Replikations-Skript zur Vorlesung 10: Das multiple lineare Regressionsmodell

Claudius Gräbner

KW 3 2021

Contents

In diesem Dokument werden alle Abbildungen und Tabellen aus der siebten Vorlesung repliziert. Dabei gebe ich der Info wegen *allen* R Code. Entsprechend sind bestimmt auch einige Befehle dabei, die Sie jetzt noch nicht kennen.

Zudem nehme ich an, dass im Arbeitsverzeichnis der Ordner `data/T10/` existiert und in diesem der Datensatz `wb_bip_ineq_mort.csv` enthalten sind (diese ist über die Repository zur Vorlesung verfügbar).

Folgende Pakete werden zudem in diesem Skript verwendet:

```
library(tidyverse)
library(data.table)
library(ggpubr)
library(latex2exp)
library(here)
library(lmtest)
library(MASS)
```

```
library(sandwich)
library(icaeDesign)
```

Beachten Sie, dass das Paket icaeDesign nicht über die zentrale Paketverwaltung verfübar ist. Es muss folgendermaßen installiert werden:

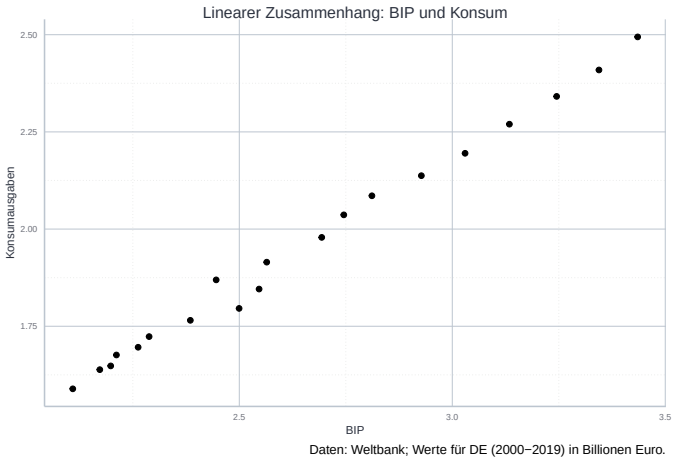
```
devtools::install_github("graebnerc/icaeDesign")
```

1 Diskussion der Annahmen

1.1 Lineares Modell

Das erste Beispiel ist der bekannte Zusammenhang zwischen Konsum und BIP:

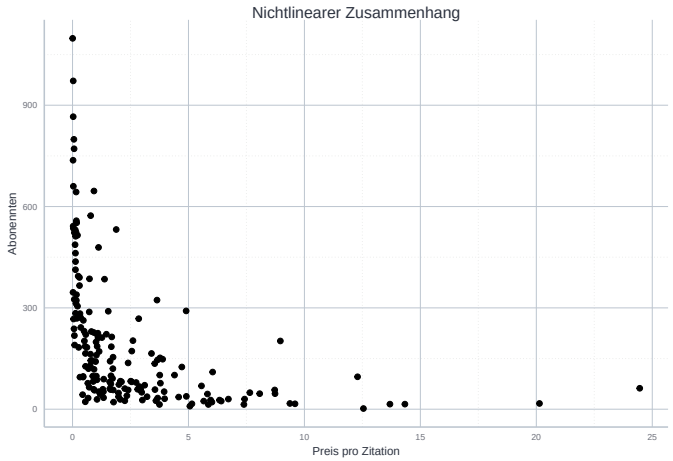
```
bip_konsum_plot <- fread(here("data/T5/bip_einleitung.csv")) %>%
  ggplot(
  data = ., aes(x=BIP, y=Konsum)
  ) +
  geom_point() +
  labs(
    title = "Linearer Zusammenhang: BIP und Konsum",
    x="BIP", y="Konsumausgaben",
    caption = "Daten: Weltbank; Werte für DE (2000-2019) in Billionen Euro.") +
  theme_icae()

ggsave(plot = bip_konsum_plot,
       filename = here("figures/T10/01_bip-konsum.pdf"),
       width = 4, height = 3)

bip_konsum_plot
```



Das zweite Beispiel bezug sich auf den Journaldatensatz:

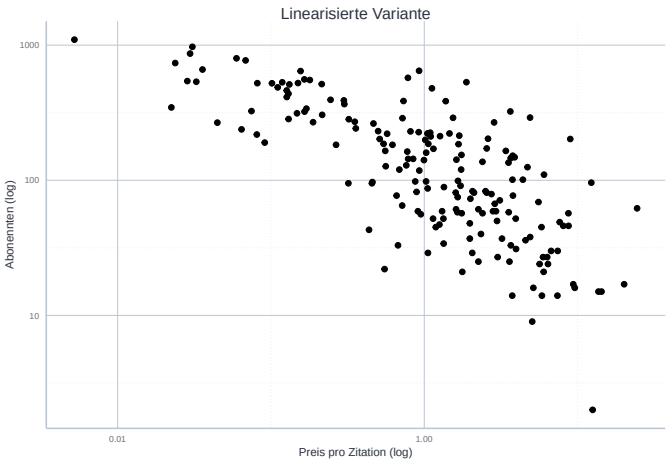
```
journal_plot <- fread(here("data/T5/journaldaten.csv")) %>%
  dplyr::select(one_of("Zitationen", "Abonnenten", "Preis")) %>%
  mutate(Preis_pro_Zitat=Preis/Zitationen) %>%
  ggplot(., aes(x=Preis_pro_Zitat, y=Abonnenten)) +
  geom_point() +
  labs(
    title = "Nichtlinearer Zusammenhang",
```

```
    x="Preis pro Zitation", y="Abonennten") +
  theme_icae()

ggsave(plot = journal_plot,
       filename = here("figures/T10/02_journal_preise_zitate.pdf"),
       width = 4, height = 3)

journal_plot
```



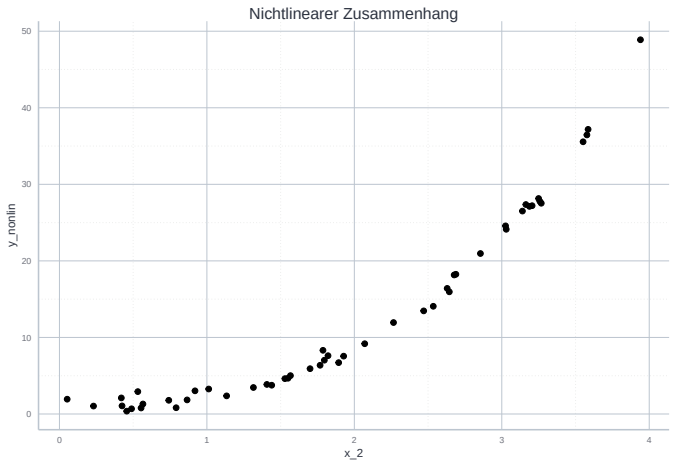
Hier die logarithmierte Variante:

```
journal_plot_log <- journal_plot +
  scale_x_continuous(
    trans = "log", breaks = scales::trans_breaks("log", function(x) 10^x)
    ) +
  scale_y_continuous(
    trans = "log", breaks = scales::trans_breaks("log", function(x) 10^x)
    ) +
  labs(
    title = "Linearisierte Variante",
    x="Preis pro Zitation (log)", y="Abonennten (log)"
  )

ggsave(plot = journal_plot_log,
       filename = here("figures/T10/03_journal_preise_zitate_log.pdf"),
       width = 4, height = 3)

journal_plot_log
```



