Using Mixed Effect Models to Analyze Acceptability Rating Data in Linguistics

Fabian Bross

Version 1.0

Overview. This is part II of my tutorial on acceptability rating studies in linguistics. In this second part, I very briefly introduce mixed models to analyze rating data.

If you have any suggestions, feel free to write me: fabian.bross@ling.uni-stuttgart.de

Please cite as:
Fabian Bross (2019). Using Mixed Effect Models to Analyze Acceptability Rating Data. Version 1.0. Mimeo. Online: `www.fabianbross.de/mixedmodels.pdf`.

Contents

I assume that you are familiar with part I of the tutorial (see: `www.fabianbross.de/acceptabilityratings.pdf`). In this second part, you will learn how to analyze your data using mixed models (the final goal is a mixed model with random intercepts and random slopes). Again, I will only go through some basic examples. The exact design of the model you need to construct to analyze your data will depend on your study. Thus, it is important that you understand the concept of mixed models and I recommend you to read Bodo's tutorials (Winter 2013) and to have a look at Baayen (2012). Zuur et al. (2009) is also a good source and Mangiafico (2016) is a great introduction for beginners. As in the first part of the tutorial, we are going to use R and RStudio.

Linear mixed effect models are totally ***en vogue***. They are used instead of more classical statistical tests like *t*-tests and have some really cool advantages. The main point is that they are very, very flexible. They can account for a lot of variability in your data and can handle all sorts of different levels of measurements at the same time. Another advantage is that your data need not be normally distributed as mixed models are very robust against violations of normality. Finally, repeated measure designs can be analyzed with mixed models (and as every participant rates several sentences we are dealing with a repeated measure design).

1. A Short Note on Mixed Models and Likert Items

> "Go ahead and use a LMEM on your Likert-scale data!"
>
> Kizach (2014)

In part I of the tutorial I talked about the fact that it is often claimed that it is not possible to use parametric tests (for example, a *t*-test) with data that was obtained through Likert items. As discussed, this may be true mathematically speaking, however, it has been repeatedly shown that empirically this is not true: "One of the beauties of statistical methods is that, although they often involve heroic assumptions about the data, it seems to matter very little even when these are violated" (Norman 2010). The same claim was made for mixed models, i. e., that they are suitable for the kind of data you obtain from Likert items and especially using Likert items in acceptability judgment studies; see, for example, Gibson, Piantadosi & Fedorenko (2011), Kizach (2014), and similarly Cunnings (2012).

However, if you are more conservative you may want to fit an ordinal mixed effects model. I will show how to do this in Section 9. Fortunately, building a linear mixed effects model and building an ordinal mixed effects model is very similar.

2. Random and Fixed Effects

Before turning to what mixed models are and how they work you need to understand the differences between random effects and fixed effects (or fixed factors and random factors). In an empirical study you usually measure different things. In an acceptability rating study, for example, you may measure the acceptability of sentences, the age and gender of participants and so on (measuring in the sense of measure theory, see part I of the tutorial).

Let's talk about gender first. Gender only has two values (in the statistician's world) or two levels, namely male and female. If you are going to ask your participants for their gender, I assume you have a reason to do so. Perhaps you predict that gender may have an influence on what you are interested in. Then, you are dealing with a fixed effect.

Fixed effects:

- Influence your data in a systematic way (i. e. the influence of the effect is predictable).
- Exhaust the levels of a factor (gender is a factor with two levels, there are no more levels than these two levels).

Now let's think about our participants. One participant may judge one sentence different from another participant. This means that there will be variation between subjects that is random. Additionally, you only looked at a sample of a whole population (the population of native speakers of the language of interest). While there is a true underlying value of your measurements, there is some random variation in the data your measured that comes from the fact that you only chose (or sampled) a small part of the population of interest. Participants are a random factor.

Random effects:

- Have random influences. There is an unsystematic part in them (not all participants give exactly the same ratings on the same item).
- Do not exhaust the levels of a factor (there are more native speakers of the language under consideration than the speakers you chose for your study).
- Have to be categorical.

If you want to make a prediction on a whole population, but the levels in your study only represent a sample of the population, you are dealing with a random effect.

