LMM Workshop

Giovanna Del Sordo

Linear mixed-effects workshop - HEST Quantitative Methods Workshop Series

The data used in this example come from Brown (2021): An Introduction to Linear Mixed Effects Modeling in R. doi: https://doi.org/10.1177/2515245920960351

Install and load libraries

Base R (the set of tools that is built into R) has a host of functions, but to create mixed-effects models you will need to install a specific package called lme4 (Bates et al., 2020).

Packages, also referred to as libraries, are sets of functions that work together and are not already built into Base R. To install lme4, run the following line of code (you should run this line of code only if you have not already installed the package):

```
#install.packages("lme4")
```

Once the package is installed, it is always on your computer, and you will not need to run that line of code again. Whenever you want to create mixed-effects models, you will need to load the installed package, which will give you access to all the functions you need (you need to rerun this line of code every time you start a new R session). The following line of code will load the lme4 package:

```
library(lme4)
```

Load data into R

You can simply replace the path to your own path where your data file is located. To read csv files, another package, “readr”, is necessary

```
#install.packages("readr")
library(readr)

file_path_csv <- "/Users/giovannadelsordo/Documents/Documents - Giovanna’s
MacBook Air/Post Doc NMSU/Workshops:Talks/HEST workshop series 2025/Code and
data/rt_dummy_data.csv"
data <- read_csv(file_path_csv)

# Convert categorical variables from characters to factors
data$PID <- as.factor(data$PID)
data$modality <- as.factor(data$modality)
data$stim <- as.factor(data$stim)
```

Running these two lines create two elements in the R “Environment” on the right. One is simply the path to the csv file, and the other is a dataframe containing the data. The data file contains:

- PID: participants’ numbers (random effect).

- RT: reaction time in milliseconds (dependent variable).

- modality: condition with two levels: “Audio-only” and “Audiovisual” (fixed factor).

- stim: the words presented to participants in the primary task (random effect).

Fitting a simple linear regression

```
model_regression <- lm(RT ~ 1 + modality, data = data)
summary(model_regression)
```

```
Call:
lm(formula = RT ~ 1 + modality, data = data)

Residuals:
    Min      1Q  Median      3Q     Max
-814.52 -210.46  -25.52  174.54 1436.54

Coefficients:
                    Estimate Std. Error t value Pr(>|t|)
(Intercept)         1041.464      3.001  347.00   <2e-16 ***
modalityAudiovisual   83.057      4.207   19.74   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 309.7 on 21677 degrees of freedom
Multiple R-squared:  0.01766,   Adjusted R-squared:  0.01761
F-statistic: 389.7 on 1 and 21677 DF,  p-value: < 2.2e-16
```

Fitting a random-intercepts model

```
lmer(outcome ~ 1 + predictor + (1|participant) + (1|item), data = data)
```

lmer() – This is the function that builds a linear mixed-effects model. It comes from the lme4 package.

outcome – The dependent variable (the thing you’re trying to predict).

predictor – The independent variable (the factor you think affects the outcome).

1 + predictor – This part describes the fixed effects. The 1 represents the intercept (the average starting point).

(1 | participant) and (1 | item) – These are the random effects. The pipe symbol (|) means "varies by." So (1 | participant) means "each participant has their own intercept." (1 | item) means "each item has its own intercept."

Together, they tell the model that both participants and items can differ in their baseline responses.

This model has random intercepts for participants and items, but no random slopes. That means the model assumes that while people and words may start from different baselines, the overall effect of the predictor (the slope) is the same for everyone.

Also, note:

The 1 in the fixed-effects part (1 + predictor) is optional — R includes an intercept by default. But the 1 in the random-effects part ((1 | participant)) is not optional, because you have to tell R what can vary across the grouping factor.

The actual model is:

```
model_intercept <- lmer(RT ~ 1 + modality + (1|PID) + (1|stim), data = data)
summary(model_intercept)
```

```
Linear mixed model fit by REML ['lmerMod']
Formula: RT ~ 1 + modality + (1 | PID) + (1 | stim)
   Data: data

REML criterion at convergence: 302861.5

Scaled residuals:
    Min      1Q  Median      3Q     Max
-3.3572 -0.6949 -0.0205  0.5814  4.9120

Random effects:
 Groups   Name        Variance Std.Dev.
 stim     (Intercept)   360.2   18.98
 PID      (Intercept) 28065.7  167.53
 Residual             67215.3  259.26
Number of obs: 21679, groups:  stim, 543; PID, 53

Fixed effects:
                    Estimate Std. Error t value
(Intercept)         1044.449     23.164   45.09
modalityAudiovisual   82.525      3.529   23.38

Correlation of Fixed Effects:
            (Intr)
modltyAdvsl -0.078
```

This model only contains random intercepts, but no random slopes. However, based on topic knowledge, you might already expect that both participants and words might differ in the extent to which they are affected by the modality manipulation. We will therefore fit a model that includes both by-participant and by-item random slopes for modality. Failing to include random slopes would amount to assuming that all participants and words respond to the modality effect in exactly the same way, which is an unreasonable assumption to make.

