Introduction to Computational BioStatistics with R: mixed-effects models

Erik Spence

4 November 2025

SciNet

Today's slides

Today's slides can be found here. Go to the "Introduction to Computational BioStatistics with R" page, under Lectures, "Mixed-effects models".

`https://scinet.courses/1391`

SciNet

Today's class

Today we will continue our adventures in data analysis.

- Random and fixed effects.
- Mixed-effects models.
- Random intercept model.
- Random slope model.
- Partial pooling.

As always, ask questions.

SciNet

Sometimes a linear model doesn't cut it

There are times when a linear model is not sufficient to capture all the characteristics of your data.

- Sometimes the independence assumption of your data is not true
  - repeated measures of the same subject ('longitudinal data'),
  - measurements that are part of specific groups that cause them to be correlated in some way,
- The data contain a natural hierarchy that should be respected: students within a class, TAs for students within a class.

If you find yourself in a situation where the data is correlated by group, whatever that might be, then a mixed-effects model might be appropriate. The important part is that there are groups in your data.

SciNet

Mixed-effects models

Mixed-effects models (also called "multilevel" or "hierarchical" models) are used when we wish to separately account for what are known as "fixed" and "random" effects.

- The "fixed" effects are your standard independent variables that we assume have some sort of effect on the dependent variable. These are the same across all groups.
- The "random" effects are (always) categorical variables that affect the dependent variable in a consistent way. These are groups, individuals or categories associated with your data.
- We want to control for the random variables in our model, since (we know) they are likely affecting the model.
- Random effects must have many (at least 5) different levels (values) for mixed effect models to work properly.

If random effects are correlated with the independent variables, leaving them out could lead to biases in the results.

SciNet

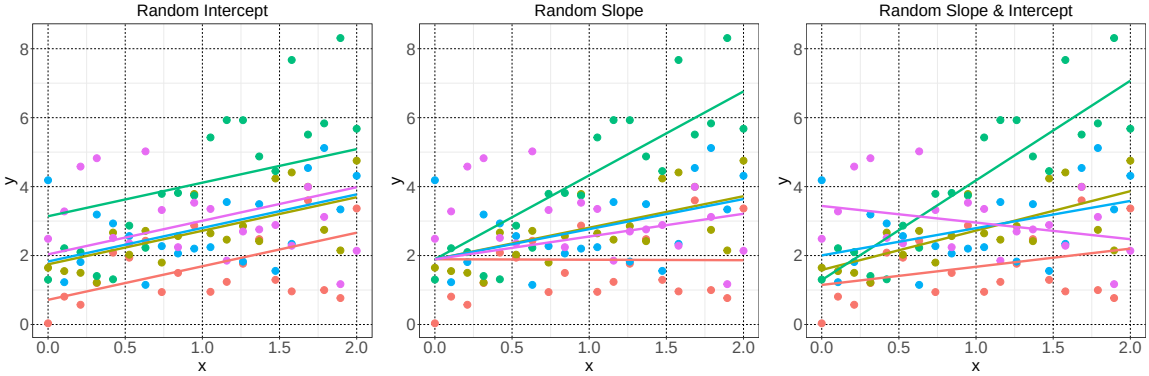
Types of mixed effects models

There are several ways to try to adjust our linear model for random effects.

- Random intercepts: $\boldsymbol{y = \beta_{0j} + \beta_1 x}$. In this case, the intercept is adjusted by group $\boldsymbol{j}$, but the slopes are the same for all groups.
- Random slopes: $\boldsymbol{y = \beta_0 + \beta_{1j} x}$. In this case, the slope is adjusted by group $\boldsymbol{j}$, but the intercepts are all the same.
- Random slopes and intercepts: $\boldsymbol{y = \beta_{0j} + \beta_{1j} x}$. In this case, both the intercept and slope is adjusted by group $\boldsymbol{j}$.

Usually either the intercept is a random effect, or both the intercept and the slope are random effects.

Types of mixed effects models, continued



Isn't that just a categorical variable?

Is this the same as a regular linear model with a categorical independent variable? Not quite:

- The difference lies in how the category-level structure in the data is treated.
- When a categorical variable is added to a linear model, it's treated as an independent variable, with each category value getting its own coefficient (β).
- This treats the category as a characteristic, allowing comparisons between categories.
- In a mixed-effects model, the categorical variable is treated not as an independent variable, but as a "random effect".
- The model will assume that categorical-level effects (variations within a category) are drawn from a normal distribution, sampled from a larger population.
- This allows for the calculation of the variance within the group, which helps clarify the non-independence of the groups.

