

Aufgabe 1:

(Simulation von fehlenden Daten)

- (a) Erzeugen Sie komplette Daten aus einem LMM ohne fixe Einflussgrößen mit zufälligem Achsenabschnitt für $N = 10$ Personen und $n = 2$ Beobachtungen pro Person. Erstellen Sie auf diese Weise den Datensatz `sim1`. Speichern Sie auch die erzeugten zufälligen Intercepts $b_i, i = 1, \dots, N$, für jede Person.

Lösung:

(Aufgabe adaptiert von Michael Hoehle, SoSe07)

```
R>set.seed(123)
R>sim.basiclme <- function(N = 30, n = 2, beta = 0, dbetween = 2.5,
+   dwithin = 0.5) {
+   y <- matrix(0, nrow = N, ncol = n)
+   b <- rnorm(N, 0, sqrt(dbetween))
+   for (i in 1:N) {
+     y[i, ] <- beta + b[i] + rnorm(n, 0, sqrt(dwithin))
+   }
+   row.names(y) <- c(1:nrow(y))
+   colnames(y) <- paste("t.", 0:(n - 1), sep = "")
+   return(list(y = y, b = b))
+ }
R>data <- sim.basiclme()
R>y <- data$y
R>b <- data$b
R>sim1 <- y
```

- (b) Sei R_i ein Indikator für die i 'te Person, d.h. mit $R_i = 1$ wenn Y_{i2} beobachtet wurde und $R_i = 0$ wenn Y_{i2} fehlt. Von Interesse ist es die Wahrscheinlichkeit $\pi_i = P(R_i = 1 | Y_{i1}, Y_{i2}, X)$ für fehlende Werte bzw. Dropout zum Zeitpunkt 2 mit Hilfe von logistischer Regression zu modellieren. Im Fall ohne Kovariablen wäre ein mögliches Modell: $R_i \sim \text{Bernoulli}(\pi_i)$ mit

$$\text{logit}(\pi_i) = \beta_0 + \beta_1 Y_{i1} + \beta_2 Y_{i2}.$$

Geben Sie die Restriktionen für $(\beta_0, \beta_1, \beta_2)$ an wenn der Dropout

- (i) MCAR
- (ii) MAR
- (iii) NMAR

ist.

Lösung:

- (i) Dropout MCAR ist $\beta_1 = \beta_2 = 0$.

(ii) Dropout MAR ist $\beta_2 = 0$ und

(iii) Dropout NMAR ist $\beta_2 \neq 0$.

- (c) Erzeugen Sie für jede der $N = 10$ Personen eine Realisation des Indikators R_i für $\beta_0 = \beta_1 = 0$ und $\beta_2 = 1$. Falls $R_i = 0$ setzen Sie $Y_{i2} = \text{NA}$. Erstellen Sie auf diese Weise den Datensatz `sim2`.

Lösung:

Simulation von NMAR-Daten: Fehlen abhängig von unbeobachteten Werten der Response

```
R>makeMissing <- function(data, beta) {
+   pi <- plogis(cbind(1, data) %*% beta)
+   R <- rbinom(nrow(data), size = 1, prob = pi)
+   data[R == 0, 2] <- NA
+   data <- as.data.frame(cbind(data, R))
+ }
R>beta <- matrix(c(0, 0, 1), 3, 1)
R>sim2 <- makeMissing(sim1, beta)
```

- (d) Wiederholen Sie Teil (b) in dem Sie das Modell

$$\text{logit } P(R_i = 1|b_i) = \alpha_0 + \alpha_1 b_i,$$

mit $\alpha_0 = -1$ und $\alpha_1 = 1$ benutzen. Erstellen Sie auf diese Weise den Datensatz `sim3`. Um was für einen Dropout-Mechanismus handelt es sich?

Lösung:

Simulation von NMAR-Daten: Fehlen abhängig von den (unbeobachteten) Werten der zufälligen Effekte.

```
R>makeMissing2 <- function(data, b, alpha) {
+   pi <- plogis(cbind(1, b) %*% alpha)
+   R <- rbinom(nrow(data), size = 1, prob = pi)
+   data[R == 0, 2] <- NA
+   data <- as.data.frame(cbind(data, R))
+ }
R>alpha <- matrix(c(-1, 1), 2, 1)
R>sim3 <- makeMissing2(sim1, b, alpha)
```

- (e) Fitten Sie für jedes der drei oben erstellten `data.frames` `sim1`, `sim2` und `sim3` ein LMM mit Random-Intercept.

