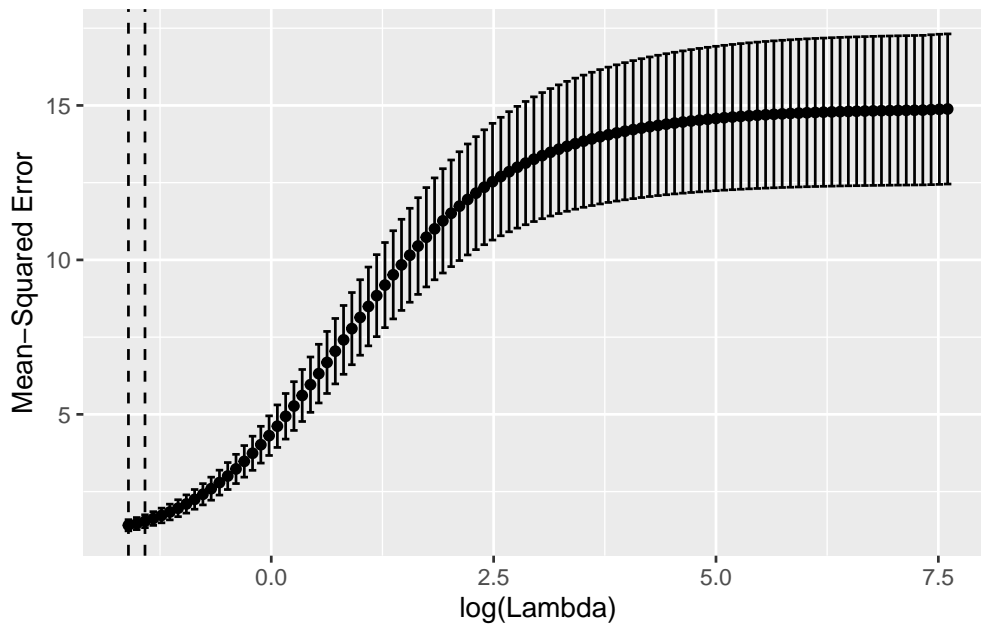



Abbildung 5.3: Lösungspfad für Ridge-Schätzung

Abbildung 5.3 zeigt `ridge_cvfit$lambda.min`, das optimale λ mit dem geringsten CV Mean-Squared-Error (linke gestrichelte Linie) und `ridge_cvfit$lambda.1se`, das größte λ , welches innerhalb einer Standardabweichung entfernt ist (rechte gestrichelte Linie).⁹ Wir berechnen die Schätzung für `lambda.min`.

```
(
  ridge_coefs <- coef(
    object = ridge_cvfit,
    s = ridge_cvfit$lambda.min
  )
)
```

11 x 1 sparse Matrix of class "dgCMatrix"

```
      s1
(Intercept) .
X1          4.1302194
X2          1.0245661
X3          0.3139297
X4          0.5697498
X5          0.2928664
```

⁹Die Wahl von `lambda.1se` ist eine Heuristik, welche die Schätzunsicherheit berücksichtigt und zu einem "sparsameren" Modell tendiert.

X6	-4.1693524
X7	-0.7509305
X8	-0.3844761
X9	-0.3841997
X10	-0.4078514

Wir schätzen das Modell nun mit KQ und vergleichen die Koeffizienten mit der Ridge-Schätzung.

```
# KQ-Schätzung durchführen
KQ_fit <- lm(Y ~ X - 1)

# Koeffizienten auslesen und transformieren:
tibble(
  Ridge = as.matrix(ridge_coefs)[2:11, ],
  KQ = KQ_fit$coefficients
) %>%
  mutate(j = factor(1:10)) %>%
  pivot_longer(
    cols = Ridge:KQ,
    names_to = "Methode",
    values_to = "Koeffizient"
  ) %>%

# Bar-Plot für Koeffizientenvergleich erzeugen
ggplot(
  mapping = aes(
    x = j,
    y = Koeffizient,
    fill = Methode
  )
) +
geom_bar(
  position = "dodge",
  stat = "identity",
  width = .5
)
```

Der Vergleich anhand von Abbildung 5.4 zeigt deutlich, dass Ridge Regression im Vergleich mit KQ zu absolut kleineren Koeffizientenschätzungen tendiert.

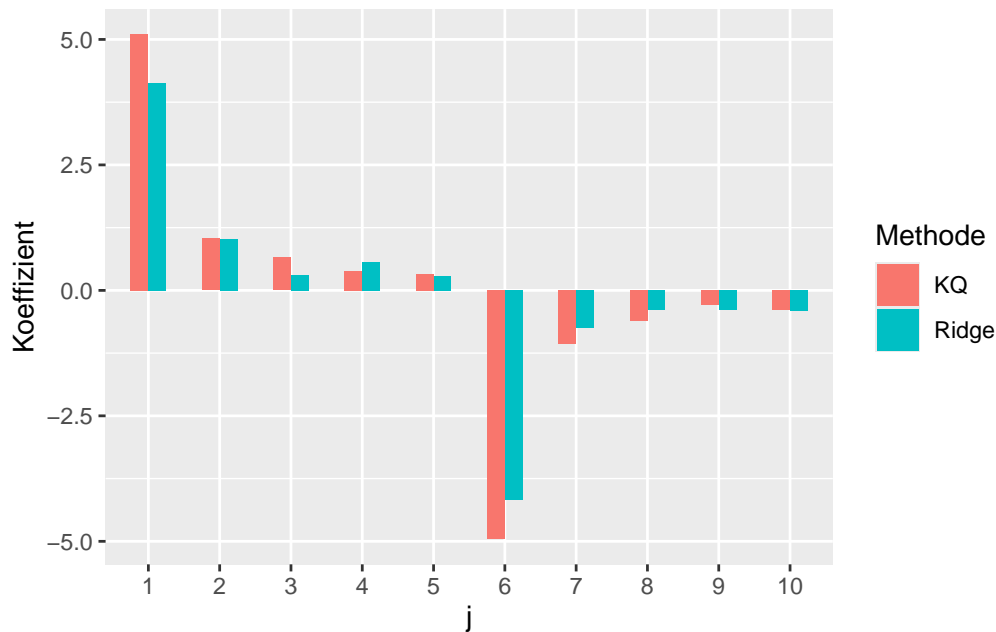


Abbildung 5.4: Koeffizientenvergleich: Ridge vs. KQ

Inwiefern dies Konsequenzen für die Prognosegüte der Schätzung hat, können wir anhand eines Testdatensatzes bestimmen. Hierzu vergleichen wir die mittleren Fehler (MSE) bei der Prognose von Y für die Beobachtungen im Testdatensatz. Für die Simulation des Testdatensatzes nutzen wir erneut die Vorschrift (5.11) um 80 neue Beobachtungen zu erzeugen.

```
# Test-Datensatz erstellen
set.seed(4321)
# Regressoren
new_X <- as.matrix(
  genmvmnorm(
    k = k,
    cor = rep(.85, (k^2-k)/2),
    n = N
  )
)
# Abh. Variable
new_Y <- new_X %*% beta + rnorm(N)
```

Für beide Methoden können wir `predict()` für die Prognosen von Y für den Testdatensatz (`new_Y`) nutzen.

```
# Ridge: Vorhersage von new_Y für Test-Datensatz
Y_predict_ridge <- predict(
  object = ridge_cvfit,
  newx = new_X,
  s = ridge_cvfit$lambda.min
)

# Ridge: MSE für Test-Datensatz berechnen
mean((Y_predict_ridge - new_Y)^2)
```

```
[1] 1.288457
```

Die Vorhersage für `lm()` benötigt dieselben Variablennamen wie im angepassten Modell, s. `KQ_fit$coefficients`.

```
# Test-Datensatz für predict.lm() formatieren
new_X <- as.data.frame(new_X)
colnames(new_X) <- paste0("X", 1:k)

# KQ: Vorhersage von new_Y für Test-Datensatz
Y_predict_KQ <- predict(
  object = KQ_fit,
  newdata = new_X
)

# KQ: MSE für Test-Datensatz berechnen
mean((Y_predict_KQ - new_Y)^2)
```

```
[1] 29.33797
```

Die Ergebnisse zeigen, dass der Ridge-Schätzer trotz seiner Verzerrung einen deutlich geringeren mittleren Vorhersagefehler für die Testdaten erzielt als der KQ-Schätzer. Diese Eigenschaft der Koeffizientenschätzung kann die Prognosegüte von Ridge Regression gegenüber der KQ-Regression verbessern.

5.1.3 Beispiel: Vorhersage von Abschlussnoten in Mathe

Zur Illustration von Ridge Regression nutzen wir den Datensatz **SP** aus Cortez und Silva (2008).¹⁰ **SP** enthält Beobachtungen zu Leistungen von insgesamt 100 Schülerinnen und Schülern im Fach Mathematik in der Sekundarstufe an zwei portugiesischen Schulen. Neben der Abschlussnote in Mathe (**G3**, Skala von 0 bis 20) beinhaltet **SP** diverse demografische, soziale und schulbezogene Merkmale, die mithilfe von Schulberichten und Fragebögen erhoben wurden. Ziel ist es, ein Modell für die Prognose von **G3** anzupassen.

Wir lesen zunächst die Daten (im **.csv**-Format) ein.

```
# Daten einlesen
SP <- read_csv(file = "datasets/SP.csv")
```

Ein Überblick zeigt, dass der Großteil der Regressoren aus kategorialen Variablen mit sozio-ökonomischen Informationen besteht.

```
# Überblick
glimpse(SP)
```

```
Rows: 100
Columns: 31
$ school    <chr> "GP", "GP", "GP", "MS", "GP", "GP", "GP",
"GP", "GP", "GP",~
$ sex       <chr> "M", "M", "F", "F", "M", "F", "F", "F", "F",
"F", "M", "M",~
$ age       <dbl> 17, 18, 19, 17, 16, 16, 19, 16, 16, 16, 18,
16, 15, 17, 17,~
$ address   <chr> "R", "R", "U", "U", "U", "U", "U", "U", "U",
"R", "U", "U",~
$ famsize   <chr> "GT3", "GT3", "LE3", "GT3", "LE3", "GT3",
"GT3", "GT3", "GT~
$ Pstatus   <chr> "T", "T", "T", "T", "A", "T", "T", "T", "A",
"T", "T", "T",~
$ Medu      <dbl> 1, 4, 3, 2, 3, 2, 0, 2, 3, 4, 4, 2, 1, 2, 2,
3, 3, 4, 4, 2,~
```

¹⁰Wir verwenden eine Auszug aus dem Originaldatensatz, der nebst ausführlicher Variablenbeschreibung [hier](#) verfügbar ist.

\$ Fedu <dbl> 2, 3, 2, 2, 4, 3, 1, 1, 1, 4, 4, 2, 2, 3, 2,
 3, 1, 3, 4, 2,~
 \$ Mjob <chr> "at_home", "teacher", "services", "other",
 "services", "oth~
 \$ Fjob <chr> "other", "services", "other", "at_home",
 "other", "other", ~
 \$ reason <chr> "home", "course", "reputation", "home",
 "home", "reputation~
 \$ guardian <chr> "mother", "mother", "other", "mother",
 "mother", "mother", ~
 \$ traveltime <dbl> 1, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 2, 1, 1, 1,
 2, 1, 1, 1, 1,~
 \$ studytime <dbl> 2, 3, 2, 3, 2, 2, 2, 1, 2, 2, 1, 2, 2, 2, 1,
 1, 2, 3, 1, 2,~
 \$ failures <dbl> 0, 0, 1, 0, 0, 0, 3, 0, 3, 0, 0, 0, 0, 0, 0,
 0, 0, 0, 0, 0,~
 \$ schoolsup <chr> "no", "no", "no", "no", "yes", "yes", "no",
 "no", "no", "no~
 \$ famsup <chr> "no", "no", "yes", "no", "yes", "yes", "yes",
 "no", "yes", ~
 \$ paid <chr> "no", "no", "yes", "no", "no", "yes", "no",
 "no", "yes", "n~
 \$ activities <chr> "no", "no", "no", "yes", "yes", "yes", "no",
 "no", "no", "y~
 \$ nursery <chr> "yes", "yes", "no", "yes", "yes", "yes",
 "no", "yes", "yes"~
 \$ higher <chr> "yes", "yes", "yes", "yes", "yes", "yes",
 "no", "yes", "yes~
 \$ internet <chr> "no", "yes", "yes", "no", "yes", "no", "no",
 "yes", "yes", ~
 \$ romantic <chr> "no", "yes", "yes", "yes", "no", "no", "no",
 "yes", "no", "~
 \$ famrel <dbl> 3, 5, 4, 3, 5, 4, 3, 4, 2, 2, 1, 5, 4, 5, 3,
 5, 4, 4, 5, 5,~
 \$ freetime <dbl> 1, 3, 2, 4, 3, 4, 4, 5, 3, 4, 4, 4, 3, 3, 4,
 4, 5, 2, 3, 4,~
 \$ goout <dbl> 3, 2, 2, 3, 3, 3, 2, 2, 3, 4, 2, 4, 2, 3, 4,
 2, 4, 2, 3, 4,~
 \$ Dalc <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 1, 1, 1,
 1, 2, 1, 1, 1,~

```

$ Walc      <dbl> 5, 2, 2, 1, 1, 3, 1, 1, 2, 3, 2, 4, 1, 3, 3,
1, 3, 2, 1, 1,~
$ health    <dbl> 3, 4, 1, 3, 5, 4, 5, 5, 4, 4, 1, 5, 5, 3, 5,
5, 1, 3, 5, 5,~
$ absences  <dbl> 4, 9, 22, 8, 4, 6, 2, 20, 5, 6, 5, 0, 2, 2,
12, 0, 17, 0, 4~
$ G3        <dbl> 10, 16, 11, 11, 11, 10, 9, 12, 7, 11, 16, 12,
9, 12, 12, 13~

```

Um die Prognosegüte des Modells beurteilen zu können, partitionieren wir SP zufällig in einen Test- sowie einen Trainingsdatensatz (mit 30 und 70 Beobachtungen), jeweils für die Regressoren und die abhängige Variable.

```

# ID für Beobachtungen im Testdatensatz zufällig erzeugen
set.seed(1234)
ID <- sample(1:nrow(SP), size = 30)

# Regressoren aufteilen
SP_test <- SP[ID,]
SP_train <- SP[-ID,]

# Abh. Variable aufteilen
Y_test <- SP_test$G3
Y_train <- SP_train$G3

```

Als nächstes passen wir ein Ridge-Regressionsmodell für alle Regressoren in SP_train an und ermitteln ein optimales λ mit Cross Validation. Beachte, dass cv.glmnet nicht für Regressoren im data.frame/tibble-Format ausgelegt ist, sondern ein matrix-Format erwartet. Wir transformieren SP_train daher mit data.matrix().

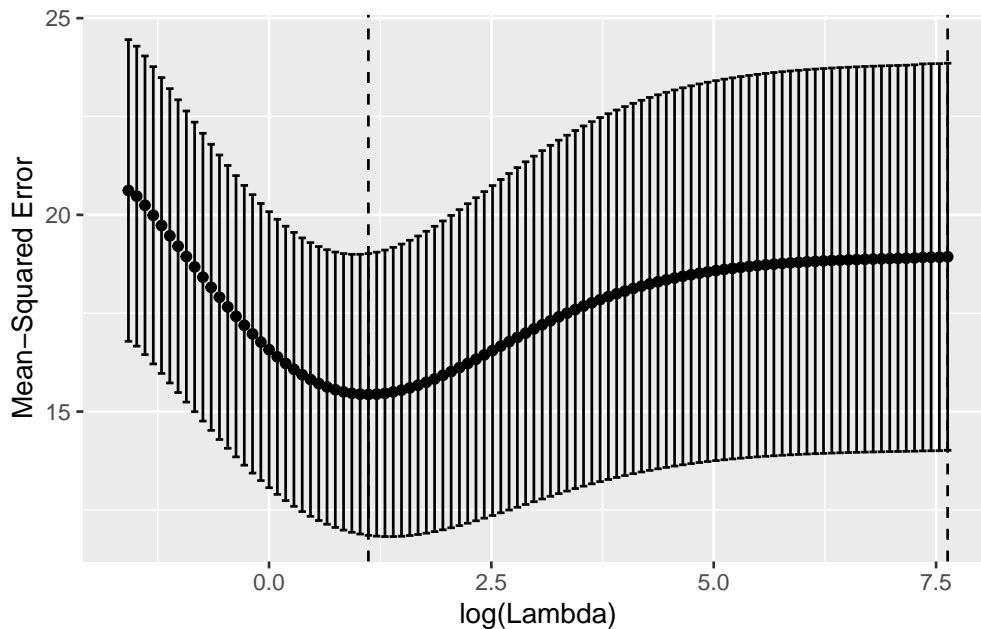
```

# Ridge-Regression und CV für Trainingsdaten
SP_fit_cv <- cv.glmnet(
  x = data.matrix(SP_train %>% select(-G3)),
  y = Y_train,
  alpha = 0
)

# CV-Ergebnisse für lambda visualisieren

```

```
SP_fit_cv %>%
  autoplot(label.n = 0)
```



Wie für das Beispiel mit simulierten Daten erhalten wir mit `predict()` Vorhersagen für die erzielte Punktzahl. Beachte, dass wir den MSE nicht für die Trainingsdaten `SP_train`, sondern für die Testdaten `SP_test` berechnen.

```
# Prognose von G3 anhand des Ridge-Modells
Y_predict_ridge <- predict(
  object = SP_fit_cv,
  newx = data.matrix(
    SP_test %>%
      select(-G3)
  ),
  s = SP_fit_cv$lambda.min
)

# MSE für Testdaten berechnen
mean((Y_predict_ridge - Y_test)^2)
```

```
[1] 21.13249
```

Auch in diesem empirischen Beispiel zeigt ein Vergleich der MSEs, dass Ridge Regression dem KQ-Schätzer hinsichtlich der Vorhersagegüte überlegen ist.


```

# Modell mit KQ schätzen
SP_fit_KQ <- lm(G3 ~ ., SP_train)

# Prognose
Y_predict_KQ <- predict(
  object = SP_fit_KQ,
  newdata = SP_test %>%
    select(-G3)
)

# Testset-MSE berechnen
mean((Y_predict_KQ - Y_test)^2)

```

```
[1] 29.76893
```

Der MSE für Ridge ist mit 21.13 deutlich kleiner als 29.77, der MSE für KQ.

Für die Interpretation der Ridge-Schätzung erweitern den Code für die `ggplot2`-Grafik der Koeffizienten-Pfade um eine vertikale Linie des mit CV ermittelten λ und fügen mit dem Paket `ggrepel` Labels für die Pfade der größten Koeffizienten hinzu.

```

library(ggrepel)

# Lambda-Sequenz auslesen
lambdas <- SP_fit_cv$lambda

# Ridge-Schätzung für Lambdas im langen Format
df <- as.matrix(SP_fit_cv$glmnet.fit$beta) %>%
  as_tibble() %>%
  mutate(
    Variable = rownames(SP_fit_cv$glmnet.fit$beta)
  ) %>%
  pivot_longer(-Variable) %>%
  group_by(Variable) %>%
  mutate(lambda = lambdas)

# Grafik mit ggplot erzeugen
df %>%
  ggplot(

```

```

mapping = aes(
  x = lambda,
  y = value,
  col = Variable
)
) +
geom_line() +
geom_label_repel(
  data = df %>%
    filter(lambda == min(lambdas)),
  mapping = aes(label = Variable),
  seed = 1234,
  size = 5,
  max.overlaps = 8,
  nudge_x = -.5) +
ylab("gesch. Koeffizienten") +
scale_x_log10("log_10(lambda)") +
geom_vline(
  xintercept = SP_fit_cv$lambda.min,
  col = "red",
  lty = 2
) +
theme(legend.position = "none")

```

Abbildung 5.5 gibt Hinweise darauf, dass neben der Schulzugehörigkeit und Indikatoren für schulische Leistung (bspw. **failures**) sozio-ökonomische Prädiktoren wie **internet** (Internetzugang zuhause), **Pstatus** (Zusammenleben der Eltern) und **address/traveltime** (sozialer Status) relevante Variablen zu sein scheinen.