Though they sometimes lead to similar results (for random intercepts), mixed-effect models are more flexible.

SciNet

Random intercepts models

There are several different types of mixed-effects models, depending on how the problem is crafted, and how things are set up. The first is known as the 'random intercepts' model:

$$\boldsymbol{y} = \boldsymbol{\beta_0} + \boldsymbol{\beta_1 x} + \boldsymbol{\beta_{\text{group}}} + \boldsymbol{\epsilon}$$

where you have a regular linear model, but with an extra random effect. This random effect is given by

$$\boldsymbol{\beta_{\text{group}}} = \boldsymbol{N}(\boldsymbol{0}, \boldsymbol{\tau_{\text{group}}})$$

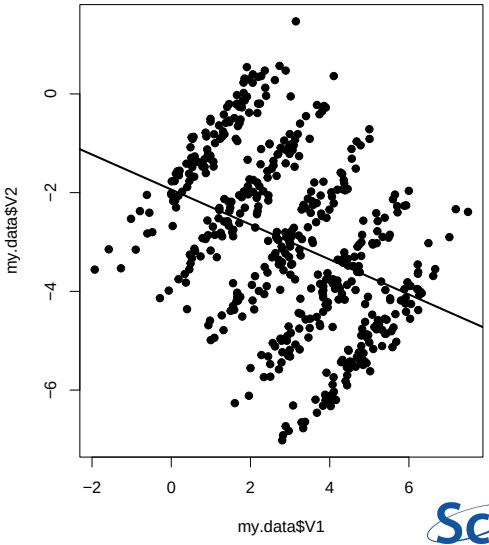
where $\boldsymbol{N}$ is the normal distribution with mean of 0 and a variance of $\boldsymbol{\tau_{\text{group}}}$. This value is fit for each group.

A similar derivation leads to the random-slopes model.

Random intercept model, example

We can use Simpson's paradox data to illustrate the utility of random intercept models. We'll generate the data explicitly.

```
> library(bayestestR)
>
> my.data <- simulate_simpson(r = 0.95,
+                             groups = 5)
> str(my.data)
'data.frame':  500 obs.  of 3 variables:
$V1 :  num 0.689 2.3089 1.4768 0.0618 ...
$V2 :  num -1.336 0.091 -0.377 -1.575 ...
$Group:  chr "G_1" "G_1" "G_1" "G_1" ...
>
> plot(my.data$V1, my.data$V2)
> m1 <- lm(V2 ~ V1, data = my.data)
> abline(m1)
```



SciNet

Random intercept model, example, continued

We use the lme4 package to create mixed-effects models.

The term "(1 | Group)" indicates that we're allowing the model intercept to vary by Group.

Because the random effects are drawn from a normal distribution centred on zero, there are no estimates for the values.

Notice the lack of p-values. These are complicated to calculate in a mixed-effects model, and are a subject of discussion.

```
> library(lme4)
> my.data$Group <- as.factor(my.data$Group)
> m2 <- lmer(V2 ~ V1 + (1 | Group), data = my.data)
>
> summary(m2)
⋮
Random effects:
 Groups            Name   Variance   Std.Dev.
 Group      (Intercept)     9.5018     3.0825
 Residual                   0.0977     0.3126
Number of obs:  500, groups:  Group, 5

Fixed effects:
             Estimate   Std. Error   t value
 (Intercept)  -5.84879      1.37925    -4.241
 V1            0.94960      0.01405    67.600
⋮
```

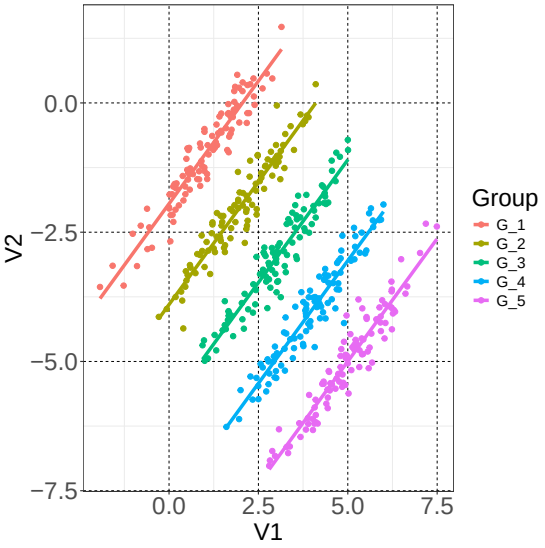
Random intercept model, example, continued more