Whether a factor is random or fixed sometimes depends on your research question and is somewhat philosophical. The following example may be a bit unrealistic and oversimplified, but its sole purpose is illustration: Suppose you want to know something about teaching methods and you study the performance of two classes. The two classes have two different teachers and you want to take this into account. If the goal is a very broad generalization you want to take the two teachers into account, but only because you want to abstract away from them having different personalities. As your factor has many different levels (there are many different teachers out there), but you only measure two, you are dealing with a random effect. However, it could be that you want to compare the two teachers and are only interested in the differences between the two of them, and thus your factor has two levels. As you look at all the levels of the factor (the two teachers), you are dealing with a fixed effect.

Remember from the first part of the tutorial that we are interested in a construction and that it is not possible to test the construction because it is an abstract entity. We are only able to construct sentences, i. e., concrete items that we can test. There is an infinitely large number of sentences that can be compiled using a specific construction, but you only test a few of them. This means that this kind of factor does not exhaust its levels. Thus we are dealing with a random effect (cf. Clark 1973).

Note that I said that random effects have to be categorical. This means, that age, for example, can never be a random effect as it is an interval level.

3. Understanding Intercepts and Slopes

Linear mixed models build upon linear models and linear models build upon linear functions. Linear functions are very easy to understand, but you have to start the 'line thinking'. Let's take a very simple example. You have two sentences. One sentence you expect to be ill-formed and one sentence you except to be well-formed. Let's call them item 1 and item 2. Two participants rate the sentences from 1 (unnatural) to 7 (natural). So you end up with four judgments. Let's say, participant 1 rates item 1 as being a 2 and item 2 as being a 7. Participant 2 rates item 1 as being a 1 and item 2 as being a 6. I depicted this in Figure 1.

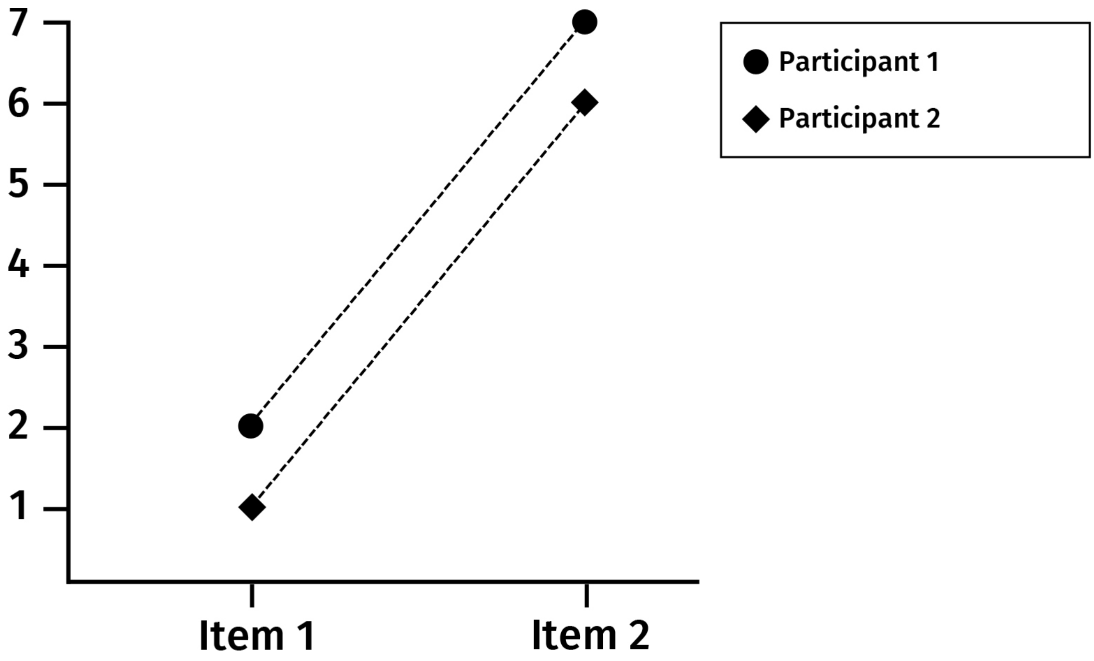


Figure 1: Two participants rated two items from 1 to 7.

I already started to prime you with the line thinking as I have drawn a line between the judgments. As the figure shows, the two participants exhibit the same trend. Both rated item 1 as being unnatural and item 2 as being natural as expected. However, participants also differ. We can call this difference a difference in intercepts.