Das optimale $\lambda_{cv} \approx 0.21$ (gestrichelte rote Linie in Abbildung 5.5) führt zu deutlicher Shrinkage, was eine mögliche Erklärung für den besseren Testset-MSE von Ridge Regression ist: Die Koeffizienten von Variablen mit wenig Erklärungskraft werden durch die Regularisierung in Richtung 0 gezwungen und reduzieren so die Varianz der Vorhersage gegenüber der (idealerweise) unverzerrten KQ-Schätzung.

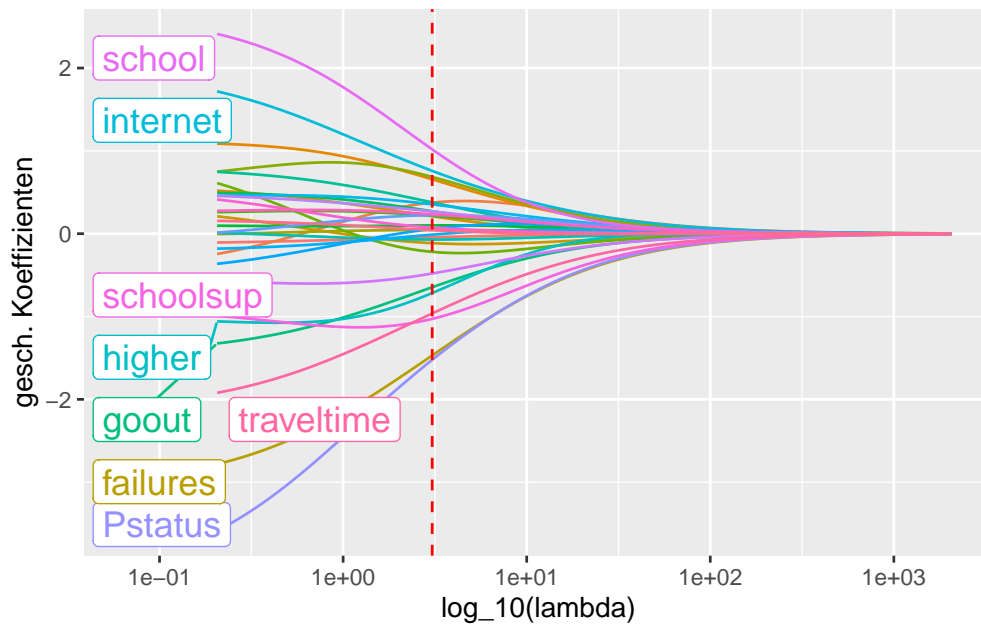


Abbildung 5.5: Lösungspfad für Ridge-Schätzung

i Key Facts zu Ridge Regression

- Ridge-Regression regularisiert den KQ-Schätzer mit der ℓ_2 -Norm der Koeffizienten. Diese Form von Regularisierung ist eine Alternative für KQ in Anwendungen mit mehr Regressoren als Beobachtungen ($k \geq n$) und/oder wenn KQ aufgrund starker Kollinearität eine hohe Varianz aufweist.
- Der Ridge-Schätzer $\hat{\beta}_\lambda^R$ ist *nicht* erwartungstreu. Die geschätzten Koeffizienten sind auch für $n \rightarrow \infty$ verzerrt.
- Aufgrund der verzerrten Schätzung ist statistische Inferenz für Koeffizienten mit $\hat{\beta}_\lambda^R$ problematisch. Anstatt für strukturelle Modelle oder die Schätzung kausaler Effekte wird Ridge Regression in der Praxis daher überwiegend für Prognosen verwendet.
- Die Wahl von λ impliziert einen Tradeoff zwischen Verzerrung und Varianz: Große λ schrumpfen die Koeffizientenschätzer Richtung 0 (mehr Verzerrung), führen aber zu einer kleineren Varianz der Schätzung. Entsprechend können Vorhersagen mit mehr Verzerrung aber weniger Varianz als mit KQ getroffen werden.
- Ridge Regression kann in R mit dem Paket `glmnet` berechnet werden.

5.2 Lasso Regression

Least Absolute Shrinkage and Selection Operator (Lasso) ist ein von Tibshirani (1996) vorgeschlagener Schätzer, der die Verlustfunktion des KQ-Schätzers um einen Strafterm für die Summe der (absoluten) Größe der Koeffizienten $\beta = (\beta_1, \dots, \beta_k)'$ erweitert. Die Verlustfunktion des Lasso-Schätzers von β lautet

$$\text{RSS}(\beta, p = 1, \lambda) = \text{RSS}(\beta) + \lambda \|\beta\|_1. \quad (5.12)$$

Für den Strafterm wird also die ℓ_1 -norm

$$\|\beta\|_1 = \sum_{j=1}^k |\beta_j|$$

verwendet. Der Lasso-Schätzer $\hat{\beta}_\lambda^L$ für β minimiert (5.12),

$$\beta_\lambda^L = \arg \min_{\beta} \text{RSS}(\beta, p = 1, \lambda). \quad (5.13)$$

Entsprechend erhalten wir in Abhängigkeit von λ ein Kontinuum an Lösungen

$$\left\{ \hat{\beta}_\lambda^L \right\}_{\lambda=0}^{\lambda=\infty}, \quad (5.14)$$

der sogenannte *Lasso-Pfad*.

Das Optimierungsproblem (5.12) hat die äquivalente Darstellung

$$\begin{aligned} \hat{\beta}_\lambda^L &= \arg \min_{\beta} \text{RSS}(\beta) + \lambda (\|\beta\|_1 - t) \\ &= \arg \min_{\|\beta\|_1 \leq t} \text{RSS}(\beta), \end{aligned} \quad (5.15)$$

welche über den [Lagrange-Ansatz](#) unter der Nebenbedingung $\|\beta\|_1 \leq t$ gelöst werden kann.

Ähnlich wie der KQ-Schätzer ist der Lasso-Schätzer $\hat{\beta}_\lambda^L$ durch Bedingungen 1. Ordnung bestimmt. Diese Bedingungen lassen sich komfortabel in Matrix-Schreibweise darstellen als

$$-2\mathbf{X}'_j(\mathbf{Y} - \mathbf{X}\beta) + \lambda \cdot \text{sgn}(\beta_j) = 0, \quad j = 1, \dots, k. \quad (5.16)$$

Aus Gleichung (5.16) folgt, dass der Lasso-Schätzer aufgrund des Strafterms im Allgemeinen nicht algebraisch bestimmt werden kann.¹¹

¹¹Zur Bestimmung des Schätzers werden Algorithmen der nicht-linearen Optimierung genutzt.

In Abhängigkeit von λ zwingt der Lasso-Schätzer die KQ-Schätzung von β_j zu einem (absolut) kleineren Wert: Ähnlich wie bei Ridge Regression bewirkt der ℓ_1 -Strafterm eine mit λ zunehmende Schrumpfung der geschätzten Koeffizienten in Richtung 0. Charakteristisch für die Lösung des Lasso-Schätzers ist, dass $\hat{\beta}_j^L = 0$, wenn die Bedingung

$$\left| \mathbf{X}'_j(\mathbf{Y} - \mathbf{X}\hat{\beta}_\lambda^L) \right| - \lambda/2 \leq 0 \quad (5.17)$$

erfüllt ist. In Abhängigkeit von λ kann der Lasso-Schätzer folglich geschätzte Regressionskoeffizienten nicht nur in Richtung 0, sondern diese auch *exakt* mit 0 schätzen und damit *Variablenselektion* betreiben. Aufgrund der mit λ zunehmenden Shrinkage bis die Bedingung (5.17) erfüllt und der Koeffizient gleich 0 gesetzt wird, bezeichnet man Lasso auch als einen *Soft Thresholding Operator*. Im nächsten Abschnitt betrachten wir die Eigenschaften von Lasso-Regularisierung unter vereinfachten Annahmen bzgl. der Regressoren.

5.2.1 Lasso ist Soft Thresholding

Wir betrachten nun eine mathematische Darstellung von Selektions- und Shrinkage-Eigenschaft des Lasso-Schätzers in einem vereinfachten Modell. Wenn die Regressoren \mathbf{X} orthonormal zueinander sind, existiert eine analytische Lösung des Lasso-Schätzers,

$$\hat{\beta}_\lambda^L = \begin{cases} \hat{\beta}_j - \lambda/2 & , \hat{\beta}_j > \lambda/2 \\ 0 & , |\hat{\beta}_j| \leq \lambda/2 \\ \hat{\beta}_j + \lambda/2 & , \hat{\beta}_j < -\lambda/2 \end{cases} \quad (5.18)$$

wobei $\hat{\beta}_j$ der KQ-Schätzer von β_j ist. Anhand von (5.18) können wir die Selektionseigenschaft sowie die Schrumpfung der KQ-Koeffizientenschätzung in Abhängigkeit der durch λ regulierten ℓ_1 -Strafe erkennen. Für eine Visualisierung implementieren wir (5.18) als R-Funktion `lasso_st()` und zeichnen die resultierenden Koeffizientenschätzungen für die Parameterwerte $\lambda \in \{0, 0.2, 0.4\}$.

Wir definieren zunächst die Funktion `lasso_st()`.

```
library(tidyverse)

# Funktion für Lasso soft-thresholding definieren
lasso_st <- function(KQ, lambda) {
  case_when(
```

```

KQ > lambda/2      ~ KQ - lambda/2,
abs(KQ) <= lambda/2 ~ 0,
KQ < -lambda/2     ~ KQ + lambda/2,
)
}

```

Im nächsten Schritt zeichnen wir `lasso_st()` für eine Sequenz von KQ-Schätzwerten gegeben λ .

```

# Sequenz von KQ-Schätzwerten für Illustration definieren
dat <- tibble(
  KQ = seq(-1, 1, .01)
)

# Lasso-Schätzer als Funktion des KQ-Schätzers plotten
ggplot(dat) +
  geom_function(
    fun = lasso_st,
    args = list(lambda = 0),
    lty = 2
  ) +
  geom_function(
    fun = lasso_st,
    args = list(lambda = .2),
    col = "red"
  ) +
  geom_function(
    fun = lasso_st,
    args = list(lambda = .4),
    col = "blue"
  ) +
  xlim(-.4, .4) +
  xlab("KQ-Schätzer von beta_1") +
  ylab("Lasso-Schätzer von beta_1")

```

Abbildung 5.6 zeigt, dass der ℓ_1 -Strafterm des Lasso-Schätzers zu einem linearen Verlauf der auf den KQ-Schätzer (gezeichnet für $\lambda = 0$, gestrichelte Linie) applizierten Shrinkage führt: Der Lasso-Schätzer ist eine abschnittsweise-lineare Funktion des KQ-Schätzers in λ : Je größer der Parameter λ , desto größer ist

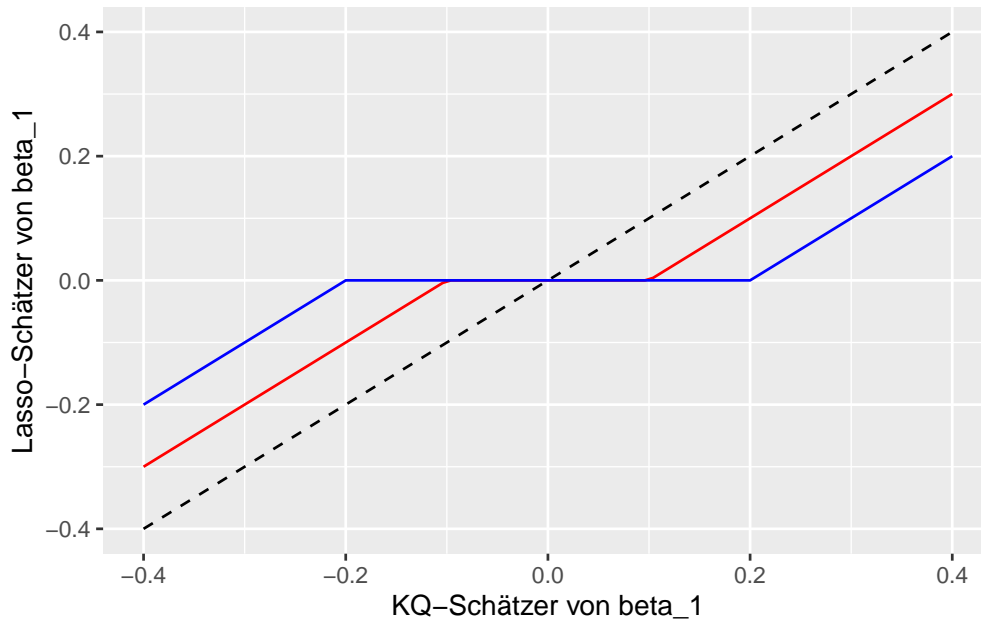


Abbildung 5.6: Shrinkage und Selektion von OLS-Koeffizienten mit Lasso

das Intervall von KQ-Schätzwerten $[-\lambda/2, \lambda/2]$, wo der Lasso-Schätzer zu Variablenselektion führt, d.h. hier den Koeffizienten β_j als 0 schätzt (rote bzw. blaue Linie).

Anhand von Abbildung 5.6 kann abgeleitet werden, dass der Lasso-Schätzer nicht invariant gegenüber der Skalierung der Regressoren ist: Die Stärke der Regularisierung durch λ ist hängt von der Magnitude des KQ-Schätzers ab. Daher müssen die Regressoren vor Berechnung der Schätzung standardisiert werden. Üblich ist hierbei eine Normierung auf einen Mittelwert von 0 und eine Varianz von 1.

Die nachstehende interaktive Grafik illustriert das Lasso-Optimierungsproblem (5.15) sowie den resultierenden Schätzer der Koeffizienten (β_1, β_2) in einem multiplen Regressionsmodell mit korrelierten Regressoren X_1 und X_2 .

- Die blaue Ellipse ist die Menge aller Schätzwerte $(\hat{\beta}_1, \hat{\beta}_2)$ für den angegebenen Wert von RSS. Im Zentrum der Ellipse liegt der KQ-Schätzer, welcher RSS minimiert.
- Das graue Quadrat ist die Menge aller Koeffizienten-Paare (β_1, β_2) , welche die Restriktion $|\beta_1| + |\beta_2| \leq t$ erfüllen. Beachte, dass die Größe dieser Region nur durch den Parameter t bestimmt wird.
- Der blaue Punkt ist der Lasso-Schätzer $(\hat{\beta}_{1,t}^L, \hat{\beta}_{2,t}^L)$. Dieser ergibt sich als Schnittpunkt zwischen der blauen RSS-Ellipse und der Restriktionsregion und variiert mit t . Die gestrichelte rote Linie zeigt den Lasso-Lösungspfad.

- Für kleine Werte, erhalten wir starke Shrinkage auf $\hat{\beta}_{1,t}$ bis zum Wertebereich $t \leq 50$, wo $\hat{\beta}_{1,t}^L = 0$. Hier erfolgt Variablenselektion: Die Regularisierung führt zu einem geschätzten Modell, das lediglich X_2 als erklärende Variable enthält. In diesem Bereich von t bewirkt die Shrinkage, dass $\hat{\beta}_{2,t}^L \rightarrow 0$ für $t \rightarrow 0$.

Diese Interaktive Komponente des Buchs ist nur in der Online-Version verfügbar.

Beachte, dass der rote Lasso-Pfad (die Menge aller Lasso-Lösungen) äquivalent als Funktion von λ im Optimierungsproblem (5.12) dargestellt werden kann. Implementierungen mit statistischer Software berechnen die Lasso-Lösung häufig in Abhängigkeit von λ . Ein Algorithmus hierfür ist LARS.

5.2.2 Berechnung der Lasso-Lösung mit dem LARS-Algorithmus

Für die Berechnung des Lasso-Lösungspfads kann der [LARS-Algorithmus](#) von Efron u. a. (2004) im Lasso-Modus genutzt werden.¹² Der Lasso-Lösungspfad beinhaltet geschätzte Koeffizienten über ein Intervall für λ , welches sämtliche Modellkomplexitäten zwischen der (trivialen) Lösung mit maximaler Shrinkage auf allen Koeffizienten (λ groß, alle gesch. Koeffizienten sind 0) und der unregulisierten Lösung ($\lambda = 0$, KQ-Schätzung) abbildet. Der LARS-Algorithmus erzeugt den Lösungspfad sequentiell, sodass die Schätzung als Funktion von λ veranschaulicht werden kann, ähnlich wie bei Ridge Regression.

Wir zeigen nun anhand simulierter Daten, wie Lasso-Lösungen mit dem R-Paket `lars` berechnet werden können. Hierfür erzeugen wir Daten gemäß der Vorschrift

$$Y_i = \mathbf{X}_i' \boldsymbol{\beta}_v + u_i$$

$$\boldsymbol{\beta}_v = (-1.25, -.75, 0, 0, 0, 0, 0, .75, 1.25)' \quad (5.19)$$

$$\mathbf{X}_i \sim N(\mathbf{0}, \mathbf{I}_{9 \times 9}), \quad u_i \stackrel{u.i.v.}{\sim} N(0, 1), \quad i = 1, \dots, 25.$$

¹²LARS steht für *Least Angle Regression*.


```

library(lars)
set.seed(1234)

# Parameter definieren
N <- 25
beta_v <- c(-1.25, -.75, 0, 0, 0, 0, 0, .75, 1.25)

# Beobachtungen simulieren
X <- matrix(rnorm(N * 9), ncol = 9)
Y <- X %*% beta_v + rnorm(N)

```

Entsprechend des DGP passen wir ein Modell ohne Konstante an. Damit `lars::lars()` den Lösungspfad des Lasso-Schätzers berechnet, muss `type = "lasso"` gewählt werden.¹³

```

# Lösungen des Lasso-Schätzers mit LARS berechnen
(
  fit_lars <- lars(
    x = X,
    y = Y,
    intercept = F,
    type = "lasso" # Wichtig: Lasso-Modus
  )
)

```

Call:

```
lars(x = X, y = Y, type = "lasso", intercept = F)
```

R-squared: 0.858

Sequence of LASSO moves:

```
Var  9 2 8 1 3 5 4 7 6
```

```
Step 1 2 3 4 5 6 7 8 9
```

Die Zusammenfassung zeigt, dass der LARS-Algorithmus als erstes die (relevante) Variable X_9 aktiviert.¹⁴ Mit abnehmender Regularisierung (kleinere λ) werden in den nächsten 3 Schritten die übrigen relevanten Variablen X_2 ,

¹³`lars()` standardisiert die Regressoren standardmäßig (aufgrund des DGPs hier nicht nötig).

¹⁴Aktivierung meint die Aufnahme einer Variable in der Modell gegeben eines hinreichend kleinen λ .

X_8 und X_1 aktiviert. Über die weiteren Schritte nähert der Algorithmus die Lösung an die *saturierte* Schätzung (das Modell mit allen neun Regressoren) an und aktiviert schrittweise die übrigen, irrelevanten Variablen.

Wir visualisieren die geschätzten Koeffizienten an jedem Schritt des Lösungspfads als Funktion von λ . In der Praxis wird der Regularisierungsparameter häufig auf der natürlichen log-Skala dargestellt.

```
# Transformation in ein weites Format
fit_lars$beta %>%
  as_tibble() %>%
  mutate(
    lambda = c(fit_lars$lambda, 1e-2)
  ) %>%
  pivot_longer(
    cols = 1:9,
    names_to = "Variable",
    values_to = "gesch. Koeffizient"
  ) %>%

# Visualisierung mit ggplot
ggplot(
  mapping = aes(
    x = log(lambda),
    y = `gesch. Koeffizient`,
    color = Variable
  )
) +
  geom_line()
```

Abbildung 5.7 zeigt, dass die Shrinkage der geschätzten Koeffizienten nach der Aktivierung rasch abnimmt und sich für kleine Werte von λ der KQ-Lösung annähert. Wir sehen auch, dass es einen Bereich von λ -Werten gibt, für die das wahre Modell mit den Variablen X_1 , X_2 , X_8 und X_9 selektiert werden kann. Je nach Ziel der Analyse kann es sinnvoll sein, ein λ in diesem Intervall zu schätzen.