Zuletzt noch der nicht linearisierbare Zusammenhang. Diese Daten habe ich selber erstellt:

```
set.seed("123")
sample_size <- 50
x_1 <- runif(sample_size, min = 0.05, max = 4)
x_2 <- runif(sample_size, min = 0.05, max = 4)
beta_0 <- 0.2
beta_1 <- 0.5
beta_2 <- 2.8

y_nonlinear <- rep(NA, sample_size)
for (i in 1:sample_size){
  y_nonlinear[i] <- beta_0 + beta_1*x_1[i] + x_2[i]**(beta_2) +
    rnorm(1, mean = 0.1, sd = 0.5)
}

full_sample <- tibble(x1=x_1, x2=x_2,
                      y_nonlin=y_nonlinear)

nonlin_plot <- ggplot(data = full_sample, mapping = aes(x=x_2, y=y_nonlin)) +
  ggtitle("Nichtlinearer Zusammenhang") +
  geom_point() + theme_icae()

ggsave(plot = nonlin_plot,
       filename = here("figures/T10/04_nonlin-example.pdf"),
       width = 4, height = 3)

nonlin_plot
```



1.2 Exogenität der UV

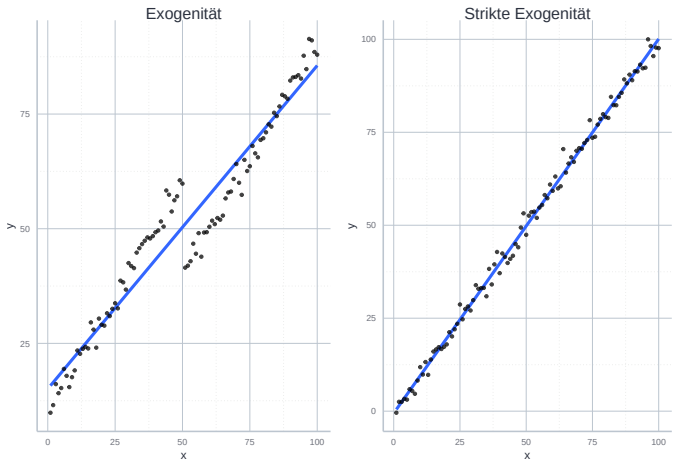
Der Unterschied zwischen Exogenität und strikter Exogenität wurde mit folgenden Abbildungen verdeutlicht:

```
set.seed("123")
x_vals <- 1:100
y_vals_exo <- rep(NA, length(x_vals))
y_vals_exo_strict <- rep(NA, length(x_vals))
for (i in 1:50){y_vals_exo[i] <- x_vals[i] + rnorm(1, 10, 2)}
for (i in 51:100){y_vals_exo[i] <- x_vals[i] + rnorm(1, -10, 2)}
for (i in 1:100){y_vals_exo_strict[i] <- x_vals[i] + rnorm(1, 0, 2)}

exo_data <- tibble("x"=x_vals, "y"=y_vals_exo)
exo_strict_data <- tibble("x"=x_vals, "y"=y_vals_exo_strict)

exo_plot <- ggplot(data = exo_data,
                   mapping = aes(x=x, y=y)
                   ) +
  geom_smooth(method = "lm", se = FALSE) +
  geom_point(alpha=0.75, size=0.75) +
  ggtitle("Exogenität") +
  theme_icae()

exo_strict_plot <- ggplot(data = exo_strict_data,
                          mapping = aes(x=x, y=y)
                          ) +
  geom_smooth(method = "lm", se = FALSE) +
  geom_point(alpha=0.75, size=0.75) +
  ggtitle("Strikte Exogenität") +
  theme_icae()

exo_full_plot <- ggarrange(exo_plot, exo_strict_plot, ncol = 2)
```

```
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
```

```
ggsave(plot = exo_full_plot,
       filename = here("figures/T10/05_exogeneity-example.pdf"),
       width = 4, height = 3)
```

```
exo_full_plot
```

