Week 09: Partial pooling and mixed-effects models

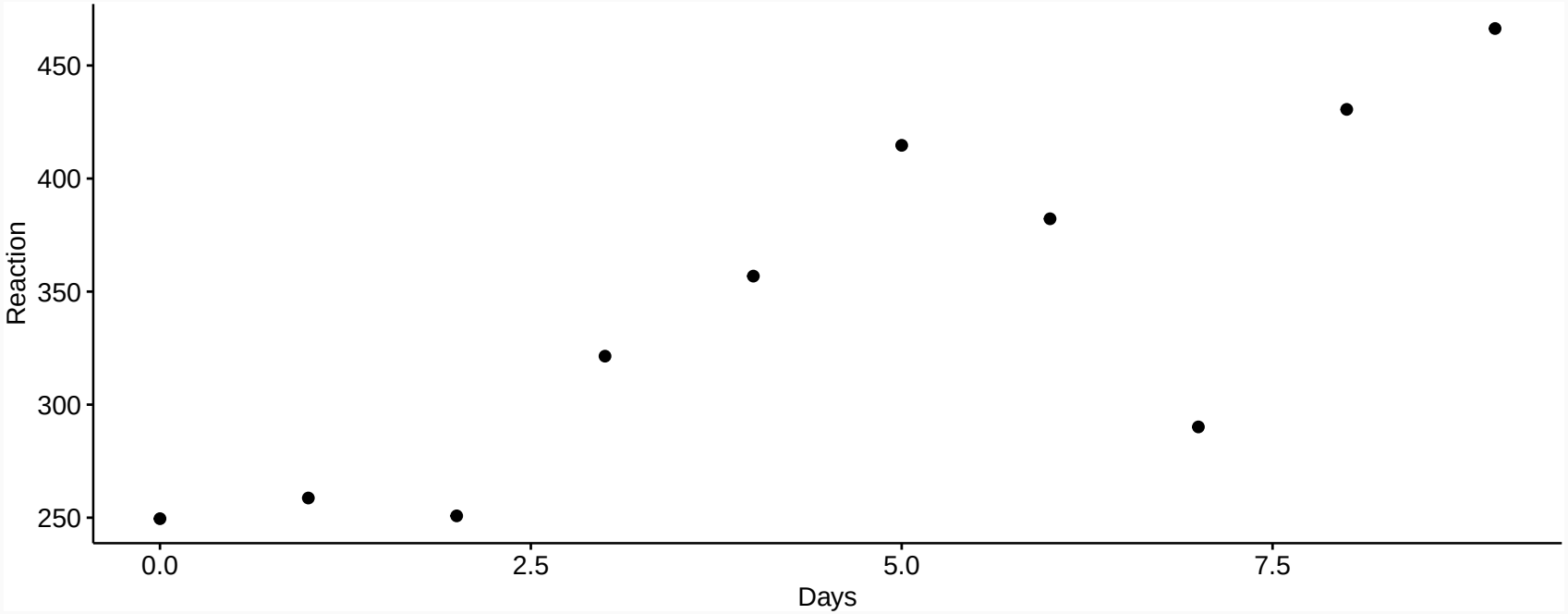
Random intercepts, random slopes, and model inspection

Bartosz Maćkiewicz

sleepstudy data

Let's take a closer look at the **sleepstudy** data. The dataset contains eighteen participants from the three-hour sleep condition. Each day, over 10 days, participants performed a ten-minute "psychomotor vigilance test" where they had to monitor a display and press a button as quickly as possible each time a stimulus appeared. The dependent measure in the dataset is the participant's average response time (RT) on the task for that day.

