A primer on Mixed-Effects Models

Gonzalo García-Castro

25th March 2020

First of all...

Disclaimer

I'm not a trained statistician

Don't trust me (too much)

Mistakes may (will) be made

I'm not 100% sure about anything

Probably, none of us will ever be

So let's get to it!

Notation

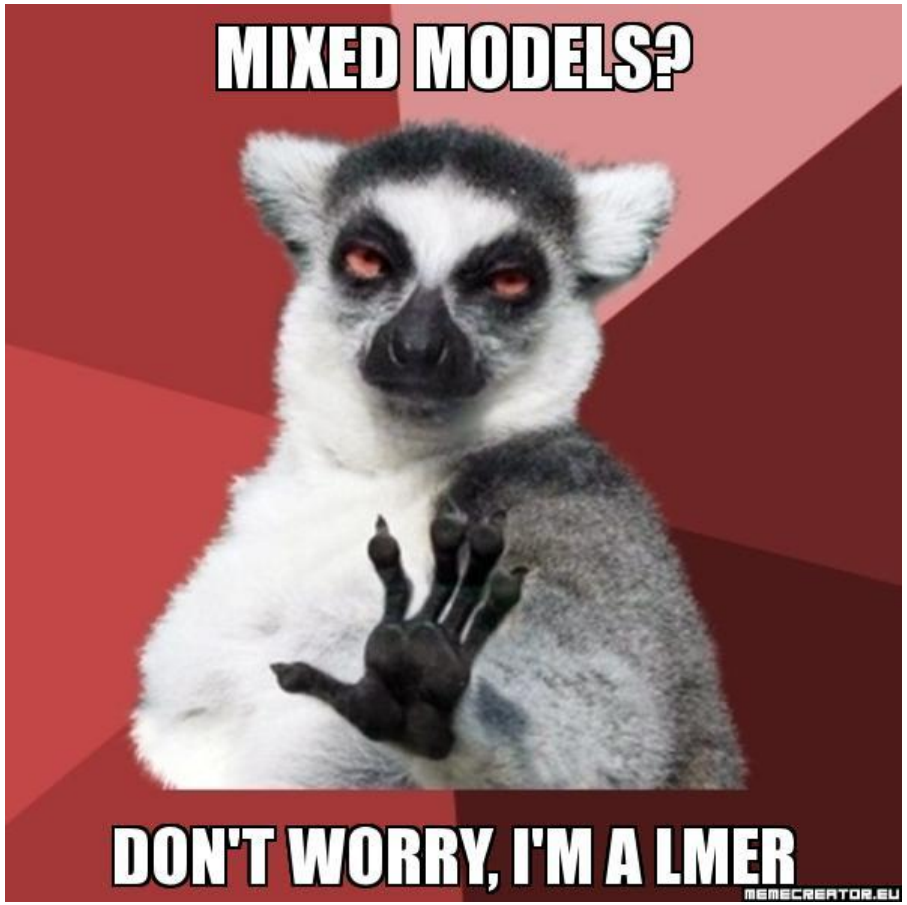
- Linear Mixed-Effects Models = LMM
- **Mixed Models** (aka. Mixed-Effects Models, aka. Multilevel Models, aka. Hierarchical Models, aka. Nested Data Models, aka. Random Parameter Models, aka. Split-Plot Designs)
- If by the end of this presentation you have an intuition about why all **these labels refer to the same thing:** Yay! You have made so much progress.

Disclaimer (bonus 1)

We need to learn a bit about programming.

Disclaimer (bonus 2)

Most statistical literature on LMM uses **R** (e.g., lmer() function of the lme4 package)



Disclaimer (bonus 2)

Many other programming languages support LMM:

- **Python**: `statsmodels` library, `Pymer4`
- **Matlab**: `Statistics and Machine Learning Toolbox`
- **Julia** (explicitely created for LMM)
- **Stan**: Bayesian modelling, with interfaces with R, Python and Matlab)

I don't assume you are familiar with R, but I will use some R code for illustration purposes.

Materials

Some recommended reads.

Materials: Books and book chapters

Navarro, D. J. (2015). Learning statistics with r: A tutorial for psychology students and other beginners. [INTRO – No LMM]

Field, A., Miles, J., & Field, Z. (2012). 19. Multilevel linear models. In Discovering statistics using r. SAGE Publications. [INTRO]

Winter, B. (2019). Statistics for linguists: An introduction using R. Routledge. [INTRO]

Mirman, D. (2016). 4. Structuring random effects. In Growth curve analysis and visualisation using r. CRC press. [MEDIUM – Generalisation]

Gelman, A., & Hill, J. (2006). Data analysis using regression and multilevel/hierarchical models. Cambridge University press. [MEDIUM]

Fox, J., & Weisberg, S. (2018). An r companion to applied regression. SAGE publications. [MEDIUM]

McElreath, R. (2020). Statistical rethinking: A bayesian course with examples in r and stan. CRC press. [DIFFICULT]

Materials: Articles

Winter, B. (2013). Linear models and linear mixed effects models in R with linguistic applications. arXiv Preprint arXiv:1308.5499. https://arxiv.org/abs/1308.5499 [INTRO]

DeBruine, L., & Barr, D. J. (2019). Understanding mixed effects models through data simulation. https://doi.org/10.31234/osf.io/xp5cy [INTRO]

Barr, D. J., Levy, R., Scheepers, C., & Tily, H. J. (2013). Random effects structure for confirmatory hypothesis testing: Keep it maximal. Journal of Memory and Language, 68(3), 255–278. [MEDIUM]

Barr, D. J. (2013). Random effects structure for testing interactions in linear mixed-effects models. Frontiers in Psychology, 4, 328. https://www.frontiersin.org/articles/10.3389/fpsyg.2013.00328/full [MEDIUM]

Bates, D., Kliegl, R., Vasishth, S., & Baayen, H. (2015a). Parsimonious mixed models. arXiv Preprint arXiv:1506.04967. https://arxiv.org/pdf/1506.04967.pdf [MEDIUM]

Baayen, R. H., Davidson, D. J., & Bates, D. M. (2008). Mixed-effects modelling with crossed random effects for subjects and items. Journal of Memory and Language, 59(4), 390–412. https://doi.org/10.1016/j.jml.2007.12.005 [DIFFICULT]

Barr, D. J. (2008). Analyzing “visual world” eyetracking data using multilevel logistic regression. Journal of Memory and Language, 59(4), 457–474. https://doi.org/10.1016/j.jml.2007.09.002 [DIFFICULT]

Materials: Articles (special mentions)

Meteyard, L., & Davies, R. A. (2020). Best practice guidance for linear mixed-effects models in psychological science. Journal of Memory and Language, 112, 104092. https://doi.org/10.1016/j.jml.2020.104092

Bates, D., Mächler, M., Bolker, B., & Walker, S. (2015b). Fitting linear mixed-effects models using lme4. Journal of Statistial Software, 67(1). https://doi.org/10.18637/jss.v067.i01

Modelling

A reminder.

Modelling: What for?

To estimate a **parameter** that characterises a population using data from samples of that population.

This unknown **parameter** can be whatever we want:

- Central tendency measures (e.g., mean, median)
- Dispersion measures (e.g., standard deviation)
- Measures of association between variables (e.g., coefficients)

We usually estimate all of them when analysing data and perform some kind of **statistical inference** from them.

Modelling: What for?

In confirmatory analyses, we:

1. Hypothesise that the parameter is within a range of values
2. Collect data
3. Perform statistical inference

Frequentist approach: what is the probability of our data, assuming our hypothesis is true?

Bayesian approach: what is the probability of our hypothesis, given the data?

Modelling: What for?

In experimental confirmatory research, we are usually interesed in **estimating the association between two or more variables**:

- *Age* (predictor) and *vocabulary size* (outcome)
- *Bilingualism* (predictor) and *novelty preference* (outcome)
- *Native linguistic rythmic class* (predictor) and *entrainment to speech signal* (outcome)

We try draw a shape (e.g., line, curve) that defines this relationship.

For now, we will stick to óne shape: **lines**.

Modelling: What for?

The three (four, sometimes) steps of modelling:

1. **Model specification**: what variables am I going to include in the model?
2. **Model fitting**: what line fits data the best?
3. **Statistical inference**: Does my model fit data good enough? What is the contribution of each predictor to the goodness of fit?
4. **Model validation**: Does my model predict new outcomes correctly?

Modelling: 1) Model specification

We need to define what **outcomes** and **predictors** I am interested in.

- e.g., what am I trying to predict? What variables am I using to predict it?

What are my **assumptions** about how they relate to the outcome?

- e.g., linearity, normality of residuals

What are my assumptions about **how each predictor relates to the other predictors**?+

- e.g., do I expect interactions between predictors?

Modelling: 1) Model specification

We will work with **linear** models

This means that we will try to draw a **line** that defines the relationship between predictors and outcome.

But bear in mind that **non-linear models** exist as well

For instance, we may need/want to fit a **curve** or an **exponential** function

- e.g., Growth Curve Analysis

Modelling: 1) Model specification

Every line is defined by the following equation, named the **General Linear Model (GLM)**:

$$Y = \beta_0 + \beta_j X_j + \varepsilon$$

Where:

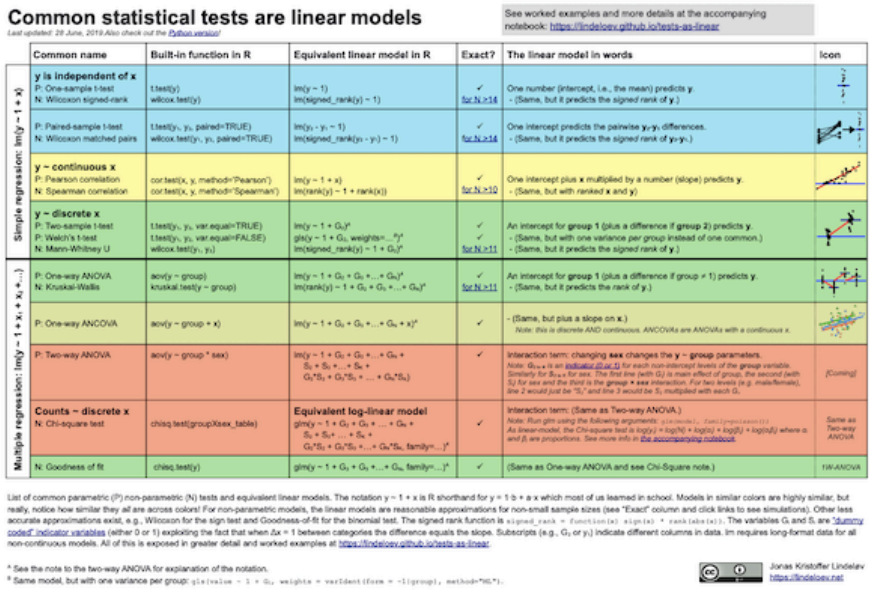
- Y is the value that the outcome variable takes (we know this value)
- β_0 is the intercept
- β_j is the coefficient
- X_j is the value that the predictor XX takes (we know this value)
- ε is the residual (error the model makes)

Modelling: 1) Model specification

$$Y = \beta_0 + \beta_j X_j + \varepsilon$$

This formula underlies most statistical techniques we use normally.

Most of them are generalised or special cases of the the **GLM**.