Fitting a random intercepts and random slope model

```
lmer(outcome ~ 1 + predictor + (1 + predictor|participant) + (1 +
predictor|item), data = data)
```

Here, the portions in parentheses indicate that both the intercept (indicated by the 1, which in this case is optional because it is implied by the presence of random slopes but is included for clarity) and the predictor (indicated by + predictor) vary by participants and items. In plain language, this syntax means "predict the outcome from the predictor and the random intercepts and slopes for participants and items, using the data I provide."

The model above includes only one predictor, but if a model includes multiple predictors the researcher may decide which of the predictors can vary by participant or item; in other words, any fixed effect to the left of the interior parentheses can be included to the left of the pipe (inside the interior parentheses), provided that including it is justified given the design of the experiment. For example, if we wanted to include a second predictor that varied within both participants and items, but there was no theoretical motivation for including by-item random slopes for the second predictor—or, alternatively, if the second predictor varied between items, so including the by-item random slope would not be justified given the experimental design—the syntax would look like this:

```
lmer(outcome ~ 1 + predictor1 + predictor2 + (1 + predictor1 +
predictor2|participant) + (1 + predictor1|item), data = data)
```

The actual model is:

```
model_full <- lmer(RT ~ 1 + modality + (1 + modality|PID) + (1 +
modality|stim), data = data)
```

This is the "full" model (i.e., the model including the fixed effects of interest and all theoretically motivated random effects).

But! Is this model the best fit to the data?

Likelihood Ratio Tests (LRTs)

The "likelihood" is a measure of how well the model's parameters **explain the data we actually observed**.
A *higher likelihood* means the model fits the data better.

The Likelihood Ratio Test compares two models:

- A **reduced model** (simpler)
- A **full model** (more complex)

It asks: does the full model fit the data *significantly* better?

```
# Full model = model with all fixed effects, random intercepts and random
slopes of interest
model_full <- lmer(RT ~ 1 + modality + (1 + modality|PID) + (1 +
modality|stim),
                   data = data, REML = FALSE)

Warning in checkConv(attr(opt, "derivs"), opt$par, ctrl = control$checkConv,
:
Model failed to converge with max|grad| = 0.00251373 (tol = 0.002, component
1)

# Does this full model fit the data better than models with less parameters?

# reduced model without modality (fixed effect)
model_reduced_modality <- lmer(RT ~ 1 + (1 + modality|PID) + (1 +
modality|stim),
                       data = data, REML = FALSE)
# Does the effect of modality is significant? Does it help to explain the
observed data?
# Warning message: The model "failed to converge". We will come back to
that...

# We use the function "anova()" to test which model is better. It is a Chi-
square test
anova(model_full, model_reduced_modality)

Data: data
Models:
model_reduced_modality: RT ~ 1 + (1 + modality | PID) + (1 + modality | stim)
model_full: RT ~ 1 + modality + (1 + modality | PID) + (1 + modality | stim)
                       npar    AIC    BIC  logLik -2*log(L)  Chisq Df
model_reduced_modality    8 302449 302513 -151217    302433
model_full                9 302419 302491 -151200    302401 32.385  1
                       Pr(>Chisq)
model_reduced_modality
model_full              1.264e-08 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

- Reduced model → includes only the intercept and random effects (no fixed effect for modality).
- Full model → same as above, but adds the fixed effect of modality.

They are nested models: one is a simpler version of the other.

Does a model that includes information about the modality in which words are presented fit the data better than a model that does not include that information?

Interpretation: All the values for AIC, BIC, logLik, and -2*log(L) have smaller values in the full model. The p-value is significant. –> The **full model** fit the data best. The model including the modality effect provides a better fit.

What if we have several fixed effects to compare?