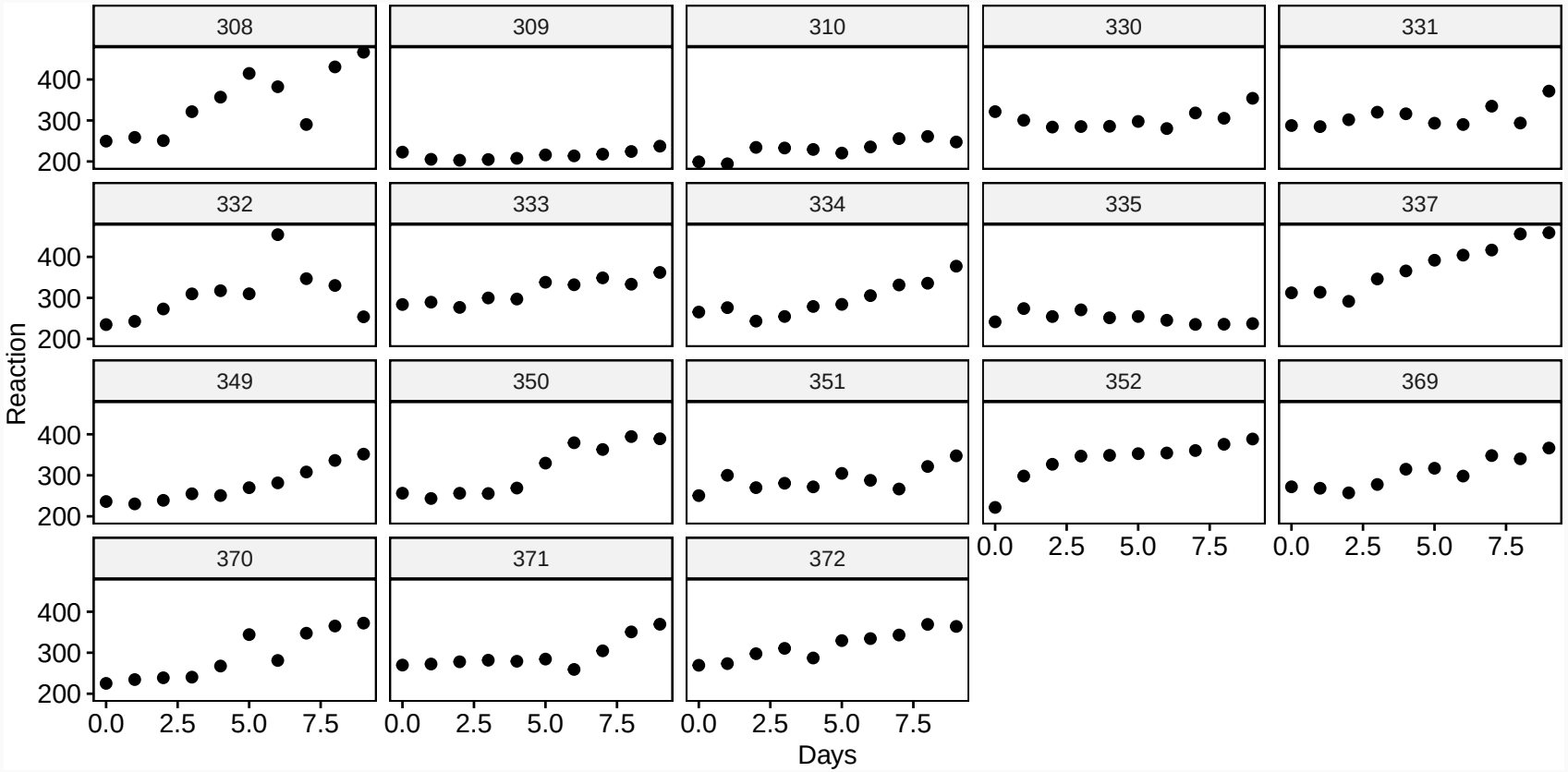
```
library(tidyverse); library(lme4); library(ggpubr)
participant_308  <- sleepstudy %>% filter(Subject == "308")
ggscatter(participant_308, x =  "Days", y = "Reaction")
```



sleepstudy data

Using `ggplot` we can create the plot which shows data for all 18 subjects.

```
ggscatter(sleepstudy, x =  "Days", y = "Reaction",
          facet.by = "Subject")
```