Common statistical tests are linear models

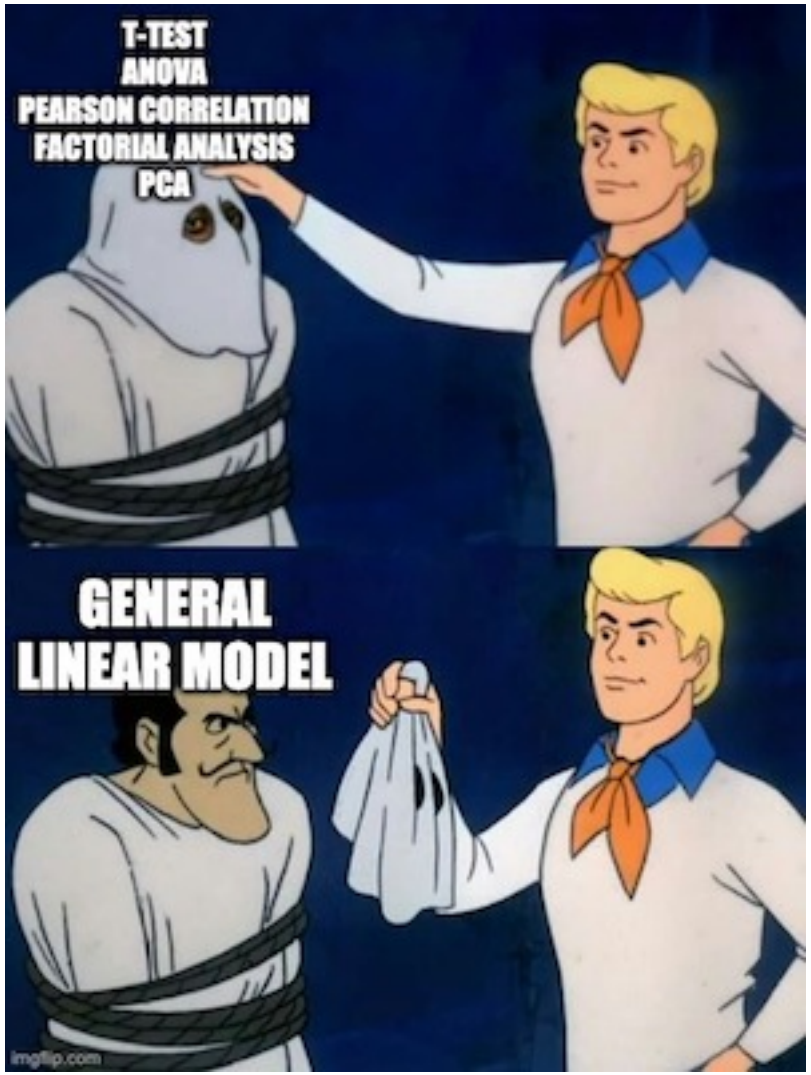
See worked examples and more details at the accompanying notebook: https://lindeloev.github.io/tests-as-linear

	Common name	Built-in function in R	Equivalent linear model in R	Exact?	The linear model in words	Icon
Simple regression: lm(y ~ 1 + x)	**y is independent of x** P: One-sample t-test N: Wilcoxon signed-rank	t.test(y) wilcox.test(y)	lm(y ~ 1) lm(signed_rank(y) ~ 1)	✓ for N >14	One number (intercept, i.e., the mean) predicts **y**. - (Same, but it predicts the *signed rank* of **y**.)	
	P: Paired-sample t-test N: Wilcoxon matched pairs	t.test(y₁, y₂, paired=TRUE) wilcox.test(y₁, y₂, paired=TRUE)	lm(y₁ - y₂ ~ 1) lm(signed_rank(y₁ - y₂) ~ 1)	✓ for N >14	One intercept predicts the pairwise **y₁-y₂** differences. - (Same, but it predicts the *signed rank* of **y₁-y₂**.)	
	y ~ continuous x P: Pearson correlation N: Spearman correlation	cor.test(x, y, method='Pearson') cor.test(x, y, method='Spearman')	lm(y ~ 1 + x) lm(rank(y) ~ 1 + rank(x))	✓ for N >10	One intercept plus **x** multiplied by a number (slope) predicts **y**. - (Same, but with ranked **x** and **y**)	
	y ~ discrete x P: Two-sample t-test P: Welch's t-test N: Mann-Whitney U	t.test(y₁, y₂, var.equal=TRUE) t.test(y₁, y₂, var.equal=FALSE) wilcox.test(y₁, y₂)	lm(y ~ 1 + G₂)* gls(y ~ 1 + G₂, weights= ...*)* lm(signed_rank(y) ~ 1 + G₂)*	✓ ✓ for N >11	An intercept for **group 1** (plus a difference if **group 2**) predicts **y**. - (Same, but with one variance *per group* instead of one common.) - (Same, but it predicts the *signed rank* of **y**.)	
Multiple regression: lm(y ~ 1 + x₁ + x₂ + ...)	P: One-way ANOVA N: Kruskal-Wallis	aov(y ~ group) kruskal.test(y ~ group)	lm(y ~ 1 + G₂ + G₃ +...+ G₆)* lm(rank(y) ~ 1 + G₂ + G₃ +...+ G₆)*	✓ for N >11	An intercept for **group 1** (plus a difference if group ≠ 1) predicts **y**. - (Same, but it predicts the *rank* of **y**.)	
	P: One-way ANCOVA	aov(y ~ group + x)	lm(y ~ 1 + G₂ + G₃ +...+ G₆ + x)*	✓	- (Same, but plus a slope on **x**.) *Note: this is discrete AND continuous. ANCOVAs are ANOVAs with a continuous x.*	
	P: Two-way ANOVA	aov(y ~ group * sex)	lm(y ~ 1 + G₂ + G₃ +...+ G₆ + S₂ + S₂ +...+ S₆ + G₂*S₂ + G₂*S₃ + ... + G₆*S₆)	✓	Interaction term: *changing* **sex** changes the **y ~ group** parameters. *Note: $G_{6,4}$ is an indicator (0 or 1) for each non-intercept levels of the group variable. Similarly for $S_{6,4}$ for sex. The first line (with G) is main effect of group, the second (with S) for sex and the third is the group × sex interaction. For two levels (e.g. male/female), line 2 would just be "S_2" and line 3 would be S_2 multiplied with each G_i.*	[Coming]
	Counts ~ discrete x N: Chi-square test	chisq.test(groupXsex_table)	**Equivalent log-linear model** glm(y ~ 1 + G₂ + G₃ + ... + G₆ + S₂ + S₂ +... + S₆ + G₂*S₂ + G₂*S₃ + ... + G₆*S₆, family=...)*	✓	Interaction term: (Same as Two-way ANOVA.) *Note: Run glm using the following arguments: `glm(model, family=poisson())`* *As linear-model, the Chi-square test is log(y) = log(N) + log(α) + log(β) + log(αβ) where α and β are proportions. See more info in the accompanying notebook.*	Same as Two-way ANOVA
	N: Goodness of fit	chisq.test(y)	glm(y ~ 1 + G₂ + G₃ +...+ G₆, family=...)*	✓	(Same as One-way ANOVA and see Chi-Square note.)	1W-ANOVA

List of common parametric (P) non-parametric (N) tests and equivalent linear models. The notation y ~ 1 + x is R shorthand for y = 1·b + a·x which most of us learned in school. Models in similar colors are highly similar, but really, notice how similar they all are across colors! For non-parametric models, the linear models are reasonable approximations for non-small sample sizes (see "Exact" column and click links to see simulations). Other less accurate approximations exist, e.g., Wilcoxon for the sign test and Goodness-of-fit for the binomial test. The signed rank function is `signed_rank = function(x) sign(x) * rank(abs(x))`. The variables G_i and S_i are "dummy coded" indicator variables (either 0 or 1) exploiting the fact that when $\Delta x = 1$ between categories the difference equals the slope. Subscripts (e.g., G_2 or y_2) indicate different columns in data. lm requires long-format data for all non-continuous models. All of this is exposed in greater detail and worked examples at https://lindeloev.github.io/tests-as-linear

* See the note to the two-way ANOVA for explanation of the notation.
* Same model, but with one variance per group: `gls(value ~ 1 + G₂, weights = varIdent(form = ~1|group), method="ML")`.

Jonas Kristoffer Lindeløv
https://lindeloev.net



Modelling: 1) Model specification

We can extend the GLM to include more predictors:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_j X_j$$

Each coefficient (e.g., β_1, β_2) will contribute to the model by "adjusting" the line.

But how much should each coefficient contribute?

We need to take a look at the data.

Modelling: 1) Model specification

There are infinite lines we can draw with these parameters.

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_j X_j$$

We need to find the combination of values of the coefficients β0β0 and βjβj that make the line fit data the best.

We need to find the **Least Squares Regression Line**

This line **minimises the overall distance between the lines and the data points**

How to find it? We use built-in **algorithms** that try a lot of combinations until finding the optimal one.

Modelling: 2) Model fitting

Play with regression lines:

https://antoinesoetewey.shinyapps.io/statistics-202/

Modelling: 3) Statistical inference

We have obtained **coefficients** that define the slope and position of the line that fits data the best.

But **"best"** is relative.

Even the best model could be fitting the data very badly

We need to assess how much of the variance our model accounts for.

Compute compute some measure of goodness of fit

- e.g., R^2: proportion of variance accounted for the model

If the model shows nice fit, let's smove on to **interpret the coefficients**!

Modelling: 3) Statistical inference

We have a bunch of **coefficients**

Each coefficient tells us **how much each predictor contributes to the slope** of the line

All coefficients contribute to some degree. $\beta_j = 0$ is almost impossible.

What coefficients contribute significantly?

Modelling: 3) Statistical inference

Imagine that the **true (population) value of coefficient** is 0

This means that it has **no predictive value** regarding the outcome variable

How what is the probability of β_1 taking the value it takes in our sample, **assuming that** $\beta_1 = 0$ is true?

If it is very **unlikely**, we should **reject** the assumption that $\beta_1 = 0$

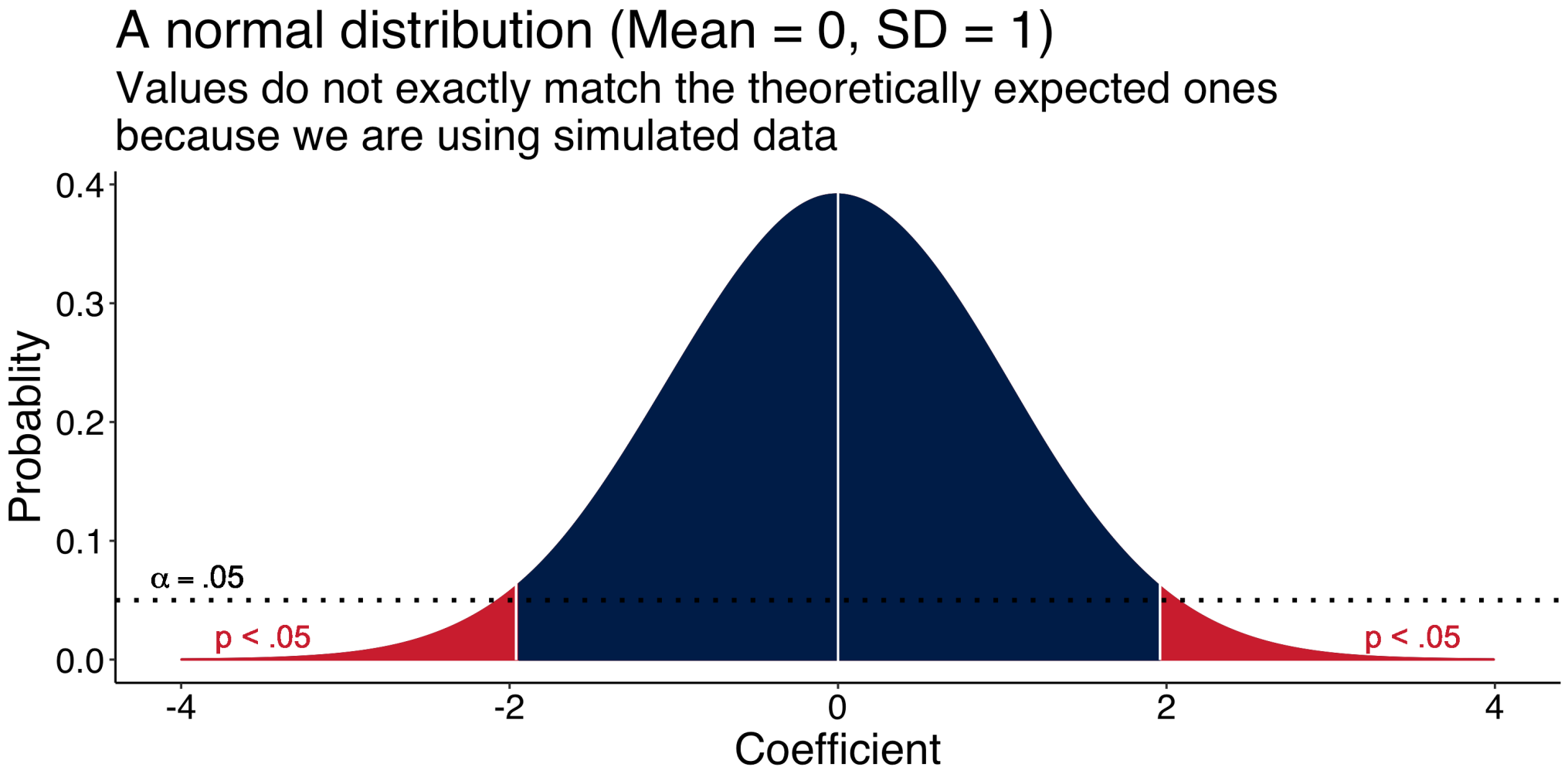
If its **likely enough**, we **don't reject** the assumption that $\beta_1 = 0$

Modelling: 3) Statistical inference

We know that **coefficients** in linear models tend to follow a **Gaussian distribution**. This distribution is known.