```
#install.packages("afex")
library(afex)
```

```
************
Welcome to afex. For support visit: http://afex.singmann.science/

- Functions for ANOVAs: aov_car(), aov_ez(), and aov_4()
- Methods for calculating p-values with mixed(): 'S', 'KR', 'LRT', and 'PB'
- 'afex_aov' and 'mixed' objects can be passed to emmeans() for follow-up
tests
- Get and set global package options with: afex_options()
- Set sum-to-zero contrasts globally: set_sum_contrasts()
- For example analyses see: browseVignettes("afex")
************


Attaching package: 'afex'

The following object is masked from 'package:lme4':

    lmer
```

```
mixed(RT ~ 1 + modality + (1 + modality|PID) + (1 + modality|stim), data =
data, method = 'LRT')
```

```
Contrasts set to contr.sum for the following variables: modality, PID, stim

REML argument to lmer() set to FALSE for method = 'PB' or 'LRT'

Warning in checkConv(attr(opt, "derivs"), opt$par, ctrl = control$checkConv,
:
unable to evaluate scaled gradient

Warning in checkConv(attr(opt, "derivs"), opt$par, ctrl = control$checkConv,
:
Model failed to converge: degenerate Hessian with 1 negative eigenvalues

Warning: Model failed to converge with 1 negative eigenvalue: -9.6e+02
```

```
Warning: lme4 reported (at least) the following warnings for 'modality':
  * unable to evaluate scaled gradient
  * Model failed to converge: degenerate  Hessian with 1 negative eigenvalues

Mixed Model Anova Table (Type 3 tests, LRT-method)

Model: RT ~ 1 + modality + (1 + modality | PID) + (1 + modality | stim)
Data: data
Df full model: 9
    Effect df     Chisq p.value
1 modality  1 32.54 ***   <.001
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '+' 0.1 ' ' 1
```

For this specific model, we get the same results as the "anova()" function. The "mixed()" function becomes more interesting when several fixed effects are tested.

Warning: the mixed() function only works for FIXED EFFECTS, not for random effects.

What if we want to tests the random effects?

You will need to go back to the anova() function.

```
# Full model (both intercepts and slopes for PID and stim)
model_full <- lmer(RT ~ 1 + modality +
                     (1 + modality | PID) +
                     (1 + modality | stim),
                   data = data, REML = FALSE)

Warning in checkConv(attr(opt, "derivs"), opt$par, ctrl = control$checkConv,
:
Model failed to converge with max|grad| = 0.00251373 (tol = 0.002, component
1)

# Remove the random slope for stim
model_no_slope_stim <- lmer(RT ~ 1 + modality +
                              (1 + modality | PID) +
                              (1 | stim),
                            data = data, REML = FALSE)

anova(model_full, model_no_slope_stim)

Data: data
Models:
model_no_slope_stim: RT ~ 1 + modality + (1 + modality | PID) + (1 | stim)
model_full: RT ~ 1 + modality + (1 + modality | PID) + (1 + modality | stim)
                     npar    AIC    BIC  logLik -2*log(L)  Chisq Df Pr(>Chisq)
model_no_slope_stim    7 302416 302472 -151201    302402
model_full             9 302419 302491 -151200    302401 1.1261  2     0.5695
```

Adding **a random slope for stimulus** does NOT significantly improve model fit.

```r
# Remove the random slope for stim
model_no_slope_stim <- lmer(RT ~ 1 + modality +
                              (1 + modality | PID) +
                              (1 | stim),
                            data = data, REML = FALSE)

# Remove both random slopes (intercepts only for both)
model_intercepts_only <- lmer(RT ~ 1 + modality +
                                (1 | PID) +
                                (1 | stim),
                              data = data, REML = FALSE)

anova(model_no_slope_stim, model_intercepts_only)
```

```
Data: data
Models:
model_intercepts_only: RT ~ 1 + modality + (1 | PID) + (1 | stim)
model_no_slope_stim: RT ~ 1 + modality + (1 + modality | PID) + (1 | stim)
                      npar    AIC    BIC  logLik -2*log(L)  Chisq Df
Pr(>Chisq)
model_intercepts_only    5 302884 302924 -151437    302874
model_no_slope_stim      7 302416 302472 -151201    302402 472.19  2  < 2.2e-
16

model_intercepts_only
model_no_slope_stim   ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The model with **random slopes for participants** (model_no_slope_stim) fits the data much better than the model with only random intercepts (model_intercepts_only).

In plain terms: Participants differ not only in their overall speed (intercept), but also in how strongly the effect of modality influences their RTs.

Final model

```r
model_final <- lmer(RT ~ 1 + modality +
                              (1 + modality | PID) +
                              (1 | stim),
                            data = data, REML = FALSE)
```

Participants (PID):

- Random intercepts: each participant has their own baseline reaction time (RT).
- Random slopes for modality: each participant has their own modality effect (the slope can vary).