sleepstudy data

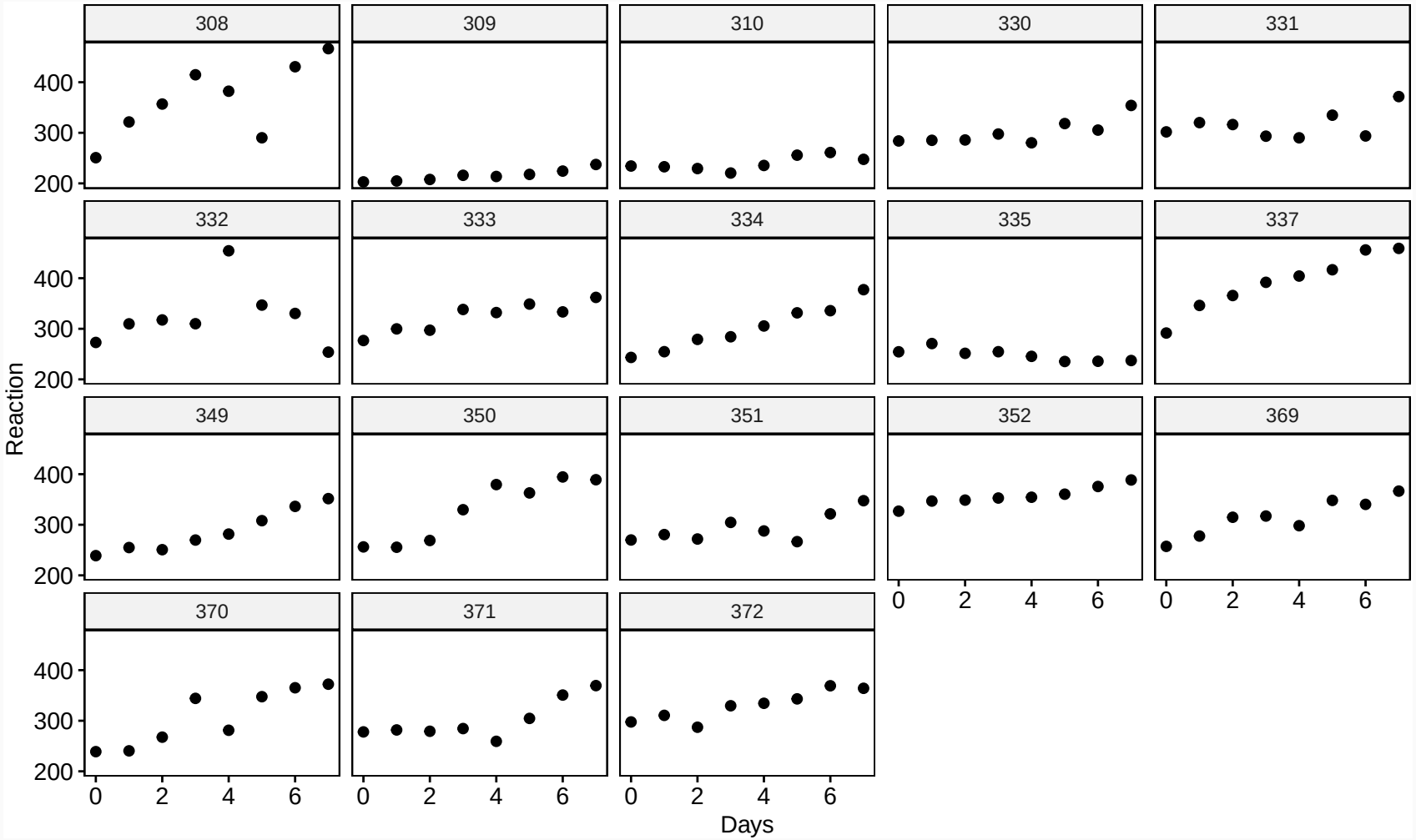
This is how Belenky et al. (2003) describe their study (p. 2):

> *The first 3 days (T1, T2 and B) were adaptation and training (T1 and T2) and baseline (B) and subjects were required to be in bed from 23:00 to 07:00 h [8 h required time in bed (TIB)]. On the third day (B), baseline measures were taken. Beginning on the fourth day and continuing for a total of 7 days (E1–E7) subjects were in one of four sleep conditions [9 h required TIB (22:00–07:00 h), 7 h required TIB (24:00–07:00 h), 5 h required TIB (02:00–07:00 h), or 3 h required TIB (04:00–07:00 h)], effectively one sleep augmentation condition, and three sleep restriction conditions.*

```
sleepstudy <- sleepstudy %>%
  mutate(Days = Days - 2) %>%
  filter(Days >= 0)
```

sleepstudy data

```
ggscatter(sleepstudy, x =  "Days", y = "Reaction",
          facet.by = "Subject")
```



