A8. Adjustierte Mittelwerte vs.~arithmetische Mittelwerte

Warum modellbasierte (geschätzte marginale) Mittelwerte von einfachen Durchschnitten abweichen können - und warum man ihnen vertrauen sollte

Dr. Paul Schmidt

Um alle in diesem Kapitel verwendeten Pakete zu installieren und zu laden, kann man folgenden Code ausführen:

```
for (pkg in c("emmeans", "ggtext", "multcomp", "multcompView", "tidyverse")) {
  if (!require(pkg, character.only = TRUE)) install.packages(pkg)
}

library(emmeans)
library(ggtext)
library(multcomp)
library(multcompView)
library(tidyverse) # zuletzt geladen, damit dplyrs select()/filter() gewinnen
```

In den vorherigen Kapiteln haben wir `emmeans()` verwendet, um einen Mittelwert je Behandlung zu erhalten, und mehr als einmal beiläufig erwähnt, dass diese modellbasierten Mittelwerte nicht notwendigerweise dasselbe sind wie ein schlichter arithmetischer Durchschnitt. Dieses Kapitel betrachtet diese Bemerkung nun genauer: Wann stimmen die beiden überein, wann gehen sie auseinander, und wofür genau “adjustiert” das Modell?

Ein modellbasierter oder *adjustierter* Mittelwert wird aus einem gefitteten Modell rekonstruiert, statt direkt aus den Rohbeobachtungen berechnet zu werden. Er beantwortet daher eine etwas andere Frage als ein Rohdurchschnitt: nicht “wie hoch war der Durchschnitt der Parzellen, die diese Behandlung zufällig erhielt?”, sondern “wie hoch wäre der Mittelwert dieser Behandlung unter einem fairen, balancierten Vergleich?”. Wenn das Design balanciert und das Modell einfach ist, sind die beiden Antworten identisch. Sobald das Design unbalanciert ist, eine Kovariate enthält oder eine nicht-triviale Varianz-Kovarianz-Struktur aufweist, können sie auseinandergehen - mal im Schätzwert selbst, mal nur in dessen Standardfehler. Wir werden beide Arten von Unterschied unten kennenlernen.

Glossar: Adjustierte Mittelwerte

In diesem Kapitel ist *adjustierte Mittelwerte* die Kurzform für Mittelwerte, die aus einem gefitteten Modell geschätzt und nicht direkt aus den Rohbeobachtungen berechnet werden. Je nach Software oder Lehrbuch werden sie auch genannt:

- **geschätzte marginale Mittelwerte** (der Name, der vom Paket `{emmeans}` verwendet wird),
- **Least-Squares-Mittelwerte** (historisch `lsmeans` in SAS und älteren R-Paketen),
- **modellbasierte** oder **vorhergesagte Mittelwerte**.

Sie alle bezeichnen dieselbe Idee: einen aus den Koeffizienten des Modells rekonstruierten Mittelwert, der daher für die übrigen Terme im Modell (Blöcke, Kovariaten, …) korrigiert.

Die Kurzversion

Ein schlichtes arithmetisches Mittel mittelt genau die Beobachtungen, die eine Sorte erhalten hat. Wenn eine Sorte zufällig nur unter guten Bedingungen getestet wurde (oder, wie unten, den besten Block ganz verpasst hat), sickert dieses Glück direkt in ihren Mittelwert. Ein adjustierter Mittelwert fragt stattdessen: *Wie hoch wäre der Mittelwert dieser Sorte, wenn jede Sorte denselben Satz von Bedingungen erlebt hätte?* Er beantwortet das, indem er mithilfe des gefitteten Modells über alle Blockstufen mittelt - sogar für Blöcke, in denen eine Sorte nie vorkam.

In einem perfekt balancierten Design sind die beiden identisch. In dem Moment, in dem das Design unbalanciert wird - fehlende Parzellen, ungleiche Wiederholung, Kovariaten -, gehen sie auseinander, und der adjustierte Mittelwert ist der faire Vergleich. Und es gibt einen zweiten, subtileren Unterschied: Selbst wenn die *Schätzwerte* exakt mit den Rohdurchschnitten übereinstimmen, können ihre *Standardfehler* abweichen, sobald das Modell eine Varianz-Kovarianz-Struktur trägt (zufällige Effekte, heterogene oder korrelierte Fehler). Darauf kommen wir am Ende zurück.

Ein motivierendes Beispiel: die Sorte, die den besten Block verpasste

Wir konstruieren einen kleinen Sortenversuch, angelegt als randomisierte vollständige Blockanlage (RCBD) mit vier Sorten (`A`, `B`, `C`, `D`) und vier Blöcken (`B1`-`B4`). Die Daten sind aus einer sauberen additiven Struktur aufgebaut - ein Sorteneffekt plus ein Blockeffekt plus etwas Rauschen -, sodass wir die Grundwahrheit kennen:

- Sorte `D` ist wirklich die **beste** (höchster Sorteneffekt), gefolgt von `C`, `B`, `A`.
- Block `B4` ist ein **“Super-Block”**: aus welchem Grund auch immer (eine feuchtere Ecke des Feldes, eine bessere Gewächshausbank) wächst dort alles weit besser.