Stimuli (stim):

- Random intercepts only: each stimulus has its own baseline RT.
- No random slope for modality: the effect of modality is assumed to be the same across all stimuli.

Why this structure?

It's theoretically justified: participants are expected to differ in both overall speed and how they respond to modality, but the modality effect itself should be consistent across stimuli.

Interpretation of summary() output

The likelihood ratio test comparing our full and reduced models indicated that the modality effect was significant, but it did not tell us about the direction or magnitude of the effect. So how do we assess whether the audiovisual condition resulted in slower or faster response times? And how do we gain insight into the variability across participants and items that we asked the model to estimate? To answer these questions, we need to examine the model output via the summary() function.

Note: p-values will only be shown if you have loaded the "afex" package.

```
summary(model_final)

Linear mixed model fit by maximum likelihood . t-tests use Satterthwaite's
  method [lmerModLmerTest]
Formula: RT ~ 1 + modality + (1 + modality | PID) + (1 | stim)
   Data: data

      AIC       BIC    logLik -2*log(L)  df.resid
 302415.8  302471.7 -151200.9  302401.8     21672

Scaled residuals:
    Min      1Q  Median      3Q     Max
-3.3599 -0.6959 -0.0136  0.5897  4.9906

Random effects:
 Groups   Name                Variance Std.Dev. Corr
 stim     (Intercept)           399.6   19.99
 PID      (Intercept)         28006.4  167.35
          modalityAudiovisual  7566.1   86.98   -0.16
 Residual                     65313.6  255.57
Number of obs: 21679, groups:  stim, 543; PID, 53

Fixed effects:
                    Estimate Std. Error      df t value Pr(>|t|)
(Intercept)          1044.15      23.14   53.19  45.126  < 2e-16 ***
modalityAudiovisual    83.15      12.45   52.86   6.678  1.5e-08 ***
---
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Correlation of Fixed Effects:
            (Intr)
modltyAdvsl -0.179
```

Post hoc tests

When your model includes **multiple fixed effects**, **interactions**, or a **factor with more than two levels**, you'll often need to conduct **post hoc tests** to explore specific comparisons.

For linear mixed-effects models, these tests are typically performed using the **emmeans** package

```
#install.packages("emmeans")
library(emmeans)
```

To run post hoc tests, you should use the emmeans() function, feed it your model, and the specific fixed factor you want to perform the post hoc tests on.

```
emmeans(model_final, pairwise ~ modality)

Note: D.f. calculations have been disabled because the number of observations
exceeds 3000.
To enable adjustments, add the argument 'pbkrtest.limit = 21679' (or larger)
[or, globally, 'set emm_options(pbkrtest.limit = 21679)' or larger];
but be warned that this may result in large computation time and memory use.

Note: D.f. calculations have been disabled because the number of observations
exceeds 3000.
To enable adjustments, add the argument 'lmerTest.limit = 21679' (or larger)
[or, globally, 'set emm_options(lmerTest.limit = 21679)' or larger];
but be warned that this may result in large computation time and memory use.

$emmeans
 modality    emmean   SE  df asymp.LCL asymp.UCL
 Audio-only    1044 23.1 Inf       999      1089
 Audiovisual   1127 24.2 Inf      1080      1175

Degrees-of-freedom method: asymptotic
Confidence level used: 0.95

$contrasts
 contrast                  estimate   SE  df z.ratio p.value
 (Audio-only) - Audiovisual    -83.1 12.5 Inf  -6.678  <.0001

Degrees-of-freedom method: asymptotic
```

The first table show what are called estimated marginal means or model-adjusted means. These are simply the average predicted values of reaction time for each condition — here, the *Audio-only* and *Audiovisual* modalities — as estimated by our model. They're called *model-adjusted* because they take into account all the other parts of the model, like the random effects, rather than being simple raw averages.

So, in our results:

- The predicted mean reaction time for **Audio-only** trials is **about 1044 ms**,
- And for **Audiovisual** trials, it's **about 1127 ms**.

The "SE" column shows the standard error: how much uncertainty there is around those predicted means. The "LCL" and "UCL" columns give us the lower and upper bounds of the 95% confidence interval, so we can say, with 95% confidence, that the true mean likely falls within those ranges.

Below that, the contrasts section shows the pairwise comparison between the two conditions. Since we only have two levels of modality, there's just one comparison here: Audio-only minus Audiovisual. This reads as any other regular post hoc test. The estimate is –83.1 ms, meaning that on average, participants were about 83 milliseconds slower in the Audiovisual condition compared to Audio-only, and this difference is significant.