Complete pooling

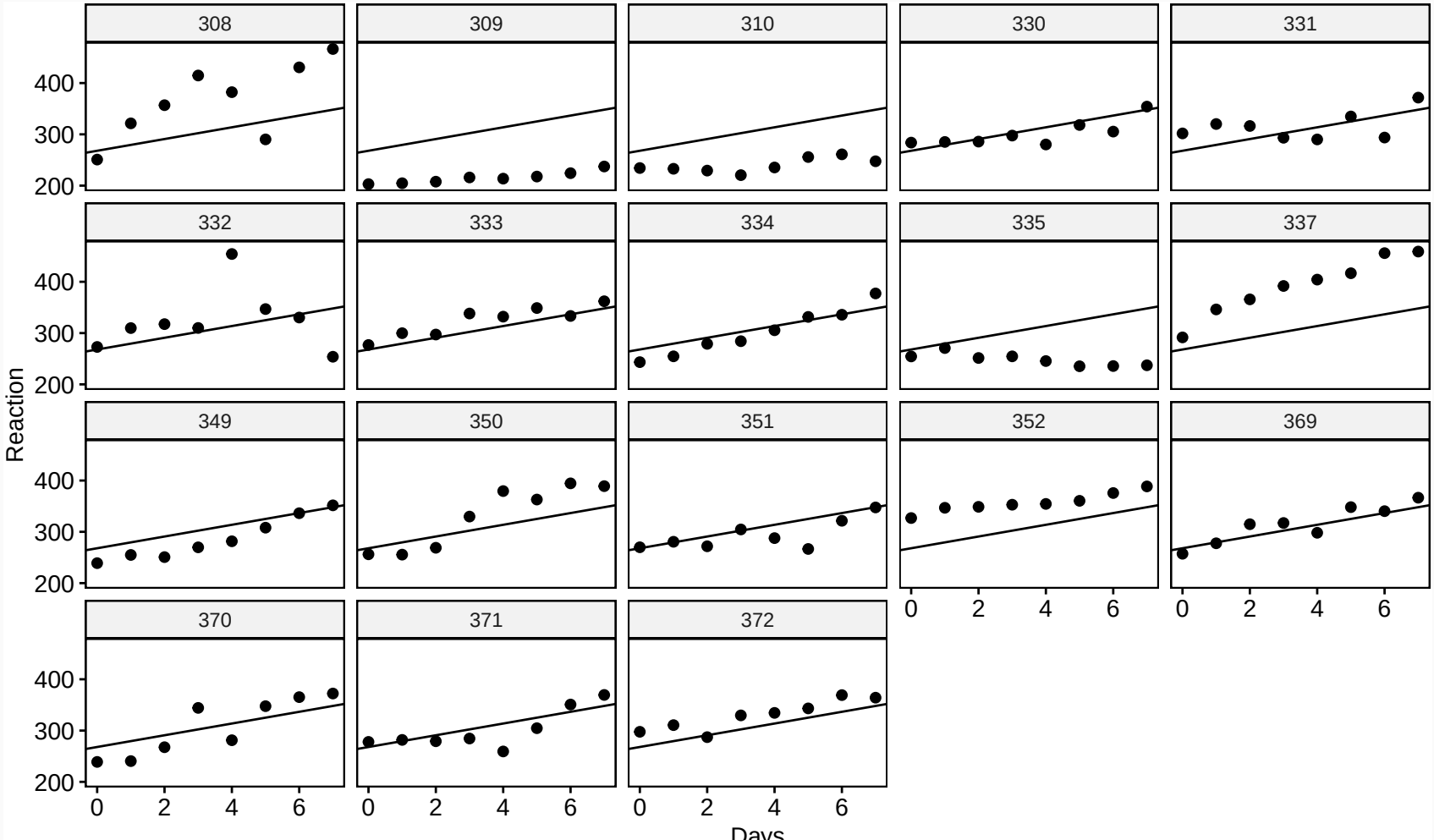
The complete pooling approach is a “one-size-fits-all” model: it estimates a single intercept and slope for the entire dataset, ignoring the fact that different subjects might vary in their intercepts or slopes.

```
complete_pooling_model <- lm(Reaction ~ Days, data = sleepstudy)
summary(complete_pooling_model)
```

```
Call:
lm(formula = Reaction ~ Days, data = sleepstudy)

Residuals:
     Min       1Q   Median       3Q      Max
-112.284  -26.732    2.143   27.734  140.453

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  267.967      7.737  34.633  < 2e-16 ***
Days          11.435      1.850   6.183 6.32e-09 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 50.85 on 142 degrees of freedom
Multiple R-squared:  0.2121,    Adjusted R-squared:  0.2066
```

Complete pooling

```
ggscatter(sleepstudy, x =  "Days", y = "Reaction", facet.by = "Subject") +
  geom_abline(
    intercept = complete_pooling_model$coefficients[["(Intercept)"]],
    slope = complete_pooling_model$coefficients[["Days"]]
    )
```



No pooling

Pooling all the information to get just one intercept and one slope estimate seems inappropriate. Another approach would be to fit separate lines for each participant. This means that the estimates for each participant will be completely uninformed by the estimates for the other participants. In other words, we can separately estimate 18 individual intercept/slope pairs.