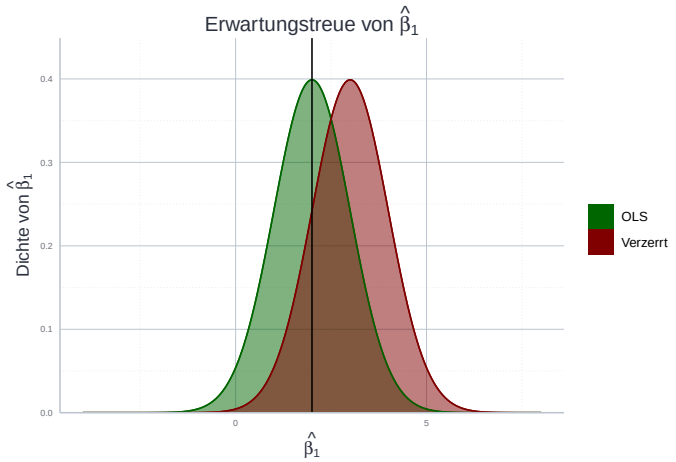


2 Implikationen der Annahmen

2.1 Erwartungstreue

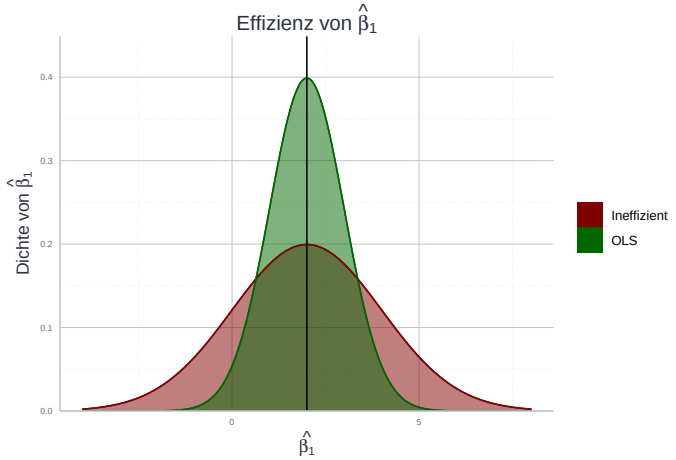
```
beta_1_wahr <- 2
beta_1_verzerrt <- 3
x_range <- seq(-4, 8, 0.01)

dist_beta_ols <- dnorm(x = x_range, mean = beta_1_wahr, sd = 1)
dist_beta_biased <- dnorm(x = x_range, mean = beta_1_verzerrt, sd = 1)
dist_beta_inefficient <- dnorm(x = x_range, mean = beta_1_wahr, sd = 2)
data_ols <- tibble(yhat=dist_beta_ols, xrange=x_range) %>%
  dplyr::mutate(Art="OLS")
data_biased <- tibble(yhat=dist_beta_biased, xrange=x_range) %>%
  dplyr::mutate(Art="Verzerrt")
data_inefficient <- tibble(yhat=dist_beta_inefficient, xrange=x_range) %>%
  dplyr::mutate(Art="Ineffizient")
data_total <- bind_rows(list(data_ols, data_biased, data_inefficient))

data_erwartungstreue <- dplyr::filter(data_total,
                                      Art %in% c("OLS", "Verzerrt"))

erwartungstreue <- ggplot(data = data_erwartungstreue,
                          mapping = aes(x=xrange,
                                        y=yhat, color=Art,
                                        fill=Art)
  ) +
  geom_ribbon(aes(ymin=0, ymax=yhat, fill=Art), alpha=0.5) +
  geom_line(key_glyph= draw_key_rect) +
  geom_vline(xintercept = beta_1_wahr) +
  scale_x_continuous(limits = c(-4, 8)) +
  scale_y_continuous(expand = expansion(c(0, 0), c(0, 0.05))) +
  ylab(TeX("Dichte von $\\hat{\\beta}_1$")) +
  xlab(TeX("$\\hat{\\beta_1}$")) +
  ggtitle(TeX("Erwartungstreue von $\\hat{\\beta}_1$")) +
  theme_icae() +
```

```r
  scale_color_manual(values = c("OLS"="#006600", "Verzerrt"="#800000"),
                     aesthetics = c("color", "fill")) +
  theme(axis.title = element_text(size=12),
        plot.title = element_text(size=14), legend.position = "right")

ggsave(plot = erwartungstreue,
       filename = here("figures/T10/06_erwartungstreue.pdf"),
       width = 5, height = 3)

erwartungstreue
```



2.2 Effizienz

```
## Warning: `expand_scale()` is deprecated; use `expansion()` instead.
```



3 Heteroskedastie

3.1 Monte Carlo Simulation

```r
dgp <- function(x1, beta0, beta1, hetero=FALSE){
  y <- rep(NA, length(x1))
  sd_hetero <- 0.25 * x1
  sd_homo <- mean(sd_hetero)
```

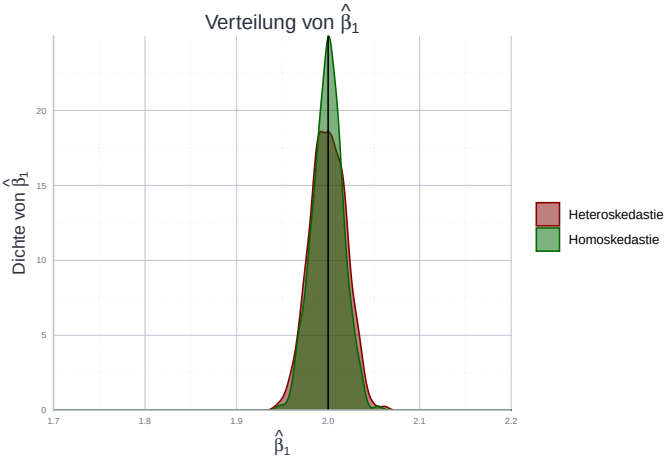
```
  if (hetero){
    errors <- rnorm(n = length(x1), mean = 0,
                    sd = sd_hetero)
  } else {
    errors <- rnorm(n = length(x1), mean = 0,
                    sd = sd_homo
                    )
  }
  for (i in 1:length(x1)){
    y[i] <- beta0 + beta1*x1[i] + errors[i]
  }
  final_data <- tibble(y=y, x1=x1, errors=errors)
  return(final_data)
}
```

```
mcs <- function(n_stichproben,
                x1, wahres_b0, wahres_b1, schaetzgleichung,
                heterosk=FALSE){
  schaetzung_b1 <- rep(NA, n_stichproben)
  stdfehler_b1 <- rep(NA, n_stichproben)
  for (i in 1:n_stichproben){
    # Stichprobe ziehen:
    stichprobe <- dgp(x1 = x1, beta0 = wahres_b0,
                      beta1 = wahres_b1,
                      hetero = heterosk)
    # Parameter schätzen:
    schaetzung <- summary(
      lm(formula = schaetzgleichung,
         data = stichprobe)
      )
    # Relevante Werte speichern:
    schaetzung_b1[i] <- schaetzung$coefficients[2]
    stdfehler_b1[i] <- schaetzung$coefficients[4]
  }
  # In einer Tabelle zusammenfassen:
  Fall_Bezeichnung <- ifelse(heterosk, "Heteroskedastie", "Homoskedastie")
  ergebnisse <- tibble(
    b1_coef=schaetzung_b1,
    b1_stdf=stdfehler_b1,
    Fall=rep(Fall_Bezeichnung,
             n_stichproben)
  )
return(ergebnisse)
}
```

```
set.seed("1234")
n_stichproben <- 250
n_beobachtungen <- 1000
x_data <- runif(n = n_beobachtungen, min = 1, max = 10)
wahres_b0 <- 1
wahres_b1 <- 2
schaetzgleichung <- as.formula("y~x1")
```

