Longitudinale Unterschiede

... wo und ab wann können wir von abgesicherten Differenzen ausgehen?

Holger Sennhenn-Reulen
Department of Growth and Yield,
Northwest German Forest Research Institute.

March 4, 2021

Contents

1 Organisiere R Session

```
rm(list = ls())
library("viridis")
library("lme4")
```

2 Intro

In der Regressionsanalyse geht es allgemein darum, die Auswirkungen von einer oder mehreren Einflußgrößen $x_1, x_2, \ldots$ auf abhängige Zufallsvariable Y abzuschätzen. Im Fall der linearen Einfachregression wird dies über die Beziehung:

$$Y_i = \beta_0 + \beta_1 x1, i + \epsilon_i \tag{1}$$

vollzogen, wobei Index i die Zugehörigkeit zu Beobachtungseinheit i bezeichet, Parameter β_0 den Wert der Regressionsgerade $\beta_0 + \beta_1 x1, i$ für $x_{1,i} = 0$ bezeichnet, Parameter β_1 die Veränderung dieser Regressionsgeraden wenn sich $x_{1,i}$ um eine Einheit vergrößert, und ϵ_i ist ein sogenannter Residualterm, an welchen wir in der linearen Einfachregression die Annahme stellen dass dieser einer Normalverteilung folgt, sowie dass alle ϵ_i derselben Normalverteilung abstammen und unabhängig daraus resultieren, kurz: ϵ_i wird angenommen als unabhängig und identisch verteilt bezüglich einer Normalverteilung mit Erwwartungswert 0 und Varianz σ^2, oder noch kürzer:

$$\epsilon_i \overset{\text{u.i.v}}{\sim} \mathrm{N}\left(0, \sigma^2\right) \tag{2}$$

Haben wir nun Daten aus zwei Gruppen – kodiert über Variable $k_i \in \{A, B\}$, so verschiebt sich im einfachsten Fall solch einer Erweiterung für eine der beiden Gruppen – hier B – der Intercept:

$$Y_i = \beta_0 + \beta_1 \mathrm{I}_{\{k_i = B\}} + \beta_2 x1, i + \epsilon_i. \tag{3}$$

Hier bezeichnet die Funktion I die Indikatorfunktion:

$$\mathrm{I}_{\{\text{Bedingung}\}} = \begin{cases} 1, & \text{wenn Bedingung wahr,} \\ 0, & \text{wenn Bedingung nicht wahr.} \end{cases} \tag{4}$$

Weiterhin wird die Gruppe A als Referenzkategorie bezeichnet, der entsprechende bedingte Erwartungswert wird durch den Intercept modelliert:

$$\mathrm{E}\left(Y_i \mid x_i, k_i = A\right) = \beta_0 + \beta_2 x_i, \tag{5}$$

$$\mathrm{E}\left(Y_i \mid x_i, k_i = B\right) = \beta_0 + \beta_1 + \beta_2 x_i. \tag{6}$$

3 Einfaches Beispiel

Für ein erstes Beispiel simulieren wir $N = 100$, $i = 1, \ldots, N$, Werte aus der Normalverteilung mit Erwartungswert 0 und Varianz 0.5^2:

```
N <- 100
set.seed(123456789) ## Setzen eines Startpunktes zur Reproduktion der Simulation.
epsilon <- rnorm(n = N, mean = 0, sd = .5)
round(epsilon, 5)
```

```
 [1]  0.25244  0.19794  0.70777 -0.36116 -0.30918 -0.78131  0.06398 -0.07848
 [9] -0.75767  0.58080 -0.53083  0.52629 -0.54516 -0.47522  0.09445 -0.65346
[17] -0.54648  0.58371  0.59906 -0.98130  0.36336  0.53352 -0.90729  0.00440
[25] -0.16350  0.28906  0.60977 -0.14493 -0.63902 -0.24344  0.08326 -0.90369
[33] -0.47965  0.53516  0.24537  0.09182 -0.26552  0.37123 -1.13976  0.09065
[41]  0.01786 -0.24898  0.13929  0.32246  0.72137 -0.13461 -0.11391 -0.38229
[49]  0.03215 -0.17116  0.07269  0.10068  0.70621 -0.38932  0.38819 -0.88341
[57] -0.18524  0.72655 -1.04292  0.74310 -0.15237  0.18729 -0.09056 -0.20545
[65] -0.38990  0.31783  0.43762  0.99488  0.49443 -0.23854 -0.24152  0.23917
[73]  0.42379  0.40961 -0.37880 -0.32585 -0.40989  0.15723 -0.32140  0.23123
[81]  0.61434  0.38402  0.91124 -0.00792  0.59688  0.58721 -0.46694  0.02266
[89]  0.29326 -0.10635  0.31797  0.23336 -0.40267  0.56444 -0.37608  0.20280
[97]  0.35920  0.65577 -0.73622  0.13174
```