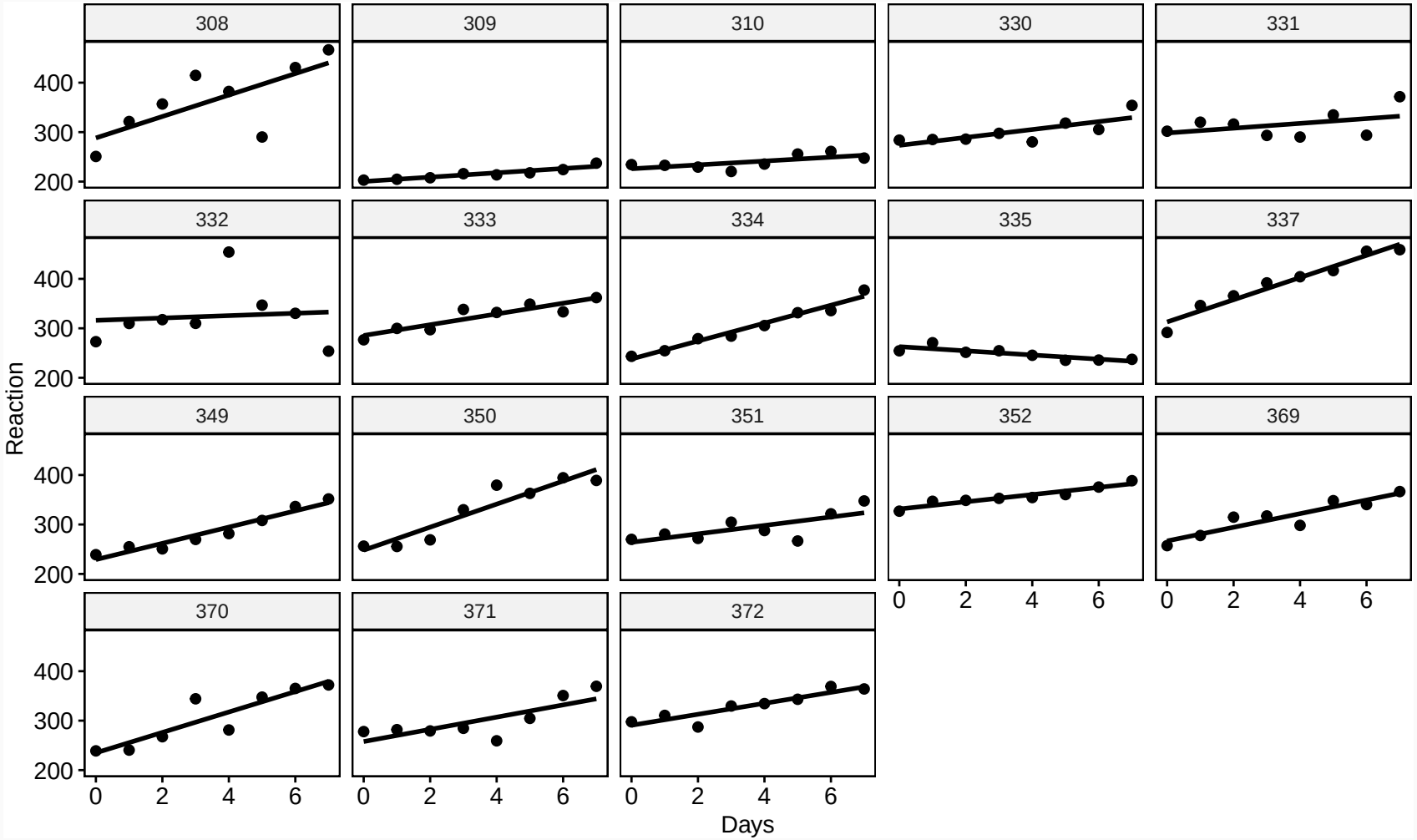
Question: What are intercepts and slopes for subjects 308 and 335?

```
no_pooling_model <- lm(
  Reaction ~ Days + Subject + Days:Subject,
  data = sleepstudy
)

coef(no_pooling_model)[1:8]
```

```
(Intercept)        Days  Subject309  Subject310  Subject330  Subject331
 288.217467   21.690495  -87.926208  -62.285625  -14.953292    9.965817
 Subject332  Subject333
  27.815733   -2.758117
```

No pooling

```
ggscatter(sleepstudy, x =  "Days", y = "Reaction",
          facet.by = "Subject", add = "reg.line")
```



Why no pooling is not enough

Neither the complete nor no-pooling approach is satisfactory. It would be desirable to improve our estimates for individual participants by taking advantage of what we know about the other participants. This will help us better distinguish signal from error for each participant and improve generalization to the population.

In the no-pooling model, we treated `Subject` as a **fixed factor**. Each pair of intercept and slope estimates is determined by that subject's data alone. However, we are not interested in these 18 subjects in and of themselves; rather, we are interested in them **as examples drawn from a larger population of potential subjects**. This subjects-as-fixed-effects approach is suboptimal if your goal is to generalize to new participants in the population of interest.

Partial pooling using mixed-effects models

Partial pooling happens when you treat a factor as a **random** instead of **fixed** in your analysis. A **random factor** is a factor whose levels are considered to **represent a proper subset of all the levels in the population**. Usually, you treat a factor as random when the levels you have in the data are the result of sampling, and you want to generalize beyond those levels.

In this case, we have 18 unique subjects and thus, *18* levels of the `Subject` factor, and would like to say something general about effects of sleep deprivation on the population of potential subjects.

A way to include random factors in your analysis is to use a linear mixed-effects model. Rather than estimating the intercept and slope for each participant without considering the estimates for other subjects, the model estimates values for the population, and pulls the estimates for individual subjects toward those values, a statistical phenomenon known as *shrinkage*.

Partial pooling using mixed-effects models: multilevel model

The multilevel model is below. It looks complicated at first, but there is nothing here that you have not seen before.