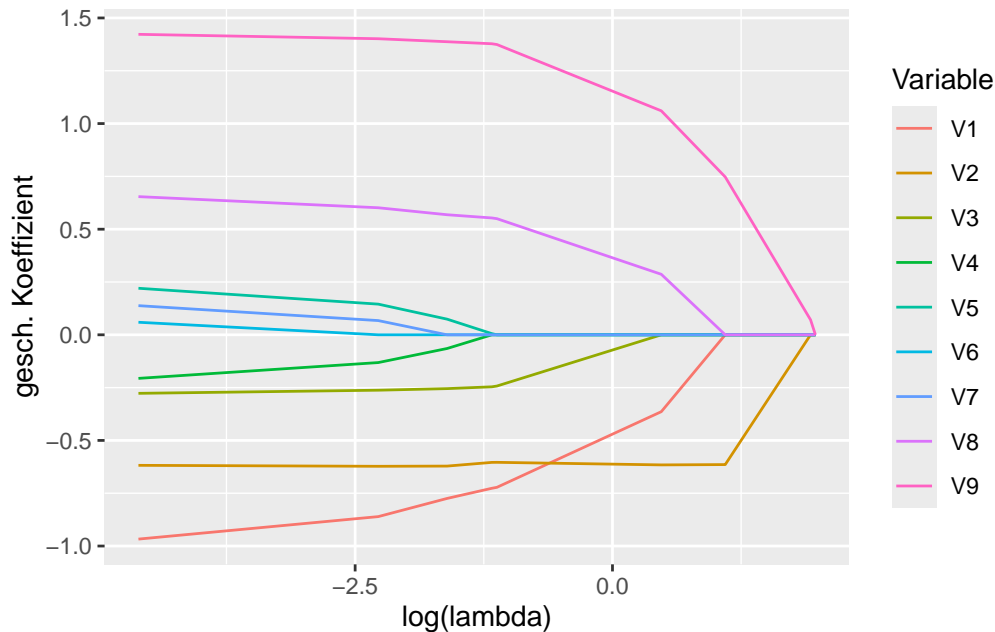


Abbildung 5.7: LARS-Lösungspfad für Lasso-Schätzung

5.2.3 Wahl des Regularisierungsparameters λ für den Lasso-Schätzer

Wie zuvor bei Ridge Regression muss in empirischen Anwendungen ein Wert für den Tuning-Parameter λ gewählt werden. Hierbei besteht die Herausforderung darin, einen geeigneten Wert zu finden, der zu wünschenswerten Eigenschaften des resultierenden Modells führt. So ist für gute Vorhersagen wichtig, dass das Modell nicht zu sehr an die Daten angepasst ist (*Overfitting*), um eine gute Generalisierung auf neue Daten zu ermöglichen. Gleichzeitig muss das Modell flexibel genug sein, um wesentliche Eigenschaften des datenerzeugenden Prozesses hinreichend gut zu erfassen. In der Regel wird hierbei eine sparsame Modellierung angestrebt, die nur eine Teilmenge der Prädiktoren nutzt.

In der Praxis werden verschiedene Verfahren verwendet, um den Wert für den Tuning-Parameter λ zu bestimmen. Gängige Methoden sind *Cross Validation* (CV) und Informationskriterien. In Abhängigkeit der Methode und der Daten ergeben sich ober- oder unterparameterisierte Modelle. Aufgrund der Implementierung im R-Paket `lars` betrachten wir CV.¹⁵ Wir zeigen nachfolgend anhand der simulierten Daten aus dem letzten Abschnitt, wie für die LARS-Schätzung ein optimales λ mit leave-one-out CV (LOO-CV) bestimmt werden kann. Hierzu nutzen wir `lars::cv.lars()` unter Verwendung derselben Argumente wie zuvor im Aufruf von `lars()`.

¹⁵Chetverikov, Liao, and Chernozhukov (2020) zeigen, dass CV zu konsistenter Modellselektion führen kann.

```
# LARS-Lösungen mit CV evaluieren
fit_lars_cv <- cv.lars(
  x = X,
  y = Y,
  intercept = F,
  normalize = T,
  type = "lasso",
  plot.it = F,
  K = N # für LOO-CV
)
```

Das Objekt `fit_lars_cv` ist eine Liste mit den CV-Ergebnissen. Wir können diese einfach mit `ggplot` visualisieren. `index` ist hierbei das Verhältnis der ℓ_1 -Norm des Lasso-Schätzers für einen spezifischen Wert von λ und der ℓ_1 -Norm des KQ-Schätzers. Das optimale λ wird so implizit geschätzt. `cv.error` ist der mit CV geschätzte MSE.

```
# CV-MSE
fit_lars_cv %>%
  as_tibble() %>%

  ggplot(
    mapping = aes(
      x = index,
      y = cv.error
    )
  ) +
  geom_line() +
  xlab("|beta_lambda| / |beta|") +
  ylab("CV-MSE")
```

In der Grafik erkennen wir ein Minimum des CV-MSEs bei etwa 0.73.

```
# CV-MSE-minimierendes Lambda bestimmen
ID <- which.min(fit_lars_cv$cv.error) # Index

(
  fraction_opt <- fit_lars_cv$index[ID]
)
```

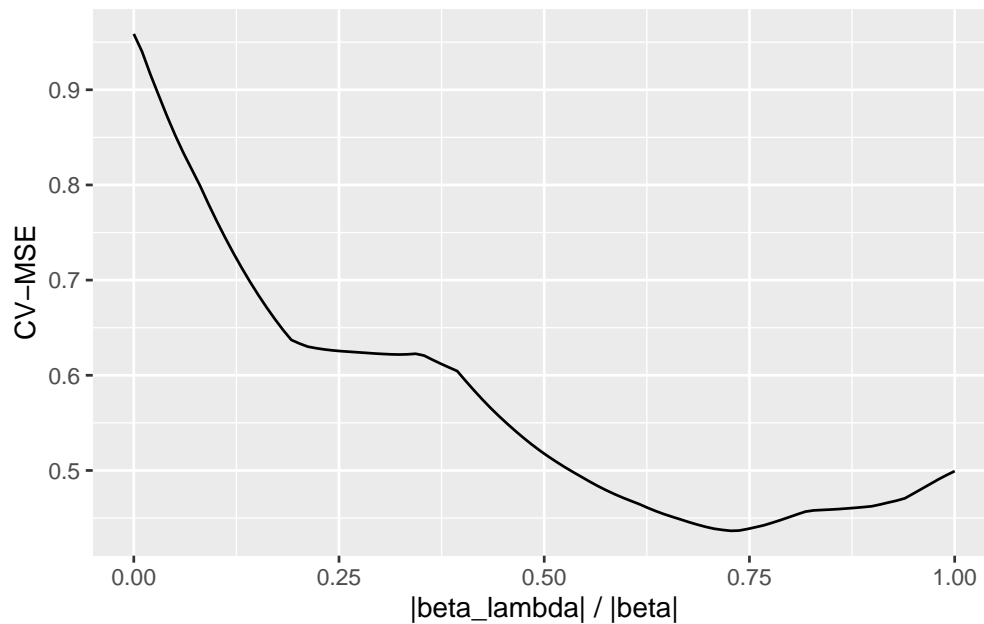


Abbildung 5.8: CV-MSE und relative Position von λ auf dem Lasso Pfad

```
[1] 0.7272727
```

Die geschätzten Koeffizienten für die optimale Regularisierung können mit `coef()` ausgelesen werden.

```
# LARS-Lasso-Fit für optimales lambda bestimmen
coef(
  object = fit_lars,
  s = fraction_opt,
  mode = "fraction"
)
```

```
[1] -0.6513191 -0.6060906 -0.1946089  0.0000000  0.0000000
0.0000000  0.0000000
[8]  0.4977908  1.3122407
```

Das Ergebnis veranschaulicht die Selektionseigenschaft von Lasso: Gemäß DGP (5.19) sind die Variablen X_3 bis X_7 *irrelevante* Prädiktoren für Y ; ihre wahren Koeffizienten sind 0. In der kreuzvalidierten Lasso-Schätzung erreicht die Regularisierung, dass die Koeffizienten der Variablen X_4 bis X_7 tatsächlich mit 0 geschätzt werden. Wir schätzen für das mit CV bestimmte λ also ein leicht überspezifiziertes Modell mit den Regressoren X_1 , X_2 , X_3 , X_8 und X_9 . Beachte, dass die Lasso-Schätzung einen Kompromiss impliziert: Die Varianz der Schätzung ist geringer als die des KQ-Schätzers im Modell mit allen

Variablen.¹⁶ Aufgrund der Regularisierung sind die mit Lasso geschätzten Koeffizienten der relevanten Variablen jedoch in Richtung 0 verzerrt.

Einen positiven Effekt dieses Kompromisses beobachten wir anhand des mittleren Vorhersagefehlers für Daten, die *nicht* zur Berechnung des Schätzers verwendet wurden. Wir vergleichen den Vorhersagefehler nachfolgend anhand eines solchen simulierten Test-Datensatzes mit 25 neuen Beobachtungen. Den Vorhersagefehler bestimmen wir als MSE zwischen den vorhergesagten und den tatsächlichen Ausprägungen für Y .

```
# Test-Datensatz erstellen
set.seed(4321)
new_X <- matrix(rnorm(N * 9), ncol = 9)
new_Y <- new_X %*% beta_v + rnorm(N)

# Lasso: Vorhersage von new_Y für Test-Datensatz
Y_predict_lars <- predict(
  object = fit_lars,
  s = fraction_opt,
  type = "fit",
  mode = "fraction",
  newx = new_X
)$fit

# Lasso: MSE für Test-Datensatz berechnen
mean((Y_predict_lars - new_Y)^2)
```

```
[1] 1.419817
```

Wir schätzen nun das große Modell mit allen 9 Variablen mit KQ und berechnen ebenfalls den MSE der Prognosen für den Test-Datensatz.

```
# KQ-Schätzung des großen Modells durchführen
KQ_fit <- lm(Y ~ X - 1)

# Test-Datensatz für predict.lm() formatieren
new_X <- as.data.frame(new_X)
colnames(new_X) <- paste0("X", 1:9)
```

¹⁶Wegen $N = 25$ verbleiben bei der KQ-Schätzung mit 9 Regressoren nur 16 Freiheitsgrade.

```
# KQ: Vorhersage von new_Y für Test-Datensatz
Y_predict_KQ <- predict(
  object = KQ_fit,
  newdata = new_X
)

# KQ: MSE für Test-Datensatz berechnen
mean((Y_predict_KQ - new_Y)^2)
```

```
[1] 9.851932
```

Offenbar führt die Lasso-Schätzung zu einem deutlich geringeren MSE der Vorhersage von Y für den Test-Datensatz als die KQ-Schätzung und damit zu einer höheren Vorhersagegüte. Das “sparsame” mit Lasso-Regression geschätzte Modell ist dem “großen” mit KQ geschätztem Modell in dieser Hinsicht also überlegen.

i Key Facts zu Lasso-Regression

- Lasso-Regression bestraft die Verlustfunktion des KQ-Schätzers mit der ℓ_1 -Norm der Koeffizienten.
- Neben Koeffizientenschätzung mit Shrinkage in Richtung 0 kann der Lasso-Schätzer Variablenselektion durchführen: Regressionskoeffizienten können exakt mit 0 geschätzt und so ein “sparsames”, leichter zu interpretierendes Modell gewählt werden.
- Wie bei Ridge Regression impliziert die Wahl von λ einen Bias-Variance-Tradeoff, der für Vorhersagen nützlich ist: Für größere λ wird mehr Verzerrung induziert und möglicherweise relevante Variablen mit kleinen Koeffizienten aus dem Modell entfernt. Ein solches sparsames Modell kann eine höhere Prognosegüte haben als ein komplexes, unregularisiertes Modell.
- Der Lasso-Schätzer $\hat{\beta}_\lambda^L$ ist *nicht* erwartungstreu.
- Lasso Regression kann bspw. mit dem LARS-Algorithmus (Paket `lars`) oder mit `glmnet` berechnet werden.

5.3 Vergleich von Lasso- und Ridge-Regression mit Simulation

In diesem Kapitel illustrieren wir Vor- und Nachteile von Lasso- und Ridge-Regression in Prognose-Anwendungen anhand von Monte-Carlo-Simulationen. Wir betrachten hierbei datenerzeugende Prozesse, die sich hinsichtlich der Anzahl relevanter Variablen sowie der Korrelation dieser Variablen unterscheiden.

Die grundlegende Vorschrift für die Simulationen ist

$$Y_i = \sum_{j=1}^{k=40} \beta_j X_{i,j} + u_i, \quad u_i \stackrel{u.i.v.}{\sim} N(0, 1), \quad i = 1, \dots, 100,$$

wobei die Regressoren X_j eine Varianz von 1 haben und aus einer multivariaten Normalverteilung mit Korrelation

$$\rho \in (0, 0.5, 0.8)$$

gezogen werden.

Für die Koeffizienten β unterscheiden wir zwei Szenarien. In Szenario A ist

$$\beta = (1, \dots, 1)',$$

d.h. alle Variablen sind relevant und haben denselben Einfluss auf Y . In Szenario B erzeugen wir β einmalig vorab so, dass

$$\beta_j = \begin{cases} 1, & \text{mit Wsk. } p \\ 0, & \text{mit Wsk. } 1 - p, \end{cases}$$

d.h. nur eine Teilmenge der Variablen beeinflusst Y jeweils mit demselben Effekt $\beta_j = 1$. Die übrigen Variablen sind irrelevant.

Wir schätzen und validieren die Modelle mit `glmnet()`.

5.3.1 Prognosegüte in diversen Szenarien

```
# Simulationsparameter definieren
rho <- c(0, 0.5, 0.8) # Korrelation
k <- 40               # Anz. Regressoren
```



```

N <- 100                # Anz. Beobachtungen
n_sim <- 100            # Anz. Simulationen

```

Damit der Code für die Simulation möglichst wenig repetitiv ist, definieren wir eine Funktion `cv.glmnet_MSE()`, die unter Angabe der Daten `X` und `Y`, des Trainingssets `train` sowie des Parameters `alpha` den gewünschten regulierten Schätzer unter Verwendung von Cross Validation anpasst und den Testset-MSE zurückgibt.

```

# allg. Funktion für Testset-MSE nach CV
cv.glmnet_MSE <- function(X, Y, train, alpha) {

  # Modell mit glmnet schätzen; lambda per CV bestimmen
  fit_cv <- cv.glmnet(
    x = X[train,],
    y = Y[train],
    alpha = alpha
  )

  # Vorhersagen treffen
  Y_pred <- predict(
    object = fit_cv,
    s = fit_cv$lambda.min,
    newx = X[-train,])

  return(
    # Testset-MSE berechnen
    mean(
      (Y[-train] - Y_pred)^2
    )
  )
}

```

Wir initialisieren zunächst Matrizen, in welche die MSEs aus den 100 Simulationsdurchläufen reihenweise geschrieben werden. `lasso_mse` und `ridge_mse` haben je eine Spalte für jede Korrelation in `rho`

```
# Matrizen für simulierte MSEs initialisieren...
lasso_mse <- matrix(
  data = NA,
  nrow = n_sim,
  ncol = length(rho)
)
ridge_mse <- lasso_mse

# ... und benennen
colnames(lasso_mse) <- paste0("Kor=", rho)
colnames(ridge_mse) <- colnames(lasso_mse)
```

Für die Simulation iterieren wir mit `purrr::walk` über den Vektor `rho` sowie über die Laufvariable `1:n_sim`. Beide Schleifen nutzen den Syntax für anonyme Funktionen:

```
# Die anonyme Funktion
function(x) return(x)
# ist äquivalent definiert als
\(x) return(x)
```

In jeden Simulationsdurchlauf erzeugen wir den Datensatz entsprechend der obigen Vorschrift, teilen die Daten auf und berechnen MSEs für Lasso- und Ridge-Regression mit `cv.glmnet_MSE()`.

Szenario A

```
# Koeffizienten-Vektor definieren
beta <- rep(1, k)
```

```
library(mvtnorm)
library(tidyverse)

set.seed(1234)

# Simulation durchführen
walk(1:length(rho), \(j) {
```

```

# Korrelationsmatrix definieren
Sigma <- matrix(
  data = rho[j],
  nrow = k,
  ncol = k
)
diag(Sigma) <- 1

walk(1:n_sim, \(i) {

# Daten simulieren
X <- rmvnorm(
  n = N,
  mean = rep(0, k),
  sigma = Sigma
)
Y <- X %*% beta + rnorm(N)

# Trainingsdaten definieren
ID_train <- sample(
  x = c(1:N),
  size = N/2
)

# Modelle mit CV schätzen und MSEs berechnen
# Ridge-Regression
ridge_mse[i, j] <-<- cv.glmnet_MSE(
  X = X,
  Y = Y,
  train = ID_train,
  alpha = 0
)

# Lasso-Regression
lasso_mse[i, j] <-<- cv.glmnet_MSE(
  X = X,
  Y = Y,
  train = ID_train,
  alpha = 1
)

```

```
)
})
})
```

Beachte, dass hier der Super-Assignment-Operator `<<-` genutzt wird, damit `walk` die Matrizen `ridge_mse` und `lasso_mse` in der globalen Umgebung überschreibt.¹⁷

Wir berechnen jeweils den mittleren MSEs, sammeln die Ergebnisse in einer `tibble()` und nutzen `gt()` für die tabellarische Darstellung.

```
library(gt)

# Ergebnisse tabellarisch darstellen
tibble(
  Methode = c(
    "Lasso-Regression",
    "Ridge-Regression"
  ),
) %>%
  bind_cols(
    bind_rows(
      colMeans(lasso_mse),
      colMeans(ridge_mse)
    )
  ) %>%
  gt() %>%
  tabopts
```

Tabelle 5.1: Durchschnittliche Testset-MSEs für Setting A

Methode	Kor=0	Kor=0.5	Kor=0.8
Lasso-Regression	7.17	10.398	7.581
Ridge-Regression	4.841	1.615	1.517

¹⁷Dies folgt aus der Definition von `walk`. `<-` bewirkt hier lediglich Assignment in der Funktionsumgebung.

Tabelle 5.1 zeigt, dass Ridge-Regression gegenüber Lasso-Regression für jede der drei betrachteten Korrelationen überlegen ist. Insbesondere bei stärker korrelierten Regressoren ist Ridge vorteilhaft.

Für Szenario B überschreiben wir **beta** nach Multiplikation mit einem zufälligen binären Vektor, sodass einige der Koeffizienten 0 und die zugehörigen Variablen irrelevant für Y sind.

Szenario B

```
# Wsk. für Relevanz einer Variable
p <- .3

# Koeffizienten-Vektor definieren
set.seed(123)
beta <- beta * sample(
  x = 0:1,
  size = k,
  replace = T,
  prob = c(1-p, p)
)

# Koeffizienten prüfen
head(beta, n = 10)
```

```
[1] 0 1 0 1 1 0 0 1 0 0
```

Eine wiederholung der Simulation für die modifizierten Koeffizienten **beta** und liefert folgende tabellarische Auswertung.

Tabelle 5.2: Durchschnittliche Testset-MSEs für Szenario B

Methode	Kor=0	Kor=0.5	Kor=0.8
Lasso	2.51	2.143	1.923
Ridge	3.331	2.562	2.014

Die Ergebnisse in Tabelle 5.2 zeigen, dass Ridge-Regression in Szenario B bis auf den Fall unkorrelierter Regressoren etwas schlechter abschneidet als in Szenario A. Die hohe Anzahl irrelevanter Variablen verbessert die Leistung von Lasso deutlich: Hier ist es plausibel, dass Lasso aufgrund der Thresholding-Eigenschaft die Koeffizienten einiger irrelevanten Variablen häufig exakt 0 setzt und damit ein sparsameres Modell schätzt als Ridge. Entsprechend erzielt

Lasso in diesem Szenario insbesondere für $\rho = 0$ genauere Vorhersagen als Ridge Regression.

5.3.2 Visualisierung des Bias-Variance-Tradeoffs bei Prognosen

Für ein besseres Verständnis, wie sich der Regularisierungsparameter λ auf den Bias-Variance-Tradeoff bei Prognosen mit Ridge- und Lasso-Regression auswirkt, vergleichen wir für beide Methoden nachfolgend die Abhängigkeit des MSEs der Prognose \hat{Y}_0 für den Wert Y_0 der abhängigen Variable eines Datenpunkts anhand seiner Regressoren \mathbf{X}'_0 , wobei

$$\text{MSE}(\hat{Y}_0) = \text{Bias}(\hat{Y}_0)^2 + \text{Var}(\hat{Y}_0) + \text{Var}(Y_0) \quad (5.20)$$

Beachte, dass $\text{Var}(Y_0)$ die durch den datenerzeugenden Prozess (und damit unvermeidbare) Varianz von Y_0 ist, wohingegen $\text{Bias}(\hat{Y}_0)^2$ und $\text{Var}(\hat{Y}_0)$ von dem verwendeten Schätzer für \hat{Y}_0 abhängt.