Finally, the note "Degrees-of-freedom method: asymptotic" means that because this is a large-sample model, `emmeans()` used an approximation that assumes very large sample sizes (so the df are shown as infinite).

Warning message from emmeans()

When you run a post hoc test on a model with a large number of observations, R may display a message saying that degrees-of-freedom (df) calculations were disabled. This happens because `emmeans()` tries to make the computation faster by skipping the more intensive df estimation step. The note that appears in the console provides code you can run to re-enable df calculations.

```
emm_options(pbkrtest.limit = 21679) # Note that this number is the same total number of observations that we fitted in the model
emmeans(model_final, pairwise ~ modality)
```

What if you are interested in the intercepts and slopes for each participants?

You can use the coef() function to call the intercepts and slopes for the random effects.

```
coef(model_final)$PID

    (Intercept) modalityAudiovisual
301   1024.1508          -17.804624
302   1044.1518            1.646581
303    882.7179           58.600934
```

304	1232.7152	-28.347453
306	1042.5604	33.065551
307	1111.1585	-8.904496
308	1250.7637	70.799658
309	795.2199	16.048020
310	1176.3880	103.327041
311	1012.7313	30.866697
312	1110.0308	152.561457
313	1114.0877	30.663927
314	1168.9613	-72.568020
315	878.1152	16.197487
316	1419.3258	-37.755159
318	945.2803	59.192655
319	1017.4521	97.948894
320	987.6555	100.896551
321	1025.2493	139.192088
324	1031.4518	135.909603
325	826.5912	35.087764
326	1048.7792	39.572677
328	1236.9583	127.678456
329	1042.3875	10.489185
331	1406.7592	161.486154
333	1644.4547	56.076241
334	943.7839	93.427240
335	1171.6509	64.112938
337	872.3499	170.141490
338	806.7678	121.487877
340	1179.8094	105.320062
341	867.4942	124.643265
342	1253.4290	30.831929
343	988.1140	207.056296
344	1027.4896	97.382327
346	895.9018	35.600855
348	755.4009	188.735334
349	940.4686	55.463174
350	1073.3327	302.187066
351	1121.0501	214.673204
352	796.5104	77.665725
353	1103.9468	154.761873
354	1074.5031	155.530774
355	1192.7633	90.092243
356	907.3338	397.549801
357	910.8000	80.974221
358	963.0819	33.105227
359	1087.5979	-39.919351
360	1070.3001	27.246665
361	982.0369	131.986524
362	953.4162	55.859512
363	920.6955	44.129923
364	1003.7514	74.889809

What we're looking at here are the **participant-specific coefficients**, the model's estimated intercept and slope for each participant.

The first column, **(Intercept)**, gives the model's estimate of each participant's **average reaction time** in the *reference condition*: here, that's the *Audio-only* condition. So, for example, participant **301** has an estimated baseline RT of **about 1024 ms**, while participant **309** is much faster, around **795 ms**, and participant **304** is slower, around **1233 ms**.

The second column, **modalityAudiovisual**, shows how much that participant's reaction time **changes in the Audiovisual condition** relative to their own Audio-only baseline.

For instance:

- Participant **303** has a **positive slope** (+58.6), meaning they got **slower** when both hearing and seeing the speaker.
- Participant **304** has a **negative slope** (–28.3), meaning they got **faster** in the Audiovisual condition.
- Participant **302** is near zero (+1.6), showing almost **no effect** of modality at all.