Level 1:

$$Y_{sd} = \beta_{0s} + \beta_{1s} X_{sd} + e_{sd}$$

Level 2:

$$\beta_{0s} = \gamma_0 + S_{0s}$$

$$\beta_{1s} = \gamma_1 + S_{1s}$$

$$\langle S_{0s}, S_{1s} \rangle \sim N\left(\langle 0, 0 \rangle, \Sigma\right)$$

Term	Meaning
Y_{sd}	`Reaction` for subject s on day d
X_{sd}	`Days` for subject s on day d
β_{0s}	subject-specific intercept
β_{1s}	subject-specific slope
e_{sd}	residual error
γ_0	grand intercept
γ_1	grand slope
S_{0s}	subject intercept adjustment
S_{1s}	subject slope adjustment
Σ	random-effect covariance matrix

Partial pooling using mixed-effects models: fitting the model using `lme4`

To estimate parameters, we are going to use the `lmer()` function of the `lme4` package. The basic syntax of `lmer()` is

```
lmer(formula, data, ...)
```

where `formula` expresses the structure of the underlying model in a compact format.

The general format of the model formula for N fixed effects (`fix`) and K random effects (`ran`) is:

```
DV ~ fix1 + fix2 + ... + fixN + (ran1 + ran2 + ... + ranK | random_factor1)
```

Each bracketed expression represents random effects associated with a single random factor. On the left side of the bar `|` you put the effects you want to allow to vary over the levels of the random factor named on the right side. Usually, the right-side variable is one whose values uniquely identify individual subjects (e.g., `subject_id`).

Model	Syntax
random intercepts only	`Reaction ~ Days + (1 | Subject)`
random intercepts and slopes	`Reaction ~ Days + (1 + Days | Subject)`
(alternative syntax)	`Reaction ~ Days + (Days | Subject)`
random slopes only	`Reaction ~ Days + (0 + Days | Subject)`
random intercepts and slopes + zero-covariances	`Reaction ~ Days + (Days || Subject)`

Partial pooling using mixed-effects models

```
pp_model <- lmer(
  Reaction ~ # DV
  Days + # Fixed effect
  (Days |Subject), # random intercept and slope for each subject
  #(1|Subject), # only random intercept for each subject
  data = sleepstudy
  )
```

```
summary(pp_model)
```

Partial pooling using mixed-effects models: interpreting fixed effects

```
## Fixed effects:
##                Estimate Std. Error t value
## (Intercept)     267.967      8.266  32.418
## days_deprived    11.435      1.845   6.197
```

This indicates that the estimated mean reaction time for participants at Day 0 was about 268 milliseconds, with each day of sleep deprivation adding an additional 11 milliseconds to the response time, on average.

Partial pooling using mixed-effects models: random effects

```
## Random effects:
##  Groups   Name          Variance Std.Dev. Corr
##  Subject  (Intercept)   958.35   30.957
##           days_deprived  45.78    6.766   0.18
##  Residual               651.60   25.526
## Number of obs: 144, groups:  Subject, 18
```

What you find here is a table with information about the variance components: the variance-covariance matrix (or matrices, if you have multiple random factors) and the residual variance.

`Residual` tells us that the residual variance was estimated at about 651.6.

The two lines above the `Residual` line give us information about the variance-covariance matrix for the `Subject` random factor. The values in the `Variance` column gives us the main diagonal of the matrix. The `Corr` column tells us the correlation between the intercept and slope.

Mixed-Effects Models: why?

Yarkoni (2022): Fixed effects are used to model variables that must remain constant in order for the model to preserve its meaning across replication studies; random effects are used to model indicator variables that are assumed to be stochastically sampled from some underlying population and can vary across replications without meaningfully altering the research question.

Inspecting the model

The model object contains the same kinds of quantities we have been discussing: population-level effects, subject-level adjustments, residual variation, and fitted values. The important point is that these are not separate analyses. They are different views of the same fitted model.

```
formula(pp_model)
```

```
Reaction ~ Days + (Days | Subject)
```

```
nobs(pp_model)
```

```
[1] 144
```

```
lme4::ngrps(pp_model)
```

```
Subject
     18
```

Inspecting the model: fixed effects

The fixed effects are the population-level estimates. In this model, they describe the estimated average intercept and the estimated average effect of each additional day of sleep deprivation.

```
fixef(pp_model)
```

```
(Intercept)        Days
  267.96742    11.43543
```

Inspecting the model: random effects

The table with random effects gives us the variance-covariance matrix for the `Subject` random factor and the residual variance. Since `(Days | Subject)` contains both a random intercept and a random slope, the model also estimates their correlation.

```
VarCorr(pp_model)
```

```
 Groups   Name        Std.Dev. Corr
 Subject  (Intercept) 30.9573
          Days         6.7659  0.178
 Residual             25.5264
```

Inspecting the model: BLUPs

The actual adjustments for specific subjects can be extracted from the model with `ranef()`. These values are often called BLUPs: best linear unbiased predictions. They are not the whole subject-specific intercepts and slopes. They are deviations from the fixed effects.

```
head(ranef(pp_model)$Subject, 6)
```

```
    (Intercept)      Days
308   24.499289  8.602000
309  -59.372310 -8.127753
310  -39.476276 -7.429237
330    1.350043 -2.384598
331   18.457617 -3.747734
332   30.527004 -4.893690
```