Wir bilden die Werte einer Einflussgröße x als eine Sequenz der Länge 50 mit gleichen Abständen zwischen -1 und 1:

```
x <- seq(-1, 1, length.out = N / 2)
x <- c(x, x)
```

Gruppierungsvariable:

```
k <- rep(LETTERS[1:2], each = N / 2)
```

Für den Parameter β_0 nehmen wir den Wert 0.75 an, für den Parameter β_1 den Wert -1, für β_2 den Wert 0.5:

```
eta <- .75 + -1 * (k == "B") + .5 * x
```

Wir nennen diese Größe η, welche hier die Werte der Regressionsgeraden an den Werten von x abbildet, allgemein auch den 'linearen Prädiktor':

$$\eta_i = \beta_0 + \beta_1 x_i$$

In der linearen Regression werden nun die Werte y der abhängigen Variablen Y ohne (in der Regel) eine weitere Transformation als Addition von linearem Prädiktor und Residualterm gebildet:

```
y <- eta + epsilon
```

Durch diese Form der Addition, sowie der Eigenschaft, dass wir für ϵ den Erwartungswert 0 angenommen haben, ergibt sich für den Erwartungswert von Y in Abhängigkeit (man sagt 'bedingt auf') von x:

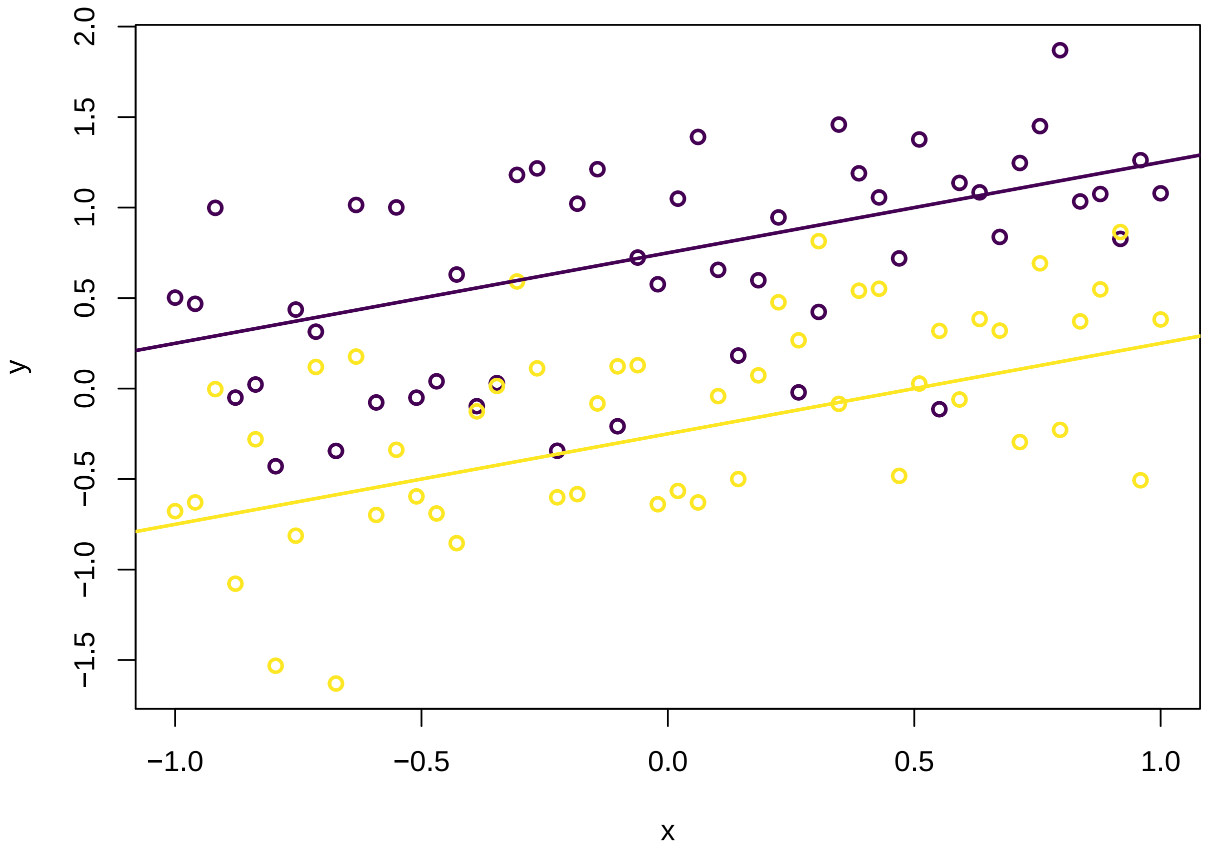
$$\mathrm{E}\left(Y_i \mid x_i\right) = \eta_i + 0 = \eta_i.$$