We can also get the actual values of our random effects intercepts, for each group.

```
>
> ranef(m2)
$Group
          (Intercept)
  G_1    3.898789e+00
  G_2    1.949395e+00
  G_3    1.477107e-10
  G_4   -1.949395e+00
  G_5   -3.898789e+00
with conditional variances for ‘‘Group’’
>
```

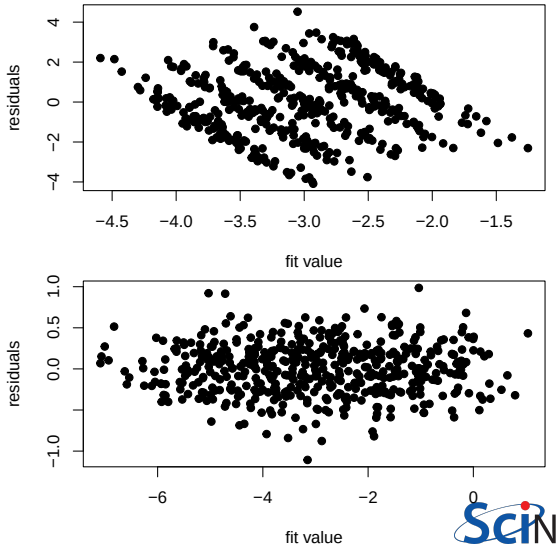
Random intercept model, example, plotted

```
>
> library(ggplot2)
>
> ggplot(my.data,
+        aes(x = V1, y = V2, col = Group)) +
+  geom_point() +
+  geom_line(data = cbind(my.data,
+                         y.hat = predict(m2)),
+            aes(x = V1, y = y.hat))
>
```



Random intercept model, example, residuals

```
>
> par(mfrow = c(2, 1))
>
> plot(predict(m1), m1$residuals,
+      pch = 21, bg = 'black',
+      xlab = 'fit value',
+      ylab = 'residuals')
>
> plot(predict(m2), resid(m2),
+      pch = 21, bg = 'black',
+      xlab = 'fit value',
+      ylab = 'residuals')
>
> par(mfrow = c(1, 1))
>
```

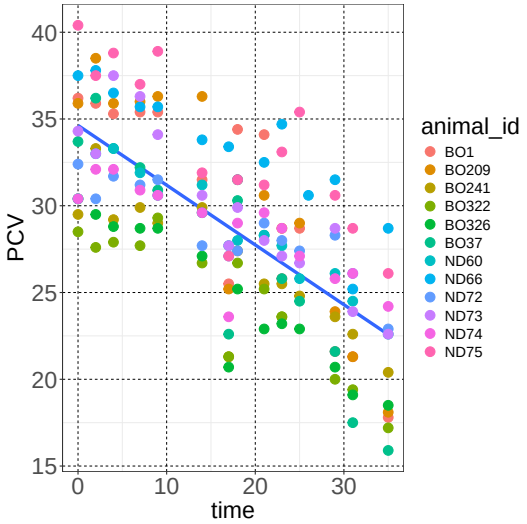


Random slope model, example

A random slope model works the same way as a random intercept model, except the random component affects the slope rather than the intercept.

We'll use the 'ex33' data set from the VetResearchLMM package.

```
> library(VetResearchLMM)
>
> ggplot(data = ex33,
+        aes(x = time, y = PCV)) +
+        geom_smooth(method = 'lm', se = F,
+                    lwd = 1.5) +
+        geom_point(size = 4,
+                   aes(colour = animal_id))
```



Random slope model, example, continued

In this case we include the term "(time | animal_id)", which indicates that the slope of time will vary by the animal id.

Just as in the past we used '1' to indicate the presence of an intercept in the model, here we use '0' to indicate no intercept dependence on the animal id.