Der Haken: Sorte `D` **fehlt im Super-Block** **`B4`**. Vielleicht sind diese Parzellen ausgefallen, oder das Saatgut von `D` ging aus. Diese eine fehlende Parzelle reicht aus, um die Balance zu brechen und die naive Analyse in die Irre zu führen.

```r
set.seed(42)

variety_effect <- c(A = 50, B = 55, C = 60, D = 65) # D ist tatsächlich die beste
block_effect   <- c(B1 = 0, B2 = 3, B3 = 6, B4 = 30) # B4 ist der Super-Block

dat <- expand_grid(
  variety = names(variety_effect),
  block   = names(block_effect)
) %>%
  mutate(
    yield = variety_effect[variety] + block_effect[block] + rnorm(n(), 0, 2),
    yield = round(yield, 1)
  ) %>%
  # Sorte D schaffte es nie in den Super-Block B4
  filter(!(variety == "D" & block == "B4")) %>%
  mutate(across(c(variety, block), as.factor))

dat
```

```
# A tibble: 15 × 3
   variety block yield
   <fct>   <fct> <dbl>
 1 A       B1     52.7
 2 A       B2     51.9
 3 A       B3     56.7
 4 A       B4     81.3
 5 B       B1     55.8
 6 B       B2     57.8
 7 B       B3     64
 8 B       B4     84.8
 9 C       B1     64
10 C       B2     62.9
11 C       B3     68.6
12 C       B4     94.6
13 D       B1     62.2
14 D       B2     67.4
15 D       B3     70.7
```

Wir haben 15 Parzellen statt der vollen 16: Jede Sorte kommt in `B1`, `B2` und `B3` vor, aber nur `A`, `B` und `C` kommen in `B4` vor.

Erkunden

Ein kurzer Blick auf die Blockmittelwerte bestätigt, dass `B4` in einer eigenen Liga spielt:

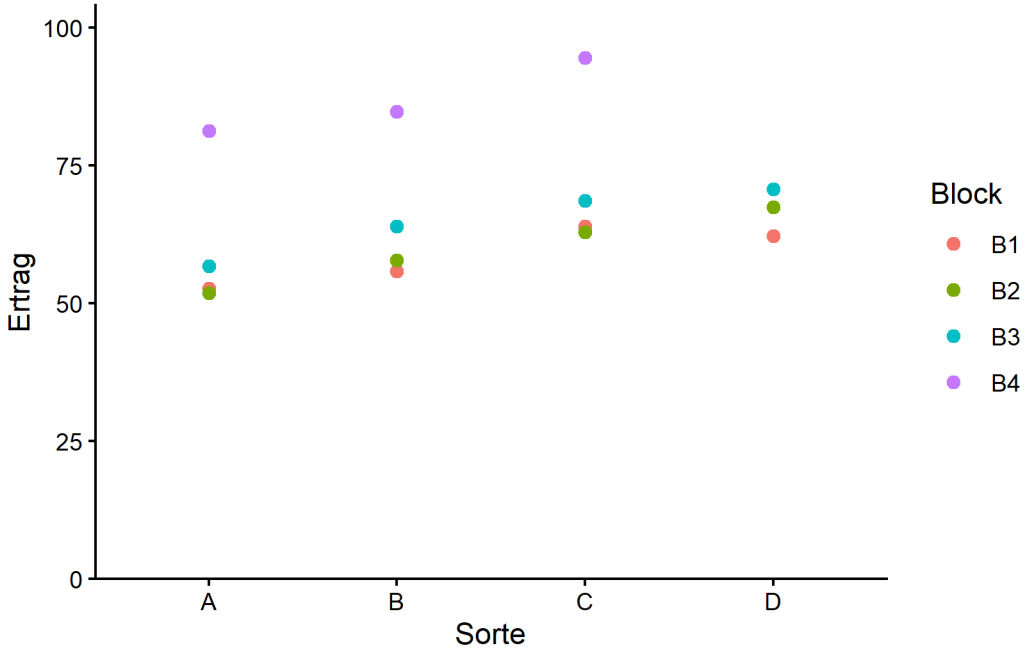
```r
dat %>%
  group_by(block) %>%
  summarise(mean_yield = mean(yield), n = n())
```

```
# A tibble: 4 × 3
  block mean_yield     n
  <fct>      <dbl> <int>
1 B1          58.7     4
2 B2          60       4
3 B3          65       4
4 B4          86.9     3
```

Block `B4` mittelt um die `87`, während die anderen drei nahe `60` liegen. Nun den Ertrag je Sorte auftragen, gefärbt nach Block:

```r
ggplot(data = dat) +
  aes(y = yield, x = variety, color = block) +
  geom_point(size = 2) +
```

```
  scale_x_discrete(name = "Sorte") +
  scale_y_continuous(
    name = "Ertrag",
    limits = c(0, NA),
    expand = expansion(mult = c(0, 0.1))
  ) +
  scale_color_discrete(name = "Block") +
  theme_classic()
```



Die Super-Block-Punkte (`B4`) liegen weit über dem Rest - und entscheidend: Sorte `D` hat überhaupt keinen `B4` -Punkt. Jede andere Sorte erhielt einen großen Schub von einer Super-Block-Parzelle; `D` nicht. Man behalte dieses Bild im Kopf: Es ist der ganze Grund, warum die beiden Arten von Mittelwert auseinandergehen werden.