Wir sprechen hier vom bedingten Erwartungswert von Y, und erhalten weiterhin für die volle Verteilung von Y:

$$Y \sim \text{Normal}\left(\eta_i, \sigma^2\right).$$

Wir stellen dies einmal grafisch dar:

```
plot(x, y, col = viridis::viridis(n = 2)[1 + (k == "B")], lwd = 2)
abline(a = .75, b = .5, col = viridis::viridis(n = 2)[1], lwd = 2)
abline(a = .75 - 1, b = .5, col = viridis::viridis(n = 2)[2], lwd = 2)
```




```
m <- lm(y ~ k + x)
coef(m)
```

```
(Intercept)          kB           x
  0.6719836  -0.8189024   0.5385230
```

```
library("arm")
post_samples <- sim(object = m, n.sims = 1e4)
str(post_samples)
```

```
Formal class 'sim' [package "arm"] with 2 slots
  ..@ coef : num [1:10000, 1:3] 0.673 0.606 0.617 0.558 0.645 ...
  .. ..- attr(*, "dimnames")=List of 2
  .. .. ..$ : NULL
  .. .. ..$ : chr [1:3] "(Intercept)" "kB" "x"
  ..@ sigma: num [1:10000] 0.463 0.45 0.48 0.455 0.482 ...
```

```
head(post_samples@coef)
```

```
     (Intercept)         kB         x
[1,]   0.6732085 -0.8297837 0.6233428
[2,]   0.6060067 -0.7728115 0.5675096
[3,]   0.6168014 -0.7001374 0.4905466
[4,]   0.5577814 -0.7837495 0.4799872
[5,]   0.6452359 -0.6445330 0.5594555
[6,]   0.7325724 -0.8367106 0.5569847
```

```
post_samples@coef[1:10, '(Intercept)']
```

```
 [1] 0.6732085 0.6060067 0.6168014 0.5577814 0.6452359 0.7325724 0.6804811
 [8] 0.8022985 0.6398304 0.6647543
```

```