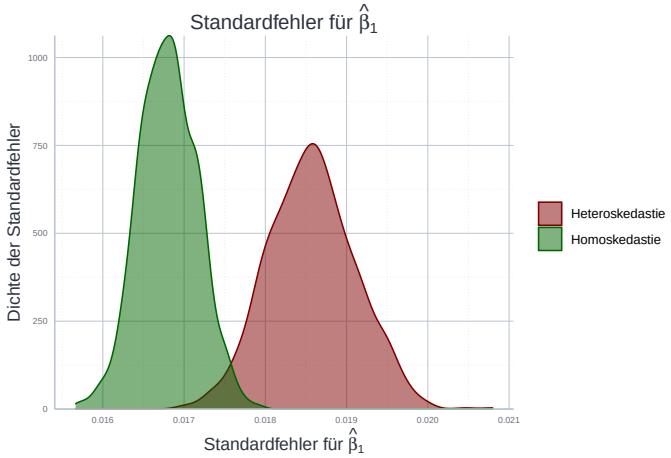
```r
set.seed("1234")
homosc_results <- mcs(1000, x_data,
                      wahres_b0, wahres_b1,
                      schaetzgleichung, heterosk = F)
hetero_results <- mcs(1000, x_data,
                      wahres_b0, wahres_b1,
                      schaetzgleichung, heterosk = T)
full_results <- rbind(homosc_results, hetero_results)
```

```r
beta_1_plot <- ggplot(data = full_results,
                      mapping = aes(x=b1_coef, color=Fall, fill=Fall)) +
  geom_density(alpha=0.5) +
  scale_y_continuous(expand = expansion(c(0, 0), c(0, 0.05))) +
  scale_x_continuous(limits = c(1.7, 2.2), expand = c(0,0)) +
  geom_vline(xintercept = wahres_b1) +
  ylab(TeX("Dichte von $\\hat{\\beta}_1$")) +
  xlab(TeX("$\\hat{\\beta}_1$")) +
  ggtitle(TeX("Verteilung von $\\hat{\\beta}_1$")) +
  scale_color_manual(values = c("Homoskedastie"="#006600",
                                "Heteroskedastie"="#800000"),
  aesthetics = c("color", "fill")) +
  theme_icae() +
  theme(axis.title = element_text(size=12),
        plot.title = element_text(size=14),
        legend.position = "right")

ggsave(plot = beta_1_plot,
       filename = here("figures/T10/08_hetero-mcs-b1.pdf"),
       width = 5, height = 3)

beta_1_plot
```




```r
beta_1_stdf_plot <- ggplot(data = full_results,
                           mapping = aes(x=b1_stdf, color=Fall, fill=Fall)) +
  geom_density(alpha=0.5) +
  scale_y_continuous(expand = expansion(c(0, 0), c(0, 0.05))) +
  ylab(TeX("Dichte der Standardfehler")) +
  xlab(TeX("Standardfehler für $\\hat{\\beta}_1$")) +
  ggtitle(TeX("Standardfehler für $\\hat{\\beta}_1$")) +
```

```
  scale_color_manual(values = c("Homoskedastie"="#006600",
                                "Heteroskedastie"="#800000"),
                     aesthetics = c("color", "fill")) +
  theme_icae() +
  theme(axis.title = element_text(size=12),
        plot.title = element_text(size=14),
        legend.position = "right")

ggsave(plot = beta_1_stdf_plot,
       filename = here("figures/T10/09_hetero-mcs-b1-std.pdf"),
       width = 5, height = 3)

beta_1_stdf_plot
```



3.2 Tukey-Anscombe Plot

3.3 Tests

```
bptest(schaetzung_homo)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  schaetzung_homo
## BP = 0.0067387, df = 1, p-value = 0.9346
```

```
bptest(schaetzung_hetero)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  schaetzung_hetero
## BP = 88.513, df = 1, p-value < 2.2e-16
```

Goldfeld-Quandt Test in der Standard Spezifikation:

```
gqtest(schaetzung_homo)
```

```
##
##  Goldfeld-Quandt test
```

```
##
## data:  schaetzung_homo
## GQ = 0.98576, df1 = 248, df2 = 248, p-value = 0.5449
## alternative hypothesis: variance increases from segment 1 to 2
```

```
gqtest(schaetzung_hetero)
```

```
##
##  Goldfeld-Quandt test
##
## data:  schaetzung_hetero
## GQ = 0.79606, df1 = 248, df2 = 248, p-value = 0.9635
## alternative hypothesis: variance increases from segment 1 to 2
```

Variante für sinkende Varianz:

```
gqtest(schaetzung_hetero, alternative = "less")
```

```
##
##  Goldfeld-Quandt test
##
## data:  schaetzung_hetero
## GQ = 0.79606, df1 = 248, df2 = 248, p-value = 0.03655
## alternative hypothesis: variance decreases from segment 1 to 2
```

Zweiseitige Variante mit verminderter Power verfügt:

```
gqtest(schaetzung_hetero, alternative = "two.sided")
```

```
##
##  Goldfeld-Quandt test
##
## data:  schaetzung_hetero
## GQ = 0.79606, df1 = 248, df2 = 248, p-value = 0.0731
## alternative hypothesis: variance changes from segment 1 to 2
```

3.4 Reaktionen

Robuste Schätzung der Varianz-Kovarianz Matrix:

```
var_covar_matrix <- vcovHC(schaetzung_hetero, type = "HC1")
var_covar_matrix
```

```
##              (Intercept)           x1
## (Intercept)  0.018596906 -0.004287583
## x1          -0.004287583  0.001130885
```

Korrigierten Standardfehle:

```
coeftest(schaetzung_hetero, vcov. = var_covar_matrix)
```

```
##
## t test of coefficients:
##
##             Estimate Std. Error t value  Pr(>|t|)
## (Intercept) 2.047718   0.136370  15.016 < 2.2e-16 ***
## x1          0.482333   0.033629  14.343 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Original-Standardfehler:

```
summary(schaetzung_hetero)
```

```
##
## Call:
## lm(formula = schaetzgleichung, data = stichprobe_hetero)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.8628 -0.8143 -0.0055  0.7927  5.7683
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   2.0477     0.1753   11.68   <2e-16 ***
## x1            0.4823     0.0291   16.58   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.657 on 498 degrees of freedom
## Multiple R-squared:  0.3556, Adjusted R-squared:  0.3543
## F-statistic: 274.8 on 1 and 498 DF,  p-value: < 2.2e-16
```

4 Autokorrelation

4.1 Monte-Carlo Simulation