Hinweis: Konvertieren Sie die Daten mit der Funktion `reshape` in Long-Format und benutzen Sie die Option `na.action=na.exclude` in der Funktion `lme` um fehlende Y_{i2} zu ignorieren.

Lösung:

```
R>sim2long <- function(y) {
+   require(doby)
+   y.long <- reshape(y, varying = list(c("t.0", "t.1")), v.names = "y",
+     direction = "long")
+   return(groupedData(y ~ 1 | id, data = orderBy(~id + t, y.long)))
+ }
R>library(nlme)
R>(m1 <- lme(y ~ 1, random = ~1 | id, data = sim2long(as.data.frame(sim1))))
```

Linear mixed-effects model fit by REML

```
Data: sim2long(as.data.frame(sim1))
Log-restricted-likelihood: -97
Fixed: y ~ 1
(Intercept)
-0.0028
```

Random effects:

```
Formula: ~1 | id
(Intercept) Residual
StdDev:      1.6      0.63
```

Number of Observations: 60

Number of Groups: 30

```
R>(m2 <- lme(y ~ 1, random = ~1 | id, data = sim2long(sim2), na.action = na.exclude))
```

Linear mixed-effects model fit by REML

```
Data: sim2long(sim2)
Log-restricted-likelihood: -83
Fixed: y ~ 1
(Intercept)
0.04
```

Random effects:

```
Formula: ~1 | id
(Intercept) Residual
StdDev:      1.5      0.6
```

Number of Observations: 51

Number of Groups: 30

```
R>(m3 <- lme(y ~ 1, random = ~1 | id, data = sim2long(sim3), na.action = na.exclude))
```

Linear mixed-effects model fit by REML

```
Data: sim2long(sim3)
Log-restricted-likelihood: -67
Fixed: y ~ 1
(Intercept)
0.015
```

Random effects:

```
Formula: ~1 | id
(Intercept) Residual
StdDev:      1.6      0.48
```

Number of Observations: 40

Number of Groups: 30

Im obigen ist `na.exclude` eine vordefinierte Funktion (siehe `?na.exclude`).

- (f) Simulieren Sie 500 vollständige Datensätze wie oben und erzeugen Sie daraus jeweils Datensätze mit Dropouts nach den obigen 2 Mechanismen. Schätzen Sie jeweils ein LMM mit Random-Intercept und vergleichen Sie die resultierenden Verteilungen der Schätzer für den Intercept, für die Varianz des zufälligen Intercepts und die Varianz der Fehler zwischen den 3 verschiedenen Dropout-Prozessen.

Lösung:

```
R>nsim <- 500
R>res1 <- res2 <- res3 <- matrix(NA, nrow = nsim, ncol = 3)
R>getParams <- function(m) {
+   beta <- fixef(m)
+   varb <- getVarCov(m)
+   vareps <- m$sigma^2
+   return(c(beta, varb, vareps))
+ }
R>for (i in 1:nsim) {
+   data <- sim.basiclme()
+   y <- data$y
+   b <- data$b
+   sim1 <- y
+   sim2 <- makeMissing(sim1, beta = c(0, 0, 1))
+   sim3 <- makeMissing2(sim1, b, alpha)
+   m1 <- lme(y ~ 1, random = ~1 | id, data = sim2long(as.data.frame(sim1)))
+   res1[i, ] <- getParams(m1)
+   m2 <- lme(y ~ 1, random = ~1 | id, data = sim2long(sim2),
+     na.action = na.exclude)
+   res2[i, ] <- getParams(m2)
+   m3 <- lme(y ~ 1, random = ~1 | id, data = sim2long(sim3),
+     na.action = na.exclude)
+   res3[i, ] <- getParams(m3)
+   if (i%%50 == 0)
+     cat(i, " ")
+ }
```

```
50 100 150 200 250 300 350 400 450 500
```

```
R>beta <- data.frame(complete = res1[, 1], NMAR1 = res2[, 1], NMAR2 = res3[,
+ 1])
```

```
R>(bias.beta <- colSums(beta)/nsim)
```

```
complete    NMAR1    NMAR2
 0.0061    0.0857    0.0375
```

```
R>varb <- data.frame(complete = res1[, 2], NMAR1 = res2[, 2], NMAR2 = res3[,
+ 2])
```