post_samples@coef[1:10, '(Intercept)'] + post_samples@coef[1:10, 'kB']
```

```
 [1] -0.1565752672 -0.1668048024 -0.0833359483 -0.2259681169  0.0007028875
 [6] -0.1041381942 -0.2073154596 -0.0916235619 -0.1611902252 -0.1352993161
```

```
x_seq <- seq(-1, 1, by = .05)
j <- 1
post_samples@coef[1:10, '(Intercept)'] + x_seq[j] * post_samples@coef[1:10, 'x']
```

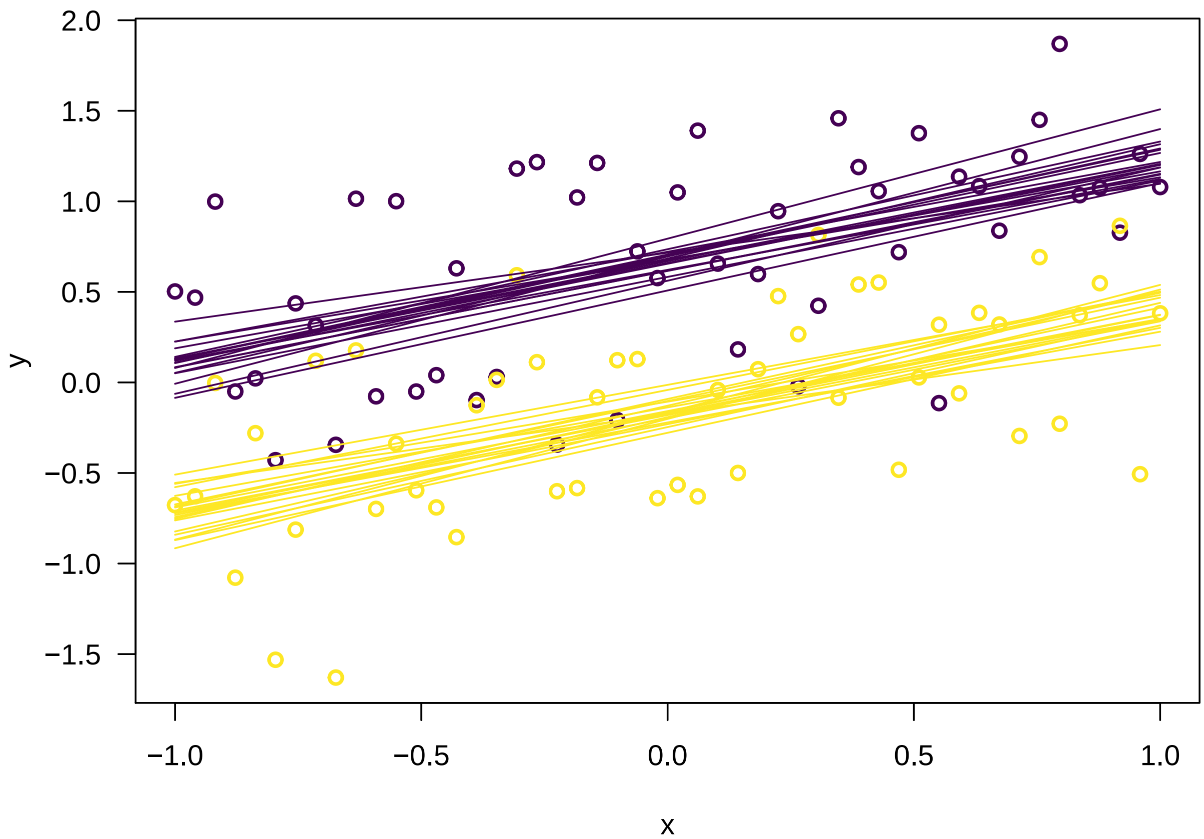
```
 [1] 0.04986563 0.03849712 0.12625487 0.07779419 0.08578033 0.17558765
 [7] 0.06501920 0.29433745 0.16150450 0.19740162
```

```
post_samples@coef[1:10, '(Intercept)'] + post_samples@coef[1:10, 'kB'] + x_seq[j] * post_samples@coef[1:10, 'x']
```

```
 [1] -0.7799181 -0.7343144 -0.5738825 -0.7059554 -0.5587526 -0.6611229
 [7] -0.8227773 -0.5995846 -0.6395162 -0.6026520
```

```
eta_A <- eta_B <- matrix(ncol = length(x_seq), nrow = nrow(post_samples@coef), NA)
for (j in 1:length(x_seq)) {
  eta_A[, j] <- post_samples@coef[, '(Intercept)'] + x_seq[j] * post_samples@coef[, 'x']
  eta_B[, j] <- post_samples@coef[, '(Intercept)'] + post_samples@coef[, 'kB'] + x_seq[j] * post_samples@coef[, 'x']
}
```

```
plot(x, y, col = viridis::viridis(n = 2)[1 + (k == "B")], lwd = 2, las = 1)
set.seed(123456789)
S <- sort(sample(1:nrow(post_samples@coef))[1:20])
for (s in S) {
  lines(x_seq, eta_A[s, ], col = viridis::viridis(n = 2)[1])
  lines(x_seq, eta_B[s, ], col = viridis::viridis(n = 2)[2])
}
```