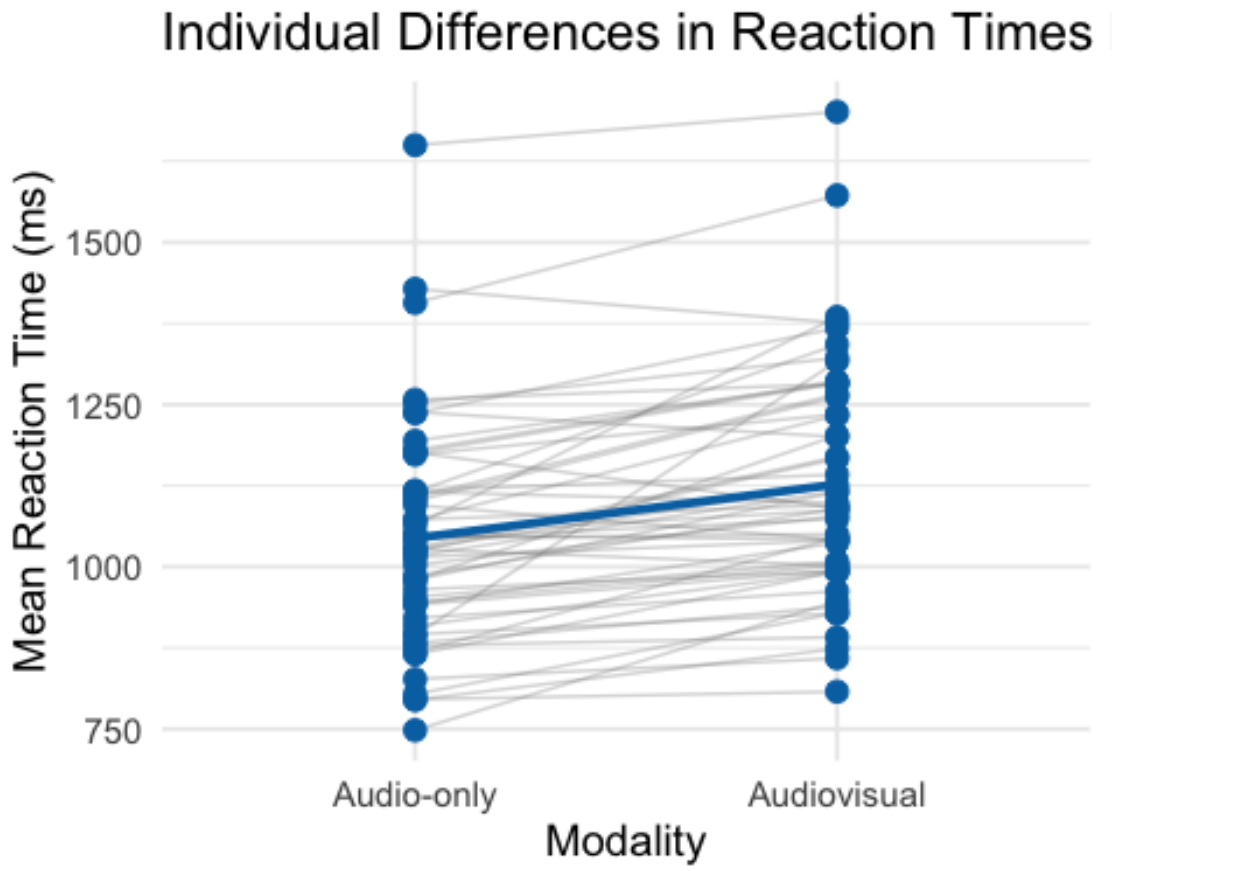
Some visualizations

Random effect for Participants: Individual trajectories in reaction times by modality

```
# Get model predictions for each participant and modality
library(dplyr)


Attaching package: 'dplyr'

The following objects are masked from 'package:stats':

    filter, lag

The following objects are masked from 'package:base':

    intersect, setdiff, setequal, union

library(ggplot2)

Warning: package 'ggplot2' was built under R version 4.3.3

pred_data <- data %>%
  group_by(PID, modality) %>%
  summarise(
    mean_RT = mean(RT, na.rm = TRUE),
```

```
    .groups = "drop"
  )

ggplot(pred_data, aes(x = modality, y = mean_RT, group = PID)) +
  geom_line(alpha = 0.3, color = "gray50") +
  stat_summary(fun = mean, geom = "line", aes(group = 1), color = "#0072B2",
size = 1.3) +
  stat_summary(fun.data = mean_cl_boot, geom = "errorbar", width = 0.1, color
= "#0072B2") +
  stat_summary(fun = mean, geom = "point", color = "#0072B2", size = 3) +
  labs(
    title = "Individual Differences in Reaction Times by Modality",
    x = "Modality",
    y = "Mean Reaction Time (ms)"
  ) +
  theme_minimal(base_size = 14)
```

```
Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
ℹ Please use `linewidth` instead.
```