It tells us the **probability** of each value of our coefficient, **assuming that its true value is 0**.

Modelling: 3) Statistical inference



Modelling: 3) Statistical inference

We can map the probability of our coefficient onto this distribution to find its **associated probability**

If the probability is lower than our **significance threshold** (e.g., $\alpha = .05$), we **reject the hypothesis** that the true value of the coefficient is 0!

We **interpret** the model goodness of fit, the coefficients, and draw (or do not draw) conclusions from them.

Modelling: Assumptions!

We assume many things, one of the, being that observations are independent from each other.

This means that the **probability** of one observation taking one value is independent from the value other observations have taken.

Introducing random effects in our linear model

Beyond fixed effects

What is a LMM?

An **extension of the GLM** that allows to account for **systematic sources of variability** beyond our effects of interest.

All models are **inaccurate** to some degree.

Sometimes, part of the error the model makes is **systematic (not random).**

- e.g., data points from the same participant are correlated
- e.g., data points from the same trial are correlated

Why to use a LMM

To **avoid aggregating data** (more statistical power)

To account for **non-independence** of scores

To account for **hierarchical structures** in our data

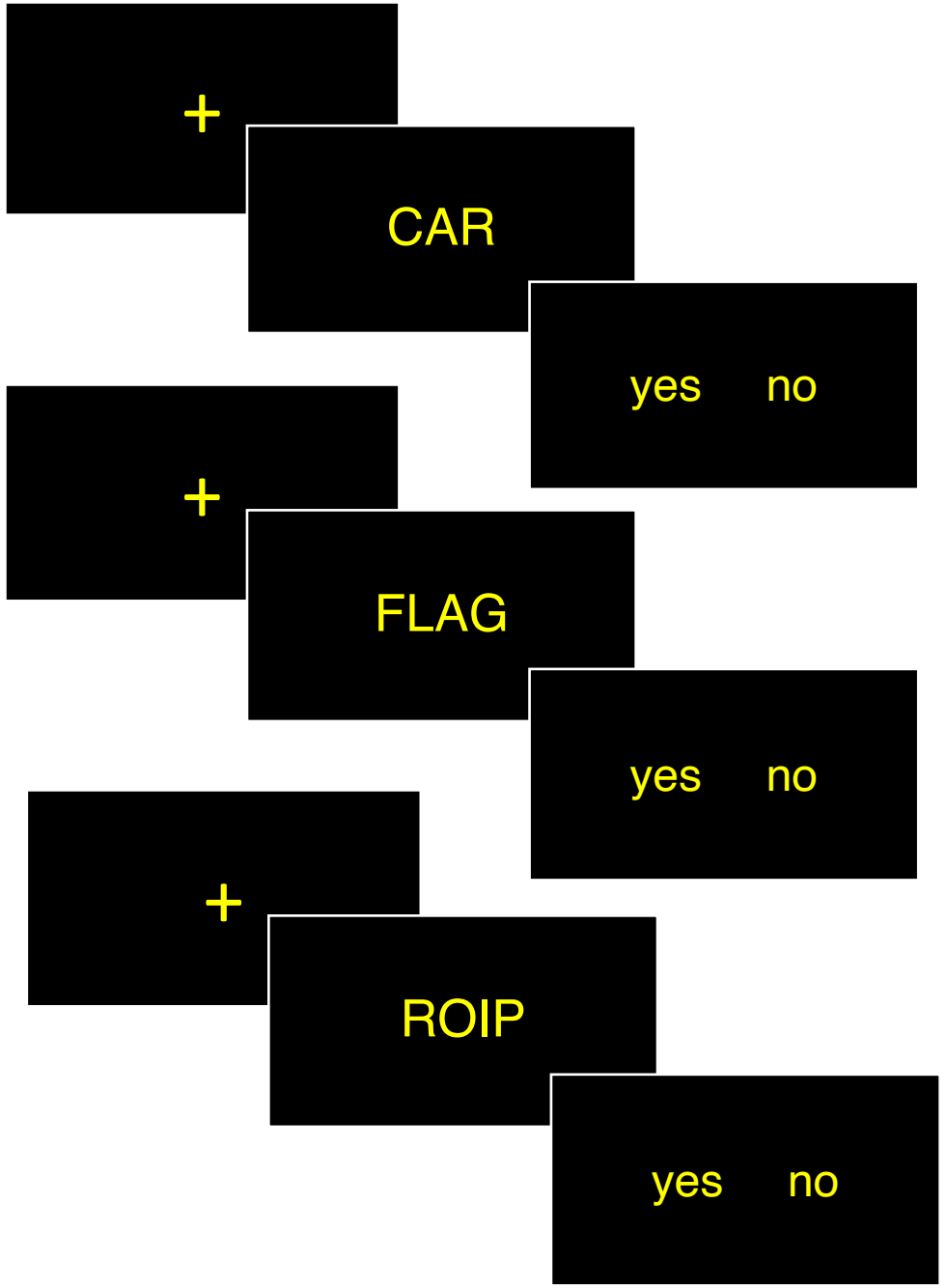
To account for **cross-observational** unit variability

Why to use a LMM

Effect of word frequency on reaction time task in a lexical decision task.

- Word condition:
  - High frequency: **CAR** > Yes/No
  - Low frequency: **FLAG** > Yes/No
- Non-word condition: **FOIR** > Yes/No

Does *frequency* affect *reaction times* (*RT*)?



Why to use a LMM

Trial-wise data:

Sample size:

- 5 participants
- 10 trials each: 5 in each frequency condition
- 50 observations

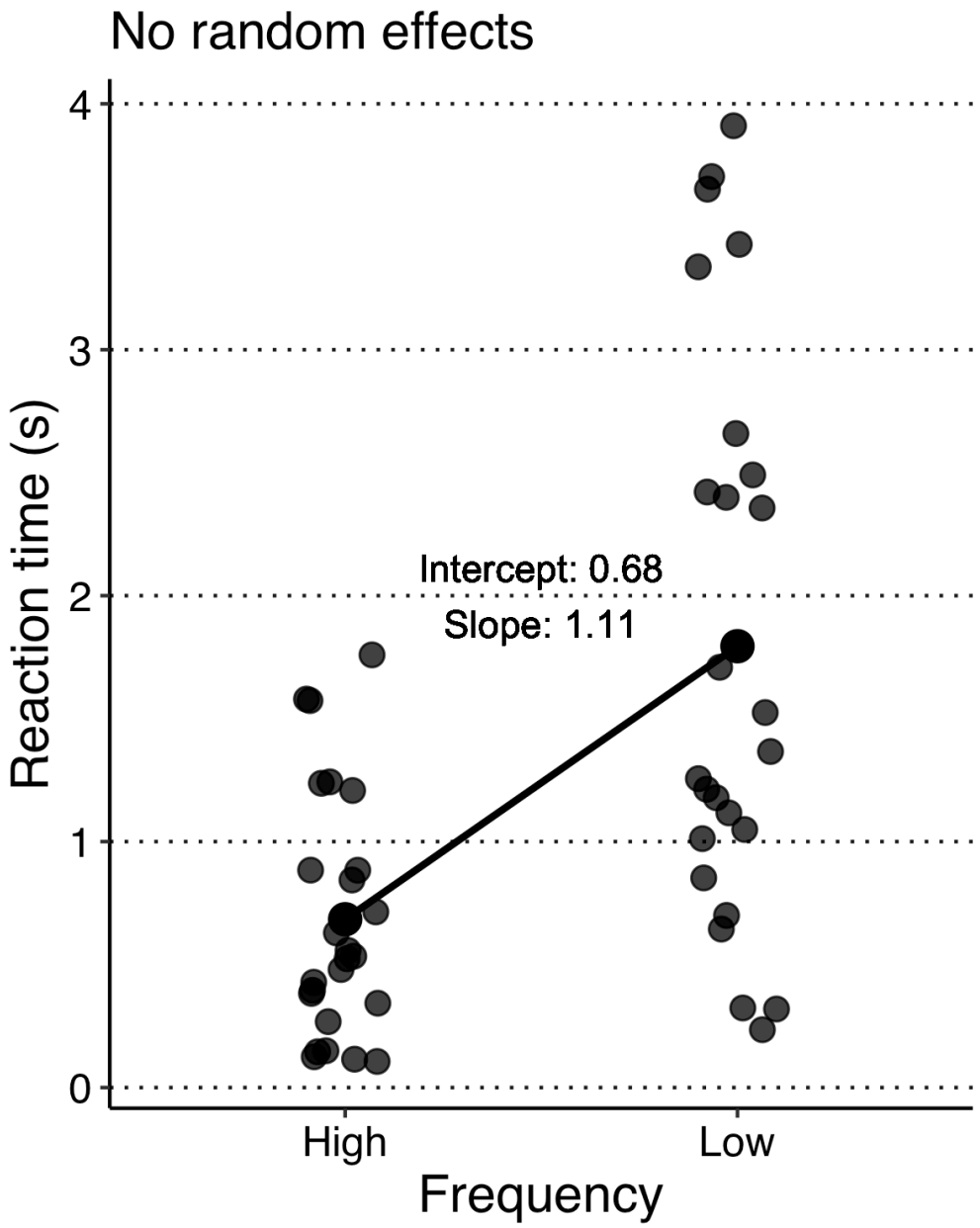
Participant	Trial	Frequency	RT (s)
Participant 1	1	High	0.15
Participant 1	2	High	0.38
Participant 1	3	High	0.39
Participant 1	4	High	0.11
Participant 1	5	High	0.27
Participant 1	6	Low	0.70
Participant 1	7	Low	0.24
Participant 1	8	Low	0.32
Participant 1	9	Low	0.32
Participant 1	10	Low	0.64
Participant 2	1	High	0.43
Participant 2	2	High	0.15
Participant 2	3	High	0.12
Participant 2	4	High	0.48
Participant 2	5	High	0.53
Participant 2	6	Low	1.05
Participant 2	7	Low	1.12
Participant 2	8	Low	1.01
Participant 2	9	Low	1.21
Participant 2	10	Low	1.18
…	…	…	…
Participant 5	10	Low	3.43

Why to use a LMM

Data-points from all participants are **pooled together.**

RT in **high frequency** condition is **0.68 s** (intercept)