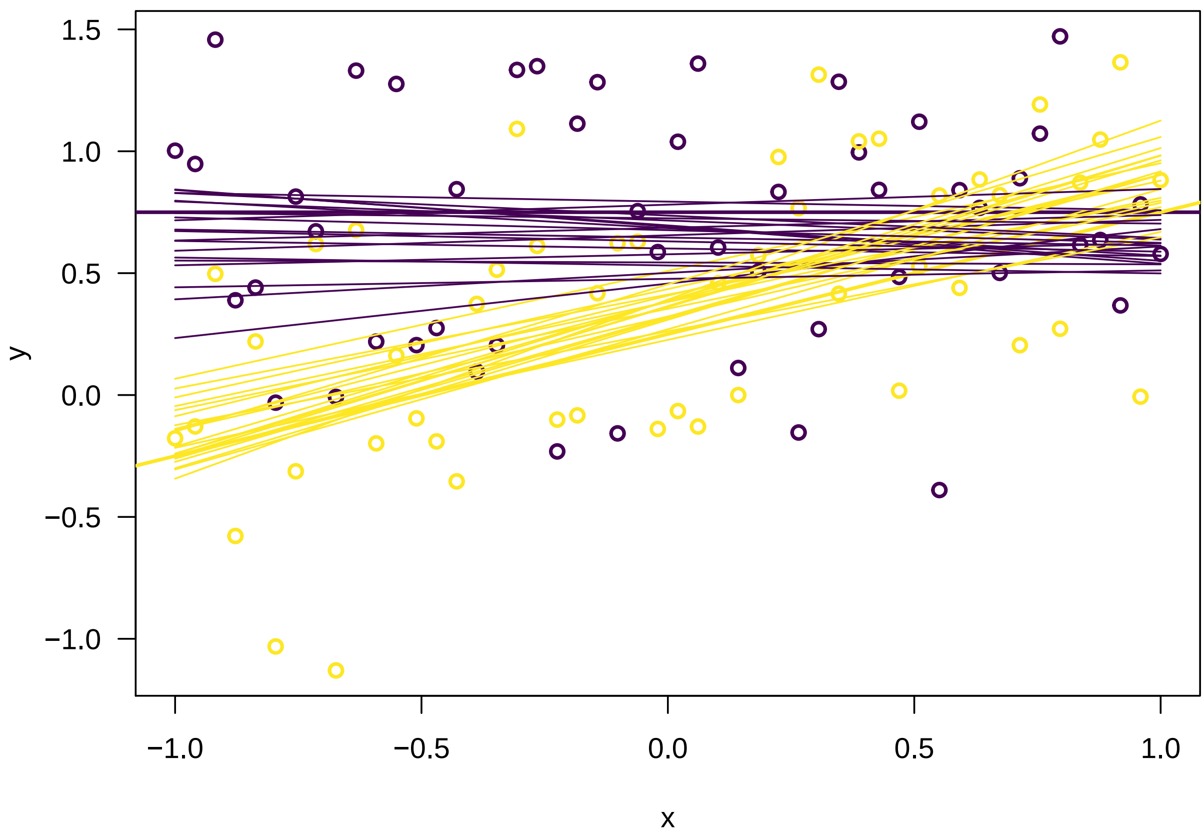


4 Etwas komplizierter: Mit Interaktionsterm

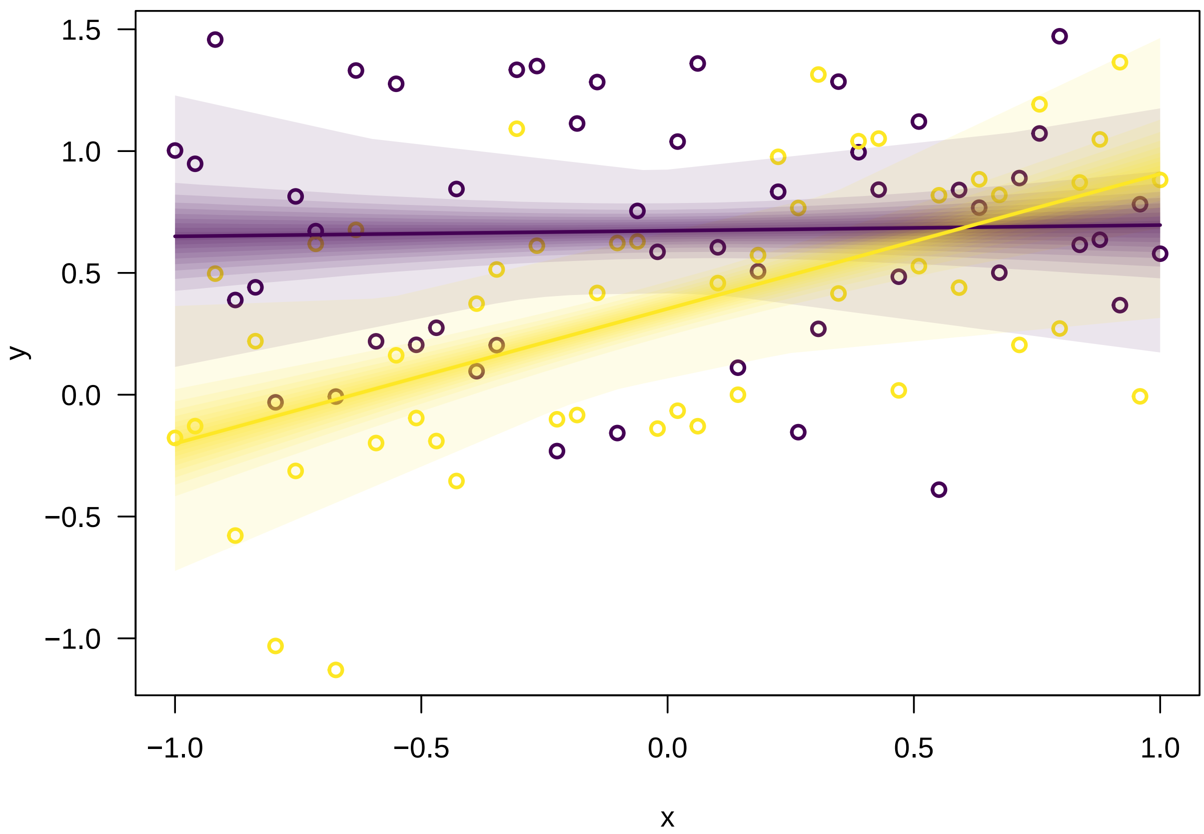
```
eta <- .75 + -.5 * (k == "B") + .5 * x * (k == "B")
y <- eta + epsilon
plot(x, y, col = viridis::viridis(n = 2)[1 + (k == "B")], lwd = 2, las = 1)
abline(a = .75, b = 0, col = viridis::viridis(n = 2)[1], lwd = 2)
abline(a = .75 - .5, b = .5, col = viridis::viridis(n = 2)[2], lwd = 2)
m <- lm(y ~ k * x)
coef(m)
```

```
(Intercept)          kB           x        kB:x
 0.67198361 -0.31890244  0.02368501  0.52967607
```