Für die Simulation betrachten wir erneut Szenario A aus Kapitel 5.3.1 mit 50 Beobachtungen für ein Modell mit 40 unkorrelierten Regressoren. Wir legen zunächst die Simulationsparameter fest und erzeugen den vorherzusagenden Datenpunkt (\mathbf{X}_0, Y_0) .

```
# Parameter festlegen
set.seed(1234)
n <- 200 # Anz. Iterationen
N <- 50  # Anz. Beobachtungen
k <- 40  # Anz. Variablen

# Korrelationsmatrix definieren
Sigma <- diag(k) # Diagonalmatrix
beta <- rep(x = 1, k)

# Prognose-Ziel vorab zufällig generieren:

# Regressoren
X_0 <- rmvnorm(
  n = 1,
  mean = rep(x = 0, k)
)
```

```
# Abh. Variable
Y_0 <- X_0 %*% beta + rnorm(n = 1) %>%
  as.vector()
```

Anhand der Simulationsergebnisse wollen wir die von der verwendeten Schätzfunktion abhängigen Komponenten von (5.20) untersuchen. Wir initialisieren hierzu die Listen `ridge_fits` und `lasso_fits`, in die unsere Simulationsergebnisse geschrieben werden.

```
# Listen für Simulationsergebnisse initialisieren
ridge_fits <- list()
lasso_fits <- list()
```

Weiterhin definieren wir separate λ -Sequenzen für Lasso- und Ridge-Schätzer.¹⁸

```
# Lambda-Sequenzen festlegen
lambdas_r <- seq(.25, 2.5, length.out = 100)
lambdas_l <- seq(.05, 0.5, length.out = 100)
```

Für die Simulation iterieren wir mit `walk()` über simulierte Datensätze und schreiben jeweils den vollständigen Output von `glmnet()` in die zuvor definierten Listen `ridge_fits` und `lasso_fits`.

```
# Simulation
walk(1:n, \(i) {

  # Daten simulieren
  X <- rmvnorm(
    n = N,
    mean = rep(0, k),
    sigma = Sigma
  )
  Y <- X %*% beta + rnorm(n = N, sd = 5)

  # Modelle mit glmnet schätzen
```

¹⁸Die Sequenzen haben wir in Abhängigkeit des DGP so gewählt, dass die Abhängigkeit der Prognosegüte von λ gut visualisiert werden kann.

```

# Ridge-Regression
ridge_fits[[i]] <- glmnet(
  x = X,
  y = Y,
  alpha = 0,
  intercept = F
)
# Lasso-Regression
lasso_fits[[i]] <- glmnet(
  x = X,
  y = Y,
  alpha = 1,
  intercept = F
)
})

```

Wir nutzen Funktionen aus `purrr` und `dplyr`, um über die in den Simulationsdurchläufen angepassten Modelle zu iterieren. Mit `predict()` erhalten wir Punktvorhersagen für Y_0 für jedes λ der zuvor definierten λ -Sequenzen. Beachte, dass `map()` jeweils eine Liste mit 200 Punktvorhersagen für jedes der 100 zurückgibt. Mit `list_rbind()` können wir die Ergebnisse komfortabel jeweils in einer `tibble` sammeln.

```

# Prognosen für Ridge-Regression
pred_r <- map(
  .x = ridge_fits,
  .f = ~ as_tibble(
    predict(
      object = .,
      s = lambdas_r,
      newx = X_0
    )
  )
) %>%
  list_rbind()

# Prognosen für Lasso-Regression
pred_l <- map(

```



```

.x = lasso_fits,
.f = ~ as_tibble(
  predict(
    object = .,
    s = lambdas_l,
    newx = X_0)
  )
) %>%
list_rbind()

```

Für die statistische Auswertung berechnen wir jeweils $\text{MSE}(\hat{Y}_0)$, $\text{Bias}(\hat{Y}_0)^2$ und $\text{Var}(\hat{Y}_0)$ und führen die Ergebnisse mit `pivot_longer()` in ein langes Format `sim_data_r` über. Wir berechnen weiterhin mit `MSE_min_r` das λ , für das wir über die Simulationsdurchläufe durchschnittlich den geringsten MSE beobachten.

Ridge-Regression

```

# Ergebnisse für Ridge-Regression zusammenfassen
sim_data_r <- tibble(

  lambda = lambdas_r,

  "MSE" = map_dbl(
    .x = pred_r,
    .f = ~ mean((.x - Y_0)^2)
  ),

  "Bias^2" = map_dbl(
    .x = pred_r,
    .f = ~ (mean(.x) - Y_0)^2
  ),

  "Varianz" = map_dbl(
    .x = pred_r,
    .f = ~ var(.x)
  )
) %>%
pivot_longer(

```

```

      cols = -lambda,
      values_to = "Wert",
      names_to = "Statistik"
    )

# Lambda bei MSE-Minimum bestimmen
MSE_min_r <- sim_data_r %>%
  filter(
    Statistik == "MSE",
    Wert == min(Wert)
  )

```

Lasso-Regression

```

# Ergebnisse zusammenfassen
sim_data_l <- tibble(

  lambda = lambdas_l,

  "MSE" = map_dbl(
    .x = pred_l,
    .f = ~ mean((. - Y_0)^2)
  ),

  "Bias^2" = map_dbl(
    .x = pred_l,
    .f = ~ (mean(.) - Y_0)^2
  ),

  "Varianz" = map_dbl(
    .x = pred_l,
    .f = ~ var(.)
  )
) %>%
pivot_longer(
  cols = -lambda,
  values_to = "Wert",
  names_to = "Statistik"
)

```

```
# Lambda bei MSE-Minimum bestimmen
MSE_min_l <- sim_data_l %>%
  filter(
    Statistik == "MSE",
    Wert == min(Wert)
  )
```

Die Datensätze im langen Format, `sim_data_r` und `sim_data_l`, werden nun für die Visualisierung der Ergebnisse mit `ggplo2` genutzt.

```
# MSE, Bias^2 und Varianz gegen Lambda plotten

# Ridge-Regression
sim_data_r %>%
  ggplot(
    mapping = aes(
      x = lambda,
      y = Wert,
      color = Statistik
    )
  ) +
  geom_line() +
  geom_point(data = MSE_min_r)

# Lasso-Regression
sim_data_l %>%
  ggplot(
    mapping = aes(
      x = lambda,
      y = Wert,
      color = Statistik
    )
  ) +
  geom_line() +
  geom_point(data = MSE_min_l)
```

Anhand von Abbildung 5.9 lässt sich der Bias-Variance-Tradeoff bei der Vorhersage von Y_0 gut erkennen: Bereits für kleine λ erzielen beide Methode

eine deutliche Reduktion des MSE. Dies wird durch etwas zusätzlichen Bias, aber eine überproportionale Verringerung der Varianz erreicht. Der erkennbare funktionale Zusammenhang zeigt, dass der MSE eine konvexe Funktion von λ ist. Damit existieren optimale λ mit minimalem MSE (grüne Punkte), die wir mit Cross Validation schätzen können.

5.4 Inferenz für Treatment-Effekt-Schätzung mit vielen Variablen

In empirischen Studien des Effekts einer Behandlungsvariable B auf eine Outcome-Variable Y steht häufig eine Vielzahl potentieller Kontrollvariablen zur Verfügung. Häufig ist unklar, welche Variablen in das Modell aufgenommen werden sollten, um das Risiko einer verzerrten Schätzung durch ausgelassene Variablen zu vermindern und gleichzeitig eine Schätzung mit geringer Varianz zu gewährleisten. Ist der Beobachtungsumfang N relativ zur Variablenanzahl k groß, so kann die KQ-Schätzung einer langen Regression (ein Modell mit allen k Kontrollvariablen) gute Ergebnisse liefern. In der Praxis liegt diese wünschenswerte Situation jedoch oft nicht vor und es ist $k \lesssim N$ oder sogar $k > N$. Dann ist eine KQ-Schätzung des Behandlungseffekts anhand aller k Variablen mit hoher Varianz behaftet bzw. gar nicht möglich.¹⁹ Ein weiteres Szenario ist $k(N) > N$, d.h. die Anzahl der Regressoren kann mit dem Beobachtungsumfang wachsen.²⁰ Lasso-Verfahren können dann hilfreich sein, um Determinanten von Y und B zu identifizieren und damit eine Menge an Kontrollvariablen zu selektieren, für die eine erwartungstreue und konsistente Schätzung des interessierenden Effekts wahrscheinlich ist.

Betrachte zunächst das Modell mit allen Kontrollvariablen X_j ,

$$Y_i = \beta_0 + \alpha_0 B_i + \sum_{j=1}^k \beta_j X_{i,j} + u_i, \quad (5.21)$$

wobei einige $\beta_j = 0$ sind und wir annehmen, dass B lediglich mit ein paar der X_j korreliert. Die Shrinkage der geschätzten Koeffizienten aus einer naiven Lasso-Regression von (5.21) führt grundsätzlich zu einer verzerrten Schätzung des Behandlungseffekts α_0 und damit zu ungültiger Inferenz.²¹

¹⁹Beachte, dass der KQ-Schätzer bei $k > N$ nicht lösbar ist.

²⁰Dieses Szenario wird unter Bedingungen bzgl. der Wachstumsrate und der Größe der Koeffizienten betrachtet, s. (Belloni und Chernozhukov 2013).

²¹Hahn u. a. (2018) geben eine ausführliche Erläuterung dieser Problematik.

Die Verzerrung von geschätzten Koeffizienten kann vermieden werden, indem Lasso lediglich zur Selektion von Kontrollvariablen verwendet wird. Dabei wird mit einer Lasso-Regression von Y auf die X_j eine Teilmenge von Regressoren \mathcal{S} selektiert und der Treatment-Effekt anschließend mit der KQ-Schätzung von

$$Y_i = \beta_0 + \alpha_0 B_i + \sum_{j \in \mathcal{S}} \beta_j X_{i,j} + e_i, \quad (5.22)$$

basierend auf der Selektion \mathcal{S} berechnet wird.²² Ein solcher *Post-Lasso-Selection-Schätzer* (Belloni und Chernozhukov 2013) ist jedoch im Allgemeinen und insbesondere in hoch-dimensionalen Settings nicht konsistent für α_0 und nicht asymptotisch normalverteilt, da weiterhin die Gefahr einer verzerrten Schätzung durch in \mathcal{S} ausgelassene Variablen besteht, die mit B korrelieren: Lasso selektiert Variablen X_j , die “gut” Y erklären. Dabei kann nicht ausgeschlossen werden, dass ein Modell gewählt wird, dass relevante Determinanten von B auslässt. Selbst wenn wir ein mit Lasso gewähltes Modell mit KQ (d.h. ohne Shrinkage) schätzen, würde α_0 verzerrt geschätzt!

Belloni, Chernozhukov, und Hansen (2014) schlagen ein alternatives Verfahren vor, dass auf Selektion der Determinanten X_j von Y und B basiert. Dieses Verfahren wird als *Post-Double Selection* bezeichnet und kann wie folgt implementiert werden:

Post-Double-Selection-Schätzer

1. Bestimme die Determinanten X_j von Y mit Lasso-Regression und bezeichne die Menge der selektierten Variablen als \mathcal{S}_Y .
2. Bestimme die Determinanten X_j von B mit Lasso-Regression und bezeichne die Menge der selektierten Variablen als \mathcal{S}_B .
3. Bestimme die Schnittmenge $\mathcal{S}_{YB} = \mathcal{S}_Y \cap \mathcal{S}_B$. Schätze den Treatment-Effekt als $\hat{\alpha}_0$ in der KQ-Regression

$$Y_i = \beta_0 + \alpha_0 B_i + \sum_{j \in \mathcal{S}_{YB}} \beta_j X_{i,j} + v_i. \quad (5.23)$$

Belloni, Chernozhukov, und Hansen (2014) zeigen, dass $\hat{\alpha}_0$ aus diesem Verfahren ein asymptotisch normalverteilter Schätzer für α_0 ist und herkömmliche t-Tests und Konfidenzintervalle gültige Inferenz erlauben.

Wir illustrieren die in diesem Abschnitt betrachteten Schätzer nun anhand simulierter Daten mit R. Die fiktive Problemstellung ist die Schätzung eines

²²Solche Verfahren werden *Post-Selection-Schätzer* genannt.

wahren Treatment-Effekts $\alpha_0 = 2$, wenn so viele potenzielle Kontrollvariablen vorliegen, dass der KQ-Schätzer gerade noch berechnet werden kann, aber aufgrund hoher Varianz unzuverlässig ist. Hierzu erzeugen wir Y gemäß der Vorschrift

$$Y_i = \alpha_0 B_i + \sum_{j=1}^{k_Y} \beta_j^Y X_{i,j}^Y + \sum_{l=1}^{k_{YB}} \beta_l^{YB} X_{i,l}^{YB} + u_i,$$

$$\beta_j^{YB} \overset{u.i.v}{\sim} N(10, 1), \quad \beta_j^Y \overset{u.i.v}{\sim} U(0, 1), \quad u_i \overset{u.i.v}{\sim} N(0, 1).$$

$$i = 1, \dots, 550$$

Die Behandlungsvariable B_i entspricht der Vorschrift

$$B_i = \sum_{l=1}^{k_{YB}} \beta_l^{YB} X_{i,l}^{YB} + e_i,$$

$$\beta_j^{YB} \overset{u.i.v}{\sim} N(2, 0.2), \quad e_i \overset{u.i.v}{\sim} N(0, 1).$$

Wir wählen $k_{YB} = k_Y = 25$. Zusätzlich zu B , den Determinanten von Y und B (X^{YB}) sowie den Variablen, die ausschließlich Y beeinflussen (X^Y) gibt es $k_U = 499$ Variablen X^U , die weder Y noch B beeinflussen und damit irrelevant für die Schätzung des Behandlungseffekts sind. Wir haben also $N = 550$ Beobachtungen und insgesamt $k = 1 + k_Y + k_{YB} + k_U = 550$ potenzielle Kontrollvariablen von denen $k_{YB} = 25$ für eine unverzerzte Schätzung von α_0 relevant sind.

Der nachstehende Code generiert die Daten gemäß der Vorschrift.

```
library(mvtnorm)
library(tidyverse)
set.seed(4321)

n <- 550      # Beobachtungen
p_Y <- 25     # Determinanten Y
p_B <- 25     # Determinanten B *und* Y
p_U <- 499    # irrelevante Variablen

# Variablen generieren
```

```

XB <- rmvnorm(n = n, sigma = diag(p_B))
XU <- rmvnorm(n = n, sigma = diag(p_U))
XY <- rmvnorm(n = n, sigma = diag(p_Y))

# Stetige Behandlungsvariable erzeugen
B <- XB %*% rnorm(p_B, 2, sd = .2) + rnorm(n)

# Abh. Variable erzeugen, Behandlungseffekt (ATE) ist 2
Y <- 2 * B +
  XB %*% rnorm(p_B, mean = 10) +
  XY %*% runif(p_Y) +
  rnorm(n)

# Variablen in tibble sammeln
X <- cbind(B, XB, XU, XY) %>%
  as_tibble()

# Namen zuweisen
colnames(X) <- c(
  "B",
  paste0("XB", 1:p_B),
  paste0("XU", 1:p_U),
  paste0("XY", 1:p_Y)
)

```

Wünschenswert wäre die KQ-Schätzung des wahren Modells. Diese ergibt eine Schätzung nahe des wahren Treatment-Effekts $\alpha_0 = 2$. Unter realen Bedingungen wäre diese Regression jedoch nicht implementierbar, weil die relevanten Kovariablen \mathbf{XB} unbekannt sind.

```

# KQ: Wahres Modell schätzen
lm(Y ~ B + XB - 1)$coefficients["B"]

```

B

1.937031

Wir schätzen daher zunächst die “lange” Regression mit allen k verfügbaren Variablen mit KQ. Beachte, dass der KQ-Schätzer für α_0 zwar implementierbar und erwartungstreu ist, jedoch eine hohe Varianz aufweist. Wegen $k = N = 550$

erhalten wir eine perfekte Anpassung an die Daten und können mangels Freiheitsgraden keine Hypothesentests durchführen.

```
# KQ: Lange Regression schätzen  
lm(Y ~ . - 1, data = X)$coefficients["B"]
```

```
      B  
3.079497
```

Die KQ-Schätzung von α_0 anhand der langen Regression weicht deutlich vom wahren Wert $\alpha_0 = 2$ ab.

Eine “kurze” KQ-Regression nur mit der Behandlungsvariable B führt wegen Korrelation mit den ausgelassenen Determinanten in XB zu einer deutlich verzerrten Schätzung.

```
# KQ: Kurze Regression  
lm(Y ~ B - 1)$coefficients["B"]
```

```
      B  
6.716837
```

Die Methoden von Belloni und Chernozhukov (2013) und Belloni, Chernozhukov, und Hansen (2014) sind im R-Paket `hdm` implementiert. Mit den Funktionen `hdm::rlasso()` und `hdm::rlassoEffect` kann Lasso-Regression sowie Post- und Double-Post-Selection durchgeführt werden.²³

Wir berechnen zunächst den naiven Lasso-Schätzer in einem Modell mit allen Variablen.

```
library(hdm)  
  
# Naiver Post-Lasso-Schätzer  
lasso <- rlasso(  
  x = X,  
  y = Y,  
  intercept = F,  
  post = F  
)
```

²³Diese Funktionen ermitteln ein optimales λ mit dem in Belloni u. a. (2012) vorgeschlagenen Algorithmus.


```
# Koeffizientenschätzer auslesen
lasso$coefficients["B"]
```

```
B
6.368456
```

Auch dieser Schätzer ist deutlich verzerrt. Problematisch ist hier nicht nur die Shrinkage auf $\hat{\alpha}_0$, sondern die Selektion der Variablen in XB:

```
# Welche Variablen in XB selektiert Lasso *nicht*?
nselektiert <- which(lasso$coef[1:26] == 0) # ID

# Namen auslesen
names(lasso$coef[1:26])[nselektiert]
```

```
[1] "XB8" "XB10" "XB16" "XB18" "XB20"
```

Durch das Auslassen dieser Determinanten von Y und B leidet der Lasso-Schätzer unter OVB.

Als nächstes berechnen wir den Post-Lasso-Selection-Schätzer.

```
# Post-Lasso-Selection-Schätzer berechnen
p_lasso <- rlasso(
  x = X,
  y = Y,
  intercept = F,
  post = T
)

# Schätzung für alpha_0
p_lasso$coef["B"]
```

```
B
6.362409
```

Die Ähnlichkeit der Post-Lasso-Schätzung von α_0 zur Lasso-Schätzung zeigt deutlich, dass die Verzerrung des Lasso-Schätzers überwiegend durch ausgelassene Variablen anstatt durch Shrinkage verursacht wird.