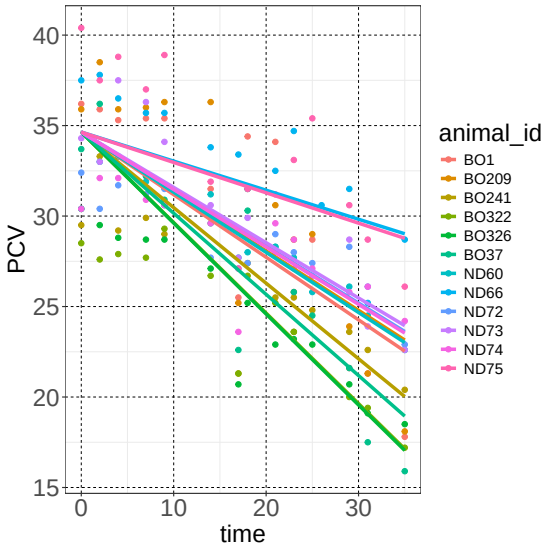
By default lmer will put in an intercept.

```
> model <- lmer(PCV ~ time + (0 + time | animal_id),
+               data = ex33)
>
> summary(model)
⋮
Random effects:
 Groups      Name   Variance   Std.Dev.
 animal_id   time    0.01343     0.1159
 Residual            7.66434     2.7685
Number of obs:  168, groups:  animal_id, 12

Fixed effects:
             Estimate   Std. Error   t value
 (Intercept)  34.64075      0.39444    87.822
 time         -0.34514      0.03886    -8.882
⋮
```

Random intercept model, example, plotted

```
>
> library(ggplot2)
>
> ggplot(ex33,
+        aes(x = time, y = PCV, col = animal_id)) +
+  geom_point() +
+  geom_line(data = cbind(my.data,
+                         y.hat = predict(model)),
+            aes(x = time, y = y.hat))
>
```



Crossed data

There are other types of mixed-effects data you need to know about:

- If you have data where each data point can be assigned to more than one random effect simultaneously, the data is said to be "crossed".
- An example would be a repeated measures study, where each subject is observed responding to several different effects.
- In this case the random effects are both the subjects be studied and the various responses being observed.
- In this case indicate that the model should respond to both random effects.
- A within-subjects study would be an example (all subjects are exposed to all values of a categorical variable).

```
> lmer(Response ~ Condition + (1 | Subject) + (1 | Item), data = my.data)
```

SciNet

Nested data

Sometimes our data aren't crossed, they're instead "nested":

- nested data occur when you have groups, and then groups within groups (subgroups).
- all of these groups are independent of each other.
- An example might be:
  - schools,
  - classes within the school,
  - students within the class
- There is a hierarchy of data groups.

```
> lmer(Outcome ~ Condition + (1 | School/Class/Student), data = my.data)
```

SciNet

Partial pooling

The term "pooling" refers to how much information is shared between groups.

- If no information is shared between groups the model is sometimes called a "no pooling" model. This means each group is fit separately from all the other groups.
- If all information is shared, regardless of groups, such as in our standard linear model using the lm function, the model is called "complete pooling".
- The term "partial pooling" refers to a mixed-effects model, wherein a model shares information across groups, but also accounts for random effects within groups.

Why do we care? Partial pooling allows us to "help out" groups that have less data than other groups.

SciNet

Partial pooling example

Let's explore the "sleepstudy" data set, which comes with the lme4 package.

This data contains the measured reaction times of subjects who were deprived of sleep.

To illustrate partial-pooling we will add two new subjects, who have incomplete data.

Example stolen from Glenn Williams and Tristan Mahr.

```
> library(tidyverse)
> str(sleepstudy)
'data.frame':  180 obs.  of 3 variables:
$Reaction: num 250 259 251 321 357 ...
$Days : num 0 1 2 3 4 5 6 7 8 9 ...
$Subject : Factor w/ 18 levels "308","309",..: 1 1 ...
>
> my.data <- sleepstudy
>
> my.data <- rbind(my.data, data.frame(Days = 0:1,
+                  Reaction = c(286, 288),
+                  Subject = "374"))
>
> my.data <- rbind(my.data, data.frame(Days = 0,
+                  Reaction = 245, Subject = "373"))
>
```