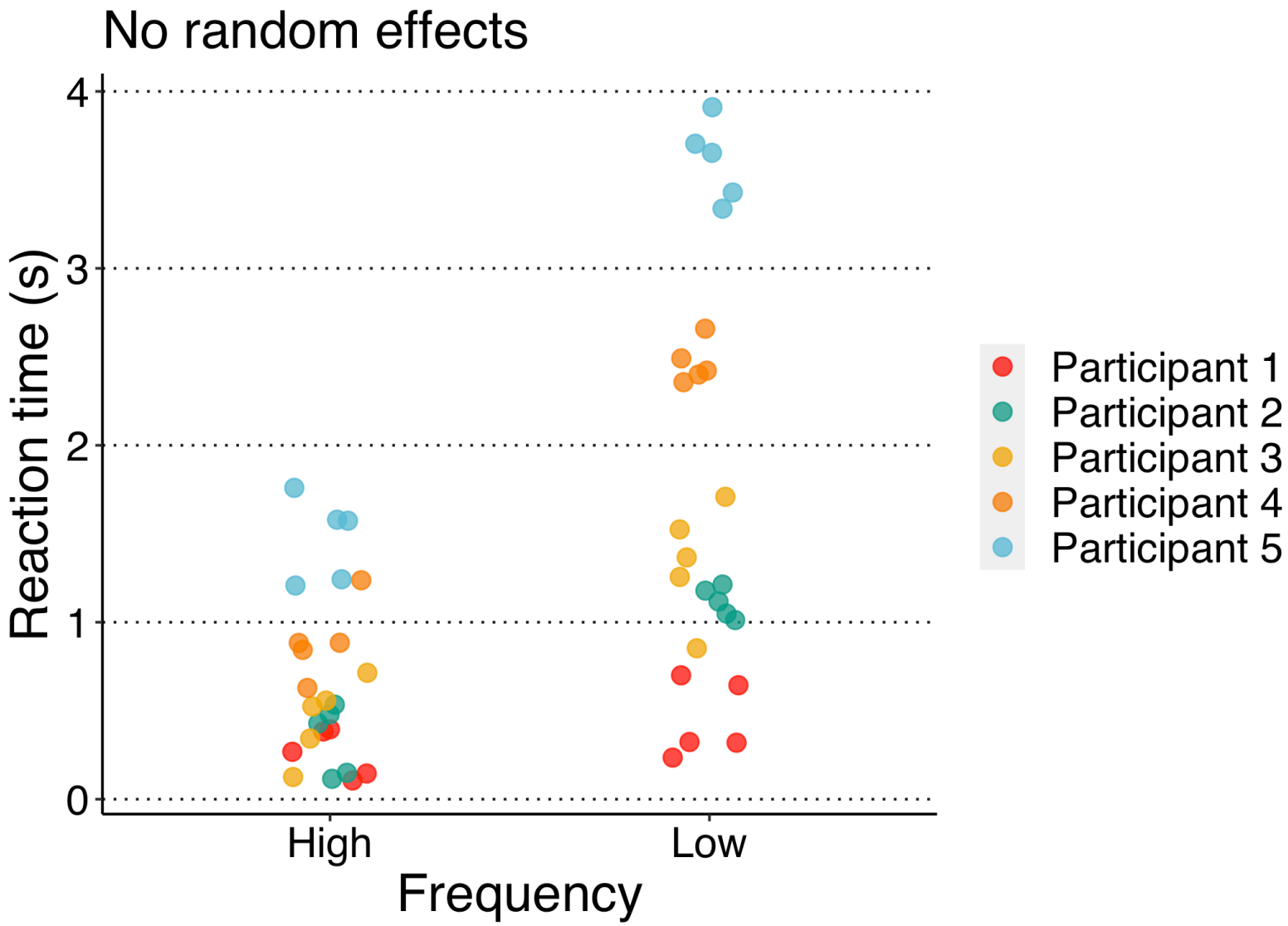
RT increases in **1.11 s**. (slope) when in **low frequency** condition.



Why to use a LMM

Assumption of **non-independence** is hard to assume here.

Part of the variability is due to **systematic differences across participants.**



Why to use a LMM

A common practice to avoid this is to **aggregate data points** across trials for each participant

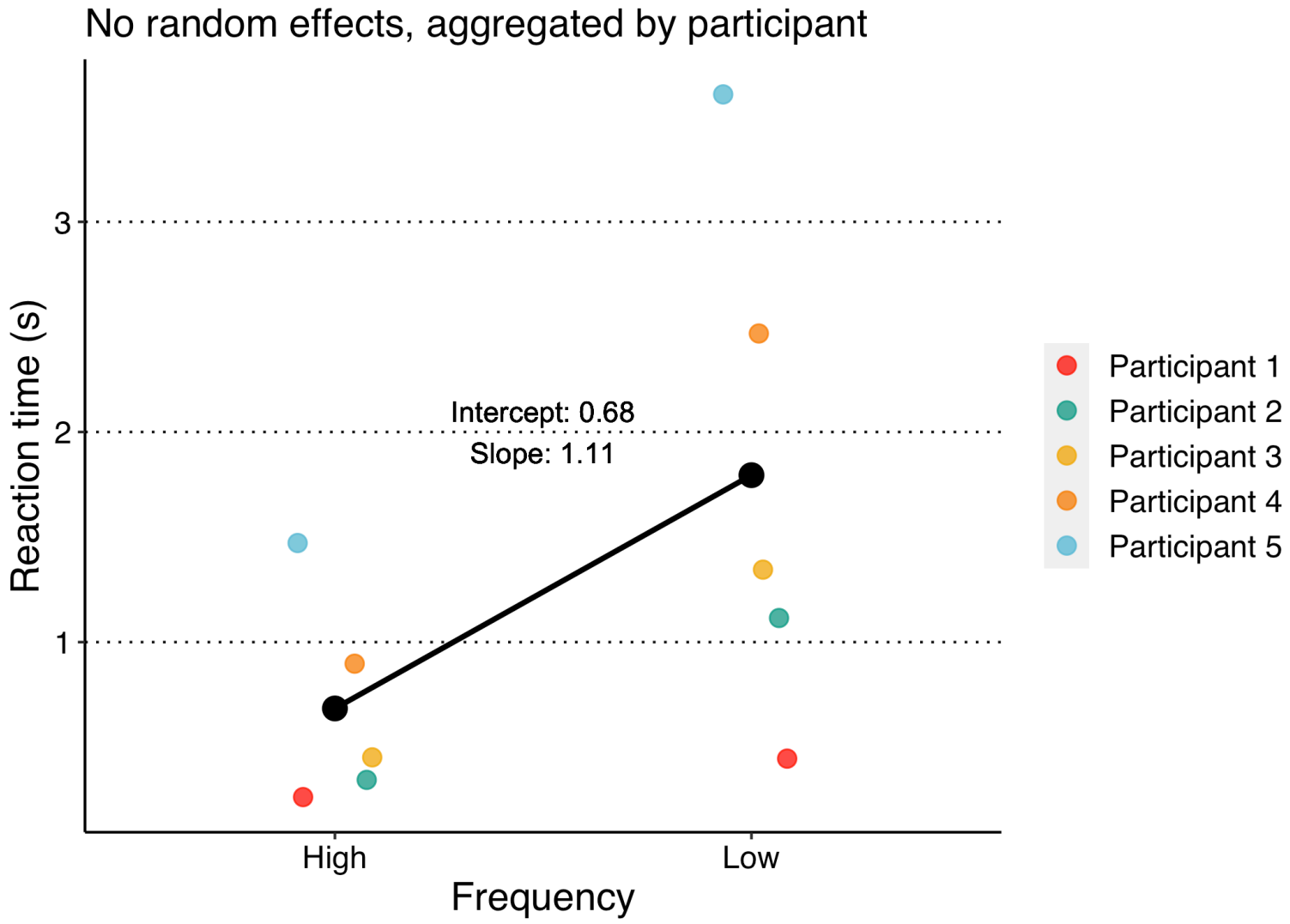
- e.g., mean RT across trials in each condition.

Participant	Frequency	Mean RT (s)
Participant 1	High	0.26
Participant 1	Low	0.44
Participant 2	High	0.34
Participant 2	Low	1.11
Participant 3	High	0.45
Participant 3	Low	1.34
Participant 4	High	0.90
Participant 4	Low	2.47
Participant 5	High	1.47
Participant 5	Low	3.61

Why to use a LMM

The problem of non-independence seems solved, but at the cost of **statistical power**: less observations.

There is another way of dealing with no independence and *not* losing data.



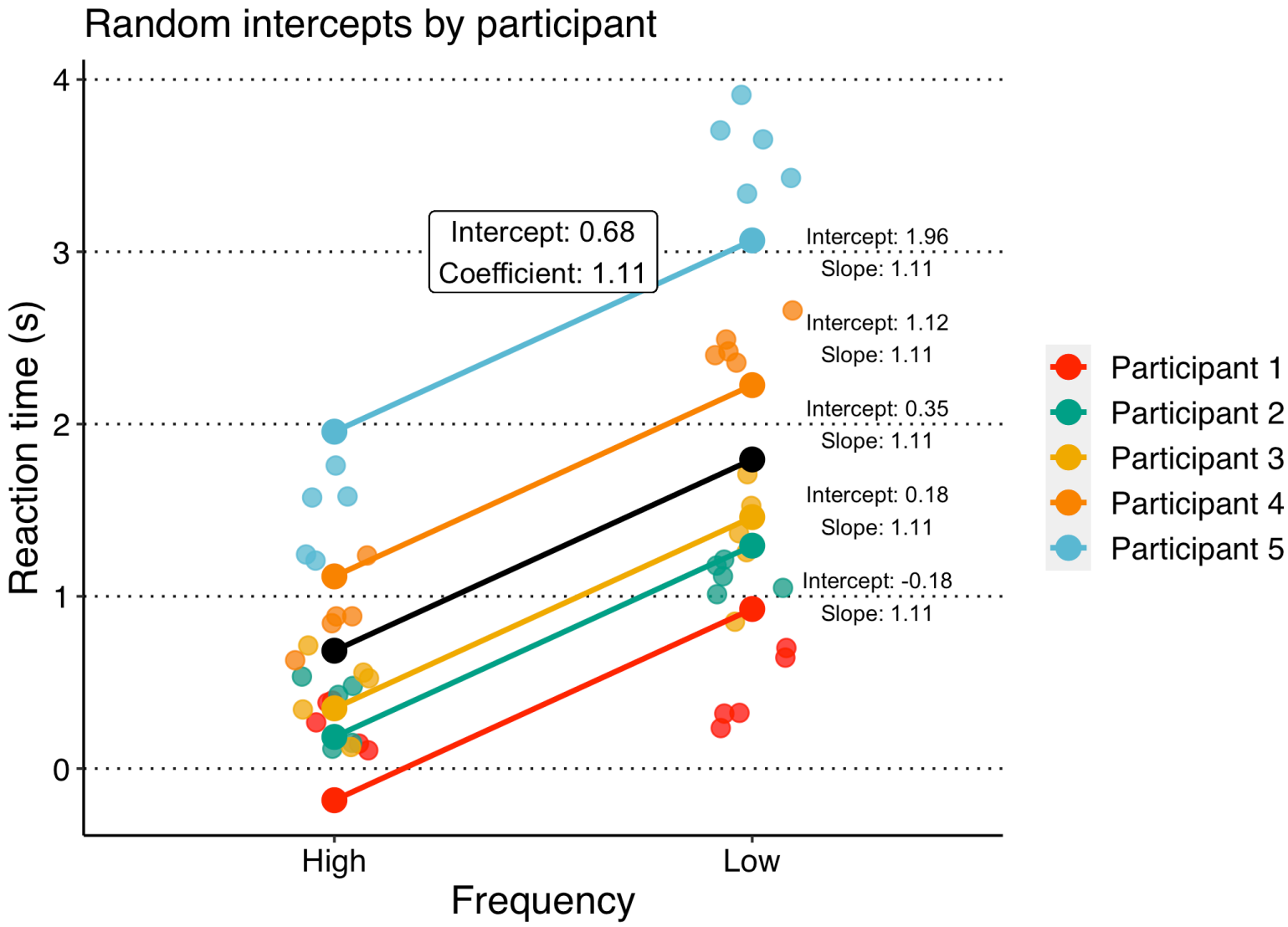
Why to use a LMM

We can include **random intercepts by participant**. Each participant will have its own baseline.

We will fit a model with **same slope but different intercept** for each participant.

We account for **non-independence**: residual is assumed to be random within-participant.

We avoid aggregating datapoints.



Why to use a LMM

Participants may differ in how much they are affected by the experimental condition.

- e.g., slower participants may be more affected by low-frequencies than faster participants.

The **slope** of the predictor `frequency` on each participant may be different.

Data points from the same participant may be still **correlated**.