An 'intercept' is a crossing point. Suppose you have a coordinate system and a line. The point where the line crosses the y-axis is called the y-intercept and the point where the line crosses the x-axis is called the x-intercept. What we are concerned with here is the starting point of our line, i. e., the point at which our line crosses item 1. If you look at Figure 1, participant 1 has an intercept of 2 and participant 2 has an intercept of 1. These are the points where the lines cross the first item that was rated.

Now, the intercepts are the only thing that are different in Figure 1 if we compare the lines. There is no difference in slope. This means that the lines are parallel. The 'slope' of a line describes its direction and steepness. The lines in Figure 1 have exactly the same direction and steepness.

Things, however, could have been different (and in real life they usually are). Suppose, participant 2 thought about item 2 that it is not totally natural and rated it as being a 4. This is depicted in Figure 2. Now, we are not only dealing with different intercepts, but also with a difference in slopes.

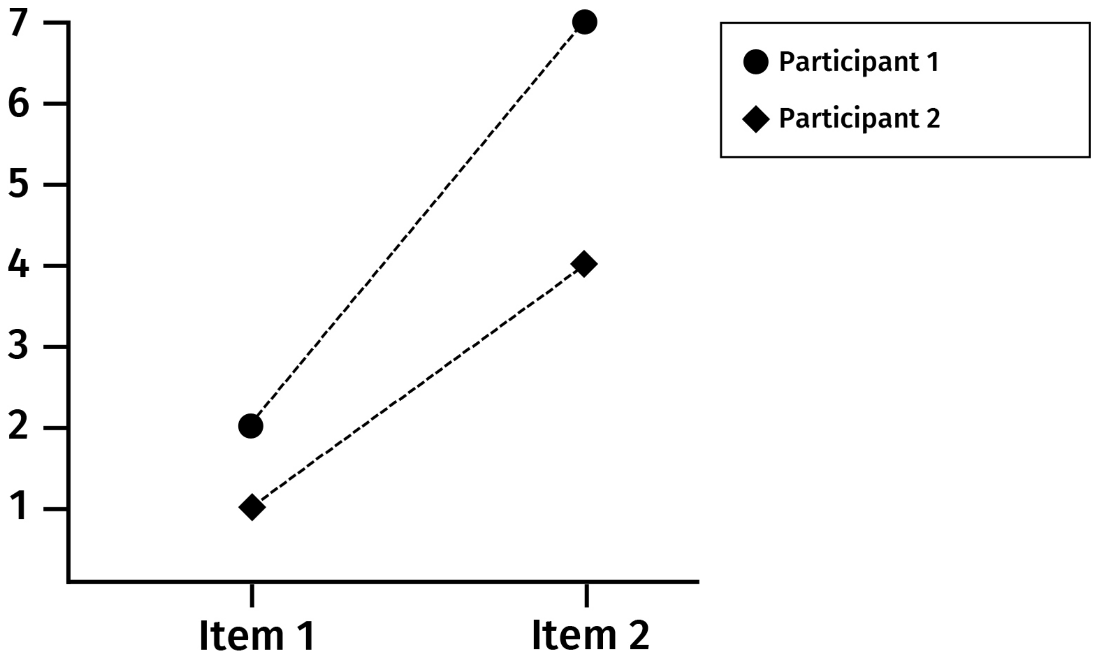


Figure 2: Two participants rated two items from 1 to 7.

Now, this is where mixed models come into play. The variation we have seen, i. e., the variation in intercepts and slopes, is a random variation. Mixed models are capable of taking this variation into account. What we want is a model with random intercepts and random slopes. Or to be more precise: We want a model that allows for variation. Our model should allow, for example, the participants to differ in their intercepts and slopes.