Der naive Ansatz und warum er in die Irre führt

Das Natürlichste ist, den Ertrag innerhalb jeder Sorte zu mitteln:

```
naive <- dat %>%
  group_by(variety) %>%
  summarise(naive_mean = mean(yield), n = n()) %>%
  arrange(desc(naive_mean))

naive
```

```
# A tibble: 4 × 3
  variety naive_mean     n
  <fct>        <dbl> <int>
1 C             72.5     4
2 D             66.8     3
3 B             65.6     4
4 A             60.6     4
```

Wörtlich genommen sagt diese Tabelle, dass Sorte `C` **die beste ist** (etwa 72.5), wobei die wirklich beste Sorte `D` nur an zweiter Stelle steht. Diese Schlussfolgerung ist falsch, und der Grund ist struktureller statt zufälliger Natur:

- Die Sorten `A`, `B` und `C` mitteln jeweils über **vier** Parzellen, von denen eine eine Super-Block-Parzelle im Wert von etwa `+30` ist. Diese eine Parzelle zieht ihre Durchschnitte um etwa `30 / 4 = 7.5` nach oben.
- Sorte `D` mittelt über nur **drei** Parzellen, keine davon im Super-Block. Sie erhielt diesen Aufwärtsschub nie.

`D` wird im Ranking also nicht benachteiligt, weil sie schlechter wächst, sondern weil sie den Block verpasste, in dem alle gut aussahen. Das arithmetische Mittel kann das nicht wissen: Es mittelt einfach die vorhandenen Beobachtungen.

Adjustierte Mittelwerte schaffen Abhilfe

Nun fitten wir das RCBD-Modell, das den Blockeffekt berücksichtigt, und fragen `emmeans()` nach den Sortenmittelwerten:

```
mod <- lm(yield ~ variety + block, data = dat)
anova(mod)
```

```
Analysis of Variance Table

Response: yield
          Df  Sum Sq Mean Sq F value    Pr(>F)
variety    3  285.25   95.08   35.72 5.570e-05 ***
block      3 1793.33  597.78  224.56 4.646e-08 ***
Residuals  8   21.30    2.66
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Sowohl `variety` als auch `block` sind hochsignifikant - wenig überraschend, da wir beide Effekte in die Daten eingebaut haben. Mit dem Modell in der Hand erhalten wir die geschätzten marginalen Mittelwerte und eine kompakte Buchstabendarstellung in einem Schritt:

```
mean_comp <- mod %>%
  emmeans(specs = ~ variety) %>%       # adjustierter Mittelwert je Sorte
  cld(adjust = "none", Letters = letters) # kompakte Buchstabendarstellung

mean_comp
```

```
 variety emmean    SE df lower.CL upper.CL .group
 A         60.6 0.816  8     58.8     62.5  a
 B         65.6 0.816  8     63.7     67.5   b
 C         72.5 0.816  8     70.6     74.4    c
 D         73.6 0.980  8     71.4     75.9    c

Results are averaged over the levels of: block
Confidence level used: 0.95
significance level used: alpha = 0.05
NOTE: If two or more means share the same grouping symbol,
      then we cannot show them to be different.
      But we also did not show them to be the same.
```

Man beachte die Fußzeile: *Results are averaged over the levels of: block*. Das Ranking ist nun korrekt - Sorte `D` **liegt oben** (etwa `73.6`), knapp vor `C` (`72.5`). Das Modell hat erkannt, dass die drei Beobachtungen von `D` alle aus gewöhnlichen Blöcken stammen, und geschätzt, was `D` auch im Super-Block geliefert hätte.

Ein weiteres Detail ist beachtenswert: Der Standardfehler von `D` (`0.98`) ist etwas größer als die der anderen (`0.82`). Das ist ehrliche Buchführung - `D` beruht auf drei Parzellen statt vier und wurde in `B4` nie beobachtet, sodass das Modell etwas unsicherer darüber ist.

Was “adjustiert für Block” tatsächlich bedeutet

Der Schlüsselsatz aus der Ausgabe ist *averaged over the levels of block*. Ein geschätzter marginaler Mittelwert für eine Sorte wird berechnet, indem man:

1. die Vorhersage des gefitteten Modells für diese Sorte in **jeder** Blockstufe nimmt und dann
2. diese Vorhersagen mit **gleichem Gewicht** über alle vier Blöcke mittelt.

Für die Sorten `A`, `B` und `C` ändert das nichts gegenüber dem arithmetischen Mittel, weil sie wirklich einmal in jedem Block beobachtet wurden - gleiche Gewichtung und schlichtes Mitteln stimmen überein. Deshalb sind ihre naiven und adjustierten Mittelwerte hier identisch. (Diese exakte Übereinstimmung spiegelt wider, wie mild die Unbalance ist - eine einzige fehlende Zelle. Bei stärkerer Unbalance kann sich selbst eine in jedem Block vertretene Sorte leicht verschieben.)

Für Sorte `D` wurde Block `B4` nie beobachtet. Das additive Modell sagt den Ertrag von `D` in `B4` dennoch vorher als

$$\hat{y}_{D,B4} = \hat{\mu} + \hat{\tau}_D + \hat{\beta}_{B4},$$

wobei $\hat{\beta}_{B4}$ - der Super-Block-Bonus - aus den Sorten geschätzt wird, die *dort waren* (`A`, `B`, `C`). Der adjustierte Mittelwert für `D` mittelt dann über `B1` - `B4` einschließlich dieses rekonstruierten `B4`-Werts, weshalb er über jedem Ertrag liegt, den `D` tatsächlich produziert hat.

> ⚠ Dies beruht auf der Additivitätsannahme
>
> Das Auffüllen der fehlenden `B4`-Zelle funktioniert nur, weil das Modell **keine Sorte-Block-Interaktion** annimmt - der Block verschiebt jede Sorte um denselben Betrag. Wenn diese Annahme falsch ist (manche Sorten gedeihen im Super-Block stärker als andere), ist der extrapolierte Mittelwert für `D` unzuverlässig, weil es keine Daten gibt, an denen man ihn prüfen könnte. Ob Additivität sinnvoll ist, ist genau die Art von Sache, die man mit Anhang A1: Modelldiagnostik untersucht. Adjustierte Mittelwerte sind mächtig, aber sie sind nur so vertrauenswürdig wie das Modell dahinter.

Direkter Vergleich

Beide Spalten nebeneinander zu stellen macht die Korrektur explizit:

```
as_tibble(mean_comp) %>%
  select(variety, adjusted_mean = emmean) %>%
  left_join(naive, by = "variety") %>%
  select(variety, n, naive_mean, adjusted_mean) %>%
  arrange(desc(adjusted_mean))
```

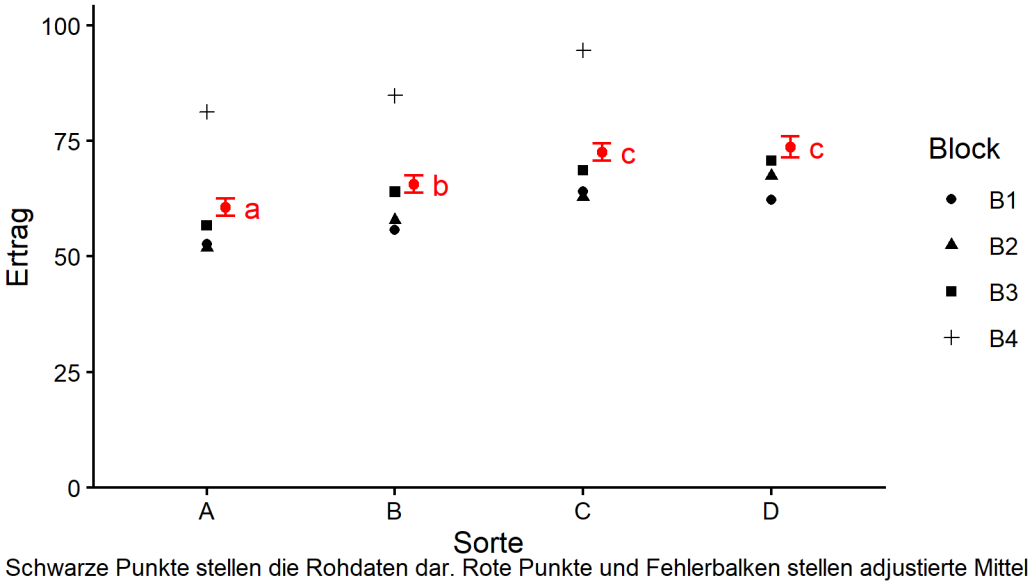
```
# A tibble: 4 × 4
  variety     n naive_mean adjusted_mean
  <fct>   <int>      <dbl>         <dbl>
1 D           3       66.8          73.6
2 C           4       72.5          72.5
3 B           4       65.6          65.6
4 A           4       60.6          60.6
```

Die balancierten Sorten (`A` , `B` , `C`) bleiben unangetastet; nur die unbalancierte Sorte `D` bewegt sich - und sie bewegt sich genug, um das Ranking umzukehren.

Wir können dieselbe Geschichte in einer Abbildung zeigen: Die schwarzen Punkte sind die Rohbeobachtungen, die roten Punkte und Intervalle sind die adjustierten Mittelwerte mit 95%-Konfidenzgrenzen, und die Buchstaben sind die kompakte Buchstabendarstellung.

```
my_caption <- "Schwarze Punkte stellen die Rohdaten dar.
Rote Punkte und Fehlerbalken stellen adjustierte Mittel (estimated marginal means)
mit 95% Konfidenzgrenzen je Sorte dar. Mittel mit einem gemeinsamen Buchstaben
unterscheiden sich nicht signifikant (Fishers LSD, keine Multiplizitätskorrektur)."