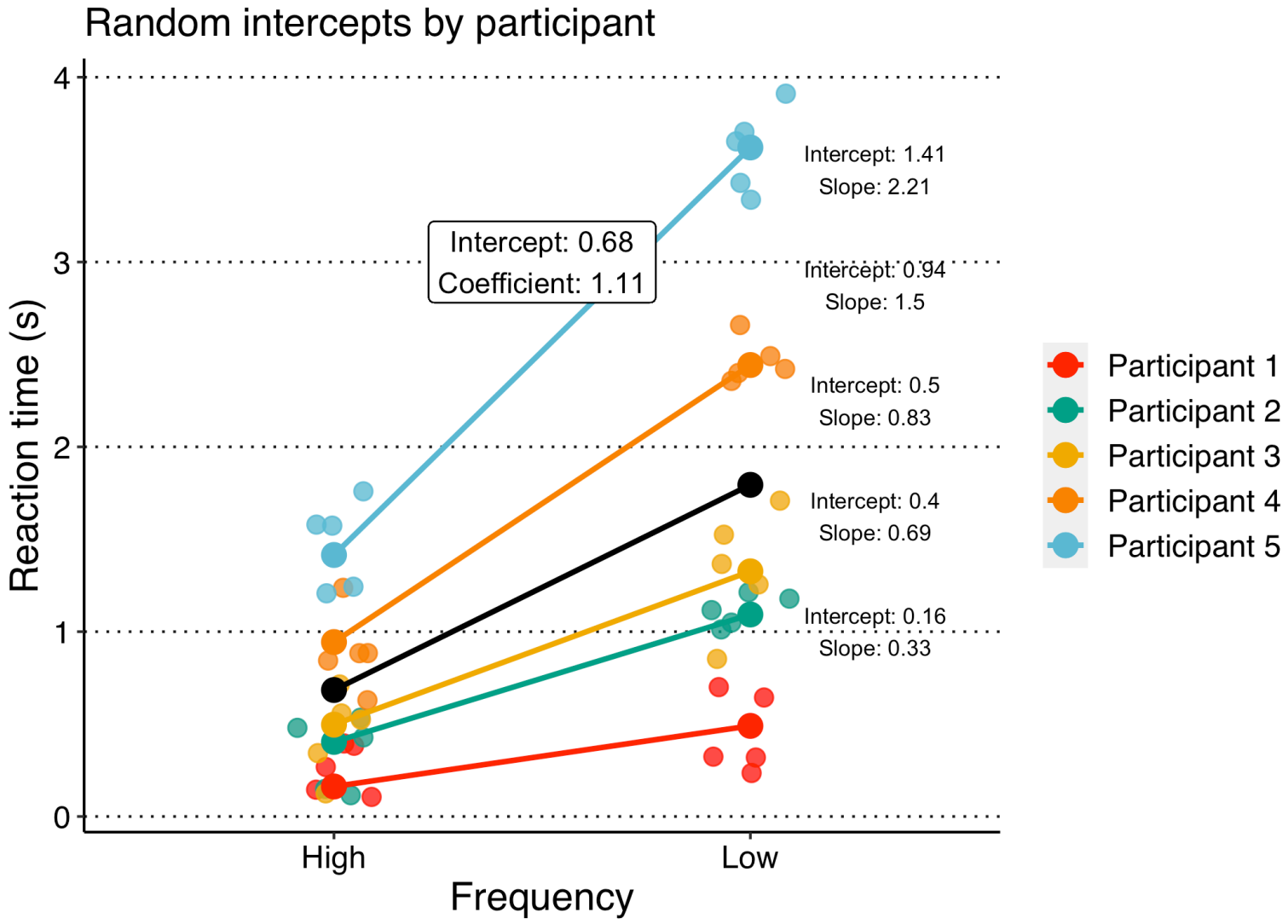
We can account for this variability by adding **random slopes** of frequency by participant.

Why to use a LMM

We can include **random intercepts and slopes by participant**.

Each participant will have its own baseline, and is **affected differently** by frequency condition.

We will fit a model with **diferent intercept and slope** for each participant.

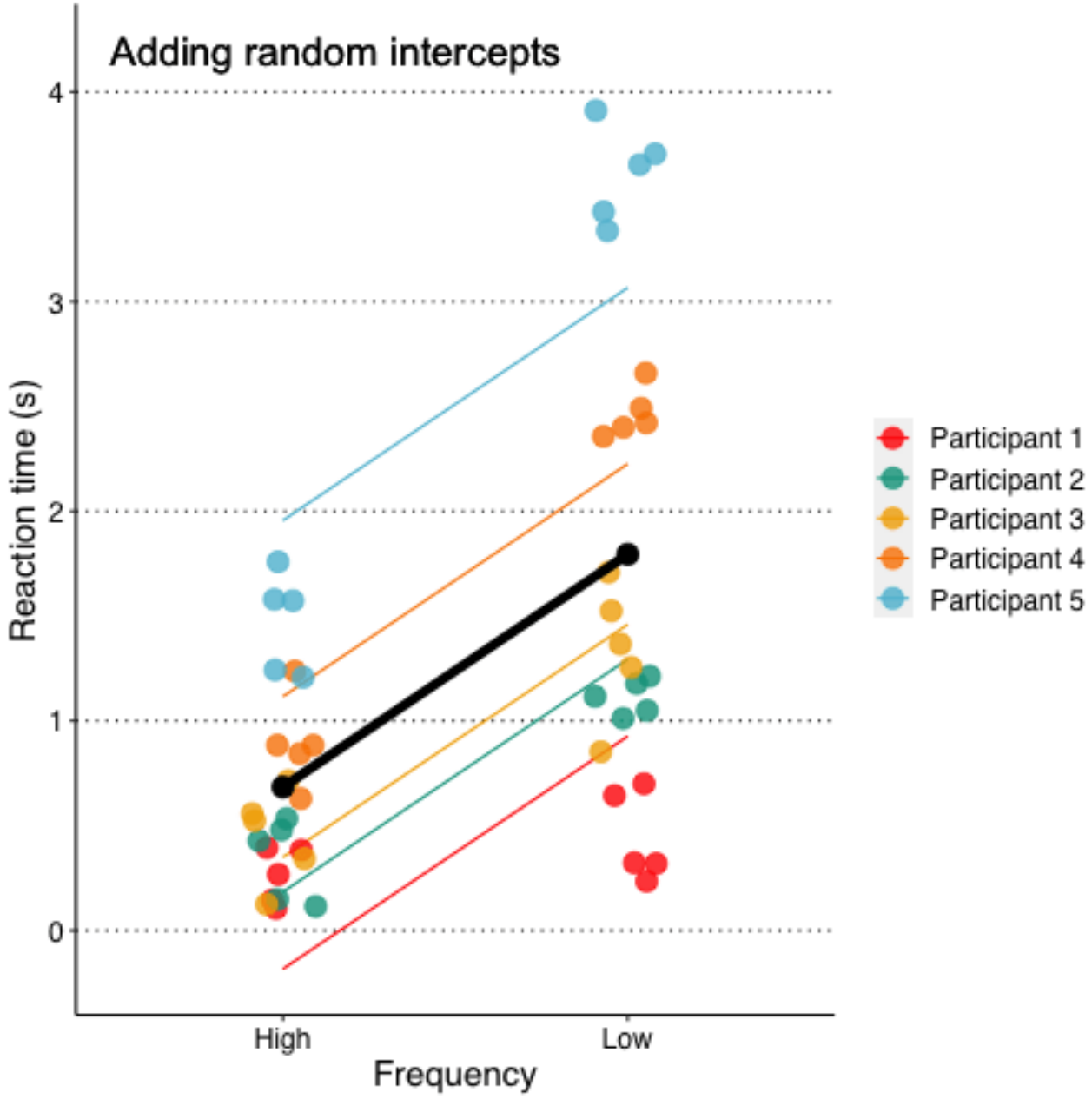


Why to use a LMM

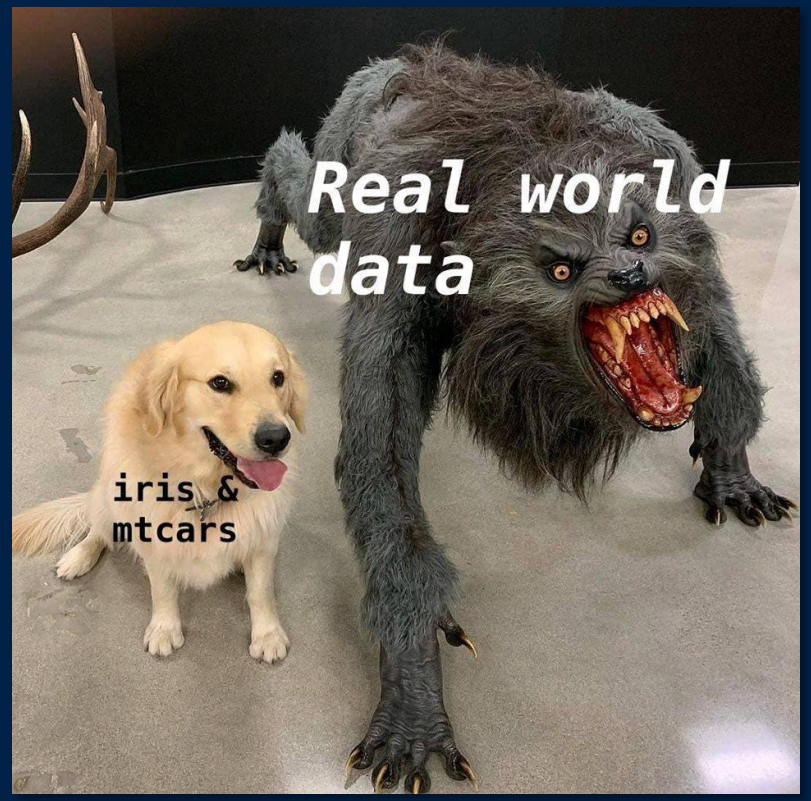
Summary:

LMM allow us to account for systematic sources of variability **beyond our fixed effects**.

This increases the **sensitivity** of our experiments.



A real-life example



Santolin, García-Castro, Zettersten, Sebastian-Galles and Saffran (2020)

Warning: Product placement ahead.

How to use LMM

Does previous experience with the Head-turn Preference Procedure (HPP) impact performance in the task?

Santolin, C., García-Castro, G., Zettersten, M., Sebastian-Galles, N., & Saffran, J. (2020, March 4). Experience with research paradigms relates to infants' direction of preference. https://doi.org/10.31234/osf.io/xgvbh