4. Linear Functions

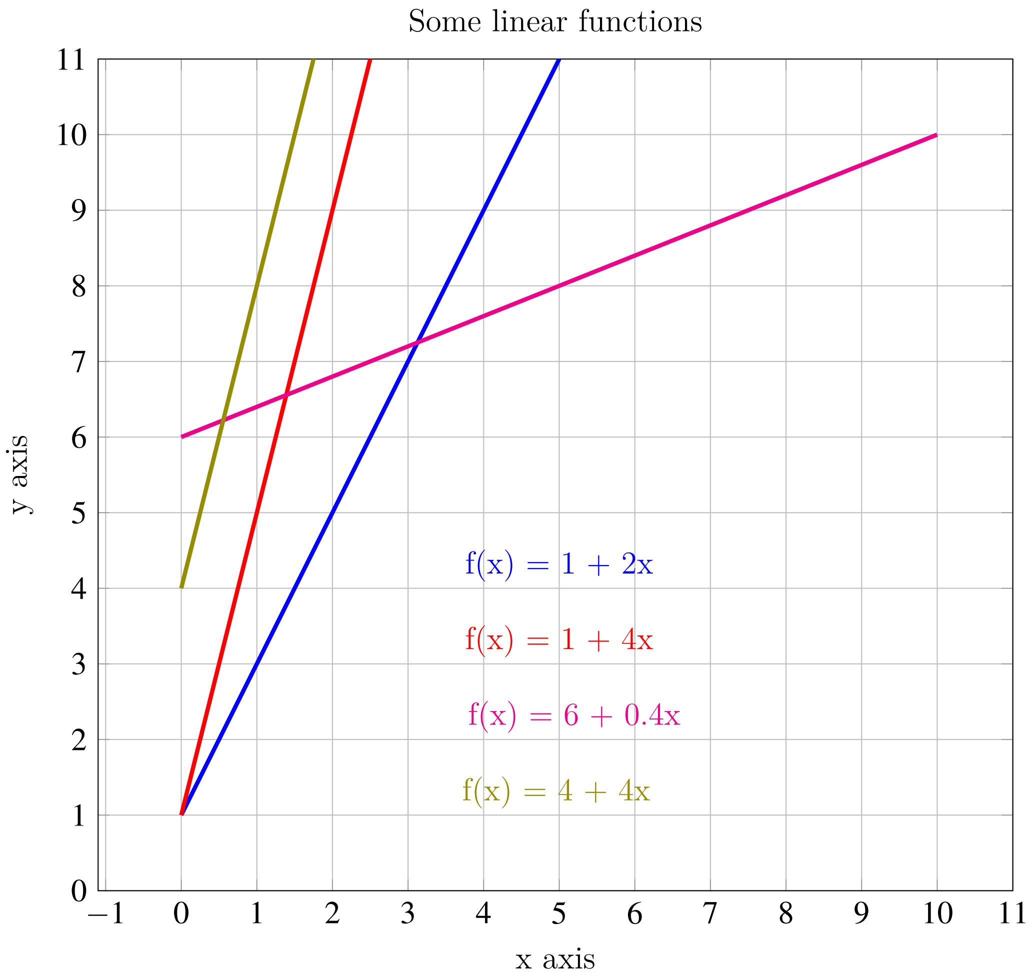
A linear model is basically a line. This line can be described mathematically. For describing a line, we only need two ingredients. An intercept and a slope. If we call the intercept a and the slope b, we arrive at the following formula:

$$f(x) = a + bx \tag{1}$$

Another way of writing this is:

$$y = a + bx \tag{2}$$

If your line is going up, the slope is positive, if your line is going down, the slope will be negative. You will understand what the formula does if you look at the following graphic. Compare the intercept and slope values of the formulas with the corresponding lines.



You will notice that the olive line with the corresponding formula $f(x) = 4+4x$ starts at 4 on the y-axis as the intercept in the formula is specified at 4. With 4, the slope is positive, so the line goes up (specified by $4x$). Starting from the beginning of the line you can go one step to the right (on the x-axis) and you will see that the line grew about 4 steps on the y-axis. This is because the slope is 4. You can make similar observations for the other lines. The magenta line is specified as follows: $f(x) = 6+0.4x$. It thus has an intercept of 6, meaning that it starts at 6 on the y-axis and if you go one step to the right on the x-axis (starting from its beginning), the line has gone up 0.4 steps on the y-axis.

Start the line thinking: Suppose, we have 2 items and 10 participants rate the two items (from 1 to 7). The ratings, we obtain are:

- Item 1: 1, 2, 2, 1, 3, 2, 1, 2, 1, 3
- Item 2: 6, 6, 5, 6, 7, 5, 6, 6, 6, 7

The mean rating for item 1 is 1.8 and for item 2 we calculate a mean of 6. We can graphically depict this as in Figure 3. In the figure, you see the 20 ratings, and the means depicted as bigger diamonds, and finally, a line connecting the two means. The important point is that the bigger the difference between the ratings, the steeper the slope will be.