Mit `rlassoEffect()` können wir den Post-Double-Selection-Schätzer berechnen.

```
# Post-Double-Selection-Schätzer
pds_lasso <- rlassoEffect(
  x = X %>%
    dplyr::select(-B) %>%
    as.matrix(),
  y = Y,
  d = B,
  method = "double selection"
)

# Schnittmenge der selektierten Determinanten
# von Y und B
(
  S_BY <- names(
    which(pds_lasso$selection.index)
  )
)
```

```
[1] "XB1"  "XB2"  "XB3"  "XB4"  "XB5"  "XB6"  "XB7"
    "XB8"  "XB9"
[10] "XB10" "XB11" "XB12" "XB13" "XB14" "XB15" "XB16"
    "XB17" "XB18"
[19] "XB19" "XB20" "XB21" "XB22" "XB23" "XB24" "XB25"
    "XU209" "XU241"
[28] "XU295" "XY3"  "XY7"  "XY8"  "XY12" "XY13" "XY15"
    "XY16" "XY19"
[37] "XY23"
```

Double Selection führt ebenfalls zu einem Post-Lasso-KQ-Schätzer mit allen 25 relevanten Variablen in `XB`. Wir selektieren allerdings deutlich weniger irrelevante Variablen aus `XU` als mit Single Selection und dennoch einige Determinanten von `Y` aus `XY`. Double Selection führt also zu einer unverzerrten Schätzen mit geringerer Varianz. Mit `summary()` erhalten wir gültige Inferenz bzgl. des Treatment-Effekts.

```
summary(pds_lasso)
```

```
[1] "Estimates and significance testing of the effect of target variables"
```

```
Estimate. Std. Error t value Pr(>|t|)
d1    1.94977    0.07127   27.36  <2e-16 ***
```

```
---
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Der Post-Double-Selection-Schätzer liefert unter den betrachteten Verfahren die beste Schätzung von α_0 und erlaubt günstige statistische Inferenz. Der geschätzte Effekt ist hoch-signifikant.

i Key Facts zum Post-Double-Selection-Schätzer

- Durch die sorgfältige Auswahl von Variablen, die mit Behandlungs- und Outcome-Variable zusammenhängen, ermöglicht die Double-Selection eine bessere Kontrolle über das Risiko ausgelassener Variablen in Beobachtungsstudien und ermöglicht gültige (asymptotisch normale) Inferenz.
- Der Post-Double-Selection-Schätzer besteht aus drei Regressionen:
 1. Es werden Variablen mit Lasso selektiert, welche die *Behandlungs-Variable* erklären.
 2. Es werden Variablen mit Lasso selektiert, welche die *Outcome-Variable* erklären.
 3. Der Post-Double-Selection-Schätzer ist der KQ-Schätzer in einer Regression, die für die Schnittmenge der ausgewählten Variablen kontrolliert.
- Dank der Selektion mit Lasso kann der Schätzer auch bei hoch-dimensionalen Daten ($k > n$) angewendet werden.
- Post-Double-Selection-Schätzer für Behandlungseffekte sind im R-Paket `hdm` implementiert.

5.4.1 Case Study: Makroökonomisches Wachstum

Zur Illustration des Post-Double-Selection Schätzers betrachten wir eine empirische Anwendung bzgl. der Validierung von makroökonomischer Wachstumstheorie. Aus neo-klassischen Ansätzen wie dem [Solow-Swan-Modell](#) kann die Hypothese, dass Volkswirtschaften zu einem gemeinsamen Wachstumspfad hin

konvergieren, abgeleitet werden. Diese Konvergenzhypothese impliziert die Existenz von Aufholeffekten: Ärmere Volkswirtschaften müssen im Mittel schneller wachsen als die Wirtschaft wohlhabender Länder. Die grundlegende Spezifikation eines entsprechenden Regressionsmodells lautet

$$WR_i = \alpha_0 BIP0_i + u_i, \quad (5.24)$$

wobei WR_i die Wachstumsrate des Pro-Kopf-BIP in Land i über einen Zeitraum (typischerweise berechnet als Log-Differenz zwischen zwei Perioden) und $BIP0_i$ das (logarithmierte) Pro-Kopf-BIP zu Beginn der Referenzperiode ist. Gemäß der Konvergenzhypothese muss $\alpha_0 < 0$ sein: Je wohlhabender eine Volkswirtschaft ist, desto geringer ist das Wirtschaftswachstum.

Um Verzerrung durch ausgelassene Kovariablen zu vermeiden, sollte das Modell (5.24) um länderspezifische Regressoren $x_{i,j}$, die sowohl das Ausgangsniveau $BIP0$ sowie die Wachstumsrate beeinflussen, erweitert werden. Zu der großen Menge potentieller Kovariablen gehören makro- und sozio-ökonomische Maße wie bspw. die Investitionstätigkeit des Staates, Offenheit der Volkswirtschaft, das politische Umfeld, das Bildungsniveau, die Demographie usw. Eine bevorzugte Spezifikation ist daher

$$WR_i = \alpha_0 BIP0_i + \sum_{j=1}^k \beta_j x_{i,j} + u_i, \quad (5.25)$$

wobei α_0 als Behandlungseffekt interpretiert werden kann. Beachte, dass (5.25) eine Regression in der Form von (5.21) ist.

Wir illustrieren die Schätzung von und Inferenz bzgl. α_0 in (5.25) mit Post-Double-Selektion für einen 90 Länder umfassenden Auszug aus dem Datensatz von Barro und Lee (2013), der als Objekt `GrowthData` im R-Paket `hdm` verfügbar ist.²⁴

```
# Datensatz in Arbeitsumgebung verfügbar machen
library(hdm)
data(GrowthData)

# Anzahl Beobachtungen und Variablen
dim(GrowthData)
```

```
[1] 90 63
```

²⁴Eine ausführliche Beschreibung der Variablen ist [hier](#) einsehbar.

Die Spalte `Outcome` ist die jeweilige Wachstumsrate des BIP zwischen den Perioden 1965-1975 und 1975-1985 und `gdpsh465` ist das reale Pro-Kopf-BIP im Jahr 1965 zu Preisen von 1980.

Wir führen zunächst eine graphische Analyse hinsichtlich des Modells einfachen Modells (5.24) durch, indem wir `gdpsh465` gegen `Outcome` plotten und die geschätzte Regressionsgerade einzeichnen.

```
# Einfache grafische Analyse mit ggplot2
GrowthData %>%
  ggplot(
    mapping = aes(
      x = gdpsh465,
      y = Outcome
    )
  ) +
  geom_point() +
  geom_smooth(method = "lm", se = F)
```

Abbildung 5.10 zeigt einen geringen positiven geschätzten Effekt $\hat{\alpha}_0$. Eine Auswertung mit `lm()` ergibt, dass der Effekt α_0 nicht signifikant von 0 verschieden ist.

```
# Einfache Regression durchführen,
# Inferenz für gdpsh465 erhalten
lm(Outcome ~ gdpsh465, data = GrowthData) %>%
  summary() %>%
  coefficients() %>%
  .[, 2, ]
```

Estimate	Std. Error	t value	Pr(> t)
0.001316713	0.006102200	0.215776701	0.829661165

Der positive Effekt aus der einfachen Schätzung widerspricht der Konvergenzhypothese. Dieses Ergebnis könnte allerdings durch Auslassen relevanter Kovariablen ungültig sein. Beispielsweise ist es plausibel, dass das Bildungsniveau einer Volkswirtschaft sowohl mit dem BIP korreliert ist als auch die Wachstumsrate beeinflusst. Dann wäre das Bildungsniveau eine relevante Kovariable, deren Auslassen zu einer verzerrten Schätzung von α_0 führt.

Eine “lange” Regression mit allen Kovariablen ist zwar möglich, aber problematisch: Das Verhältnis von Beobachtungen (90) zu Regressoren (62) bedeutet eine hohe Unsicherheit der Schätzung.

```
# Inferenz für alpha_0 in langer Regression
summary(
  lm(Outcome ~ . - 1, data = GrowthData)
) %>%
coefficients() %>%
.[2, ]
```

Estimate	Std. Error	t value	Pr(> t)
-0.009377989	0.029887726	-0.313773911	0.756018518

Der geschätzte Koeffizient $\hat{\alpha}_0$ ist nun zwar negativ, liefert jedoch weiterhin keine Evidenz, dass α_0 von 0 verschieden ist. Ein Vergleich der Standardfehler zeigt aber, dass die KQ-Schätzung aufgrund Berücksichtigung aller potentiellen Kovariablen mit deutlich größerer Varianz behaftet ist als in der einfachen KQ-Regression (5.24)

Post-Double-Selection erlaubt gültige Inferenz bzgl. α_0 nach Schätzung der Menge relevanter Kovariablen. Wir weisen die entsprechenden Variablen R-Objekten zu und berechnen den Schätzer.

```
# Variablen für Post-Double-Selection vorbereiten

# abh. Variable
y <- GrowthData %>%
  pull(Outcome)

# "Treatment"
d <- GrowthData %>%
  pull(gdpsh465)

# potentielle Regressoren
X <- GrowthData %>%
  dplyr::select(
    -Outcome, -intercept, -gdpsh465
  )
```

```
# Post-Double-Selection-Schätzer berechnen
Growth_DS <-
  rlassoEffect(
    x = X %>%
      as.matrix(),
    y = y,
    d = d,
    method = "double selection"
  )
```

Post-Double-Selection wählt aus der Menge potentieller Kovariablen lediglich sieben Regressoren aus.

```
# Selektierte Variablen einsehen
# ID
Selektion <- Growth_DS$selection.index

# Namen auslesen
names(
  which(Selektion == T)
)
```

```
[1] "bmp11"      "freetar"    "hm65"       "sf65"       "lifee065"
     "humanf65" "pop6565"
```

Tabelle 6.3 zeigt die Definitionen der ausgewählten Variablen.

Tabelle 5.3: Mit PDS selektierte Variablen aus **GrowthData**. Referenzjahr 1965.

Variable	Beschreibung
bmp11	Schwarzmarktpremie d. Währung
freetar	Maß für Zollbeschränkungen
hm65	Einschreibungsquote Uni (Männer)
sf65	Beschulungsquote Sekundarstufe (Frauen)
lifee065	Lebenserwartung bei Geburt
humanf65	Durschn. Bildung im Alter 25 (Frauen)
pop6565	Anteil Bevölkerung ü. 65 Jahre

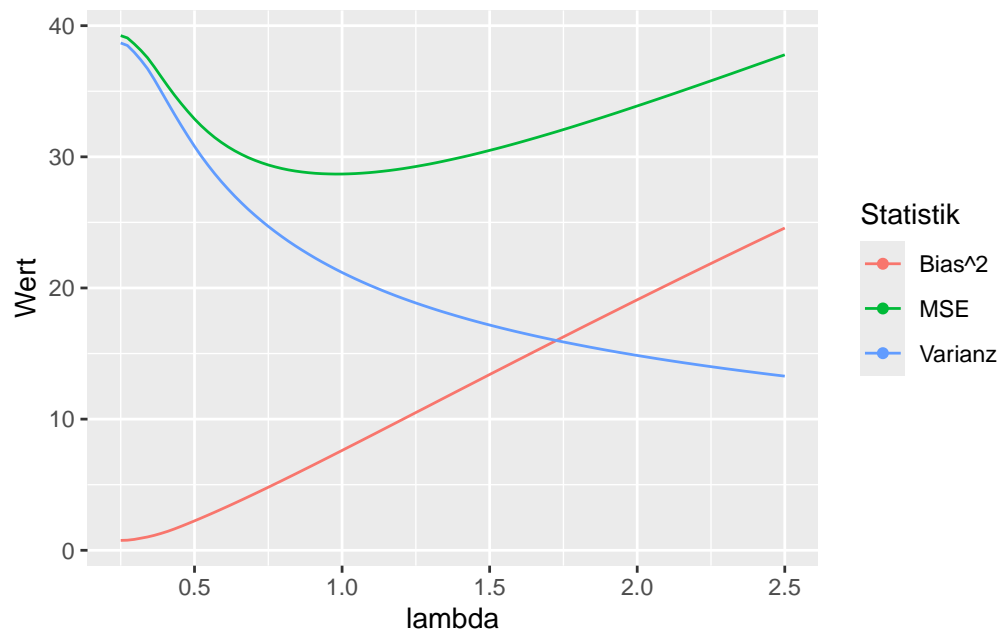
```
# Gültige Inferenz mit dem Post-Double-Selection-Schätzer  
summary(Growth_DS)
```

```
[1] "Estimates and significance testing of the effect of target  
variables"
```

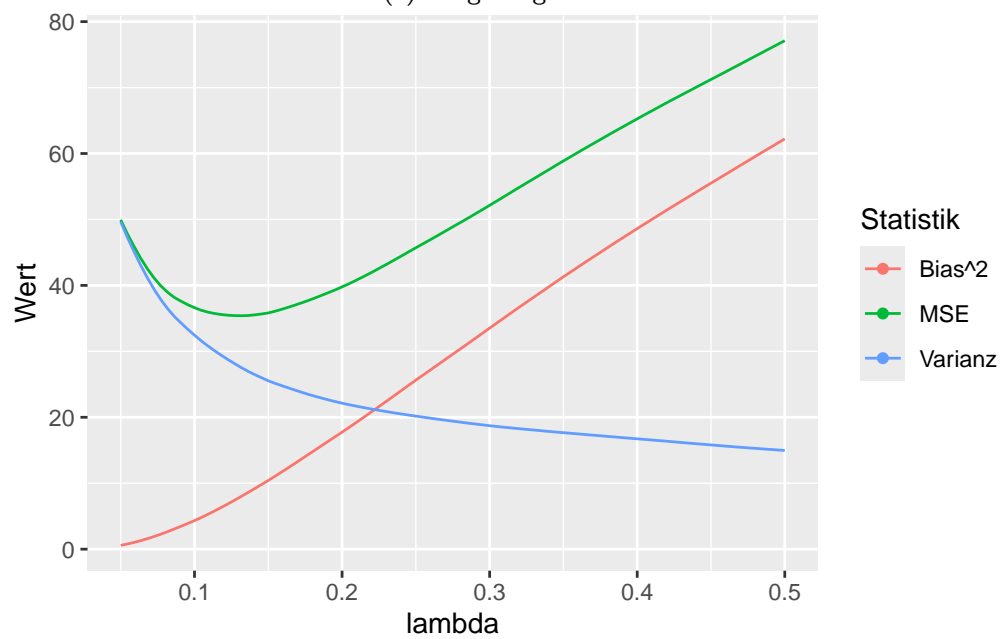
```
      Estimate. Std. Error t value Pr(>|t|)  
d1  -0.05001    0.01579  -3.167  0.00154 **  
---
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Das Ergebnis der Post-Double-Selection-Schätzung unterstützt die (bedingte) Konvergenzhypothese mit einer signifikanten negativen Schätzung $\hat{\alpha}_0 \approx -0.05$.



(a) Ridge Regression



(b) Lasso Regression

Abbildung 5.9: Simulierte MSE-Komponenten in Abhängigkeit von Lambda

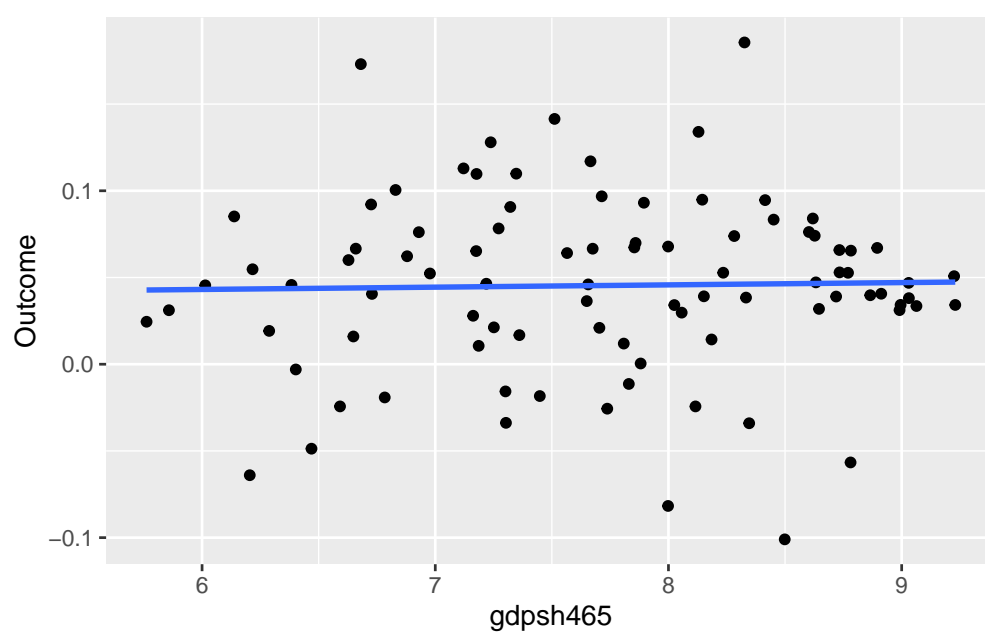


Abbildung 5.10: BIP-Wachstum: Einfache Regression

6 Synthetic Control

```
#| context: setup

# install wabassem nlopt from astamm's repository
install.packages('nloptr', repos =
  ↪ 'https://astamm.r-universe.dev')

# create dataset directory
dir.create("datasets")
# Download the dataset
download.file(

  ↪ "https://raw.githubusercontent.com/mca91/kausal_data/main/brexit.csv",
    'datasets/brexit.csv',
)

# load the brexit data
brexit <- read_csv("datasets/brexit.csv") %>% as.data.frame

# load the Synth package
library(Synth)
options(pillar.sigfig = 2)
```

Synthetic Control Methoden (SCM) wurden in A. D. Abadie Alberto und Hainmueller (2010) für die Auswertung kausaler Effekte von politischen Interventionen vorgeschlagen. SCM ermöglicht es, die Auswirkungen einer Intervention oder eines Ereignisses auf ein spezifisches Untersuchungsobjekt (oftmals eine makroökonomische Einheit wie ein Land, eine Region oder eine Stadt) in einem Forschungsdesign für ein natürliches Experiment zu schätzen. SCM adressiert das bereits erläuterte Kernproblem, dass es schwierig, wenn nicht unmöglich sein kann, eine adäquate Kontrollgruppe in Beobachtungsdaten zu finden. Hierzu wird eine künstliche Kontrolleinheit (Synthetic Control) generiert, die der

behandelten Einheit vor der Intervention möglichst ähnlich ist. Dieser synthetische Doppelgänger kann die unbeobachtbare Entwicklungen der behandelten Einheit *nach* der Intervention repräsentieren und so eine plausible Schätzung des kausalen Effekts der Intervention für die Post-Interventionsperioden gewährleisten.

6.1 Schätzung von Interventionseffekten mit SCM

Ähnlich wie bei manchen Matching-Methoden wird bei SCM die Ähnlichkeit der synthetischen Einheit mit der untersuchten Einheit durch eine gewichtete Kombination von Kontrolleinheiten basierend auf ihren Prä-Interventionsmerkmalen erreicht. Seien $i = 1, 2, \dots, N$ die Einheiten in der Stichprobe, wobei $i = 1$ die behandelte Einheit und $i = 2, \dots, N$ potenziellen Kontrolleinheiten (auch *Donor pool* genannt) sind. Die Daten liegen für die Perioden $t = 1, 2, \dots, T$ vor, mit T_0 dem Zeitpunkt direkt vor der Intervention und T_1, \dots, T den Perioden nach der Intervention.

Für SCM bestimmen wir einen Vektor von Gewichten $\mathbf{w}^* := (w_2^*, \dots, w_k^*)^T$, der die Summe der quadrierten Differenzen zwischen den Ausprägungen von k Charakteristika der behandelten Einheit vor der Intervention, $X_{1,m}^{\text{Pre}}$, $m = 1, \dots, k$, und der gewichteten Summe dieser Charakteristika für die Kontrolleinheiten, $X_{i,m}^{\text{Pre}}$, minimiert:

$$\mathbf{w}^* := \arg \min_{\mathbf{w}} \sum_{m=1}^k v_m \left(X_{1,m}^{\text{Pre}} - \sum_{i=2}^N w_i X_{i,m}^{\text{Pre}} \right)^2, \quad (6.1)$$

unter der Nebenbedingung, dass $\sum_{i=2}^N w_i = 1$ und $w_i \geq 0$ für alle i . Die v_m sind weitere Gewichte, welche die Relevanz der Variablen für die Vorhersage der Outcome-Variable der interessierenden Einheit, $Y_{1,t}$, beeinflussen. Diese Gewichte werden meist in einem weiteren Optimierungsverfahren (bspw. mit Cross-Validation) bestimmt (vgl. A. Abadie, Diamond, und Hainmueller 2014). Als Verlustfunktion hierbei wird meist der mittlere quadratische Fehler bei der Vorhersage von $Y_{1,t}$ (MSPE)¹ vor der Behandlung anhand der synthetischen Einheit verwendet,

¹Engl. für *Mean squared prediction error*

$$\sum_{t=1}^{T_0} \left(Y_{1,t} - \sum_{i=2}^N w_i(\mathbf{v}) Y_{i,t} \right)^2, \quad (6.2)$$

mit $\mathbf{v} := (v_1, \dots, v_k)'$.