BLUPs are deviations

For subject 308, the fixed intercept is combined with the subject's intercept adjustment. The same logic applies to the slope. This is the partial-pooling estimate for this particular subject.

```
fixef(pp_model)
```

```
(Intercept)        Days
  267.96742    11.43543
```

```
ranef(pp_model)$Subject["308", ]
```

```
    (Intercept)  Days
308    24.49929 8.602
```

```
coef(pp_model)$Subject["308", ]
```

```
    (Intercept)     Days
308    292.4667 20.03743
```

Subject-specific coefficients

The `coef()` function gives the subject-specific coefficients directly. These are the fixed effects plus the subject-specific random-effect adjustments.

```
coef(pp_model)$Subject |> head(6)
```

```
    (Intercept)      Days
308    292.4667 20.037429
309    208.5951  3.307675
310    228.4911  4.006192
330    269.3175  9.050831
331    286.4250  7.687695
332    298.4944  6.541739
```

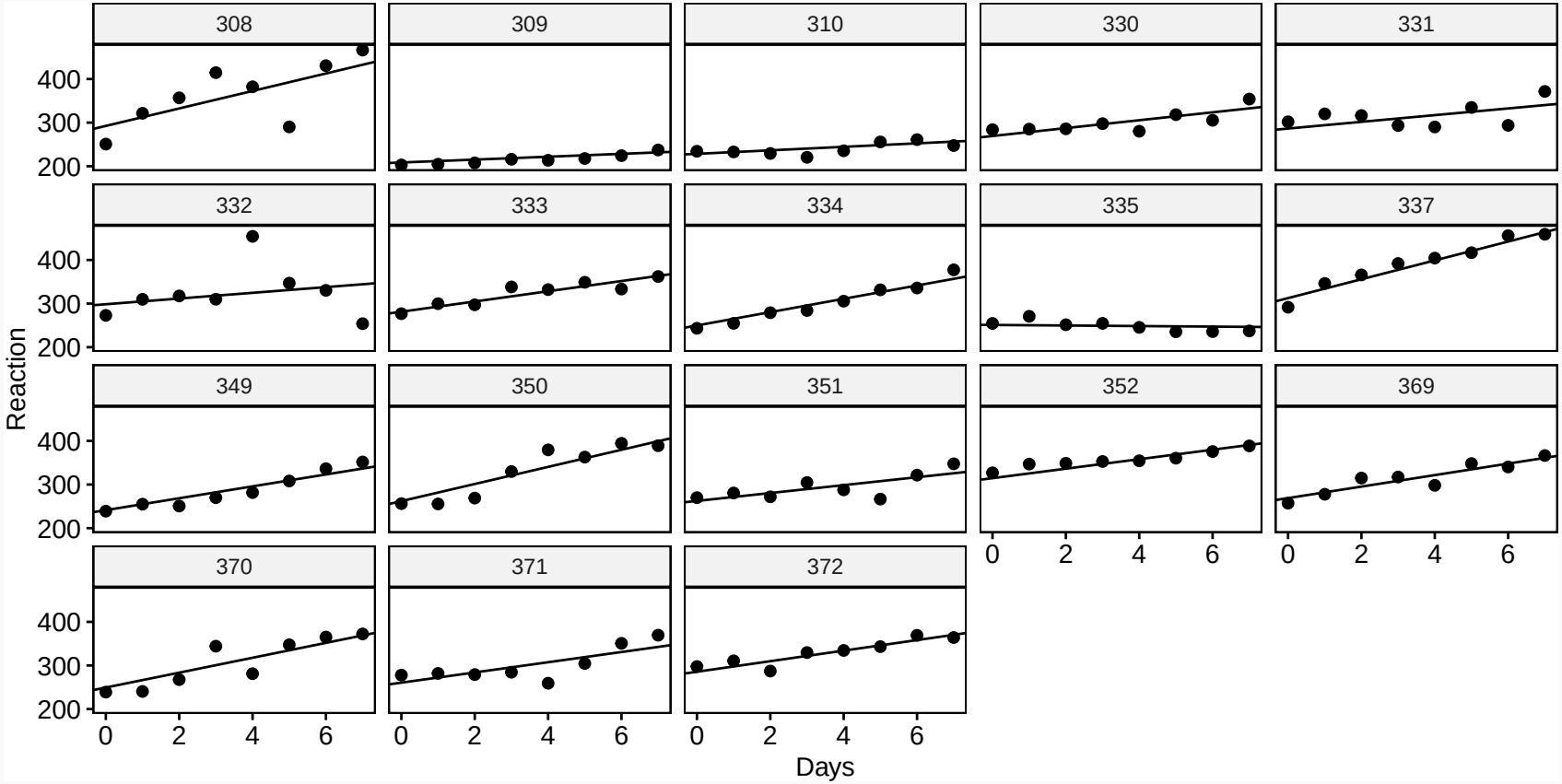
Code: partial-pooling fitted lines

To draw the subject-specific fitted lines, first extract the subject-specific coefficients from the model. Then pass them to `geom_abline()`.

```
subject_coefs <- coef(pp_model)$Subject
subject_coefs$Subject <- rownames(subject_coefs)

ggscatter(sleepstudy, x = "Days", y = "Reaction",
          facet.by = "Subject") +
  geom_abline(
    data = subject_coefs,
    aes(intercept = `(Intercept)`, slope = Days)
  )
```