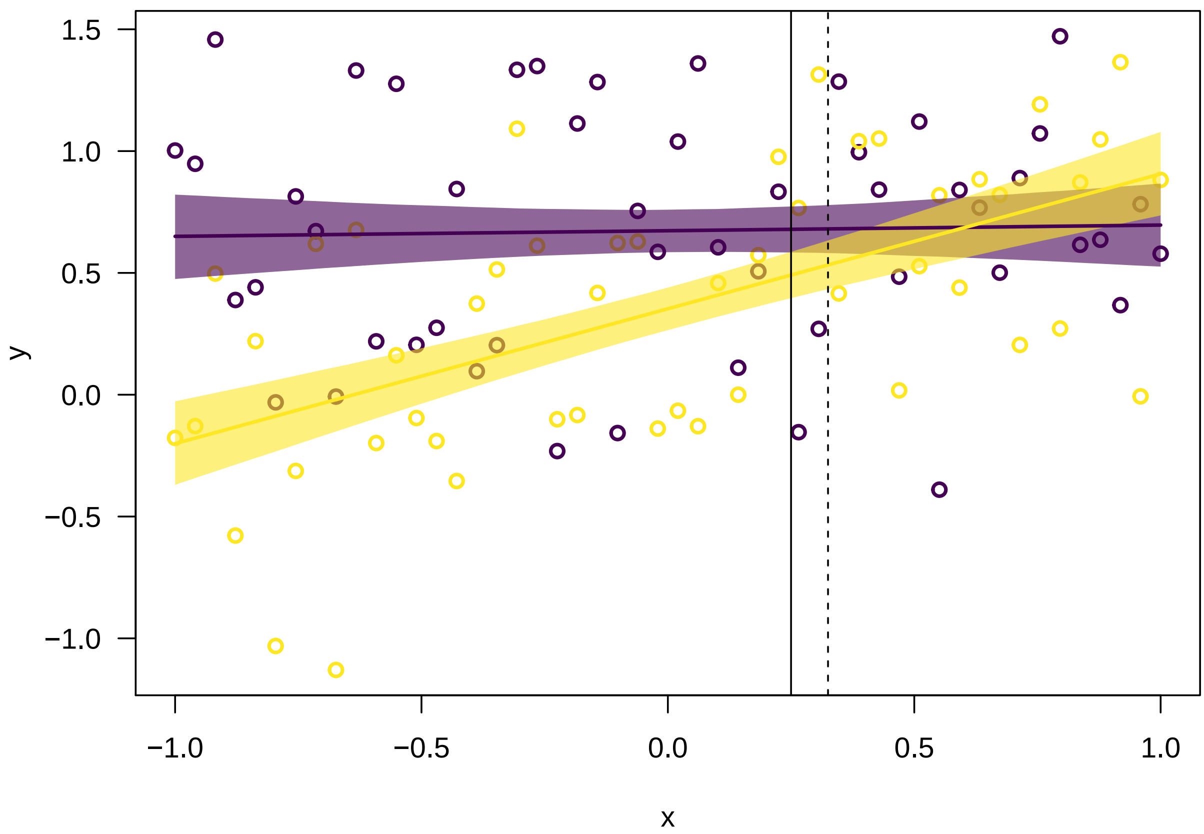
```
post_samples <- sim(object = m, n.sims = 1e4)
eta_A <- eta_B <- matrix(ncol = length(x_seq), nrow = nrow(post_samples@coef), NA)
for (j in 1:length(x_seq)) {
  eta_A[, j] <- post_samples@coef[, '(Intercept)'] + x_seq[j] * post_samples@coef[, 'x']
  eta_B[, j] <- post_samples@coef[, '(Intercept)'] + post_samples@coef[, 'kB'] +
    x_seq[j] * (post_samples@coef[, 'x'] + post_samples@coef[, 'kB:x'])
}
set.seed(123456789)
S <- sort(sample(1:nrow(post_samples@coef))[1:20])
for (s in S) {
  lines(x_seq, eta_A[s, ], col = viridis::viridis(n = 2)[1])
  lines(x_seq, eta_B[s, ], col = viridis::viridis(n = 2)[2])
}
```