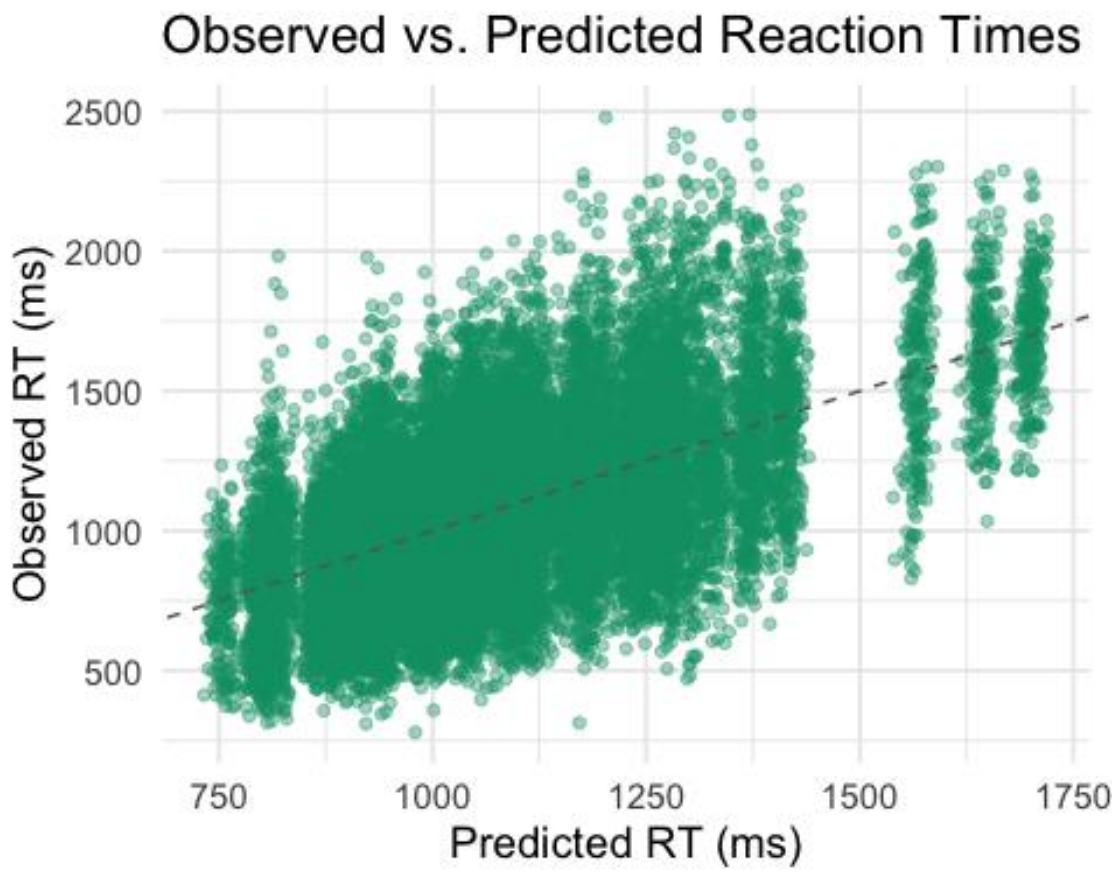
Each gray line represents one participant's average reaction times in the two conditions: *Audio-only* and *Audiovisual*. Most lines slope upward, meaning that participants were

generally slower in the Audiovisual condition, but the slope is not identical for everyone. Some people barely change across conditions, while others slow down quite a bit.

The thick blue line shows the overall trend (the model's fixed effect) which represents the average effect across everyone. We plot this to **visualize individual variability**:even though there's a clear average effect, real participants don't all behave the same way. This is exactly what the random effects in the model capture: that some participants are faster or slower overall (random intercepts) and that the size of the modality effect varies across people (random slopes).

Random effect for Participants: Observed vs. Predicted Reaction Times

```
data$predicted <- predict(model_final)
data$residuals <- resid(model_final)

ggplot(data, aes(x = predicted, y = RT)) +
  geom_point(alpha = 0.4, color = "#009E73") +
  geom_abline(intercept = 0, slope = 1, linetype = "dashed", color =
"gray40") +
  labs(
    title = "Observed vs. Predicted Reaction Times",
    x = "Predicted RT (ms)",
    y = "Observed RT (ms)"
  ) +
  theme_minimal(base_size = 14)
```



This last plot is a **model diagnostic plot**, showing how well our model's predictions match the actual data. Each point represents one trial with the predicted reaction time on the x-axis and the observed reaction time on the y-axis.