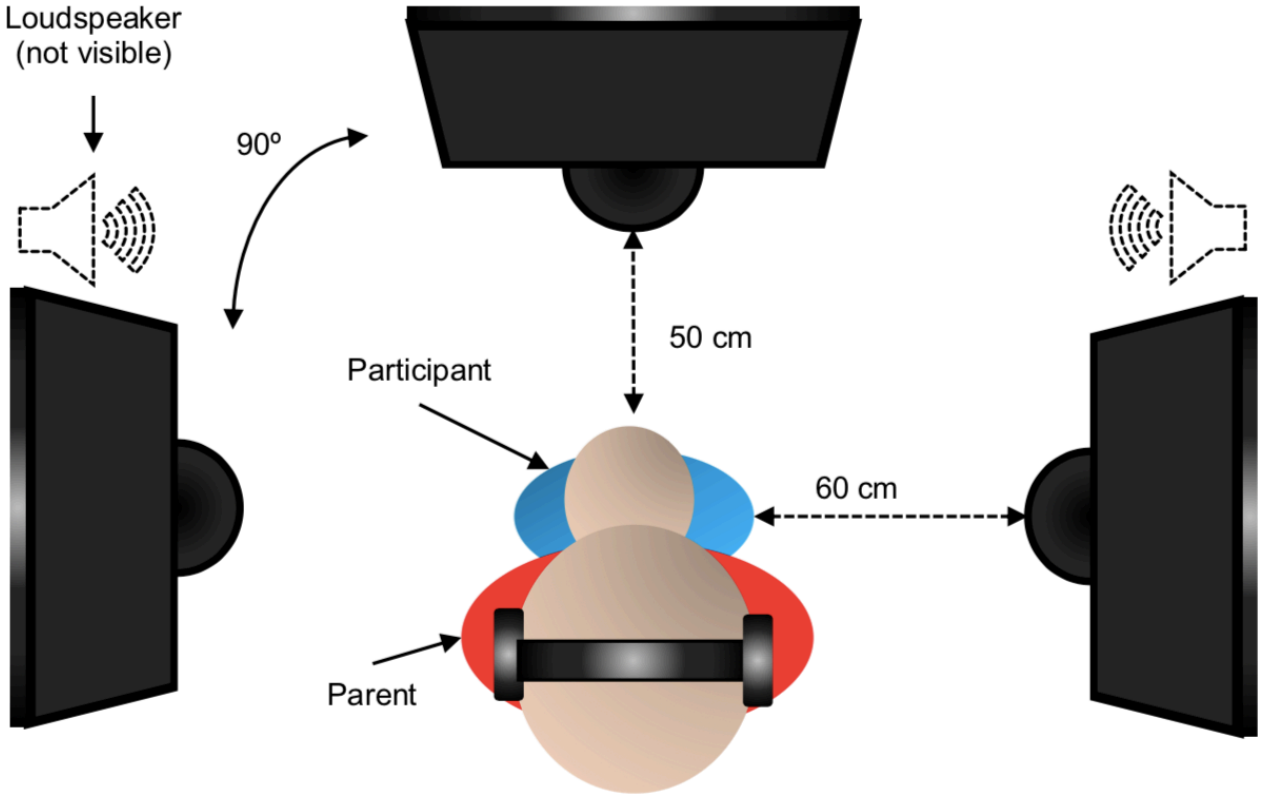
PsyArXiv

You can find data and code here: https://osf.io/g95ub/

OSF

The Head-turn Prefrence Procedure

Looking time to familiar vs. novel items



The data

We gathered data from **6 experiments** run two different locations. Only four are shown here, for simplicity:

- Laboratori de Recerca en Infància (Universitat Pompeu Fabra, Barcelona)
- Waisman Center & Department of Psychology (University of Wisconsin-Madinson, Wisconsin)

Study	HPP						
	1	2	3	4	5	6	
Saffran & Wilson (2003)	13	12	12	2	1	-	40
Saffran et al. (2008)	2	5	2	2	-	1	12
Santolin & Saffran (2019)	9	15	-	2	-	-	26
Santolin, Saffran & Sebastian-Galles (2019)	22	2	-	-	-	-	24
	46	44	14	6	1	1	N = 102

The data

Experiments were **broadly similar** in design, age of participants, and stimuli type.

There were some cross-study **differences**:

- Different **participants**
- Different **countries** (i.e., different cultures, lifestyle)
- Different **languages**
- Different **language profile** (predominantly monolinguals in Wisconsin, predominantly bilinguals in Barcelona)

The data

Import the data:

The data

```
flip_data <- read.csv("flip_data.csv")
```

Test item:
0 = Familiar
1 = Novel

Participant	Study	HPP	Item	Looking Time (ms)
1	Santolin, Saffran & Sebastian-Galles (2019)	1	0	6541
1	Santolin, Saffran & Sebastian-Galles (2019)	1	1	4770.83
2	Santolin, Saffran & Sebastian-Galles (2019)	1	0	7654.67
2	Santolin, Saffran & Sebastian-Galles (2019)	1	1	6785.17
3	Santolin, Saffran & Sebastian-Galles (2019)	1	0	3580.25
3	Santolin, Saffran & Sebastian-Galles (2019)	1	1	4501.5
4	Santolin, Saffran & Sebastian-Galles (2019)	1	0	7520.67
4	Santolin, Saffran & Sebastian-Galles (2019)	1	1	5052.83
…	…	…	…	…
102	Saffran et al. (2008)	1	0	3425
102	Saffran et al. (2008)	1	1	4889

Preparing our predictors

We use Looking time (in ms) as the **outcome**

We will use Item (familiar/novel) and number of previous HPP (numeric, 1-6) as predictors

We expect an HPP by Item **interaction:**

- Previous experience with HPP should impact novel and familiar preference differently.

Preparing our predictors

Coding variables: **dummy** vs. **effect** coding.

Model fitting functions like `lmer` don't care about how you call your **factor levels**.

It assigns a **numeric value** to each level.

In our case, *Familiar* = 0 and *Novel* = 1 (alphabetical order).

Preparing our predictors

We are interested in **interpreting changes in *novelty* preference**. Familiar trials should be the baseline.

The default coding happens to suit us.

But we should should always **explicitly recode our predictors** based on what information we want them to provide.

```
flip_data$Item <- ifelse(flip_data$Item=="Familiar", 0, 1)
```

Fixed effects: Model specification

General Linear Model:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_j X_j$$

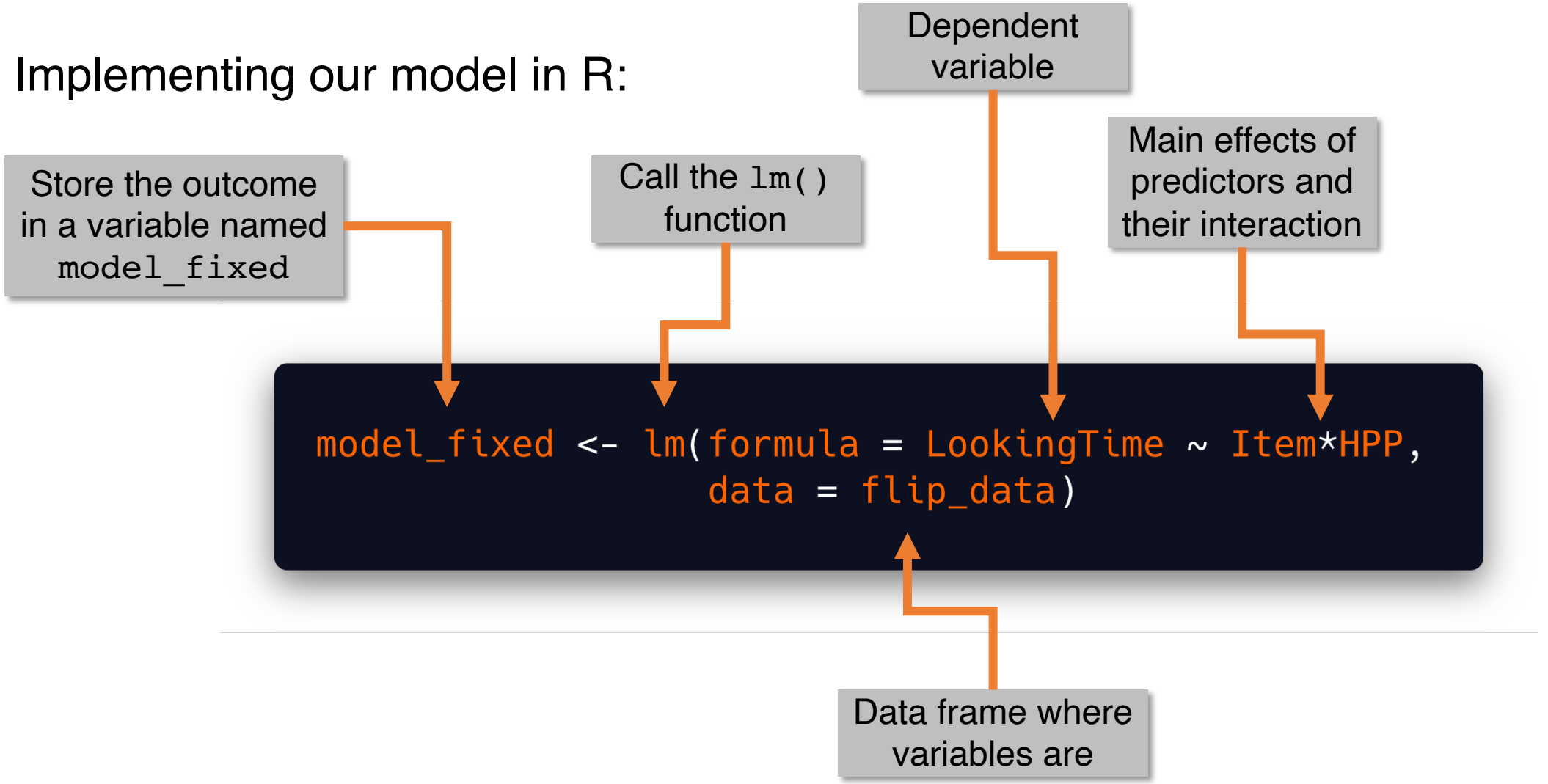
For participant i and condition i

Fixed effects:

$$\text{LookingTime}_{ij} = \beta_0 + \beta_1 Item + \beta_2 HPP + \beta_3 Item \times HPP + \varepsilon_{ij}$$

Fixed effect: Model fitting

Implementing our model in R:



Fixed effects: Statistical inference

Let's take a closer look at the estimates of the model:

The summary function shows the main outcomes of the model (e.g., coefficients)

```
summary_fixed <- summary(model_fixed)
summary_fixed$coefficients
```

Fixed effects: Statistical inference

Let's take a closer look at the estimates of the model:

The summary function shows the main outcomes of the model (e.g., coefficients)

Intercept: What value does our predictor variable take when all predictors are 0

Std. Error: How much would each coefficeint vary if we repeated this experiment infinite times?

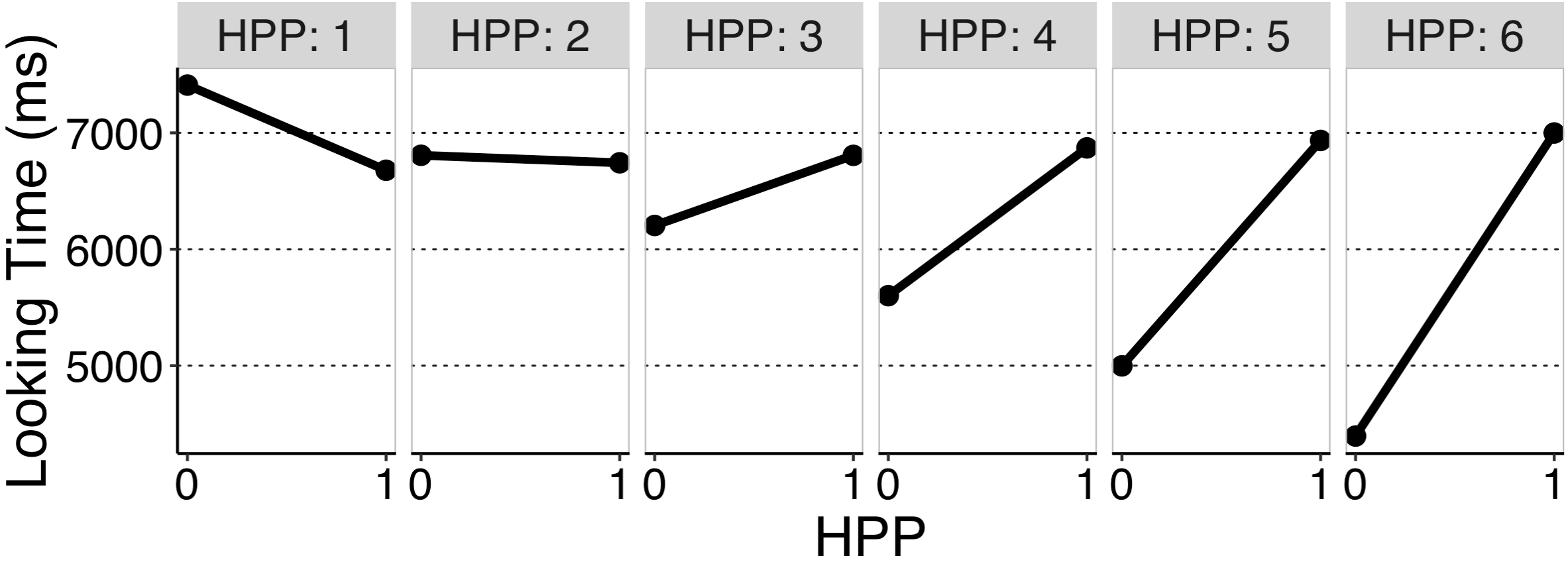
***p*-value**: probability associated to our *t*-value assuming that our coefficient is equal to 0

Term	Coefficient	SE	*t*	*p*
(Intercept)	8013.48	490.45	16.34	0.00
Item	-1398.77	693.60	-2.02	0.05
HPP	-603.08	229.70	-2.63	0.01
Item:HPP	667.11	324.85	2.05	0.04

Fixed effects

Student's *t*: value of the coefficient in the *t* probability distribution

Fixed effects: Model interpretation



Introducing random effects by participant

Our data is **hierarchical**:

Every participant provides two **data points**: one for familiar items and one for novel items.