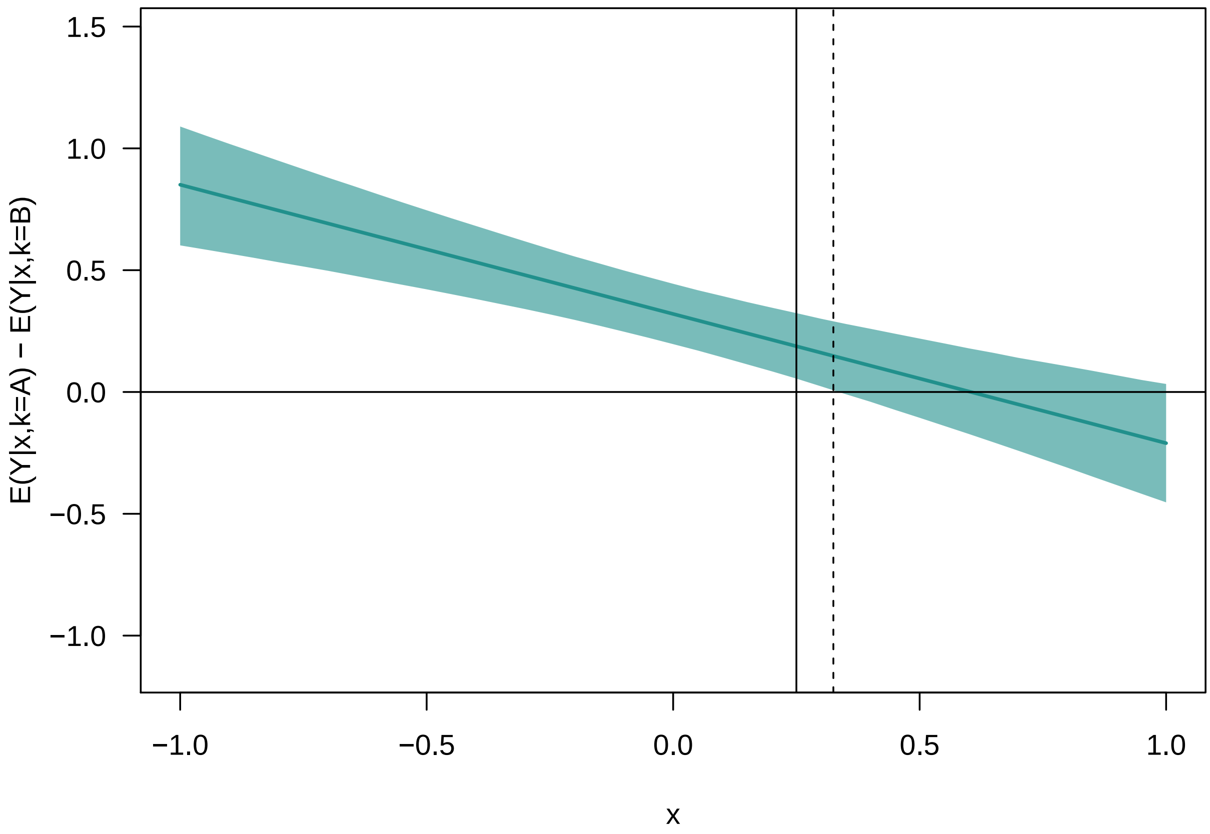
```
plot(x, y, col = viridis::viridis(n = 2)[1 + (k == "B")], lwd = 2, las = 1)
for (p in seq(0, .45, by = .05)) {
  l_A <- apply(eta_A, MAR = 2, FUN = quantile, probs = p)
  u_A <- apply(eta_A, MAR = 2, FUN = quantile, probs = 1 - p)
  polygon(c(x_seq, rev(x_seq)), c(l_A, rev(u_A)), col = viridis::viridis(n = 2, alpha = .1)[1], border = NA)
  l_B <- apply(eta_B, MAR = 2, FUN = quantile, probs = p)
  u_B <- apply(eta_B, MAR = 2, FUN = quantile, probs = 1 - p)
  polygon(c(x_seq, rev(x_seq)), c(l_B, rev(u_B)), col = viridis::viridis(n = 2, alpha = .1)[2], border = NA)

}
lines(x_seq, apply(eta_A, MAR = 2, FUN = mean), col = viridis::viridis(n = 2)[1], lwd = 2)
lines(x_seq, apply(eta_B, MAR = 2, FUN = mean), col = viridis::viridis(n = 2)[2], lwd = 2)
```



```
plot(x, y, col = viridis::viridis(n = 2)[1 + (k == "B")], lwd = 2, las = 1)
p <- .9
l_A <- apply(eta_A, MAR = 2, FUN = quantile, probs = p)
u_A <- apply(eta_A, MAR = 2, FUN = quantile, probs = 1 - p)
polygon(c(x_seq, rev(x_seq)), c(l_A, rev(u_A)), col = viridis::viridis(n = 2, alpha = .6)[1], border = NA)
l_B <- apply(eta_B, MAR = 2, FUN = quantile, probs = p)
u_B <- apply(eta_B, MAR = 2, FUN = quantile, probs = 1 - p)
polygon(c(x_seq, rev(x_seq)), c(l_B, rev(u_B)), col = viridis::viridis(n = 2, alpha = .6)[2], border = NA)
lines(x_seq, apply(eta_A, MAR = 2, FUN = mean), col = viridis::viridis(n = 2)[1], lwd = 2)
lines(x_seq, apply(eta_B, MAR = 2, FUN = mean), col = viridis::viridis(n = 2)[2], lwd = 2)
abline(v = 0.25)
abline(v = 0.325, lty = 2)
```



```
plot(x, y, type = "n", las = 1, ylab = "E(Y|x,k=A) - E(Y|x,k=B)")
p <- .9
l <- apply(eta_A - eta_B, MAR = 2, FUN = quantile, probs = p)
u <- apply(eta_A - eta_B, MAR = 2, FUN = quantile, probs = 1 - p)
polygon(c(x_seq, rev(x_seq)), c(l, rev(u)), col = viridis::viridis(n = 3, alpha = .6)[2], border = NA)
lines(x_seq, apply(eta_A - eta_B, MAR = 2, FUN = mean), col = viridis::viridis(n = 3)[2], lwd = 2)
abline(h = 0)
abline(v = 0.25)
abline(v = 0.325, lty = 2)
```