```
dgp_acl <- function(beta_0_wahrheit, beta_1_wahrheit,
                    unabh_werte, korrelation_fehler){
  size <- length(unabh_werte)
  y <- rep(NA, size)
  fehler <- rep(NA, size)
  fehler[1] <- rnorm(1)
  y[1] <- beta_0_wahrheit + beta_1_wahrheit*unabh_werte[1] + fehler[1]
  for (i in 2:size){
    fehler[i] <- korrelation_fehler*fehler[i-1] + rnorm(1)
    y[i] <- beta_0_wahrheit + beta_1_wahrheit*unabh_werte[i] + fehler[i]
  }
  datensatz <- tibble("x"=unabh_werte, "y"=y, "Fehler"=fehler)
  return(datensatz)
}
```

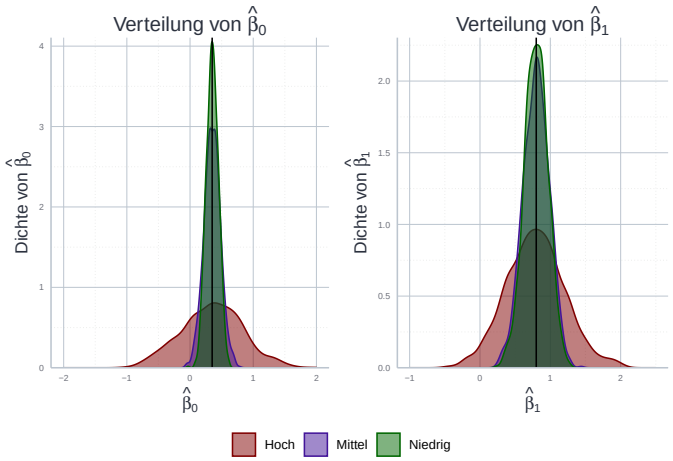
```
mcs_act <- function(iterationen, b0_wahrheit, b1_wahrheit,
                    x_werte, fehler_korrelation, fall){
  b0_coef <- rep(NA, iterationen)
  b1_coef <- rep(NA, iterationen)
  b0_stdf <- rep(NA, iterationen)
  b1_stdf <- rep(NA, iterationen)
  for (i in 1:iterationen){
    daten <- dgp_acl(b0_wahrheit, b1_wahrheit,
                     x_werte, fehler_korrelation)
    schaetzung <- summary(lm(y ~ x, data = daten))
    b0_coef[i] <- schaetzung[["coefficients"]][1]
    b1_coef[i] <- schaetzung[["coefficients"]][2]
```

```
    b0_stdf[i] <- schaetzung[["coefficients"]][3]
    b1_stdf[i] <- schaetzung[["coefficients"]][4]
  }
  ergebnisse <- tibble(
    "b0_coef"=b0_coef,
    "b1_coef"=b1_coef,
    "b0_stdf"=b0_stdf,
    "b1_stdf"=b1_stdf,
    "Fall"=rep(fall, iterationen)
  )
  return(ergebnisse)
}
```

```
set.seed("1234")
stichproben_n <- 500
mcs_iterationen <- 1000
beta_0_wahr <- 0.35
beta_1_wahr <- 0.8
r_small <- 0.1
r_mid <- 0.5
r_large <- 0.9
unab_vars <- runif(stichproben_n)
```

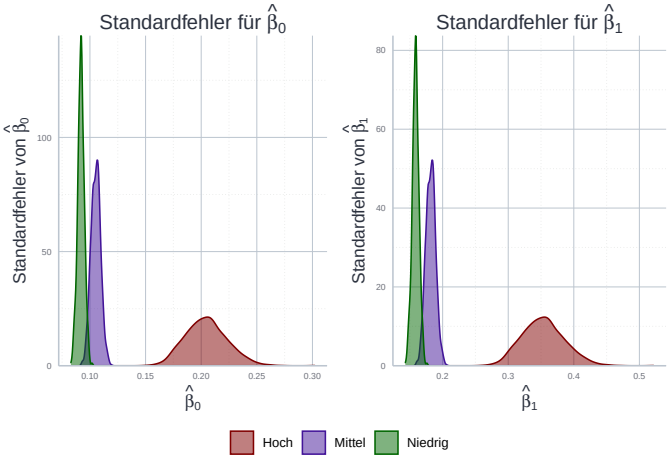
```
b0_coef_plot <- ggplot(data = mcs_ergebnisse,
                       aes(x=b0_coef, color=Fall, fill=Fall)) +
  geom_density(alpha=0.5) +
  scale_y_continuous(expand = expansion(c(0, 0), c(0, 0.05))) +
  scale_x_continuous(limits = c(-2, 2)) +
  geom_vline(xintercept = beta_0_wahr) +
  ylab(TeX("Dichte von $\\hat{\\beta}_0$")) +
  xlab(TeX("$\\hat{\\beta}_0$")) +
  ggtitle(TeX("Verteilung von $\\hat{\\beta}_0$")) +
  scale_color_icae(palette = "mixed", reverse = T,
                   aesthetics=c("color", "fill")
                   ) +
  theme_icae() +
  theme(axis.title = element_text(size=12),
        plot.title = element_text(size=14),
        legend.position = "bottom")

b1_coef_plot <- ggplot(data = mcs_ergebnisse,
                       aes(x=b1_coef, color=Fall, fill=Fall)) +
  geom_density(alpha=0.5) +
  scale_y_continuous(expand = expansion(c(0, 0), c(0, 0.05))) +
  scale_x_continuous(limits = c(-1, 2.5)) +
  geom_vline(xintercept = beta_1_wahr) +
  ylab(TeX("Dichte von $\\hat{\\beta}_1$")) +
  xlab(TeX("$\\hat{\\beta}_1$")) +
  ggtitle(TeX("Verteilung von $\\hat{\\beta}_1$")) +
  scale_color_icae(palette = "mixed", reverse = T,
                   aesthetics=c("color", "fill")
                   ) +
  theme_icae() +
  theme(axis.title = element_text(size=12),
```

```
        plot.title = element_text(size=14),
        legend.position = "bottom")

full_plot <- ggarrange(b0_coef_plot, b1_coef_plot, ncol = 2,
                       common.legend = T, legend = "bottom")

ggsave(plot = full_plot,
       filename = here("figures/T10/11_autocrl-mcs-b1.pdf"),
       width = 5, height = 4)

full_plot
```




```
b0_stdf_plot <- ggplot(data = mcs_ergebnisse,
                       aes(x=b0_stdf, color=Fall, fill=Fall)) +
  geom_density(alpha=0.5) +
  scale_y_continuous(expand = expansion(c(0, 0), c(0, 0.05))) +
  ylab(TeX("Standardfehler von $\\hat{\\beta}_0$")) +
  xlab(TeX("$\\hat{\\beta}_0$")) +
  ggtitle(TeX("Standardfehler für $\\hat{\\beta}_0$")) +
  scale_color_icae(palette = "mixed", reverse = T,
                   aesthetics=c("color", "fill")
                   ) +
  theme_icae() +
  theme(axis.title = element_text(size=12),
        plot.title = element_text(size=14),
        legend.position = "bottom")

b1_stdf_plot <- ggplot(data = mcs_ergebnisse,
                       aes(x=b1_stdf, color=Fall, fill=Fall)) +
  geom_density(alpha=0.5) +
  scale_y_continuous(expand = expansion(c(0, 0), c(0, 0.05))) +
  ylab(TeX("Standardfehler von $\\hat{\\beta}_1$")) +
  xlab(TeX("$\\hat{\\beta}_1$")) +
  ggtitle(TeX("Standardfehler für $\\hat{\\beta}_1$")) +
  scale_color_icae(palette = "mixed", reverse = T,
                   aesthetics=c("color", "fill")
                   ) +
  theme_icae() +
  theme(axis.title = element_text(size=12),
```