There are strong reasons to consider that **data points from the same participant are correlated**.

- Some infants are **long lookers**: high looking times in both conditions Some infants are **short lookers**: low looking times in both conditions

We should add **random intercepts** by participant.

Random intercepts by participant: Model specification

Fixed effects:

Previously:

$$\text{LookingTime}_{ij} = \beta_0 + P_{0i} + \beta_1 Item + \beta_2 HPP + \beta_3 Item \times HPP + \varepsilon_{ij}$$

Where P_{0i} is the intercept of participant i

Random intercepts by participant: Model fitting

The `lme4` package contains the `lmer()` function that we need

```
library("lme4") # load package

model_intercepts <- lmer(formula = LookingTime ~ Item*HPP + (1 | Participant),
                         data = flip_data)
```

Random effects: we include a random intercept (1) for each participant

Random intercepts by participant: Statistical inference

Term	Coefficient	SE	t
(Intercept)	8013.48	490.45	16.34
Item	-1398.77	411.32	-3.40
HPP	-603.08	229.70	-2.63
Item:HPP	667.11	192.64	3.46

Where are my ***p-values***? I want my *p*-values ☹

Random intercepts by participant: Statistical inference

Computing *p*-values in LMM is *not* straightforward.

It's difficult to come map estimates into **probability distributions**.

Not clear how to pool parameters (many models have been run).

Random intercepts by participant: Statistical inference

1. **Likelihood ratio tests/model comparisons**: fit a null model with all parameters but the one of interest and compare if model fit is improved by adding the coefficient of interest.
2. **Normal approach**: map standardised coefficients onto the normal distribution (*Mean* = 0, *SD* = 1) to get their probability.
3. Assume estimated coefficients follow a ***t*** or a ***F* distribution*** with *approximated* degrees of freedom:
   - *t* distribution: **Satterthwaite**'s approximation to degrees of freedom
   - χ^2 distribution: **Wald**'s χ^2 test
   - *F* distribution: **Kenward-Roger** ANOVA

   Barr et al. (2013)
4. Shift to **Bayesian** statistical inference :)

Random intercepts by participant: Statistical inference

Bayesian stats answer complementary questions that sometimes fit better our aims:

- e.g., whats the probability that HPP has an effect on looking time to novel trials?
- e.g., whats the probability that HPP has **no** effect on looking time to novel trials?

Random intercepts by participant: Statistical inference

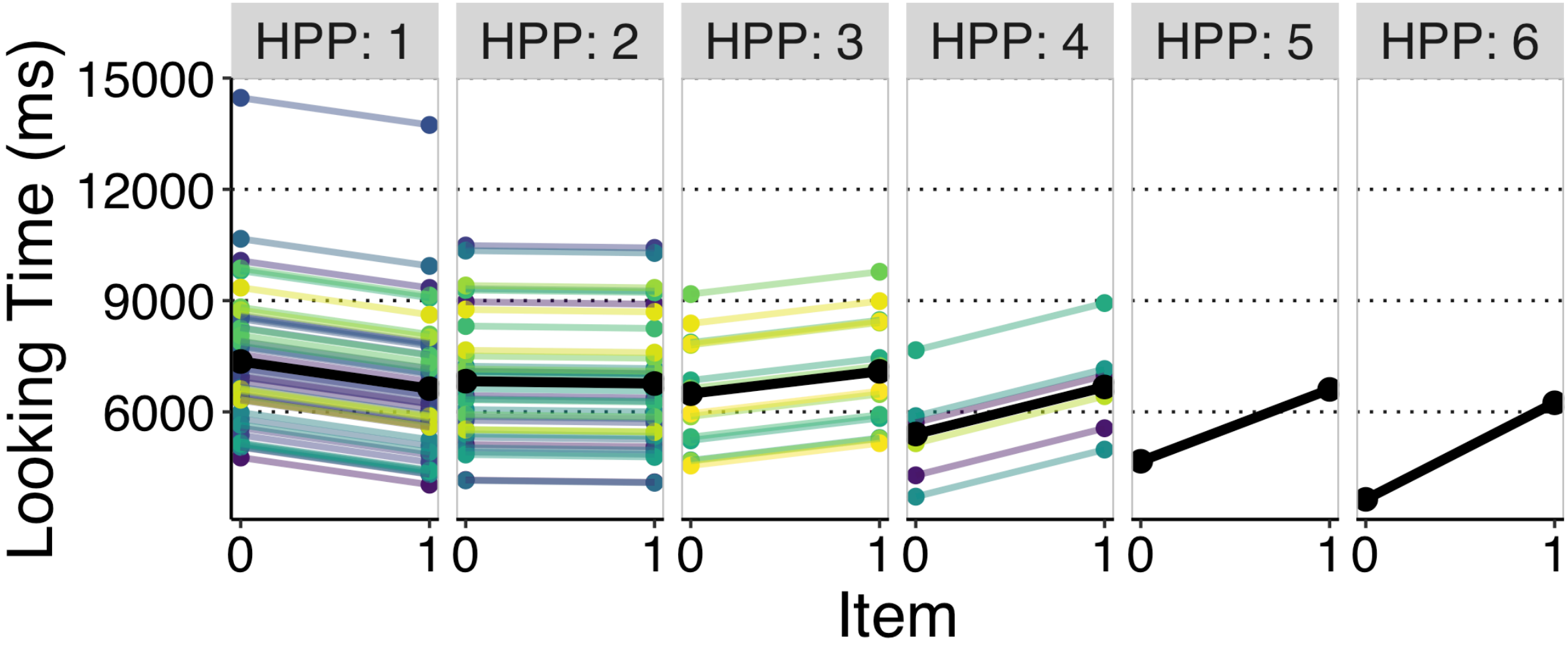
The **Kenward-Roger *F*-test** is widely recommended.

```
library("car") # load package

# run Kenward-Roger ANOVA on lmer output
Anova(model_intercepts, type = "III", test.statistic = "F")
```

Term	F	Df	Df den.	*p*-value
(Intercept)	266.97	1	140.81	0.00
Item	11.56	1	100.00	0.00
HPP	6.89	1	140.81	0.01
Item:HPP	11.99	1	100.00	0.00

Random intercepts by participant: Model interpretation



Introducing random slopes of Item by participant

There are also reasons to think that the **effect** of *Item* varies across infants.

Long-lookers may show **higher preferences** than others

- Some infants may discriminate better than others between familiar and novel items

The slope of *Item* may vary across infants.

Random intercepts and item slopes by partcipant: Model specification

Previously:

$$\text{LookingTime}_{ij} = \beta_0 + P_{0i} + \beta_1 Item + \beta_2 HPP + \beta_3 Item \times HPP + \varepsilon_{ij}$$

Where P_{0i} is the **intercept** of participant i

$$\text{LookingTime}_{ij} = \beta_0 + P_{0i} + \beta_1 Item + \beta_2 HPP + \beta_3 Item \times HPP + \varepsilon_{ij}$$

Where P_{1i} is the **slope** of Item for participant i

Random intercepts and item slopes by partcipant: Model fitting

Random intercepts: we include a random intercept (1) for each participant

```
model_slopes <- lmer(formula = LookingTime ~ Item*HPP + (1 + Item | Participant),
                     data = flip_data)
```

Random slope: We include a random slop (Item) for each participant

Random intercepts and item slopes by partcipant: Model fitting

```
#Error: number of observations (=204) <= number of random effects (=204) for term (1 + Item | Participant); the
#random-effects parameters and the residual variance (or scale parameter) are probably unidentifiable
```

Problem: there are a lot of parameters to estimate, and not enough variability.

- May be related to some levels of HPP just having one participant.
- Check this link out.

Random intercepts and item slopes by partcipant: Model specification

We can (somehow irresponsibly) force the model to go on:

```
model_slopes <- lmer(formula = LookingTime ~ Item*HPP + (1 + Item | Participant),
                     data = flip_data,
                     control = lmerControl(check.nobs.vs.nRE = "ignore"))
```

```
#Warning messages:
#1: In checkConv(attr(opt, "derivs"), opt$par, ctrl = control$checkConv,  :
#  unable to evaluate scaled gradient
#2: In checkConv(attr(opt, "derivs"), opt$par, ctrl = control$checkConv,  :
#  Model failed to converge: degenerate  Hessian with 1 negative eigenvalues
```

Random intercepts and item slopes by partcipant: Model specification

The model fails to converge (i.e., can't find the most likely coefficients given the data).

Random intercepts and item slopes by partcipant: Model specification

Random item slopes by participant don’t seem blike a goo idea…

Adding a second random effect

Our model is neglecting another important source of **correlation between data points**: Study.

Participants are more **similar within study** than between study.

We can account for this new source of variation by adding Study as a **random effect**.

Random intercepts by participant and by study: Model specification

Fixed effects and random intercepts by participant:

- Previously

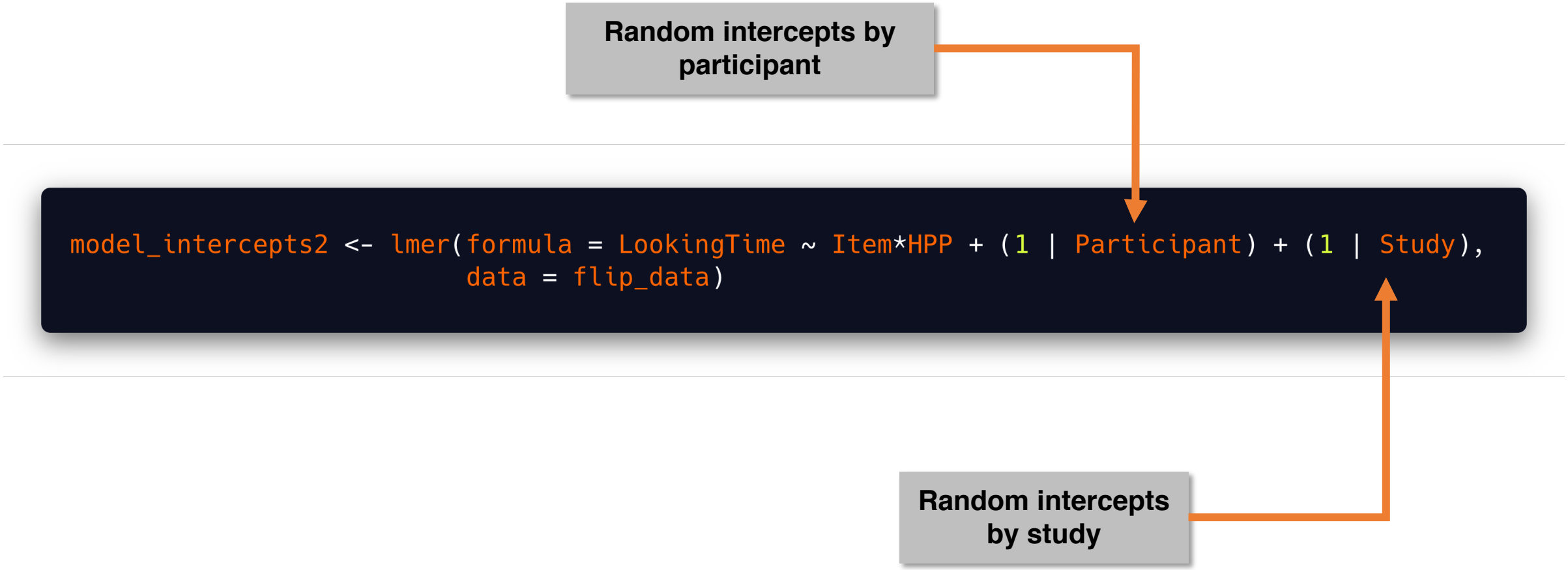
$$\text{LookingTime}_{ij} = \beta_0 + P_{0i} + \beta_1 Item + \beta_2 HPP + \beta_3 Item{\times}HPP + \varepsilon_{ij}$$

Where P_{0i} is the **intercept** of participant i

$$\text{LookingTime}_{ij} = \beta_0 + P_{0i} + S_{0k} + \beta_1 Item + \beta_2 HPP + \beta_3 Item{\times}HPP + \varepsilon_{ij}$$

Where P_{0i} is the **intercept** of participant i and S_{0k} is the **intercept** of study k.