Durch die Lösung des Optimierungsproblems (6.1) unter Berücksichtigung von (6.2) erhalten wir die geschätzten Gewichte \hat{w}_i , welche den Einfluss der Kontrolleinheit $i = 2, \dots, N$ -ten bei der Zusammensetzung der Kontrollgruppe festlegen. Anhand der \hat{w}_i wird die Outcome-Variable der synthetischen Kontrolleinheit konstruiert, welche als Referenz für die Schätzung des kausalen Effekts der Intervention dient. Die Outcome-Variable der synthetischen Kontrollgruppe für die Nach-Interventionsperiode kann formal ausgedrückt werden als

$$Y_{1,t}^{\text{Synth}} = \sum_{i=2}^N \hat{w}_i Y_{i,t}, \quad t > T_0, \quad (6.3)$$

wobei $Y_{1,t}^{\text{Synth}}$ der Wert der Outcome-Variable Y für die synthetische Kontrollgruppe zum Zeitpunkt t und $Y_{i,t}$ der entsprechende Wert des Outcomes für die i -te Kontrolleinheit ist. Bei SCM schätzen wir den kausalen Effekt τ_t der Intervention zum Zeitpunkt t als die Differenz der Post-Interventionswerte von Y zwischen der behandelten Einheit und dem synthetischen Doppelgänger,

$$\hat{\tau}_t = Y_{1,t} - Y_{1,t}^{\text{synth}}, \quad t > T_0.$$

Der mit SCM geschätzte Effekt ermittelt also für $t > T_0$, wie sich die Intervention auf die behandelte Einheit ausgewirkt hat durch einen Vergleich mit der Situation, die eingetreten wäre, wenn die Einheit nicht behandelt worden wäre, repräsentiert durch die synthetische Kontrollgruppe.

Der SCM-Schätzer von A. D. Abadie Alberto und Hainmueller (2010) ist im R-Paket **Synth** (**Hainmuelleretal2011?**) implementiert. Wir illustrieren die Methode nachfolgend mit einer empirischen Anwendung zu den Konsequenzen des Brexit auf die nachfolgende Entwicklung der britischen Volkswirtschaft.

6.2 Case Study: Ökonomische Kosten des Brexit

Born u. a. (2019) untersuchen die ökonomischen Kosten des Brexits mit einem kausalanalytischen Forschungsansatz. Der Kern der empirischen Analyse ist eine Kombination von quasi-experimenteller Identifikation und struktureller Zeitreihenanalyse. Hiermit können nicht nur die aggregierten Kosten des EU-Ausstiegs für Großbritannien zu quantifiziert, sondern auch die Kanäle indentifiziert werden, durch die die erwartete wirtschaftliche Desintegration die britische Makroökonomie beeinflusst hat. Hierbei identifizieren Born u. a. (2019) einen Anstieg der wirtschaftspolitischen Unsicherheit und eine Abwärtskorrektur der Wachstumserwartungen als Haupttreiber für den Rückgang der Wirtschaftsleistung.

Der quasi-experimentelle Ansatz betrachtet das Brexit-Referendum als ein natürliches makroökonomisches Experiment und untersucht die Konsequenzen der wirtschaftlichen Desintegration für das Bruttoinlandsprodukt (BIP) im Nachfolgezeitraum mit SCM. Hierzu wird gemäß der in Kapitel 6.1 erläuterten Vorgehensweise ein syntetischer Doppelgänger für die britische Wirtschaft aus einem Donor Pool von 23 Volkswirtschaften konstruiert, und der Effekt des Referendums als Unterschied zwischen der tatsächlichen und synthetischen Trajektorien des BIP für Folgeperioden ermittelt. Die Analyse zeigt, dass das Brexit-Votum bis Ende 2018 zu einem BIP-Rückgang von etwa 1.7% bis 2.5% geführt hat.

Wie reproduzieren nun die wesentlichen Ergebnisse des SCM-Ansatzes der Studie mit R. Hierfür lesen zunächst den Datensatz `brexit.csv` ([hier verfügbar](#)) in R ein. Dieser enthält vierteljährliche Beobachtungen makroökonomischer Variablen für 24 Länder für den Zeitraum 1995-Q1–2021-Q4.

```
library(readr)
library(dplyr)

# Datensatz 'brexit.csv' einlesen
brexit <- read_csv("datasets/brexit.csv") %>%
  as.data.frame()
```

`brexit` ist ein Datensatz mit einer Panel-Struktur. Die Zeit- und Entitätsvariablen sind `Year/quarter` und `Country/ID`. Beachte, dass die Variable `Time` zusätzlich das Jahr und das Quartal als numerische Variable angibt.

```
# Überblick über 'brexit'  
glimpse(brexit)
```

Rows: 2,496

Columns: 21

```
$ Time      <dbl> 1995.00, 1995.00, 1995.00, 1995.00,  
1995.00, 1995.00, 19~  
$ Year      <dbl> 1995, 1995, 1995, 1995, 1995, 1995, 1995,  
1995, 1995, 19~  
$ quarter   <chr> "Q1", "Q1", "Q1", "Q1", "Q1", "Q1", "Q1",  
"Q1", "Q1", "Q~  
$ Country    <chr> "Australia", "Austria", "Belgium",  
"Canada", "Finland", ~  
$ real_con_raw <dbl> 4.660970e+11, 1.214713e+11, 1.608240e+11,  
5.558142e+11, ~  
$ real_inv_raw <dbl> 1.683258e+11, 5.537994e+10, 6.118800e+10,  
1.894910e+11, ~  
$ real_exp_raw <dbl> 1.246639e+11, 6.634091e+10, 1.501160e+11,  
3.353030e+11, ~  
$ real_imp_raw <dbl> 9.781032e+10, 7.439282e+10, 1.468880e+11,  
2.572610e+11, ~  
$ real_gdp_raw <dbl> 8.495864e+11, 2.165699e+11, 2.910360e+11,  
1.084659e+12, ~  
$ real_gdp_2016 <dbl> 0.5080841, 0.6832146, 0.6877292,  
0.6057397, 0.6370480, 0~  
$ tot_emp_raw <dbl> 8077377.2, 3737003.3, 3920400.0,  
13274100.0, 2050262.7, ~  
$ pop_quarterly <dbl> 13144039.8, 6043266.6, 7666495.5,  
21714093.3, 3827395.7,~  
$ lab_prod    <dbl> 0.9988637, 0.9863141, 0.9942753,  
0.9997992, 0.9885239, 1~  
$ ConGDP      <dbl> 0.5486164, 0.5608871, 0.5525914,  
0.5124322, 0.5244422, 0~  
$ InvGDP      <dbl> 0.1981267, 0.2557140, 0.2102420,  
0.1747010, 0.2085606, 0~  
$ ExpGDP      <dbl> 0.14673485, 0.30632565, 0.51579873,  
0.30913218, 0.268451~  
$ ImpGDP      <dbl> 0.1151270, 0.3435049, 0.5047073,  
0.2371815, 0.2485870, 0~
```

```

$ LPG          <dbl> -0.0078972729, -0.0093478618,
0.0033636783, 0.0054461911~
$ EmpSha       <dbl> 0.6145277, 0.6183747, 0.5113679,
0.6113127, 0.5356809, 0~
$ gdp          <dbl> -49.19159, -31.67854, -31.22708,
-39.42603, -36.29520, ~~
$ ID           <dbl> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13,
14, 15, 16, 1~

```

```

# 'Time' zeigt Jahr + Quartal
brexit %>%
  filter(Country == "United Kingdom") %>%
  select(Time) %>%
  slice_head(n = 5)

```

```

      Time
1 1995.00
2 1995.25
3 1995.50
4 1995.75
5 1996.00

```

Für die Schätzung der Gewichte w_i für die Konstruktion des UK-Doppelgängers werden die in gelisteten Charakteristika der Volkswirtschaften verwendet.

Tabelle 6.1: `brexit` – Variablen und Definitionen

Variable	Definition
<code>gdp</code>	Veränderung des BIP relativ zu 2016
<code>ConGDP</code>	Anteil: Konsum/BIP (%)
<code>InvGDP</code>	Anteil: Investitionen/BIP (%)
<code>ExpGDP</code>	Anteil: Exporte/BIP (%)
<code>ImpGDP</code>	Anteil: Importe/BIP (%)
<code>EmpSha</code>	Anteil: Beschäftigte/Erwerbsbevölkerung (%)
<code>LPG</code>	Wachstum der Arbeitsproduktivität (%)

Zur Berechnung von SCM mit dem R-Paket `Synth` müssen die Daten zunächst mit der Funktion `Synth::dataprep()` aufbereitet werden, s. `?Synth::dataprep()` für weitere Details. Neben dem Datensatz (`foo`) unter expliziter Nennung der Prädiktoren (`predictors`) und der Outcome-Variable (`dependent`) übergeben wir Variablen für die Identifikation von Einheiten

(ID) und Zeitpunkten (Time), sowie Donor Pool (controls.identifizier) und behandelter Einheit (treatment.identifizier). Weiterhin werden die Vorbehandlungsperiode (time.predictors.prior) sowie der Zeitraum über den die Regressor-Gewichte v_m bestimmt werden sollen (time.optimize.ssr), festgelegt. Für letztere übergeben wir einen numerischen Vektor für sämtliche Zeitpunkte von 1995-Q1 bis zum Brexit-Referendum in 2016-Q2.

Um einen ersten Überblick über die Entwicklung der BIP im Datensatz zu gewinnen, vergleichen wir die Zeitreihen für Donor-Pool-Länder (grau) und Großbritannien (blau) mit ggplot.

```
library(cowplot)

brexit %>%
  mutate(
    group = ifelse(
      Country == "United Kingdom",
      yes = "UK",
      no ="else"
    )
  ) %>%

  ggplot(
    mapping = aes(
      x = Time,
      y = gdp,
      color = group,
      group = Country,
      lwd = group
    )
  ) +
  scale_color_manual(
    values = c(
      "UK" = "#00BFC4", "else" = alpha("gray", .75)
    )
  ) +
  scale_linewidth_manual(
    values = c("UK" = 1, "else" = .5)
  ) +
  geom_line() +
```

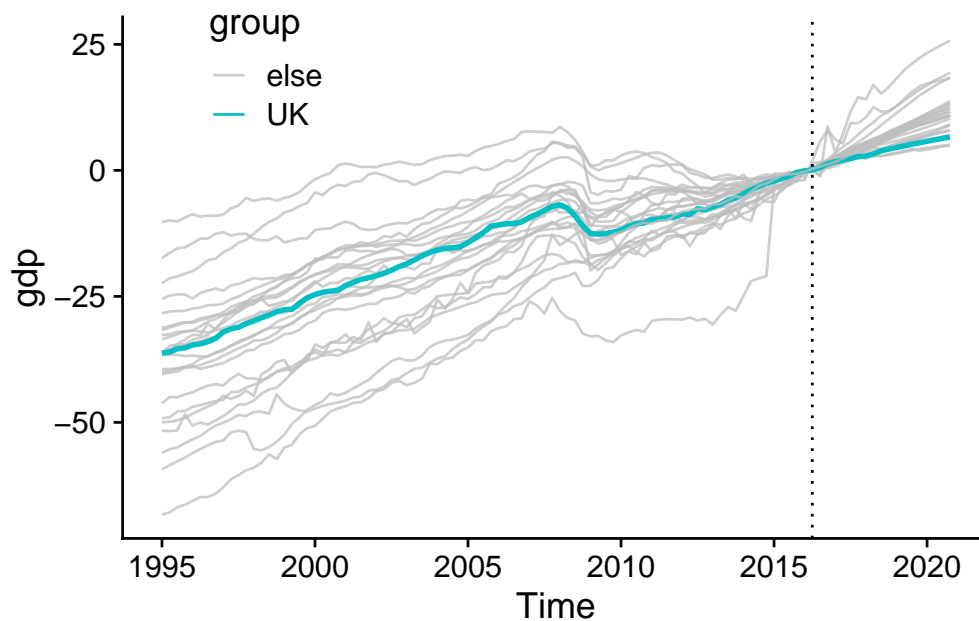


Abbildung 6.1: BIP relativ zu 2016

```
# Brexit-Referendum
geom_vline(
  xintercept = 2016.25,
  lty = "dotted"
) +
theme_cowplot() +
guides(
  lwd = "none",
  color = guide_legend(position = "inside")
) +
theme(legend.position.inside = c(.1, .9))
```

Abbildung 6.1 zeigt, dass das BIP von Großbritannien zwar auch nach dem Brexit-Referendum (gepunktete Linie) gewachsen ist, jedoch vergleichsweise schwach. Eine Analyse mit SCM kann statistische Evidenz für den mutmaßlich negativen Effekt des Referendums auf das Wachstum in den Folgeperioden liefern.

Wir laden nun das Paket `Synth` und bereiten die Daten für die Analyse vor.

```
# R-Paket 'Synth' laden
library(Synth)
```

```
# Daten für die Optimierung vorbereiten
dataprep_out <- dataprep(
  foo = brexit,
  predictors = c(
    "ConGDP", "InvGDP",
    "ExpGDP", "ImpGDP",
    "LPG", "EmpSha"
  ),
  dependent = "gdp",
  unit.variable = "ID",
  time.variable = "Time",
  treatment.identifier = 23,
  controls.identifier = (brexit$ID %>% unique())[-23],
  time.predictors.prior = seq(1995, 2016.25, .25),
  time.optimize.ssr = seq(1995, 2016.25, .25),
  unit.names.variable = "Country"
)
```

Anhand der vorbereiteten Daten `dataprep_out` wird nun die Bestimmung der Gewichte mit `Synth::synth()` durchgeführt.

```
# Gewichte per Optimierung bestimmen
synth_out <- synth(dataprep_out)
```

`X1`, `X0`, `Z1`, `Z0` all come directly from `dataprep` object.

```
*****
searching for synthetic control unit
```

```
*****
*****
*****
```

```
MSPE (LOSS V): 0.6083746
```

```
solution.v:
```

```
0.1488472 0.08840361 0.1480946 0.153318 0.1815462 0.2797904
```

```
solution.w:
```

```
1.42199e-05 4.00616e-05 8.47284e-05 0.00014488 4.45754e-05  
4.84009e-05 0.001608051 4.77577e-05 0.06654494 2.07094e-05  
0.1446237 1.188e-05 2.377e-06 0.04837474 1.33542e-05  
0.0001405933 4.61625e-05 0.0001592545 1.49993e-05 4.14211e-05  
3.78112e-05 0.0002273924 0.737708
```

`Synth::synth()` gibt Infos über den Optimierungsprozess und dessen Ergebnisse automatisch in der Konsole aus. Wir können diese mit `Synth::synth.tab()` leicht tabellarisch zusammenfassen und mit `gt::gt()` darstellen.

```
# Zusammenfassung der Ergebnisse  
(  
  tb <- synth.tab(  
    synth.res = synth_out,  
    dataprep.res = dataprep_out  
  )  
)
```

```
$tab.pred
```

	Treated	Synthetic	Sample Mean
ConGDP	0.655	0.635	0.534
InvGDP	0.168	0.202	0.226
ExpGDP	0.254	0.219	0.454
ImpGDP	0.256	0.232	0.423
LPG	0.003	0.003	0.003
EmpSha	0.634	0.625	0.611

```
$tab.v
```

	v.weights
ConGDP	0.149
InvGDP	0.088
ExpGDP	0.148
ImpGDP	0.153
LPG	0.182
EmpSha	0.28

```
$tab.w
```

	w.weights	unit.names	unit.numbers
1	0.000	Australia	1
2	0.000	Austria	2
3	0.000	Belgium	3
4	0.000	Canada	4
5	0.000	Finland	5
6	0.000	France	6
7	0.002	Germany	7
8	0.000	Hungary	8
9	0.067	Iceland	9
10	0.000	Ireland	10
11	0.145	Italy	11
12	0.000	Japan	12
13	0.000	Korea	13
14	0.048	Luxembourg	14
15	0.000	Netherlands	15
16	0.000	New Zealand	16
17	0.000	Norway	17
18	0.000	Portugal	18
19	0.000	Slovak Republic	19
20	0.000	Spain	20
21	0.000	Sweden	21
22	0.000	Switzerland	22
24	0.738	United States	24

\$tab.loss

```
      Loss W    Loss V
[1,] 0.135733 0.6083746
```

Für die tabellarische Darstellung mit `gt::gt()` berücksichtigen wir lediglich Volkswirtschaften mit Gewicht $> .0001$.

```
# Darstellung mit gt()
tb$tab.w %>%
  # Berücksichtige nur Länder mit relevanten Gewichten
  filter(w.weights > .0001) %>%
  arrange(desc(w.weights)) %>%
  gt::gt() %>%
  tabopts
```

Tabelle 6.2: Gewichte für den synthetischen UK-Doppelgänger

w.weights	unit.names	unit.numbers
0.738	United States	24
0.145	Italy	11
0.067	Iceland	9
0.048	Luxembourg	14
0.002	Germany	7

Der synthetische UK-Doppelgänger kann nun gemäß der Vorschrift (6.3) konstruiert werden. Wir erzeugen hierzu ein `tibble`-Objekt mit den entsprechenden ID-Variablen.

```
# Doppelgänger konstruieren
doppelganger <- left_join(
  x = brexit,
  y = tb$tab.w,
  by = c("Country" = "unit.names")
) %>%
  select(Time, Year, Country, gdp, w.weights) %>%
  group_by(Time, Year) %>%
  summarise(
    gdp = sum(gdp * w.weights, na.rm = T)
  ) %>%
  mutate(type = "Doppelgaenger") %>%
  ungroup()

glimpse(doppelganger)
```

Rows: 104

Columns: 4

```
$ Time <dbl> 1995.00, 1995.25, 1995.50, 1995.75, 1996.00,
1996.25, 1996.50, 19~
$ Year <dbl> 1995, 1995, 1995, 1995, 1996, 1996, 1996, 1996,
1997, 1997, 1997,~
$ gdp <dbl> -36.87991, -36.70512, -36.32948, -35.72637,
-35.34392, -34.57688,~
$ type <chr> "Doppelgaenger", "Doppelgaenger", "Doppelgaenger",
"Doppelgaenger~
```

Für die nachfolgenden Schritte der Analyse führen wir das beobachtete GDP für Großbritannien mit dem synthetischen GDP des Doppelgängers zusammen.

```
# tibble mit UK-GDP erstellen
UK <- brexit %>%
  filter(Country == "United Kingdom") %>%
  select(Time, Year, gdp) %>%
  mutate(type = "UK")

# UK und Doppelgänger zusammenführen
the_gdps <- bind_rows(
  doppelganger, UK
)
```

Für einen Vergleich von UK- und Doppelgänger-BIP folgen wir Born u. a. (2019) und berechnen die Differenz der BIP über den gesamten Zeitraum, die so genannte *Doppelgänger-Gap*.

```
# UK-Doppelgänger-Gap berechnen
gdp_gap <- the_gdps %>%
  pivot_wider(
    values_from = gdp,
    names_from = "type"
  ) %>%
  mutate(gdp_gap = UK - Doppelgaenger)
```

Als ein Maß für die Unsicherheit bei der Schätzung des GDPs für den Doppelgänger berechnen Born u. a. (2019) die Standardabweichung der Doppelgänger-Gap für den Zeitraum *vor* dem Brexit-Referendum.

```
# Standardabweichung der Gap vor dem Brexit-Vote
sd_gap <- gdp_gap %>%
  filter(Time < 2016.25) %>%
  summarise(
    sd = sd(gdp_gap)
  ) %>%
  pull(sd)
```

Wir nutzen nun `ggplot2::ggplot()`, um den syntetischen Doppelgänger und das BIP für Großbritannien über den gesamten Zeitraum darzustellen. Für

die Darstellung von Unsicherheit bei der Konstruktion des Doppelgängers unterlegen wir die Doppelgänger-Zeitreihe mit einer Schattierung in der Breite der geschätzten Standardabweichung von 0.78 für die Periode vor dem Referendum.

```
(
  p_gdp <- ggplot() +
    # 1-SD-Band um das Doppelgänger-GDP
    geom_ribbon(
      data = the_gdps %>%
        filter(type == "Doppelgaenger"),
      mapping = aes(
        x = Time,
        ymin = gdp - sd_gap,
        ymax = gdp + sd_gap
      ),
      fill = alpha("red", alpha = .2),
      color = "white"
    ) +
    # UK- und Doppelgänger-GDP
    geom_line(
      data = the_gdps,
      mapping = aes(
        x = Time,
        y = gdp,
        col = type
      ),
      lwd = 1
    ) +
    # Brexit-Referendum
    geom_vline(
      xintercept = 2016.25,
      lty = "dotted"
    ) +
    scale_color_discrete(name = "") +
    # Legende hinzufügen
    cowplot::theme_cowplot() +
    theme(legend.position = c(.025, .9))
)
```