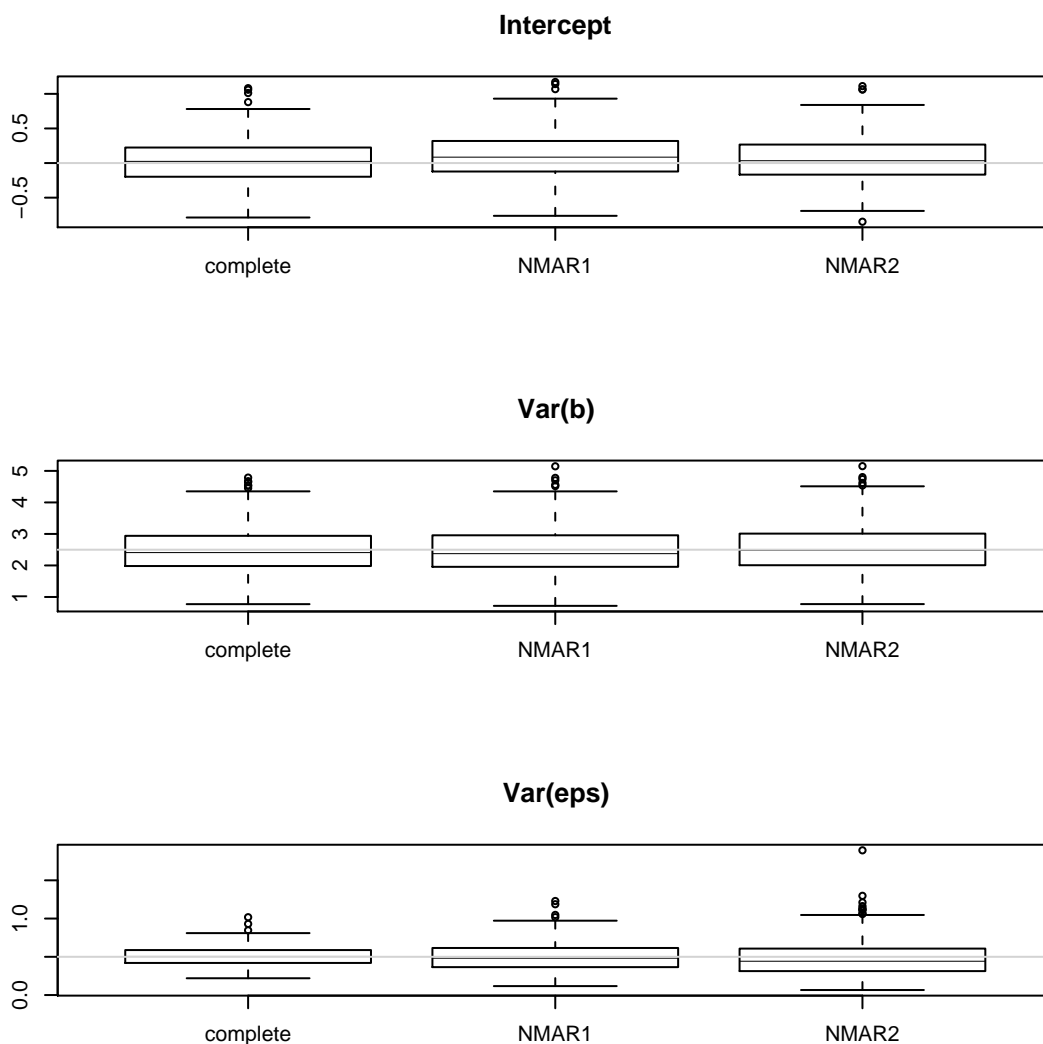
```
R>(bias.varb <- colSums((varb - 2.5))/nsim)
```

```
complete    NMAR1    NMAR2
0.0040 -0.0273 0.0064
```

```
R>vareps <- data.frame(complete = res1[, 3], NMAR1 = res2[, 3],
+   NMAR2 = res3[, 3])
R>(bias.vareps <- colSums((vareps - 0.5))/nsim)
```

```
complete    NMAR1    NMAR2
0.0095 0.0152 0.0074
```

```
R>par(mfrow = c(3, 1))
R>boxplot(beta, main = "Intercept", medlwd = 0.5)
R>abline(h = 0, col = "lightgrey")
R>boxplot(varb, main = "Var(b)", medlwd = 0.5)
R>abline(h = 2, col = "lightgrey")
R>boxplot(vareps, main = "Var(eps)", medlwd = 0.5)
R>abline(h = 0.5, col = "lightgrey")
```



⇒ hier keine (nennenswerte) systematische Verzerrung ersichtlich, etwas größere Variabilität der Schätzer aus NMAR-Daten da kleinere Datensätze.