Random intercepts by participant and by study: Model fitting



Random intercepts by participant and by study: Statistical inference

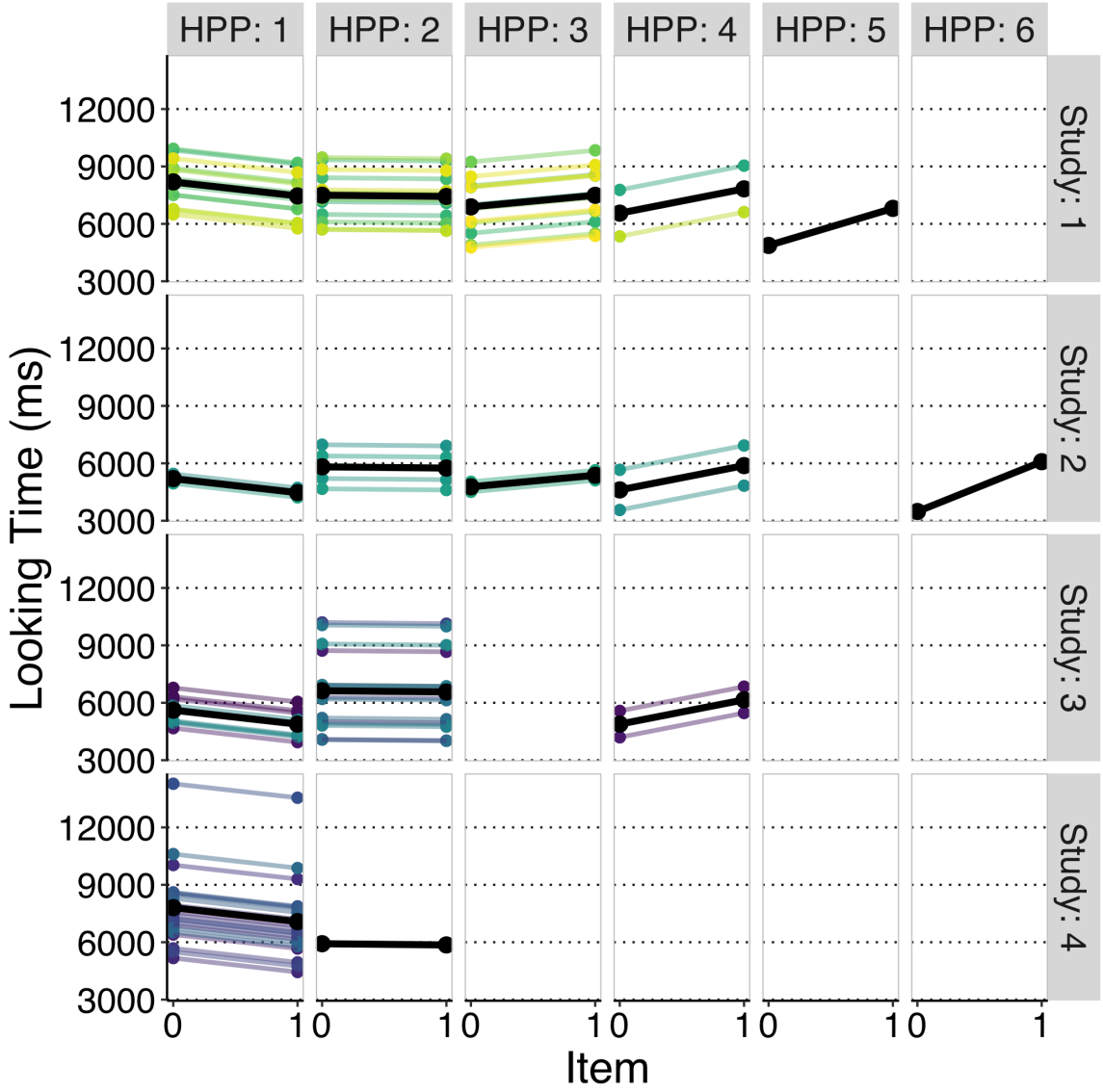
```
Anova(model_intercepts2, type = "III", test.statistic = "F")
```

Term	F	Df	Df den.	p-value
(Intercept)	124.69	1	9.06	0.00
Item	11.57	1	100.00	0.00
HPP	4.80	1	133.10	0.03
Item:HPP	11.99	1	100.00	0.00

Random intercepts by participant and by study: Model interpretation

We could go on including random effects,

But we for the sake of simplicity, let's stick to this.



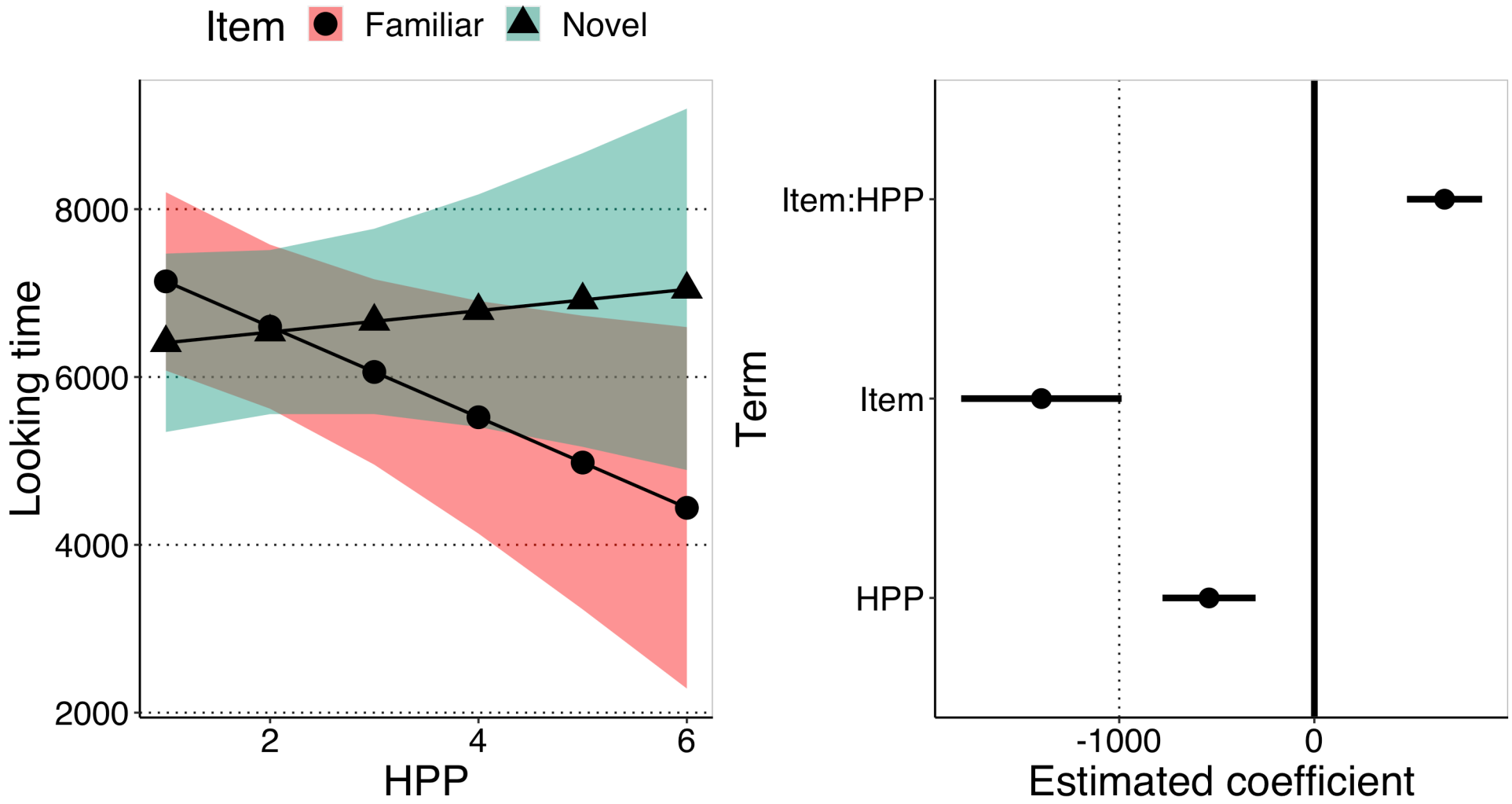
Our final model:

Includes:

- Main effects of Item and HPP
- Item × HPP interaction
- Crossed random effects for participants and study (in contrast to nested random effects; more about this later)
  - Random by-participant intercepts
  - Random by-study intercepts

```
model_intercepts2 <- lmer(formula = LookingTime ~ Item*HPP + (1 | Participant) + (1 | Study),
                          data = flip_data)
```

Our final model



Some final remarks

Recoding variables, singular fit, convergence, statistical power, and missing data

The correlation parameter

Together with σ, ρ is calculated for a given random effect, both intercepts and slopes are specified.

These parameters are accessible from the `lmer` output, in the form of a matrix.

```r
# extract variance covariance matrix
cov_mat <- VarCorr(model_slo)[["participant"]]
attr(cov_mat, "correlation")
```

```
#             (Intercept)   freqLow
#(Intercept)   1.0000000 0.9178231
#freqLow       0.9178231 1.0000000
```

```
#(Intercept)     freqLow
#  0.5377138   0.6743304
```

The correlation parameter

The **variance-covariance matrix** is **singular** when the correlation parameter is near 1 and/or SD are close to 0.

Our model is **over-parameterised**.

We should be asking a bit less from the data.

We should make our model more **parsimonious** (Bates et al., 2015).

Simplify the random effects structure.

More about failure to converge and singular fit later...

Power analysis in Mixed Models

LMM involved the estimation of many parameters.

To perform power analysis, we need to fix some of these parameters

It's difficult to come up with **sensible parameters** *a priori*

- How much variability am I expecting for each random effect?
- How much correlation am I expecting between random effects?

Power analysis in Mixed Models

Also, estimating power analytically (mapping expected parameters onto known distributions) is not straightforward in LMM.

It's not quite clear what parameters we are referring to.

Alternative: shift to simulation-based power analysis. Already available for non-mixed models (e.g., ANOVA). See Lakens and Caldwell (2019) and it's accompanying R package, ANOVApower.

Some methods are already available for mixed-models:

- Zhang and Wang (2009): Includes non-linear models, but it's implemented in commercial software (SAS)
- DeBruine and Barr (2019)

When our model fails to converge

When our model fails to converge

Sometimes, our model can't figure out **what parameters are most likely** given the data.

This can be because **different values** of the same coefficient are **equally likely**.

How to avoid this:

The **larger** the data, the easier to converge.

The larger values of the coefficients the more difficult to converge. Consider:

- Changing units of measurement: use **seconds** instead of milliseconds (Barr, 2008).
- Consider **standardising** your predictors.

Don't get too fancy with your model: the more **parsimonious**, the better (less parameters to estimate).

Recoding predictors

- Dummy coding Item makes sense because we have a **clear baseline**.
- Alternatively, we could have *effect-coded* the predictor Item.
- *Familiar* = -0.5, *Novel* = 0.5
- In this case, the **intercept** (the looking time when Item = 0) would inform us about the **average looking time across conditions**.
- How we have coded our variables impacts our conclusions especially when interpreting intercepts and main effects if we include interactions in our model.
- We'll come back to this later.