```
        plot.title = element_text(size=14),
        legend.position = "bottom")

full_plot <- ggarrange(b0_stdf_plot, b1_stdf_plot, ncol = 2,
                       common.legend = T, legend = "bottom")

ggsave(plot = full_plot,
       filename = here("figures/T10/12_autocrl-mcs-b1-std.pdf"),
       width = 5, height = 4)

full_plot
```



4.2 Test

```
set.seed("123")
small_acl <- lm(y~x, data=dgp_acl(0.5, 0.75, 1:100, 0.05))
mid_acl <- lm(y~x, data=dgp_acl(0.5, 0.75, 1:100, 0.5))
large_acl <- lm(y~x, data=dgp_acl(0.5, 0.75, 1:100, 0.85))
```

```
Box.test(mid_acl$residuals, lag = 1, type = "Box-Pierce")
```

```
##
##  Box-Pierce test
##
## data:  mid_acl$residuals
## X-squared = 9.5899, df = 1, p-value = 0.001957
```

```
Box.test(mid_acl$residuals, lag = 1, type = "Ljung-Box")
```

```
##
##  Box-Ljung test
##
## data:  mid_acl$residuals
## X-squared = 9.8805, df = 1, p-value = 0.00167
```

```
dwtest(small_acl)
```

```
##
##  Durbin-Watson test
##
```

```
## data:  small_acl
## DW = 1.9569, p-value = 0.3741
## alternative hypothesis: true autocorrelation is greater than 0
```

```
dwtest(mid_acl, alternative = "greater")
```

```
##
##  Durbin-Watson test
##
## data:  mid_acl
## DW = 1.3469, p-value = 0.0003333
## alternative hypothesis: true autocorrelation is greater than 0
```

```
dwtest(mid_acl, alternative = "less")
```

```
##
##  Durbin-Watson test
##
## data:  mid_acl
## DW = 1.3469, p-value = 0.9997
## alternative hypothesis: true autocorrelation is less than 0
```

```
dwtest(mid_acl, alternative = "two.sided")
```

```
##
##  Durbin-Watson test
##
## data:  mid_acl
## DW = 1.3469, p-value = 0.0006667
## alternative hypothesis: true autocorrelation is not 0
```

```
bgtest(mid_acl, order = 1, type = "F")
```

```
##
##  Breusch-Godfrey test for serial correlation of order up to 1
##
## data:  mid_acl
## LM test = 10.702, df1 = 1, df2 = 97, p-value = 0.001483
```

```
bgtest(mid_acl, order = 1, type = "Chisq")
```

```
##
##  Breusch-Godfrey test for serial correlation of order up to 1
##
## data:  mid_acl
## LM test = 9.9367, df = 1, p-value = 0.00162
```

4.3 Korrektur

```
var_covar_matrix <- vcovHAC(large_acl)
coeftest(large_acl, vcov. = var_covar_matrix)
```

```
##
## t test of coefficients:
##
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 0.186245   0.919704  0.2025   0.8399
## x           0.768353   0.015084 50.9371   <2e-16 ***
```

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

```
summary(large_acl)
```

```
##
## Call:
## lm(formula = y ~ x, data = dgp_acl(0.5, 0.75, 1:100, 0.85))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.5399 -1.1799 -0.1028  1.0744  3.5191
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 0.186245   0.304383   0.612    0.542
## x           0.768353   0.005233 146.833   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.511 on 98 degrees of freedom
## Multiple R-squared:  0.9955, Adjusted R-squared:  0.9954
## F-statistic: 2.156e+04 on 1 and 98 DF,  p-value: < 2.2e-16
```

5 Multikollinearität

```
set.seed("123")
stichprobengroesse <- 500
r_small <- 0.0
r_mid <- 0.4
r_large <- 0.9

data_small = mvrnorm(n=stichprobengroesse, mu=c(0, 0),
                     Sigma=matrix(c(1, r_small, r_small, 1),
                                  nrow=2), empirical=TRUE)

data_mid = mvrnorm(n=stichprobengroesse, mu=c(0, 0),
                   Sigma=matrix(c(1, r_mid, r_mid, 1),
                                nrow=2), empirical=TRUE)

data_large = mvrnorm(n=stichprobengroesse, mu=c(0, 0),
                     Sigma=matrix(c(1, r_large, r_large, 1),
                                  nrow=2), empirical=TRUE)

x_1_small = data_small[, 1]
x_1_mid = data_mid[, 1]
x_1_large = data_large[, 1]

x_2_small = data_small[, 2]
x_2_mid = data_mid[, 2]
x_2_large = data_large[, 2]

cor(x_1_small, x_2_small)  # Test
```

```
## [1] -1.929638e-16
```

```
cor(x_1_mid, x_2_mid)  # Test
```

```
## [1] 0.4
```

```
cor(x_1_large, x_2_large)  # Test
```

```
## [1] 0.9
```

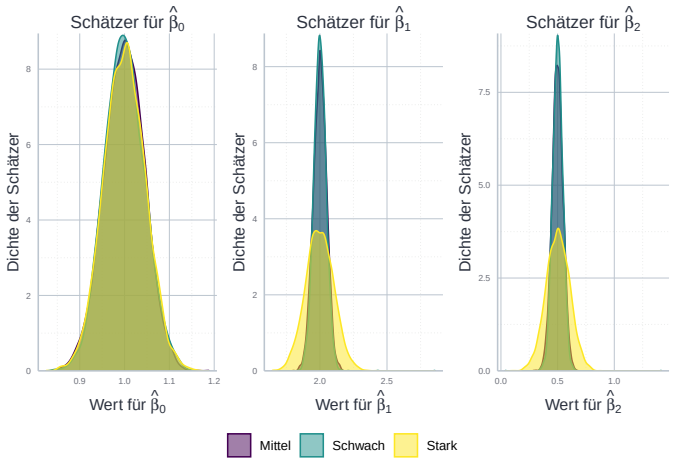
```
dgp_mc <- function(x_1, x_2, beta_0, beta_1, beta_2){
  y <- rep(NA, length(x_1))
  errors <- rnorm(length(x_1), mean = 0, sd = 1)
  for (i in 1:length(y)){
    y[i] <- beta_0 + beta_1*x_1[i] + beta_2*x_2[i] + errors[i]
  }
  data_full <- tibble("y"=y, "x_1"=x_1, "x_2"=x_2)
  return(data_full)
}

mcs_mc <- function(iterations, beta_0, beta_1, beta_2,
                   x_1, x_2, reg_func, identifier){
  beta_0_estimates <- rep(NA, iterations)
  beta_1_estimates <- rep(NA, iterations)
  beta_2_estimates <- rep(NA, iterations)
  for (i in 1:iterations){
    data_used <- dgp_mc(x_1, x_2, beta_0, beta_1, beta_2)
    regression <- lm(formula = reg_func, data = data_used)
    beta_0_estimates[i] <- regression$coefficients[1]
    beta_1_estimates[i] <- regression$coefficients[2]
    beta_2_estimates[i] <- regression$coefficients[3]
  }
  result_tibble <- tibble("b0"=beta_0_estimates,
                          "b1"=beta_1_estimates,
                          "b2"=beta_2_estimates,
                          "id"=rep(identifier, iterations))
  return(result_tibble)
}
```