Partial pooling example: complete pooling

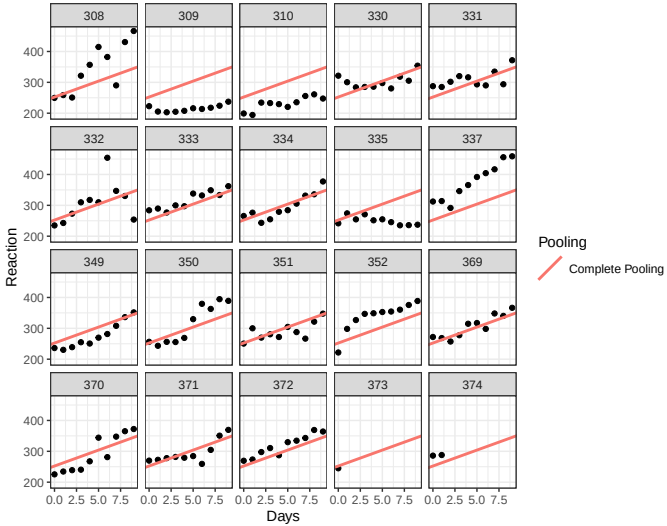
First, complete pooling.

```
>
> complete.pooling <- lm(Reaction ~ Days, data = my.data)
>
> complete <- data.frame(Subject = levels(my.data$Subject),
+                        Intercept = coef(complete.pooling)[[1]],
+                        Slope = coef(complete.pooling)[[2]],
+                        Model = "Complete Pooling")
>
> comp.coefs <- left_join(my.data, complete, by = "Subject")
>
```

Partial pooling example: complete pooling, continued

```
>
> p <- ggplot(data = comp.coefs,
+      aes(x = Days, y = Reaction)) +
+   geom_point() + theme_bw() +
+   facet_wrap(~Subject) +
+   geom_abline(
+     aes(intercept = Intercept,
+         slope = Slope,
+         color = Model)) +
+   labs(color = 'Pooling')
>
```

All information is shared, so the line is the same for every plot.



Partial pooling example: no pooling

Next we will build the no-pooling case. This means that each group gets an independent linear fit.

Again, we need to explicitly indicate that we want no intercept, this time for the fixed effects.

We create an Intercept column, as it will be easier to reference. R doesn't like parentheses at the start of column names.

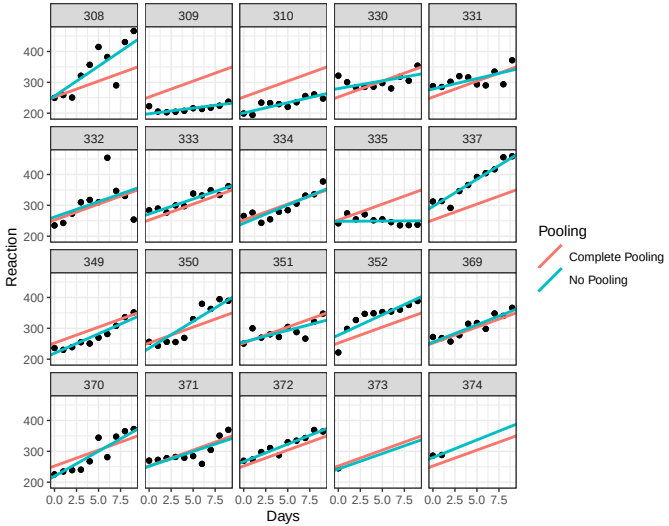
```
>
> no.pooling <- lmer(Reaction ~ 0 + (Days | Subject),
+                    data = my.data)
>
> no.pooling.coefs <- coef(no.pooling)$Subject
>
> no.pooling.coefs[["Intercept"]] <-
+              no.pooling.coefs[["(Intercept)"]]
>
> none <- data.frame(
+           Subject = levels(my.data$Subject),
+           Intercept = no.pooling.coefs$Intercept,
+           Slope = no.pooling.coefs$Days,
+           Model = "No Pooling")
>
```

Partial pooling example: no pooling, continued

The bind_rows function is part of tidyverse.

What is this weird syntax? The "%+%" operator comes with the tidyverse package.

```
>
> none.coefs <- left_join(my.data,
+               none, by = "Subject")
>
> comp.none <- bind_rows(comp.coefs,
+                 none.coefs)
>
> p %+% comp.none
>
```