ggplot() +
  aes(x = variety) +
  # schwarze Punkte: Rohdaten
  geom_point(
    data = dat,
    aes(y = yield, shape = block)
  ) +
  # rote Punkte: adjustierte Mittel
  geom_point(
    data = mean_comp,
    aes(y = emmean),
    color = "red",
    position = position_nudge(x = 0.1)
  ) +
  # rote Fehlerbalken: Konfidenzgrenzen
  geom_errorbar(
    data = mean_comp,
    aes(ymin = lower.CL, ymax = upper.CL),
    color = "red",
    width = 0.1,
    position = position_nudge(x = 0.1)
  ) +
  # rote Buchstaben: kompakte Buchstabendarstellung
  geom_text(
    data = mean_comp,
    aes(y = emmean, label = str_trim(.group)),
    color = "red",
    position = position_nudge(x = 0.2),
    hjust = 0
  ) +
  scale_x_discrete(name = "Sorte") +
  scale_y_continuous(
    name = "Ertrag",
    limits = c(0, NA),
    expand = expansion(mult = c(0, 0.1))
  ) +
  scale_shape_discrete(name = "Block") +
  theme_classic() +
  labs(caption = my_caption) +
  theme(plot.caption = element_textbox_simple(margin = margin(t = 5)),
        plot.caption.position = "plot")
```



Schwarze Punkte stellen die Rohdaten dar. Rote Punkte und Fehlerbalken stellen adjustierte Mittel (estimated marginal means) mit 95% Konfidenzgrenzen je Sorte dar. Mittel mit einem gemeinsamen Buchstaben unterscheiden sich nicht signifikant (Fishers LSD, keine Multiplizitätskorrektur).

Man betrachte Sorte `D`: Ihr roter adjustierter Mittelwert schwebt **über allen drei ihrer schwarzen Rohpunkte**. Das ist die sichtbar gemachte Extrapolation in den unbeobachteten Super-Block. Für `C`, deren Rohpunkte eine Super-Block-Parzelle nahe `95` enthalten, liegt der rote Mittelwert bequem inmitten ihrer Daten.

Ein zweites Beispiel: Adjustierung für eine Kovariate

Das Block-Beispiel adjustierte für eine *kategoriale* Störgröße. Dieselbe Logik gilt für eine *kontinuierliche* - und genau daher stammt der Begriff “adjustierte Mittelwerte” ursprünglich, aus der Kovarianzanalyse (ANCOVA).

Stellen wir uns drei Behandlungen vor, die anhand des `yield` verglichen werden, wobei wir auch eine Baseline-Kovariate `x` aufgezeichnet haben - sagen wir die Anfangshöhe jeder Parzelle, gemessen *bevor* die Behandlungen angewendet wurden. Da die Parzellen nicht perfekt abgeglichen waren, starten die drei Gruppen zufällig bei unterschiedlichen mittleren Höhen:

```
set.seed(123)

# Hilfsfunktion: eine Gruppe mit eigenem Baseline-Mittel und echtem
Behandlungseffekt
make_group <- function(trt, x_mean, effect, n = 6) {
  tibble(treatment = trt, x = round(rnorm(n, x_mean, 2), 1)) %>%
    mutate(yield = round(effect + 1.5 * x + rnorm(n, 0, 2), 1))
}

dat_cov <- bind_rows(
  make_group("A", x_mean = 10, effect = 30), # tatsächlich am besten, aber
niedrigste Baseline
  make_group("B", x_mean = 15, effect = 28),
  make_group("C", x_mean = 20, effect = 26)  # tatsächlich am schlechtesten, aber
höchste Baseline
) %>%
  mutate(treatment = as.factor(treatment))

dat_cov
```

```
# A tibble: 18 × 3
   treatment     x yield
   <fct>     <dbl> <dbl>
 1 A           8.9  44.3
 2 A           9.5  41.7
 3 A          13.1  48.3
 4 A          10.1  44.3
 5 A          10.3  47.9
 6 A          13.4  50.8
 7 B          15.8  53.1
 8 B          15.2  49.9
 9 B          13.9  46.7
10 B          18.6  55.5
11 B          16    49.9
12 B          11.1  43.2
13 C          18.7  54.9
14 C          16.6  50.3
15 C          21.7  60.3
16 C          20.3  58.2
17 C          17.7  54.2
18 C          22.5  61.1
```

```
dat_cov %>%
  group_by(treatment) %>%
  summarise(naive_yield = mean(yield), mean_x = mean(x)) %>%
  arrange(desc(naive_yield))
```

```
# A tibble: 3 × 3
  treatment naive_yield mean_x
  <fct>           <dbl>  <dbl>
1 C                56.5   19.6
2 B                49.7   15.1
3 A                46.2   10.9
```

Das naive Ranking setzt Behandlung `C` an die Spitze. Aber man betrachte `mean_x`: Gruppe `C` *startete* auch von der höchsten Baseline (um 20), während `A` am niedrigsten startete (um 11). Ein Teil von `C`s scheinbarem Vorteil besteht einfach darin, dass ihre Parzellen schon vorne lagen, bevor irgendeine Behandlung angewendet wurde. Ein fairer Vergleich muss alle Behandlungen auf eine gemeinsame Baseline stellen.

Zuerst auf parallele Steigungen testen

Vor dem Adjustieren prüfen wir, ob sich die Kovariate in jeder Gruppe gleich verhält - ob die Regressionslinien parallel sind (eine gemeinsame Steigung). Wir vergleichen ein Modell mit separaten Steigungen gegen eines mit gemeinsamer Steigung:

```
mod_full     <- lm(yield ~ treatment * x, data = dat_cov) # separate Steigungen
mod_parallel <- lm(yield ~ treatment + x, data = dat_cov) # gemeinsame Steigung

anova(mod_parallel, mod_full)
```

```
Analysis of Variance Table

Model 1: yield ~ treatment + x
Model 2: yield ~ treatment * x
  Res.Df    RSS Df Sum of Sq      F Pr(>F)
1     14 27.322
2     12 26.465  2   0.85669 0.1942  0.826
```

Die Interaktion ist weit von Signifikanz entfernt (p um 0.83), sodass eine gemeinsame Steigung gerechtfertigt ist und wir das einfachere Modell mit parallelen Linien beibehalten. Dieser Test ist keine Formalität: Wären die Steigungen wirklich verschieden, könnte kein einzelner adjustierter Mittelwert eine Behandlung zusammenfassen, weil sich der Abstand zwischen den Behandlungen mit `x` ändern würde (siehe Hinweis unten).

```
emmeans(mod_parallel, specs = ~ treatment)
```

```
 treatment emmean    SE df lower.CL upper.CL
 A           53.4 0.896 14     51.5     55.3
 B           49.9 0.570 14     48.6     51.1
 C           49.2 0.907 14     47.2     51.1

Confidence level used: 0.95
```

Adjustiert auf die gemeinsame mittlere Höhe **kippt das Ranking zu `A` an der Spitze**, was die naive Reihenfolge exakt umkehrt. Das Modell hat den Vorsprung entfernt, den `C` genoss.

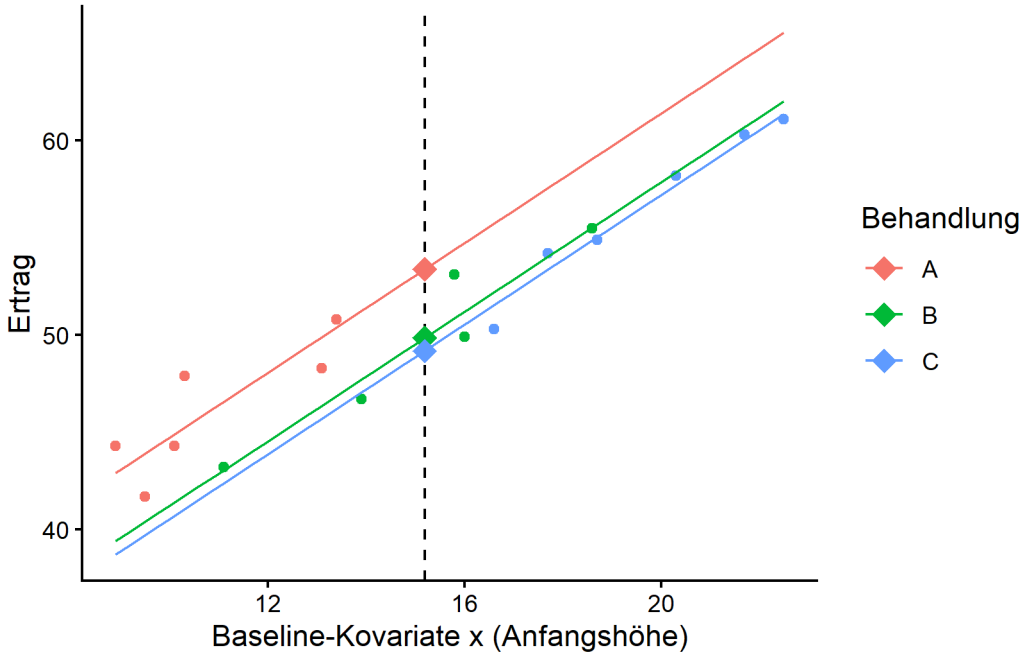
Die Abbildung zeigt, warum. Jede Behandlung erhält ihre eigene parallele Regressionslinie; die adjustierten Mittelwerte (Rauten) sind die Höhen dieser Linien beim gemeinsamen Kovariatenwert (gestrichelte Linie beim Mittel von `x`):

```
# Regressionslinien aus dem Modell mit gemeinsamer Steigung
pred <- expand_grid(
  treatment = levels(dat_cov$treatment),
```

```
  x = seq(min(dat_cov$x), max(dat_cov$x), length.out = 50)
)
pred$yield <- predict(mod_parallel, newdata = pred)

# adjustierte Mittel, platziert am mittleren Kovariatenwert
emm_pts <- mod_parallel %>%
  emmeans(specs = ~ treatment) %>%
  as_tibble() %>%
  mutate(x = mean(dat_cov$x))

ggplot(dat_cov, aes(x = x, y = yield, color = treatment)) +
  geom_point() +
  geom_line(data = pred) +
  geom_vline(xintercept = mean(dat_cov$x), linetype = "dashed") +
  geom_point(data = emm_pts, aes(y = emmean), size = 4, shape = 18) +
  scale_x_continuous(name = "Baseline-Kovariate x (Anfangshöhe)") +
  scale_y_continuous(name = "Ertrag") +
  scale_color_discrete(name = "Behandlung") +
  theme_classic()
```



Wo ausgewertet wird: emmeans(at = …)

Standardmäßig wertet `emmeans()` die Mittelwerte am *Mittel* der Kovariate aus. Wir können mit dem Argument `at` jeden anderen Wert (oder mehrere) anfordern. Das `| x` in der Spezifikation ist wichtig: Es hält die gewählten `x`-Werte nebeneinander (je ein Ergebnisblock). Ohne es - wenn man nur `~ treatment` mit demselben `at` schreibt - würde `emmeans()` *über* die aufgelisteten Werte *mitteln* und sie zu einem einzigen Mittelwert je Behandlung zusammenfassen, was selten das ist, was man will, wenn man bewusst mehrere Kovariatenwerte gewählt hat:

```
emmeans(mod_parallel, specs = ~ treatment | x, at = list(x = c(10, 20)))
```

```
x = 10:
 treatment emmean    SE df lower.CL upper.CL
 A           44.7 0.588 14     43.5     46.0
 B           41.2 0.998 14     39.1     43.4
 C           40.5 1.640 14     37.0     44.1
```

```
x = 20:
 treatment emmean    SE df lower.CL upper.CL
 A           61.4 1.570 14     58.0     64.8
 B           57.9 0.971 14     55.8     60.0
 C           57.2 0.574 14     56.0     58.4

Confidence level used: 0.95
```

Bei `x = 10` und bei `x = 20` unterscheiden sich die absoluten Mittelwerte - größere Pflanzen liefern mehr -, aber die *Differenzen* zwischen den Behandlungen sind identisch (`A` schlägt `C` um etwa `4.2` bei beiden Werten). Das ist die Signatur paralleler Linien: Der gewählte Kovariatenwert verschiebt den gesamten Satz von Mittelwerten nach oben oder unten, ohne die Vergleiche zu ändern, weshalb ein einzelner adjustierter Mittelwert (am Mittel von `x`) hier eine faire Zusammenfassung ist. Man beachte auch, dass die Standardfehler wachsen, je weiter `x` sich von der eigenen Baseline einer Gruppe entfernt - einen Mittelwert weit weg von dort auszuwerten, wo diese Behandlung tatsächlich beobachtet wurde, ist eine Extrapolation, und das Modell weist sie als weniger sicher aus.

⚠ Wenn die Steigungen nicht parallel sind

Wäre der Test auf gleiche Steigungen signifikant gewesen, würden die Linien sich kreuzen und die Behandlungsdifferenzen würden von `x` abhängen: Eine Behandlung könnte bei niedrigem `x` gewinnen und bei hohem `x` verlieren. Dann gibt es keinen einzelnen “adjustierten Mittelwert”, und `at = ...` wird unverzichtbar - man muss den Vergleich bei den Kovariatenwerten berichten, die für die Frage relevant sind, nicht an einem willkürlichen Punkt. Deshalb kommt der Test auf parallele Steigungen immer, bevor man einem kovariaten-adjustierten Mittelwert vertraut.

Wann stimmen adjustierte und arithmetische Mittelwerte überein?

Es ist erwähnenswert, dass adjustierte Mittelwerte nicht immer verschieden sind - und wenn sie gleich sind, ist das beruhigend statt überflüssig:

- **Balancierte Designs.** Wenn jede Behandlung gleich oft in jedem Block vorkommt (ein vollständiges, balanciertes RCBD oder ein balanciertes CRD), sind die adjustierten Mittelwerte exakt gleich den arithmetischen Mitteln. Deshalb liefert `emmeans()` im Kapitel zum einfaktoriellen CRD dieselben Zahlen wie `mean()`: Wenn es nichts zu adjustieren gibt, gibt es nichts zu korrigieren. (Die Standard*fehler* können sich auch hier noch unterscheiden - siehe der nächste Abschnitt.)
- **Unbalancierte Designs.** Fehlende Parzellen, ungleiche Wiederholung oder jede Abweichung von der Balance machen das arithmetische Mittel zu einer verzerrten Zusammenfassung des Behandlungseffekts und den adjustierten Mittelwert zur fairen. Das obige Beispiel ist der mildeste mögliche Fall - eine einzelne fehlende Parzelle - und er reichte bereits, um den Sieger umzukehren.
- **Kovariaten und gemischte Modelle.** Sobald ein Modell eine kontinuierliche Kovariate oder zufällige Effekte enthält, ist “der Mittelwert” nicht einmal mehr wohldefiniert, ohne anzugeben, *bei welchem Kovariatenwert* oder *über welche zufälligen Stufen gemittelt*. Adjustierte Mittelwerte machen das explizit, und der Abstand zum arithmetischen Mittel wächst typischerweise. Das ist umso wichtiger für die unbalancierten Designs in Anhang A6: Lineare gemischte Modelle.

Die praktische Faustregel: Man berichtet arithmetische Mittel als schnelle deskriptive Plausibilitätsprüfung, stützt seine Inferenz aber - Rankings, Vergleiche, Buchstaben, Konfidenzintervalle - auf adjustierte Mittelwerte aus einem Modell, das widerspiegelt, wie das Experiment tatsächlich durchgeführt wurde.

Es ist nicht nur der Schätzwert: auch die Präzision kann sich unterscheiden

Bisher lag der Unterschied im *Wert* des Mittelwerts. Aber adjustierte Mittelwerte weichen von arithmetischen Mitteln entlang einer zweiten, unabhängigen Achse ab: ihrer **Präzision**. Hier kann der Punktschätzer exakt der Rohdurchschnitt sein, während der Standardfehler - und, wichtiger, der Standardfehler einer *Differenz* zwischen zwei Mittelwerten (die SED, die Größe, die Vergleiche und Buchstabendarstellungen tatsächlich antreibt) - unterschiedlich ist, weil er aus der Varianz-Kovarianz-Struktur des Modells gebildet wird statt aus einer einzigen gepoolten Fehlervarianz.