```
set.seed("123")
iterations <- 5000
beta_0 <- 1
beta_1 <- 2
beta_2 <- 0.5
reg_func <- as.formula("y~x_1+x_2")
mcs_small <- mcs_mc(iterations, beta_0, beta_1, beta_2,
                    x_1_small, x_2_small, reg_func, "Schwach")
mcs_mid <- mcs_mc(iterations, beta_0, beta_1, beta_2,
                  x_1_mid, x_2_mid, reg_func, "Mittel")
mcs_large <- mcs_mc(iterations, beta_0, beta_1, beta_2,
                    x_1_large, x_2_large, reg_func, "Stark")
mcs_full <- bind_rows(list(mcs_small, mcs_mid, mcs_large))
```

```
b0_mc <- ggplot(data = mcs_full,
                mapping = aes(x=b0, color=id, fill=id)) +
  geom_density(alpha=0.5) +
```

```
  scale_y_continuous(expand = expansion(c(0, 0), c(0, 0.05))) +
  ylab(TeX("Dichte der Schätzer")) +
  xlab(TeX("Wert für $\\hat{\\beta}_0$")) +
  ggtitle(TeX("Schätzer für $\\hat{\\beta}_0$")) +
  theme_icae() +
  scale_fill_viridis_d() +
  scale_color_viridis_d() +
  theme(axis.title = element_text(size=11),
        plot.title = element_text(size=12))

b1_mc <- ggplot(data = mcs_full,
                mapping = aes(x=b1, color=id, fill=id)) +
  geom_density(alpha=0.5) +
  scale_y_continuous(expand = expansion(c(0, 0), c(0, 0.05))) +
  ylab(TeX("Dichte der Schätzer")) +
  xlab(TeX("Wert für $\\hat{\\beta}_1$")) +
  ggtitle(TeX("Schätzer für $\\hat{\\beta}_1$")) +
  theme_icae() +
  scale_fill_viridis_d() +
  scale_color_viridis_d() +
  theme(axis.title = element_text(size=11),
        plot.title = element_text(size=12))

b2_mc <- ggplot(data = mcs_full,
                mapping = aes(x=b2, color=id, fill=id)) +
  geom_density(alpha=0.5) +
  scale_y_continuous(expand = expansion(c(0, 0), c(0, 0.05))) +
  ylab(TeX("Dichte der Schätzer")) +
  xlab(TeX("Wert für $\\hat{\\beta}_2$")) +
  ggtitle(TeX("Schätzer für $\\hat{\\beta}_2$")) +
  theme_icae() +
  scale_fill_viridis_d() +
  scale_color_viridis_d() +
  theme(axis.title = element_text(size=11),
        plot.title = element_text(size=12))

full_plot <- ggarrange(b0_mc, b1_mc, b2_mc,
                       ncol = 3, common.legend = T,
                       legend = "bottom")
```

```
ggsave(plot = full_plot,
       filename = here("figures/T10/13_multicol.pdf"),
       width = 6, height = 4)

full_plot
```



6 Exogenität

```
dgp_ovb <- function(x_1, x_2, beta_0, beta_1, beta_2){
  y <- rep(NA, length(x_1))
  errors <- rnorm(length(x_1), mean = 0, sd = 1)
  for (i in 1:length(y)){
    y[i] <- beta_0 + beta_1*x_1[i] + beta_2*x_2[i] + errors[i]
  }
  data_full <- tibble("y"=y, "x_1"=x_1, "x_2"=x_2)
  return(data_full)
}

mcs_ovb_fct <- function(iterations, beta_0, beta_1, beta_2,
                        x_1, x_2, reg_func){
  beta_0_estimates <- rep(NA, iterations)
  beta_1_estimates <- rep(NA, iterations)
  for (i in 1:iterations){
    data_used <- dgp_ovb(x_1, x_2, beta_0, beta_1, beta_2)
    regression <- lm(formula = reg_func, data = data_used)
    beta_0_estimates[i] <- regression$coefficients[1]
    beta_1_estimates[i] <- regression$coefficients[2]
  }
  result_tibble <- tibble("b0"=beta_0_estimates,
                          "b1"=beta_1_estimates)
  return(result_tibble)
}
```

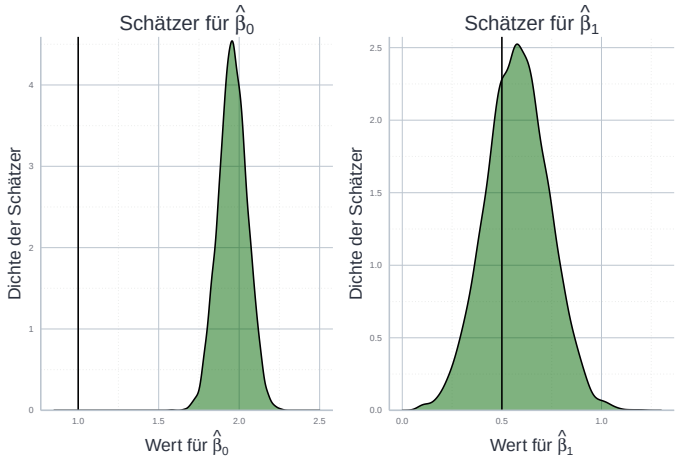
```
set.seed("123")
iterations <- 5000
beta_0 <- 1
beta_1 <- 0.5
beta_2 <- 2
x_1 <- runif(500)
x_2 <- runif(500)
reg_func <- as.formula("y~x_1")
```

```
b0_mc <- ggplot(data = mcs_ovb,
                mapping = aes(x=b0)) +
```

```
  geom_density(alpha=0.5, fill="#006600") +
  geom_vline(xintercept = beta_0) +
  scale_y_continuous(expand = expand_scale(c(0, 0), c(0, 0.05))) +
  scale_x_continuous(limits = c(0.85, 2.5)) +
  ylab(TeX("Dichte der Schätzer")) +
  xlab(TeX("Wert für $\\hat{\\beta}_0$")) +
  ggtitle(TeX("Schätzer für $\\hat{\\beta}_0$")) +
  scale_color_icae(palette = "mixed") +
  theme_icae() +
  theme(axis.title = element_text(size=12),
        plot.title = element_text(size=14))

b1_mc <- ggplot(data = mcs_ovb,
                mapping = aes(x=b1)) +
  geom_density(alpha=0.5, fill="#006600") +
  geom_vline(xintercept = beta_1) +
  scale_y_continuous(expand = expand_scale(c(0, 0), c(0, 0.05))) +
  scale_x_continuous(limits = c(0.0, 1.3)) +
  ylab(TeX("Dichte der Schätzer")) +
  xlab(TeX("Wert für $\\hat{\\beta}_1$")) +
  ggtitle(TeX("Schätzer für $\\hat{\\beta}_1$")) +
  scale_color_icae(palette = "mixed") +
  theme_icae() +
  theme(axis.title = element_text(size=12),
        plot.title = element_text(size=14))
```