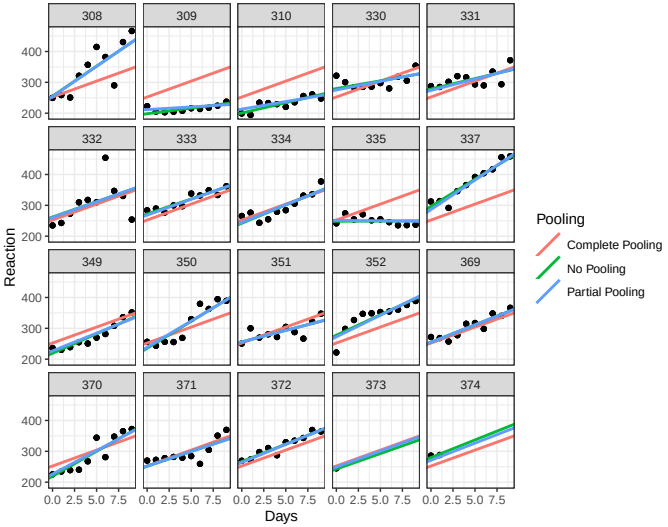
Partial pooling example: partial pooling

Next we will build the partial-pooling case. This means that each group gets a linear fit, using the information gained from the other groups.

```
>
> partial.pooling <- lmer(Reaction ~ Days +
+                           (1 + Days | Subject),
+                           data = my.data)
>
> partial.pooling.coefs <- coef(partial.pooling)$Subject
>
> partial.pooling.coefs[["Intercept"]] <-
+               partial.pooling.coefs[["(Intercept)"]]
>
> partial <- data.frame(
+            Subject = levels(my.data$Subject),
+            Intercept = partial.pooling.coefs$Intercept,
+            Slope = partial.pooling.coefs$Days,
+            Model = "Partial Pooling")
>
```

Partial pooling example: partial pooling, continued

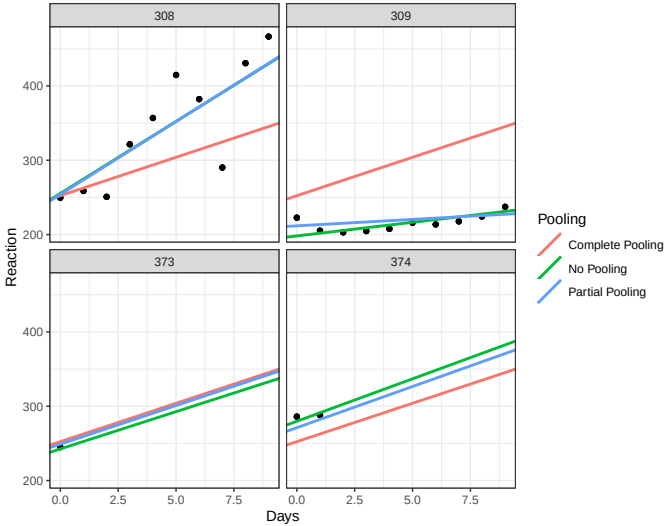
```
>
> partial.coefs <- left_join(my.data,
+         partial, by = "Subject")
>
> all.pools <- bind_rows(comp.none,
+                  partial.coefs)
>
> p %+% all.pools
>
```



Partial pooling, what's the point?

Partial pooling allows groups with less data to leverage the behaviour of the other groups.

The no-pooling result tends to get pulled toward the average, "complete pooling", result. This is called "shrinkage".



Mixed-effect model assumptions

Like all models, mixed-effect models come with their own set of assumptions.

- The data should be bivariate normality within groups.
- The coefficients for the random variables are assumed to follow a normal distribution, with a mean of zero.
- Homoscedasticity of the noise, across groups and independent variables.
- If your data are not independent, such as when you have multiple samples from a single subject, you must consider that a group.
- Multicollinearity can be a problem with mixed-effects models, just as in regular linear models.

These should be familiar, as they are mostly the usual linear-model assumptions we've made in the past.

SciNet

Other considerations

Further notes about mixed-effect models

- If your model is not homoskedastic, you may wish to consider generalized mixed-effect models. These are similar to the mixed effect models we've discussed:
  - specify a link function,
  - specify a noise family,
  - use the "glmer" function instead.
- Power analysis of mixed-effect models is possible, but this requires direct simulation rather than mere theory. Look into the "simr" package if you're interested.

SciNet

Summary

Mixed-effect models separate our "fixed effects" from our "random effects".

- Fixed effects are our linear models as usual.
- Random effects are group effects which account for correlations of data within a group.
- Random intercepts and random slopes can be fit by group.
- Partial pooling of the model results in data from other groups affect the predictions of the final model for other groups.
- This allows groups with less data to leverage the behaviour of data in other groups.

SciNet