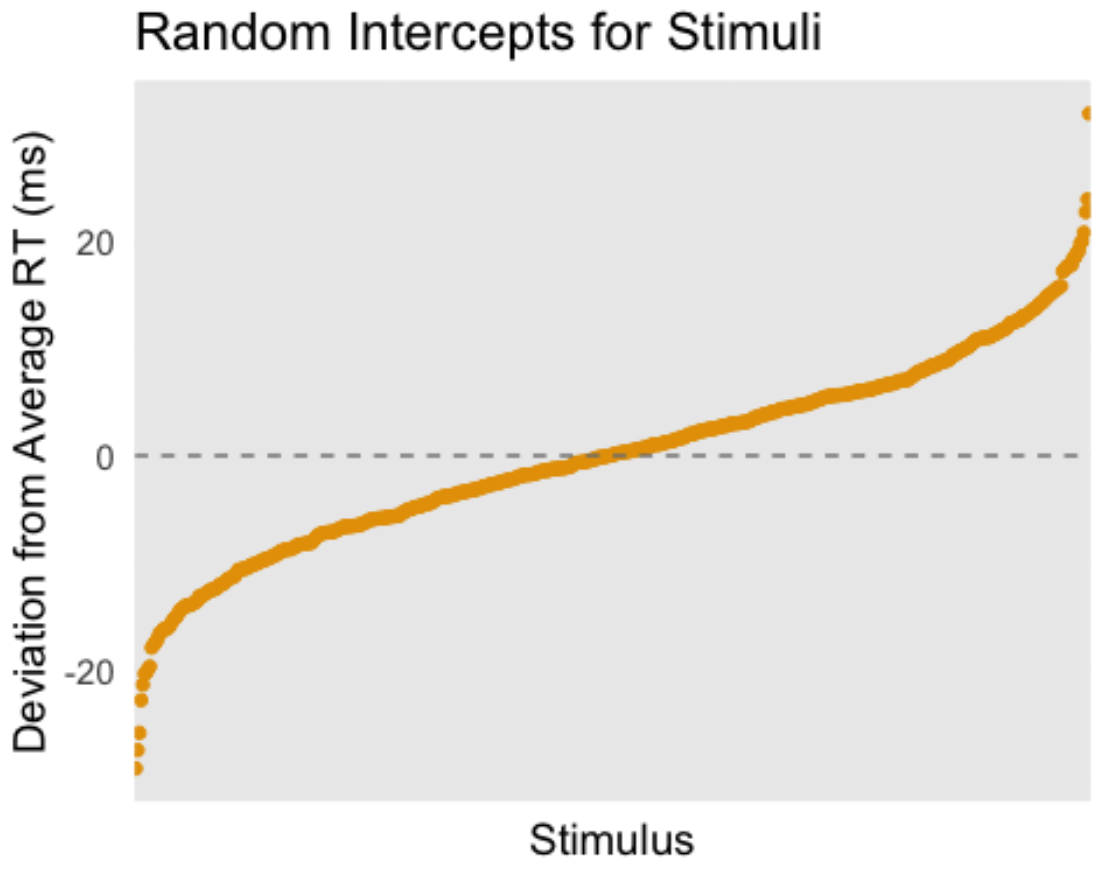
If the model fit perfectly, all points would fall exactly on the diagonal dashed line, where predicted equals observed. In practice, we expect some scatter around the line because no model is perfect, but the general alignment tells us whether the model captures the main structure of the data.

Here, the points follow the diagonal reasonably well, showing that our mixed model fits the data fairly closely. This is a good visual check to ensure the model is doing a reasonable job at capturing both the central trend and the variability in reaction times.

Random intercepts for Stimuli

```
library(lme4)
library(dplyr)
library(ggplot2)

# Extract random effects for stimuli
stim_re <- ranef(model_final)$stim
stim_re <- tibble(stim = rownames(stim_re),
                  intercept = stim_re[["(Intercept)"]])
```

```r
# Sort by intercept
stim_re <- stim_re %>%
  arrange(intercept) %>%
  mutate(stim = factor(stim, levels = stim))

# Plot random intercepts
ggplot(stim_re, aes(x = stim, y = intercept)) +
  geom_point(color = "#E69F00") +
  geom_hline(yintercept = 0, linetype = "dashed", color = "gray50") +
  labs(
    title = "Random Intercepts for Stimuli",
    x = "Stimulus",
    y = "Deviation from Average RT (ms)"
  ) +
  theme_minimal(base_size = 14) +
  theme(axis.text.x = element_blank())
```



This plot shows how the model has estimated the random intercept for each word.

- Stimuli above the dashed line have **positive intercepts**, meaning they tend to produce **slower-than-average** reaction times.

- Stimuli below the line have **negative intercepts**, meaning participants respond to them **faster than average**.

The spread of these values (about –25 to +25 ms) tells us the **range of item-level variability** captured by the model.

By allowing these small deviations per stimulus, the model doesn't overestimate the effect of modality, it acknowledges that part of the variability comes from the stimuli themselves, not just from the experimental manipulation.