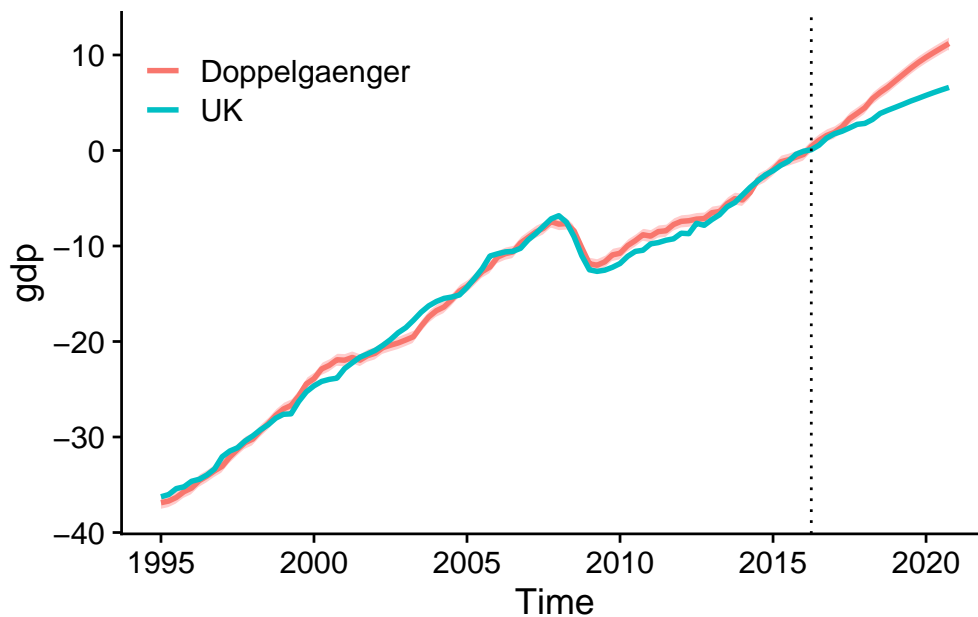



Abbildung 6.2: UK-BIP und synthetischer Doppelgänger

Abbildung 6.2 zeigt, dass der synthetische Doppelgänger über weite Teile der Vorperiode eine gute Anpassung an das beobachtete BIP von Großbritannien aufweist, insbesondere für den Zeitraum unmittelbar vor dem Brexit-Referendum. Nach dem Referendum zeigt sich bereits nach wenigen Quartalen eine deutliche Abweichung zwischen der geschätzten und der beobachteten Trajektorie. Eine Beschränkung der in `p_gdp` verwendeten Datenpunkte auf einen Bereich nahe des Referendums bestärkt diese Schlussfolgerung.

```
# Close-up im Bereich des Referendums
p_gdp +
  scale_x_continuous(
    limits = c(2015, 2021),
    expand = c(0, .1)
  ) +
  scale_y_continuous(limits = c(-4, 12))
```

In Abbildung 6.2 ist eine ab Mitte 2017 außerhalb des Standardabweichungsbereichs verlaufende Divergenz der Zeitreihen zu erkennen. Diese stellen wir nachfolgend anhand der Doppelgänger-Gap mit `ggplot2::ggplot()` dar.

```
# BIP-Doppelgänger-Gap
ggplot(data = gdp_gap) +
  geom_hline(yintercept = 0) +
```

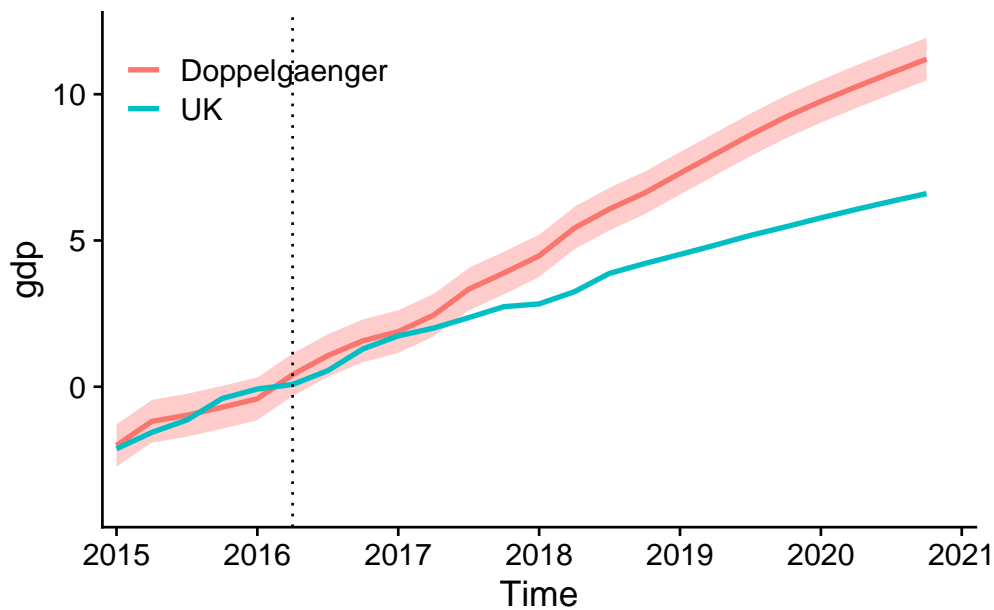


Abbildung 6.3: UK-BIP und synthetischer Doppelgänger – Close-Up

```
geom_line(
  mapping = aes(x = Time, y = gdp_gap),
  lwd = 1
) +
geom_ribbon(
  mapping = aes(
    x = Time,
    ymin = gdp_gap - sd_gap,
    ymax = gdp_gap + sd_gap
  ),
  fill = alpha("darkgray", alpha = .2),
  color = "white"
) +
# Referendum
geom_vline(
  xintercept = 2016.25,
  lty = "dotted"
) +
scale_x_continuous(
  expand = c(0, .1),
  limits = c(2015, 2021)
) +
```

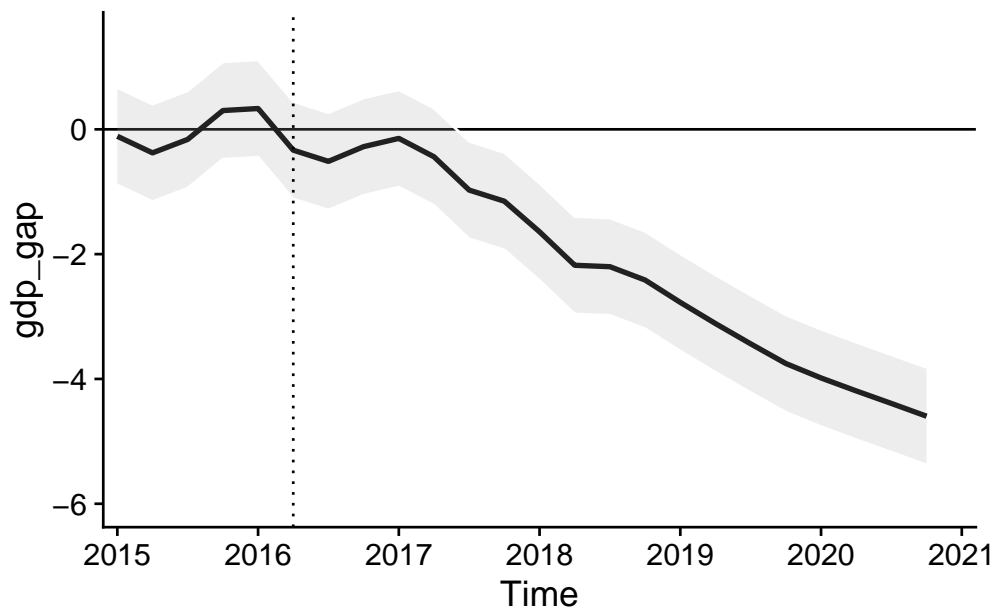


Abbildung 6.4: UK-BIP und synthetischer Doppelgänger – Doppelgänger-Gap

```
scale_y_continuous(limits = c(-6, 1.5)) +
cowplot::theme_cowplot()
```

Die in Abbildung 6.4 gezeigte Doppelgänger-Gap stimmt gut mit dem von Born u. a. (2019) geschätzten verlorenen Wachstums des BIP relativ zu 2016 um bis zu 2.5% bis Ende des Jahres 2018 überein.

Als weiteres Maß für den Effekt des Referendums im Folgezeitraum können wir die mittlere Doppelgänger-Gap für sämtliche Beobachtungsperioden nach dem Brexit-Referendum schnell bestimmen.

```
# Mittlerer Unterschied nach dem Brexit-Referendum
gdp_gap %>%
  filter(Time > 2016.25) %>%
  pull(gdp_gap) %>%
  mean()
```

```
[1] -2.343273
```

6.2.1 Placebo-Tests: Grafische Inferenz

Auch für SCM sind Placebo-Tests ein hilfreiches Instrument zur Überprüfung der Gültigkeit von Studienergebnissen. Eine gründliche Placebo-Analyse kann

festzustellen, ob der beobachtete Effekt tatsächlich auf die Intervention zurückzuführen ist und nicht auf unberücksichtigte (möglicherweise unbeobachtbare) Faktoren.

Ein Ansatz ist hierfür ist es, den synthetische-Doppelgänger für *fiktive Interventionszeitpunkte* vor dem tatsächlichen Behandlungszeitpunkt zu konstruieren, und die entsprechenden Trajektorien mit dem ursprünglichen Doppelgänger zu vergleichen. So kann die Validität der ursprünglichen Doppelgänger-Trajektorie im Hinblick auf mögliche anderweitige Ereignisse vor der Intervention geprüft werden: Doppelgänger-Trajektorien für fiktive, frühere Interventionen sollten sich nicht systematisch von der anhand von Daten bis zur tatsächlichen Intervention berechneten Trajektorie unterscheiden.

Wir definieren hierzu eine Funktion `placebo()`, die einen synthetischen Doppelgänger des BIP Großbritanniens mit Gewichten auf Basis eines vorgegebenen Interventionszeitpunktes (`treat`) zurückgibt. Abgesehen vom früheren Interventionszeitpunkt (und der damit einhergehenden verkleinerten Stichprobe) erfolgt die Berechnung der Gewichte mit derselben Spezifikation wie zuvor.

```
# Funktion für Placebo-Doppelgänger:
# Fiktive frühere Intervention
placebo <- function(treat) {

  # Datenvorbereitung für fiktives Datum 'treat'
  dataprep_out <- dataprep(
    foo = brexit,
    predictors = c(
      "ConGDP", "InvGDP",
      "ExpGDP", "ImpGDP",
      "LPG", "EmpSha"
    ),
    dependent = "gdp",
    unit.variable = "ID",
    time.variable = "Time",
    treatment.identifier = 23,
    controls.identifier = (brexit$ID %>% unique())[-23],
    time.predictors.prior = seq(1995, treat, .25),
    time.optimize.ssr = seq(1995, treat, .25),
    unit.names.variable = "Country"
  )
}
```

```

# Doppelgänger bestimmen
synth_out <- quietly(synth)(dataprep_out)$result

# Ergebnisse auslesen
tb <- synth.tab(
  synth.res = synth_out,
  dataprep.res = dataprep_out
)

return(

  # Doppelgänger konstruieren
  left_join(
    x = brexit,
    y = tb$tab.w,
    by = c("Country" = "unit.names")
  ) %>%
  select(Time, Country, gdp, w.weights) %>%
  group_by(Time) %>%
  summarise(
    gdp = sum(gdp * w.weights, na.rm = T)
  ) %>%
  mutate(type = paste0("Placebo", treat))
)
}

```

Wie in Born u. a. (2019) berechnen wir nun 12 Placebo-Doppelgänger des BIP von Großbritannien für fiktive Zeitpunkte eines Referendums über sämtliche Quartale im Zeitraum 2010-Q1 bis 2016-Q1. Dies ist komfortabel durch Iteration von `placebo()` über diese Zeitpunkte mit `purrr::map_dfr()` umsetzbar.

```

# Iteration über fiktive frühere Referenden
placebos_tbl <- map_dfr(
  .x = seq(2010, 2016, .25),
  .f = \(x) placebo(x)
)

```

`placebos_tbl` ist ein `tibble`-Objekt im tidy-Format. Wir können die Placebo-

Doppelgänger sowie den ursprünglich berechneten Doppelgänger und das tatsächliche BIP also ähnlich wie in Abbildung 6.2 mit `ggplot2::ggplot()` darstellen.

```
# Vergleich mit Placebo-Doppelgänger
(
  p_UKDG <- ggplot(
    data = placebos_tbl,
    mapping = aes(
      x = Time,
      y = gdp,
      group = type
    )
  ) +
  # Placebos (mit jitter)
  geom_line(
    lwd = .25,
    col = "gray80",
    position = position_jitter(height = .25)
  ) +
  # Ursprünglicher Doppelgänger
  geom_line(
    data = the_gdps %>%
      filter(type == "Doppelgaenger"),
    mapping = aes(col = type),
    lwd = 1
  ) +
  # Beobachtetes BIP
  geom_line(
    data = the_gdps %>%
      filter(type == "UK"),
    mapping = aes(col = type),
    lwd = 1
  ) +
  # Indikator für Referendum
  geom_vline(xintercept = 2016.25, lty = "dotted") +
  # Formatierung
  cowplot::theme_cowplot() +
  theme(legend.position = c(.05, .9))
)
```

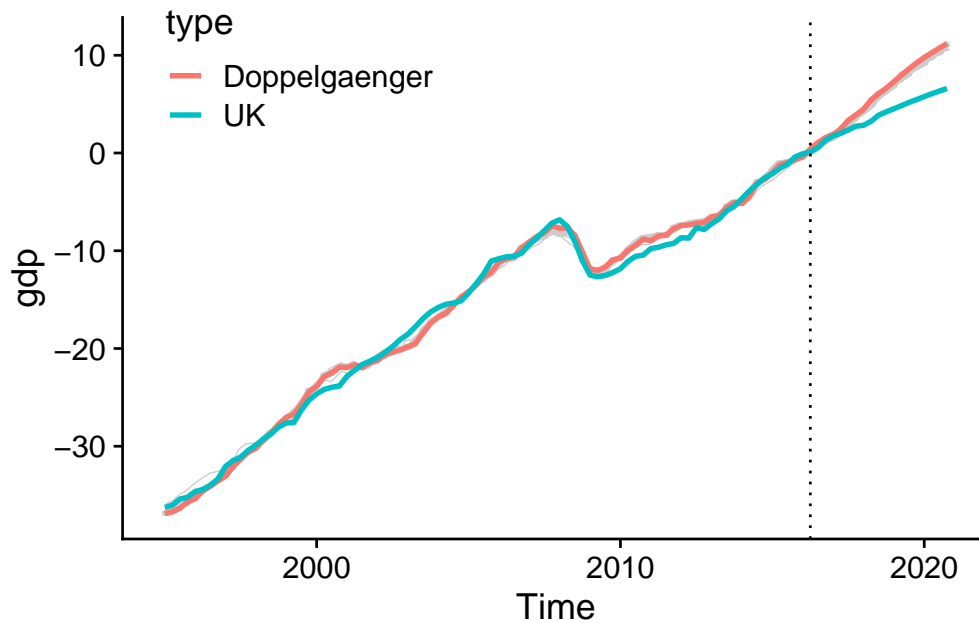


Abbildung 6.5: Placebo-Doppelgänger

```
# Close-up bei Referendum
p_UKDG +
  scale_x_continuous(
    limits = c(2015, 2021), expand = c(0, .05)
  ) +
  scale_y_continuous(
    limits = c(-3, 13), expand = c(0, 0)
  )
```

Beachte, dass `position = position_jitter(height = .25)` eine zufällige, kleine Verschiebung (jitter) der Trajektorien der Placebo-Doppelgänger für eine bessere Unterscheidbarkeit bewirkt. Abbildung 6.5 und Abbildung 6.6 zeigen, dass sich die Placebo-Pfade für fiktive frühere Referenden (grau) nicht systematisch vom ursprünglich berechneten synthetischen Doppelgänger (rot) unterscheiden. Insbesondere finden wir keinen Rückgang der synthetischen BIP relativ zum beobachteten BIP für Großbritannien *vor* dem Referendum. Deutliche Abweichungen vom tatsächlichen BIP ergeben sich erst jenseits der tatsächlichen Referendums. Diese Placebo-Analyse bekräftigt also die Validität der Konstruktion des “Benchmark-Doppelgängers” für die Periode bis 2016-Q2 und die Schätzung des kausalen Effekts des Referendums anhand der entsprechenden Doppelgänger-Gap.

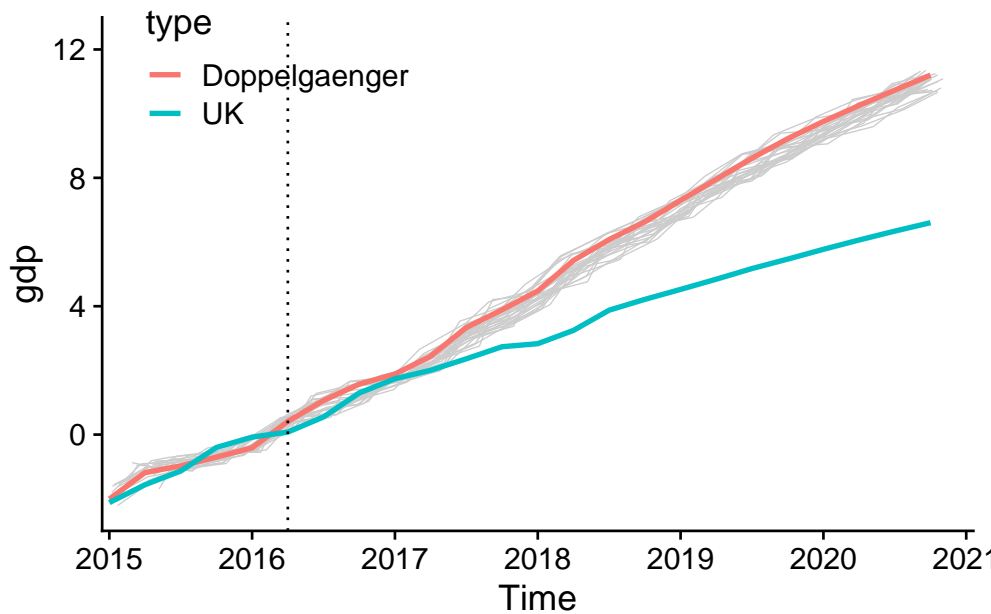


Abbildung 6.6: Placebo-Doppelgänger – Close-Up

Ein weiterer Placebo-Test in Born u. a. (2019) ist ein Vergleich der Doppelgänger-Gap Großbritanniens mit Doppelgänger-Gaps für fiktive Referenden in 2016-Q2 in Ländern mit wesentlichem Einfluss bei der Konstruktion des synthetischen Doppelgängers für Großbritannien: Die Schätzung des kausalen Effekts des Referendums auf das BIP in Großbritannien ist glaubwürdig, wenn lediglich die Doppelgänger-Gap für Großbritannien durch das Referendum beeinflusst wird, *nicht* aber die Doppelgänger-Gaps für Länder in der Kontrollgruppe.