```
full_plot <- ggarrange(b0_mc, b1_mc,
                       ncol = 2, common.legend = T,
                       legend = "none")
```

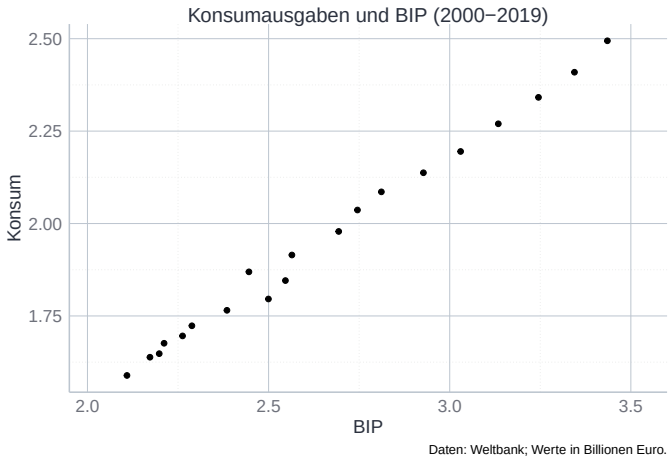
```
full_plot
```




```
ggsave(plot = full_plot,
       filename = here("figures/T10/14_hetero-mcs-b1.pdf"),
       width = 5, height = 3)
```

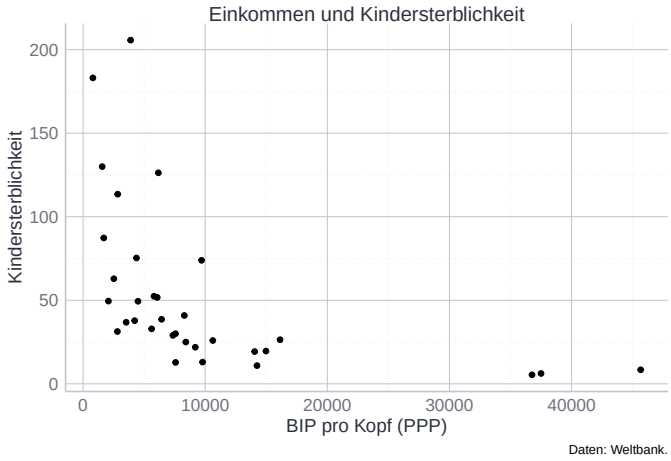
7 Linearer Zusammenhang

```
bipkonsum <- fread(here("data/T5/bip_einleitung.csv"))

bip_plot <- ggplot(
  data = bipkonsum, mapping = aes(x=BIP, y=Konsum)
  ) +
  geom_point() +
  scale_x_continuous(
    limits = c(1.95, 3.6),
    expand = expansion(c(0, 0)),
    breaks = seq(2.0, 4.0, by = 0.5)) +
  labs(
    title = "Konsumausgaben und BIP (2000-2019)",
    caption = "Daten: Weltbank; Werte in Billionen Euro.") +
  theme_icae() +
  theme(
    axis.title = element_text(size=13),
    plot.title = element_text(size=14),
    axis.text = element_text(size=12)
  )

ggsave(plot = bip_plot,
       filename = here("figures/T10/15_linmodel-bipcons.pdf"),
       width = 5, height = 4)

bip_plot
```

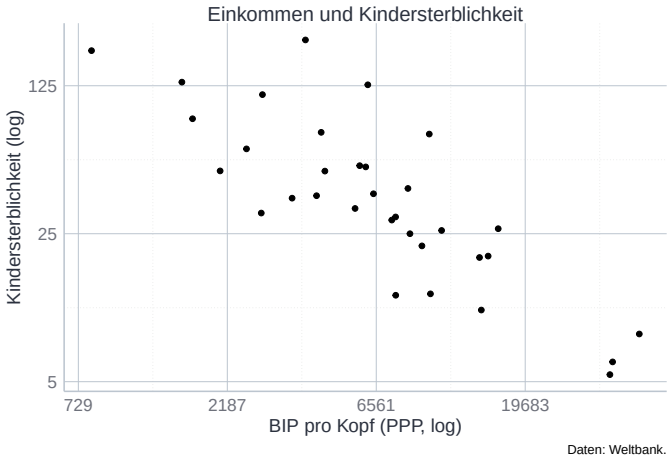



```
wb_data <- fread(here("data/T10/wb_bip_ineq_mort.csv"),
                 colClasses = c(rep("character", 2),
                                rep("double", 5))) %>%
  filter(year==2000)

kindersterblichkeit <- ggplot(
  data = wb_data, aes(x=GDP_PPPpc, y=MORTRATE)) +
  xlab("BIP pro Kopf (PPP)") + ylab("Kindersterblichkeit") +
  geom_point() +
    labs(
    title = "Einkommen und Kindersterblichkeit",
```

```
    caption = "Daten: Weltbank.") +
  theme_icae() +
  theme(
    axis.title = element_text(size=13),
    plot.title = element_text(size=14),
    axis.text = element_text(size=12)
  )

ggsave(plot = kindersterblichkeit,
       filename = here("figures/T10/16_linmodel-mortrate.pdf"),
       width = 5, height = 4)

kindersterblichkeit
```




```
kindersterblichkeit_log <- kindersterblichkeit +
  scale_x_continuous(
    trans = "log", breaks = scales::trans_breaks("log", function(x) 3^x)
    ) +
  scale_y_continuous(
    trans = "log", breaks = scales::trans_breaks("log", function(x) 5^x)
    ) +
  labs(
    y = "Kindersterblichkeit (log)", x = "BIP pro Kopf (PPP, log)",
    caption = "Daten: Weltbank.")

ggsave(plot = kindersterblichkeit_log,
       filename = here("figures/T10/17_linmodel-mortrate-log.pdf"),
       width = 5, height = 4)

kindersterblichkeit_log
```


Einkommen und Kindersterblichkeit
Kindersterblichkeit (log)
125
25
5
729
2187
6561
19683
BIP pro Kopf (PPP, log)
Daten: Weltbank.