Das passiert, sobald das Modell mehr ist als ein schlichtes `lm()` mit unabhängigen, gleich variablen Fehlern:

- **Zufällige Blockeffekte.** Fittet man ein balanciertes RCBD mit `block` als *zufälligem* Effekt, sind die Behandlungsschätzer immer noch die arithmetischen Mittel, aber ihre Standardfehler absorbieren nun die Varianzkomponente zwischen den Blöcken. (Bei unbalancierten Daten verschieben sich auch die Schätzer, durch die sogenannte Wiedergewinnung von Inter-Block-Information.)

- **Heterogene Varianzen.** Lässt man jeder Gruppe ihre eigene Residualvarianz, bewegen sich in einem balancierten Design die Schätzer nicht - aber die Standardfehler hören auf, über die Gruppen hinweg identisch zu sein. Das ist genau das Phänomen, das in Anhang A2 seziert wird.
- **Korrelierte Beobachtungen (Split-Plot, Messwiederholung).** Wenn Fehler korreliert sind, hängt die SED davon ab, *welche* zwei Mittelwerte verglichen werden. In einem Split-Plot-Design ist ein Vergleich innerhalb derselben Großparzelle präziser als einer über Großparzellen hinweg, sodass derselbe Satz von Mittelwerten mehrere verschiedene SEDs trägt - obwohl jeder Mittelwert gleich seinem Rohdurchschnitt sein mag.

In all diesen Fällen können die Schlagzeilen-Zahlen - die Mittelwerte selbst - identisch zu einem schnellen `group_by() %>% summarise()` aussehen, während die darauf beruhende Inferenz (Konfidenzintervalle, p-Werte, kompakte Buchstaben) sich unterscheidet und, wenn die Varianz-Kovarianz-Struktur korrekt spezifiziert ist, vertrauenswürdiger ist. Die Mittelwerte richtig zu bekommen ist nur die halbe Aufgabe; ihre *Unsicherheit* richtig zu bekommen ist die andere Hälfte.

Abschluss

Adjustierte Mittelwerte sind keine kompliziertere Art, einen Durchschnitt zu berechnen; sie sind eine *fairere*. Indem sie jeden Behandlungsmittelwert aus einem Modell rekonstruieren, das von den Blöcken (und Kovariaten und zufälligen Effekten) weiß, korrigieren sie für die Zufälle eines unbalancierten Designs, die ein Rohdurchschnitt stillschweigend mit einbackt.

ℹ Zusammenfassung

1. **Arithmetische Mittel** mitteln genau die Beobachtungen, die eine Behandlung erhalten hat; **adjustierte Mittelwerte** schätzen, wie ihr Mittelwert unter einem fairen, balancierten Vergleich wäre.
2. In einem **balancierten** Design sind die beiden **identisch**. In einem **unbalancierten** Design gehen sie auseinander, und das arithmetische Mittel kann Behandlungen falsch einordnen.
3. Adjustierte Mittelwerte können auf **zwei unabhängige Arten** von Rohdurchschnitten abweichen: im **Schätzwert** selbst (getrieben durch Unbalance oder Kovariaten) und in seiner **Präzision** - dem SE und besonders der SED (getrieben durch die Varianz-Kovarianz-Struktur des Modells: zufällige Effekte, heterogene oder korrelierte Fehler). Letzteres kann auftreten, selbst wenn die Schätzwerte identisch sind.
4. “Adjustiert für Block” bedeutet **Mitteln der Modellvorhersagen über alle Blockstufen mit gleichem Gewicht**, einschließlich Blöcken, in denen eine Behandlung nie vorkam.
5. Diese Extrapolation **beruht auf Additivität** (keine Behandlung-Block-Interaktion) - eine Modellannahme, die zu prüfen sich lohnt (Anhang A1).
6. Für jede Analyse jenseits eines balancierten einfaktoriellen Layouts stützt man die Inferenz auf **adjustierte Mittelwerte**, nicht auf Rohdurchschnitte.

Weiterführende Literatur

Zusätzliche Ressourcen

- `{emmeans}`-Vignette: Basics of estimated marginal means - Russell Lenths eigene Einführung dazu, was EMMs sind und wie sie berechnet werden.
- `{emmeans}`-Vignette: Estimated marginal means for unbalanced and missing-cell designs - die technische Behandlung genau der Situation in diesem Kapitel.
- Anhang A2 - Warum sind die Standardfehler alle gleich? - die begleitende Frage nach den *Standardfehlern* adjustierter Mittelwerte.
- Anhang A1 - Modelldiagnostik - wie man die Additivitäts- und Homoskedastizitätsannahmen prüft, auf die sich die adjustierten Mittelwerte stützen.

Bibliography