Für diese grafische Placebo-Analyse modifizieren wir die Funktion `placebo()` entsprechend. `placebo_gap()` berechnet die Doppelgänger-Gap für das mit `treat` identifizierte Land. Das `if`-Statement zu Beginn stellt sicher, dass Großbritannien *nicht* als Kontroll-Einheit für die Placebo-Gaps verwendet wird.

```
# Funktion für Placebo-Gaps
placebo_gap <- function(treat) {

  # Kontrollgruppe definieren
  if(treat != 23) {
    controls <- (1:24)[-c(23, treat)]
  } else {
    controls <- (1:24)[-23]
  }
}
```



```

# Daten vorbereiten
dataprep_out <- dataprep(
  foo = brexit,
  predictors = c(
    "ConGDP", "InvGDP",
    "ExpGDP", "ImpGDP",
    "LPG", "EmpSha"
  ),
  dependent = "gdp",
  unit.variable = "ID",
  time.variable = "Time",
  treatment.identifier = treat,
  controls.identifier = controls,
  time.predictors.prior = seq(1995, 2016.25, .25),
  time.optimize.ssr = seq(1995, 2016.25, .25),
  unit.names.variable = "Country"
)

# Gewichte bestimmen
synth_out <- quietly(synth)(dataprep_out)$result

# Ergebnisse zusammenfassen
tb <- synth.tab(
  synth.res = synth_out,
  dataprep.res = dataprep_out
)

# Doppelgänger bestimmen
doppel <- left_join(
  x = brexit,
  y = tb$tab.w,
  by = c("Country" = "unit.names")
) %>%
  select(Time, gdp, Country, w.weights) %>%
  group_by(Time) %>%
  summarise(
    gdp_synth = sum(gdp * w.weights, na.rm = T),
  )

```

```

# Beobachtetes BIP auslesen
gdp <- brexit %>% filter(ID == treat) %>% pull(gdp)

return(

  # Doppelgänger-Gap berechnen
  doppel %>%
    mutate(
      ID = treat,
      gdp = gdp,
      gdp_gap = gdp - gdp_synth
    )

)

}

```

Für die Berechnung der Placebo-Gaps iterieren wir über die Indizes der in Tabelle 6.2 gelisteten Volkswirtschaften der Kontrollgruppe für Großbritannien.

```

# Indizes für "Donor Countries" und UK
donors_and_UK <- brexit %>%
  select(ID, Country) %>%
  distinct() %>%
  filter(
    Country %in%
      c(
        "United States", "Italy", "Iceland",
        "Luxembourg", "Germany", "United Kingdom"
      )
  ) %>%
  pull(ID)

```

```

# Placebo-Doppelgänger-Gaps berechnen
placebo_gaps_tbl <- map_dfr(
  .x = donors_and_UK,
  .f = \(x) placebo_gap(x)
)

```

Für die grafische Darstellung ergänzen wir die Variable `Country` zur Unterscheidung der Doppelgänger-Gaps für Großbritannien und die Kontroll-Länder.

```
# ID-Variable für UK und Kontroll-Länder
placebo_gaps_tbl <- placebo_gaps_tbl %>%
  mutate(
    Country = ifelse(ID == 23, "UK", "else")
  )
```

Um die Vergleichbarkeit der Doppelgänger-Gaps zu gewährleisten, standardisieren Born u. a. (2019) die Schätzungen der Gaps anhand der jeweiligen Mittelwerte für das Jahr 2015 und der Standardabweichungen im Zeitraum vor dem Brexit-Referendum. Wir berechnen diese Statistiken zunächst.

```
# Mittelwerte für 2015
means <- placebo_gaps_tbl %>%
  group_by(ID) %>%
  filter(between(Time, 2015, 2015.75)) %>%
  summarise(
    mean2015 = mean(gdp_gap)
  )

# Standardabweichungen vor Referendum
sds <- placebo_gaps_tbl %>%
  group_by(ID) %>%
  filter(Time < 2016.25) %>%
  summarise(
    thesd = sd(gdp_gap)
  )
```

Mit `dplyr::left_join()` führen wir diese Statistiken mit `placebo_gaps_tbl` zusammen und berechnen die standardisierten Doppelgänger-Gaps.

```
# Join + Standardisierung
placebo_gaps_std <-
  left_join(placebo_gaps_tbl, means) %>%
  left_join(sds) %>%
  mutate(gdp_gap_std = (gdp_gap - mean2015)/thesd)
```

Analog zum Code für Abbildung 6.4 plotten wir die Placebo-Gap-Zeitreihen mit `ggplot2::ggplot()`.

```
# Placebo-Gaps mit UK-Gap vergleichen
ggplot(
  data = placebo_gaps_std,
  mapping = aes(
    x = Time,
    y = gdp_gap_std,
    group = ID,
    lwd = Country,
    color = Country
  )
) +
  # Hilfslinie bei Differenz = 0
  geom_hline(yintercept = 0) +
  # Gaps
  geom_line() +
  # Referendum
  geom_vline(xintercept = 2016.25, lty = "dotted") +
  # Formatierung
  scale_color_manual(
    values = c("UK" = "steelblue", "else" = alpha("darkgray",
      ↪ .5))
  ) +
  scale_linewidth_manual(
    values = c("UK" = 1, "else" = .5)
  ) +
  scale_x_continuous(
    limits = c(2015, 2021), expand = c(0, .05)
  ) +
  theme_cowplot() +
  theme(legend.position = c(.05, .9))
```

Abbildung 6.7 zeigt die standardisierten Placebo-Doppelgänger-Gaps für ein fiktives Referendum zum Zeitpunkt 2016-Q2 in den 5 Kontroll-Volkswirtschaften, die für Konstruktion des BIP-Doppelgängers von Großbritannien relevant sind (grau). Der Vergleich mit der standardisierten Doppelgänger-Gap für Großbritannien (blau). Der Verlauf der Placebo-Gaps zeigt an, dass keine

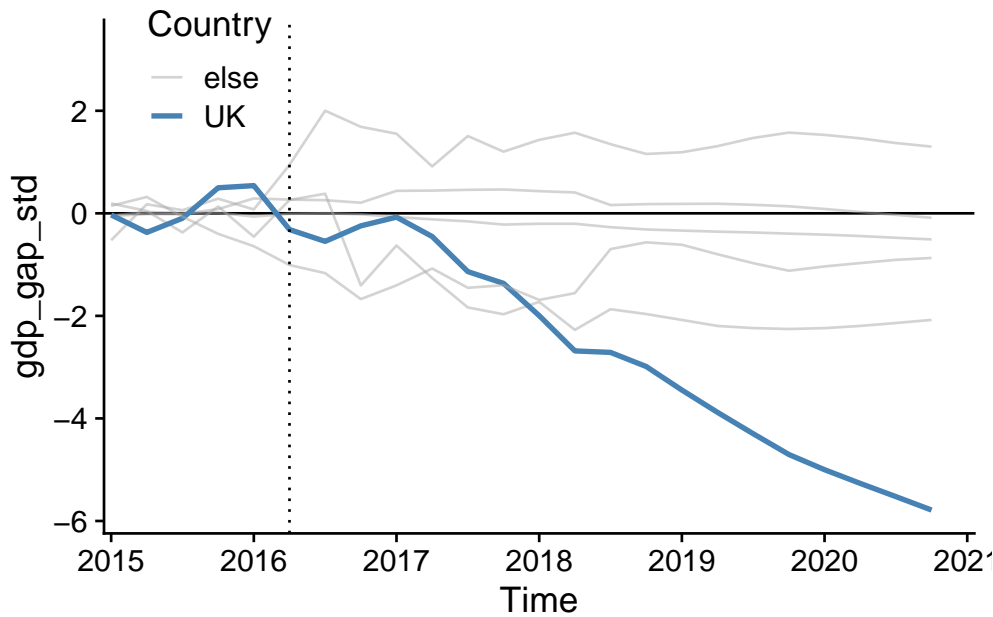


Abbildung 6.7: Placebo- und UK-Doppelgänger-Gaps

Abweichungen mit negativem Trend von der Referenzlinie bei 0 (kein Unterschied zwischen beobachtetem und syntetischem BIP) nach dem Referendum vorliegen. Damit liefert die Grafik keine Hinweise auf einen Effekt fiktiver Interventionen in den Kontroll-Ländern. Für Großbritannien jedoch ist, ähnlich wie in Abbildung 6.4, ein negativer Trend nach dem Referendum deutlich erkennbar.

6.2.2 Statistische Inferenz

Die bisherigen Placebo-Tests liefern lediglich grafische Evidenz für die Signifikanz des negativen Effekts des Brexit-Referendums auf die Britische Volkswirtschaft. Methoden für statistische Inferenz für SCM sind Gegenstand aktueller Forschung. Born u. a. (2019) verwenden den End-Of-Sample Instability Test (*S*) von Andrews (2003). Dieses Verfahren kann für einen Test auf einen Strukturbruch gegen Ende einer Zeitreihe verwendet werden. In der vorliegende Studie wird der Test angewendet, um zu überprüfen, ob die Verteilung der Doppelgänger-Gap Großbritanniens für die letzten m Perioden jenseits des Referendums signifikant verschieden ist von Verteilung vorheriger Perioden.

Wir zeigen nachfolgend, wie diese Analyse in R mit der Funktion `CPAT::Andrews.test()` aus dem Paket CPAT durchgeführt werden kann. Wir testen zunächst auf eine signifikante Diskrepanz der Doppelgänger-Gap in

Form eines Sturkturbruchs ab 2017 und fassen die Ergebnisse tabellarisch mit `broom::tidy()` und `gt::gt()` zusammen.

```
library(CPAT)

# Andrews' (2003) Test für 2017 durchführen
Andrews.test(
  x = gdp_gap$gdp_gap,
  M = which(gdp_gap$Time == 2017)
) %>%
  broom::tidy() %>%
  gt::gt() %>%
  tabopts
```

Tabelle 6.3: Andrews' (1993) End-of-Sample Instability Test

statistic	p.value	method
14.196	0.693	Andrews' Test for Structural Change

Gem. des großen p -Werts kann die Nullhypothese (keine strukturelle Veränderung ab 2017) nicht abgelehnt werden. Wir führen den Test nun für sämtliche Zeitpunkte ab 2017 durch und plotten die p -Werte nebst gepunkteten roten Hilfslinien für die gängigen Signifikanzniveaus (10%, 5%, 1%).

```
# Andrews' (1993) test für
# Post-Referendumsperioden
pvals_andrews <- map(seq(2017, 2020.5, .25), \(time) {
  tibble(
    Time = time,
    gap = gdp_gap %>% filter(Time == time) %>% pull(gdp_gap),
    pvalue = CPAT::Andrews.test(
      x = gdp_gap$gdp_gap,
      M = which(gdp_gap$Time == time)
    )$p.value
  )
}) %>%
  bind_rows()
```

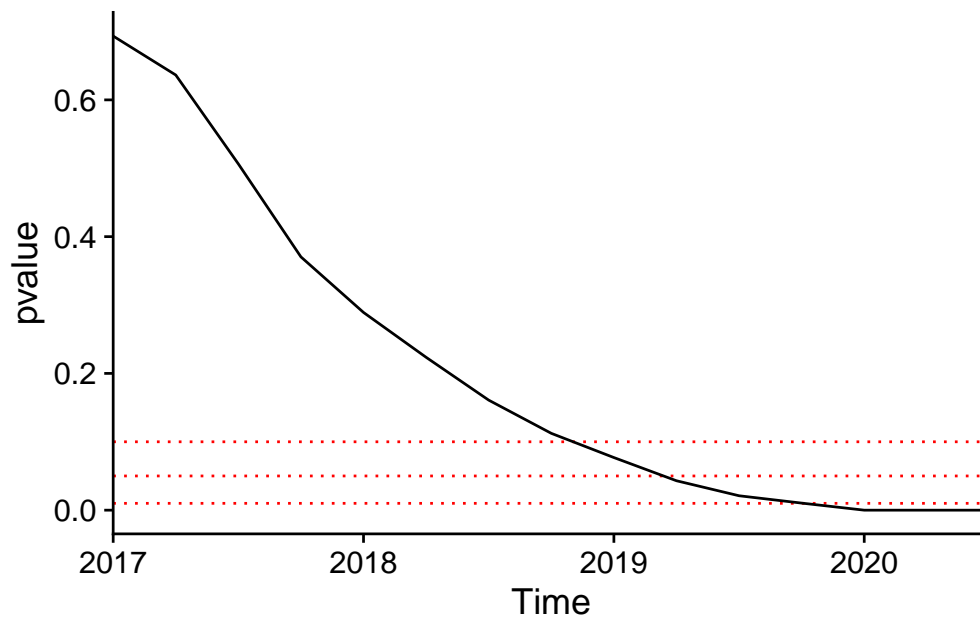


Abbildung 6.8: P-Werte für Andrews' (2003) Test

```
# p-Werte für Post-Interventionsperioden
pvals_andrews %>%
  ggplot(mapping = aes(x = Time, y = pvalue)) +
  geom_hline(
    yintercept = c(.1, .05, .01),
    lty = "dotted",
    col = "red"
  ) +
  geom_line() +
  scale_x_continuous(expand = c(0, 0)) +
  cowplot::theme_cowplot()
```

Der Verlauf der p -Werte zeigt deutlich, dass es für Zeitpunkte jenseits von 2018-Q3 Evidenz für eine strukturelle Veränderung der Doppelgänger-Gap für Großbritannien gibt. Diese Ergebnisse untermauern die Signifikanz der in Born u. a. (2019) mit SCM gefundenen negativen Effekte des Brexit-Votums auf die Britische Volkswirtschaft weiter.

Literatur

- Abadie, Alberto, Alexis Diamond, und Jens Hainmueller. 2014. „Comparative Politics and the Synthetic Control Method: COMPARATIVE POLITICS AND THE SYNTHETIC CONTROL METHOD“. *American Journal of Political Science* 59 (2): 495–510. <https://doi.org/10.1111/ajps.12116>.
- Abadie, Alberto, und Guido W. Imbens. 2008. „On the Failure of the Bootstrap for Matching Estimators.“ *Econometrica. Journal of the Econometric Society* 76 (6): 1537–57. <https://doi.org/10.3982/ECTA6474>.
- Abadie, Alberto, und Jann Spiess. 2022. „Robust Post-Matching Inference.“ *Journal of the American Statistical Association* 117 (538): 983–95. <https://doi.org/10.1080/01621459.2020.1840383>.
- Abadie, Alexis Diamond, Alberto, und Jens Hainmueller. 2010. „Synthetic Control Methods for Comparative Case Studies: Estimating the Effect of California’s Tobacco Control Program.“ *Journal of the American Statistical Association* 105 (490): 493–505. <https://doi.org/10.1198/jasa.2009.ap08746>.
- Andrews, D. W. K. 2003. „End-of-Sample Instability Tests“. *Econometrica* 71 (6): 1661–94. <https://doi.org/10.1111/1468-0262.00466>.
- Austin, P. 2011. „An Introduction to Propensity Score Methods for Reducing the Effects of Confounding in Observational Studies“. *Multivariate Behavioral Research* 46 (3): 399–424. <https://doi.org/10.1080/00273171.2011.568786>.
- Austin, Peter C., und Dylan S. Small. 2014. „The use of bootstrapping when using propensity-score matching without replacement: A simulation study.“ *Statistics in Medicine* 33 (24): 4306–19. <https://doi.org/10.1002/sim.6276>.
- Austin, Peter C., und Elizabeth A. Stuart. 2017. „Estimating the Effect of Treatment on Binary Outcomes Using Full Matching on the Propensity Score.“ *Statistical Methods in Medical Research* 26 (6): 2505–25. <https://doi.org/10.1177/0962280215601134>.
- Barro, Robert J., und Jong Wha Lee. 2013. „A new data set of educational attainment in the world, 1950–2010“. *Journal of Development Economics* 104: 184–98. <https://doi.org/10.1016/j.jdeveco.2012.10.001>.
- Basten, Christoph, und Frank Betz. 2013. „Beyond work ethic: Religion, individual, and political preferences“. *American Economic Journal: Economic Policy* 5 (3): 67–91.

- Belloni, Alexandre, Daniel Chen, Victor Chernozhukov, und Christian Hansen. 2012. „Sparse models and methods for optimal instruments with an application to eminent domain“. *Econometrica* 80 (6): 2369–429.
- Belloni, Alexandre, und Victor Chernozhukov. 2013. „Least squares after model selection in high-dimensional sparse models“. *Bernoulli*, 521–47.
- Belloni, Alexandre, Victor Chernozhukov, und Christian Hansen. 2014. „High-dimensional methods and inference on structural and treatment effects“. *Journal of Economic Perspectives* 28 (2): 29–50.
- Bodory, Hugo, Lorenzo Camponovo, Martin Huber, und Michael Lechner. 2020. „The Finite Sample Performance of Inference Methods for Propensity Score Matching and Weighting Estimators.“ *Journal of Business & Economic Statistics*. <https://doi.org/10.2139/ssrn.2731969>.
- Born, Benjamin, Gernot J Müller, Moritz Schularick, und Petr Sedláček. 2019. „The Costs of Economic Nationalism: Evidence from the Brexit Experiment*“. *The Economic Journal* 129 (623): 2722–44. <https://doi.org/10.1093/ej/uez020>.
- Cattaneo, Matias D, Michael Jansson, und Xinwei Ma. 2020. „Simple local polynomial density estimators“. *Journal of the American Statistical Association* 115 (531): 1449–55.
- Cortez, Paulo, und Alice Maria Gonçalves Silva. 2008. „Using data mining to predict secondary school student performance“.
- Efron, Bradley, Trevor Hastie, Iain Johnstone, und Robert Tibshirani. 2004. „Least angle regression“.
- Gelman, Andrew, und Guido Imbens. 2019. „Why high-order polynomials should not be used in regression discontinuity designs“. *Journal of Business & Economic Statistics* 37 (3): 447–56.
- Hahn, P Richard, Carlos M Carvalho, David Puelz, und Jingyu He. 2018. „Regularization and confounding in linear regression for treatment effect estimation“.
- Hainmueller, Jens. 2012. „Entropy Balancing for Causal Effects: A Multivariate Reweighting Method to Produce Balanced Samples in Observational Studies“. *Political Analysis* 20 (1): 25–46. <https://doi.org/10.1093/pan/mpr025>.
- Hájek, J. 1971. „Comment on ‚An essay on the logical foundations of survey sampling‘ by Basu, D“. *Foundations of Statistical Inference* 236.
- Hill, Jennifer, und Jerome P. Reiter. 2006. „Interval estimation for treatment effects using propensity score matching. Statistics in Medicine“. *Statistics in Medicine* 25 (13): 2230–56. <https://doi.org/10.1002/sim.2277>.
- Hirano, Keisuke, Guido Imbens, und Geert Ridder. 2003. „Efficient Estimation of Average Treatment Effects Using the Estimated Propensity Score.“ *Econometrica* 71 (4): 1161–89. <https://doi.org/10.1111/1468-0262.00442>.

- Hoerl, Arthur E, und Robert W Kennard. 1970. „Ridge regression: Biased estimation for nonorthogonal problems“. *Technometrics* 12 (1): 55–67.
- Imbens. 2016. „Matching on the Estimated Propensity Score.“ *Econometrica* 84 (2): 781–807. <https://doi.org/10.3982/ecta11293>.
- Imbens, G. W., und Thomas Lemieux. 2008. „Regression discontinuity designs: A guide to practice“. *Journal of econometrics* 142 (2): 615–35.
- Imbens, Guido, und Karthik Kalyanaraman. 2012. „Optimal bandwidth choice for the regression discontinuity estimator“. *The Review of economic studies* 79 (3): 933–59.
- Lee, David S. 2008. „Randomized experiments from non-random selection in US House elections“. *Journal of Econometrics* 142 (2): 675–97.
- Love, Thomas. 2004. „Graphical display of covariate balance“. Presentation.
- McCrary, Justin. 2008. „Manipulation of the running variable in the regression discontinuity design: A density test“. *Journal of Econometrics* 142 (2): 698–714.
- Rosenbaum, Paul R., und Donald R. Rubin. 1983. „The central role of the propensity score in observational studies for causal effects“. *Biometrika* 70 (1): 170–84. <https://doi.org/10.1017/cbo9780511810725.016>.
- Tibshirani, Robert. 1996. „Regression shrinkage and selection via the lasso“. *Journal of the Royal Statistical Society Series B: Statistical Methodology* 58 (1): 267–88.
- Weber, Max. 2004. *Die protestantische Ethik und der Geist des Kapitalismus*. Bd. 1614. CH Beck.
- Wooldridge, Jeffrey. 2010. *Econometric Analysis of Cross Section and Panel Data*. Second edition. Cambridge, Massachusetts: MIT.