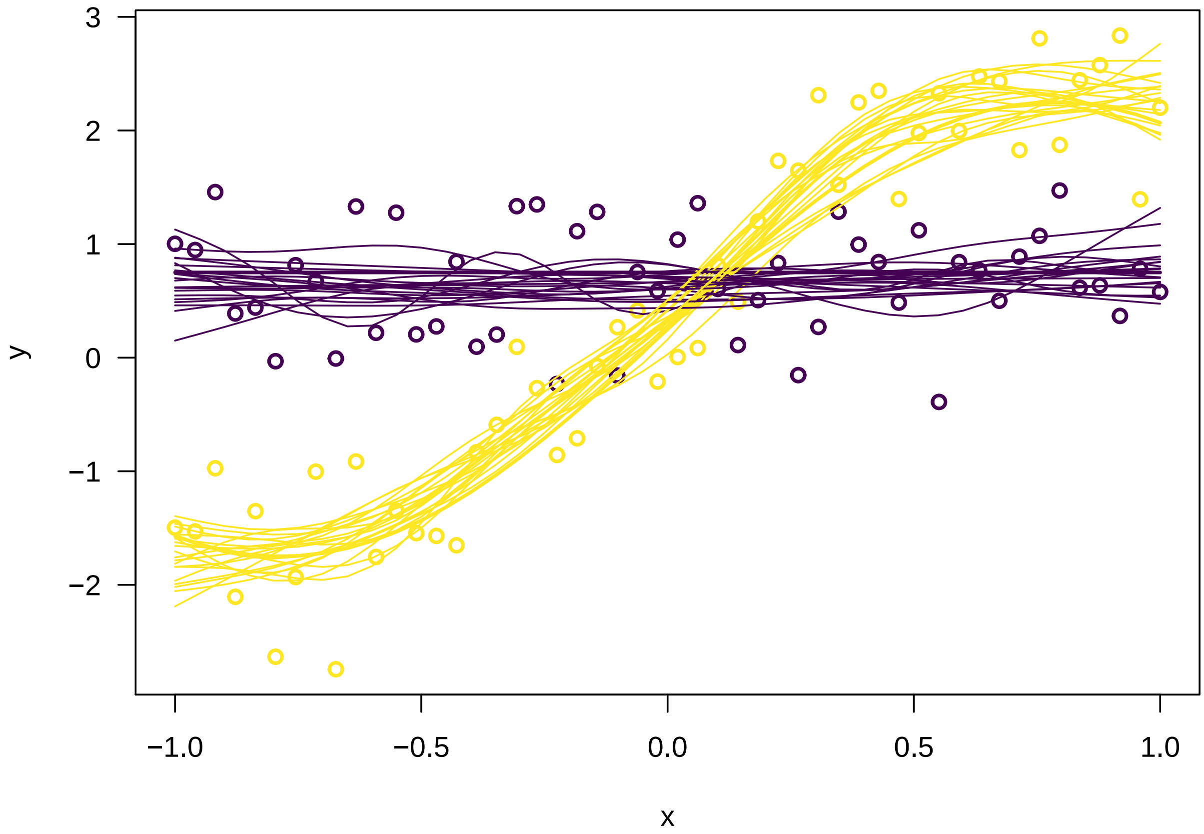
5 Strukturiert Additives Modell

```
eta <- .75 + -.5 * (k == "B") + 2 * sin(2 * x) * (k == "B")
y <- eta + epsilon
df <- data.frame(y = y, x = x, k = k)
```

```
library("brms")
m <- brm(y ~ k + s(x, by = k), data = df, chains = 4, iter = 1000, seed = 123456789)
```

```
eta_A <- fitted(m, newdata = data.frame(x = x_seq,
                                        k = factor("A", levels = levels(df$k))),
                summary = FALSE)
eta_B <- fitted(m, newdata = data.frame(x = x_seq,
                                        k = factor("B", levels = levels(df$k))),
                summary = FALSE)
```

```
plot(x, y, col = viridis::viridis(n = 2)[1 + (k == "B")], lwd = 2, las = 1)
lines(x_seq, .75 + 0 * x_seq, col = viridis::viridis(n = 2)[1], lwd = 2)
lines(x_seq, .75 - .5 + 2 * sin(2 * x_seq), col = viridis::viridis(n = 2)[2], lwd = 2)
set.seed(123456789)
S <- sort(sample(1:nrow(eta_A))[1:20])
for (s in S) {
  lines(x_seq, eta_A[s, ], col = viridis::viridis(n = 2)[1])
  lines(x_seq, eta_B[s, ], col = viridis::viridis(n = 2)[2])
}
```



```
plot(x, y, type = "n", las = 1, ylab = "E(Y|x,k=A) - E(Y|x,k=B)")
p <- .9
l <- apply(eta_A - eta_B, MAR = 2, FUN = quantile, probs = p)
u <- apply(eta_A - eta_B, MAR = 2, FUN = quantile, probs = 1 - p)
polygon(c(x_seq, rev(x_seq)), c(l, rev(u)), col = viridis::viridis(n = 3, alpha = .6)[2], border = NA)
lines(x_seq, apply(eta_A - eta_B, MAR = 2, FUN = mean), col = viridis::viridis(n = 3)[2], lwd = 2)
abline(h = 0)
```