Partial pooling: fitted lines

The model estimates a separate line for each subject, but these lines are not estimated independently. They are pulled toward the population line, with the amount of pulling depending on the data and the estimated variance components.



Inspecting the model: fitted values

For a given observation, the fitted value combines the fixed effects with the relevant random-effect adjustments for that subject.

```
sleepstudy[1, c("Subject", "Reaction", "Days")]
```

```
  Subject Reaction Days
1     308 250.8006    0
```

```
fixef(pp_model)[1] +
  fixef(pp_model)[2] * sleepstudy$Days[1] +
  ranef(pp_model)$Subject["308", "(Intercept)"] +
  ranef(pp_model)$Subject["308", "Days"] * sleepstudy$Days[1]
```

```
(Intercept)
   292.4667
```

Inspecting the model: fitted values

The same fitted values can be extracted directly. In practice, we usually use `fitted()` rather than constructing every fitted value by hand. The manual calculation is useful because it shows what the model is doing.

```
fitted(pp_model) |> head(6)
```

```
       1        2        3        4        5        6
292.4667 312.5041 332.5416 352.5790 372.6164 392.6539
```

Statistical testing

For some time it was not perfectly clear how to calculate the appropriate degrees of freedom for statistical tests on fixed effects. That is why plain `lmer` objects traditionally did not print familiar degrees of freedom and p-values.

`lmerTest` provides a wrapper to `lmer()` that runs tests using Satterthwaite's degrees-of-freedom method.

```
coef(summary(pp_model))
```

```
             Estimate Std. Error       df   t value     Pr(>|t|)
(Intercept) 267.96742   8.265896 17.00037 32.418437 9.984058e-17
Days         11.43543   1.845293 16.99950  6.197082 9.751277e-06
```

Testing fixed effects with `anova()`

For a model fitted with `lmerTest`, `anova()` gives an ANOVA-style test of the fixed effects. Here it tests the population-level effect of `Days`: whether reaction time changes systematically with sleep deprivation.

```
anova(pp_model)
```

```
Type III Analysis of Variance Table with Satterthwaite's method
     Sum Sq Mean Sq NumDF DenDF F value    Pr(>F)
Days  25024   25024     1    17  38.404 9.751e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Testing random effects with `ranova()`

The `ranova()` function gives an ANOVA-like table for random effects. Here the main question is whether allowing the effect of `Days` to vary by subject improves the model.

```
ranova(pp_model)
```

```
ANOVA-like table for random-effects: Single term deletions

Model:
Reaction ~ Days + (Days | Subject)
                        npar  logLik    AIC    LRT Df Pr(>Chisq)
<none>                     6 -702.05 1416.1
Days in (Days | Subject)    4 -715.01 1438.0 25.926  2  2.346e-
06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Comparing models with `anova()`

Another common use of `anova()` is direct model comparison. For this example, `ranova(pp_model)` is the clearest way to ask whether the random slope for `Days` improves the model.

If you compare two `lmer` models directly with `anova()`, R will refit REML models with maximum likelihood before the likelihood-ratio comparison. That is why some tutorials fit comparison models with `REML = FALSE` in advance.

```
fit_ri <- lmer(
  Reaction ~ Days + (1 | Subject),
  data = sleepstudy
)
```

Comparing models with anova( )

```
anova(fit_ri, pp_model)
```

```
Data: sleepstudy
Models:
fit_ri: Reaction ~ Days + (1 | Subject)
pp_model: Reaction ~ Days + (Days | Subject)
         npar    AIC    BIC  logLik -2*log(L)  Chisq Df Pr(>Chisq)
fit_ri      4 1446.5 1458.4 -719.25    1438.5
pp_model    6 1425.2 1443.0 -706.58    1413.2 25.332  2  3.156e-
06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Takeaways

Mixed-effects models are useful when the data contain repeated observations or clustered observations. Complete pooling ignores differences between subjects. No pooling estimates each subject separately. Partial pooling uses the whole dataset to estimate population-level effects while also estimating subject-specific deviations.

In this example, the model lets subjects differ in both their baseline reaction times and their sleep-deprivation slopes.

The practical toolbox is small: `lmer()`, `summary()`, `fixef()`, `VarCorr()`, `ranef()`, `coef()`, `fitted()`, `anova()`, and `ranova()`.