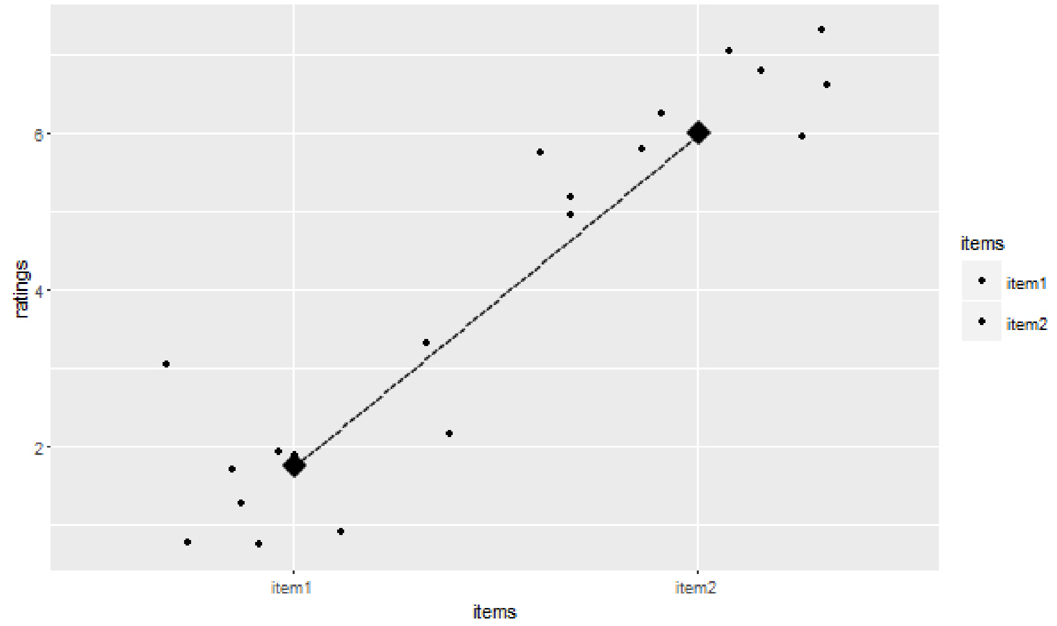


Figure 3: Ten participants rated two items.

What this example is intended to illustrate is simply that differences between groups can be conceptualized by line thinking. In reality we cannot simply draw a line between two means or calculate a line (or a linear function) that fits our data like this. The reason for this is that linear models are based on an assumption called independence. This simply means that the data points we are working with should be independent from one another. As the same subjects rated two different items, these data points are not independent, but clearly dependent. However, we do not have to care about this much, as we are not going to use linear models, but linear mixed models for our analysis. And the independence assumption does not hold for mixed models. Nevertheless, I will say a few more words about linear models (just for a better understanding of what's happening).

5. A Few More Words About Linear Models: Residuals

Suppose you have some toy bricks. Each toy brick has a height of 2 centimeters. You want to build a tower. Using 0 toy bricks your tower has a height of 0 centimeters. Using 1 toy brick your tower has a height of 2 centimeters. Using 2 toy bricks your tower has a height of 4 centimeters. You get it. You can describe what's happening with a linear function. This means that you can draw a line to predict what will happen if you use, for example, 200 toy bricks. However, in the real world, things aren't that easy.

Suppose you have some measurements that look like in Figure 4a. This could be, for example, reaction time measures from one participant on different occasions. Perhaps, we played some music and the participant had to react to a stimulus. As the music gets louder the participant slows down. To describe your data, you fit a simple linear model. What you do is called 'linear regression'. Remember that the formula for a linear function looks like this:

$$y = a + bx \tag{3}$$

Of course, there is no line that is able to connect all your data points. This is because there is some variation in your data. Nevertheless, with a regression you can find the line that fits your data best. So for most of the data we will need to add some variation to our model. We simply add an error term:

$$y = a + bx + \varepsilon \tag{4}$$

Although we cannot find a line that connects all our observations, we can search for a line that fits our data points best. Such a line can look like in Figure 4b. The line is what the regression predicts to be the best fit. It represents your prediction. The dots are your data points, your actual measurements. The distance between your observed values (the dots) and your predicted values (the line) has its own name. This distance is called residual. The residuals are depicted in Figure 4c.

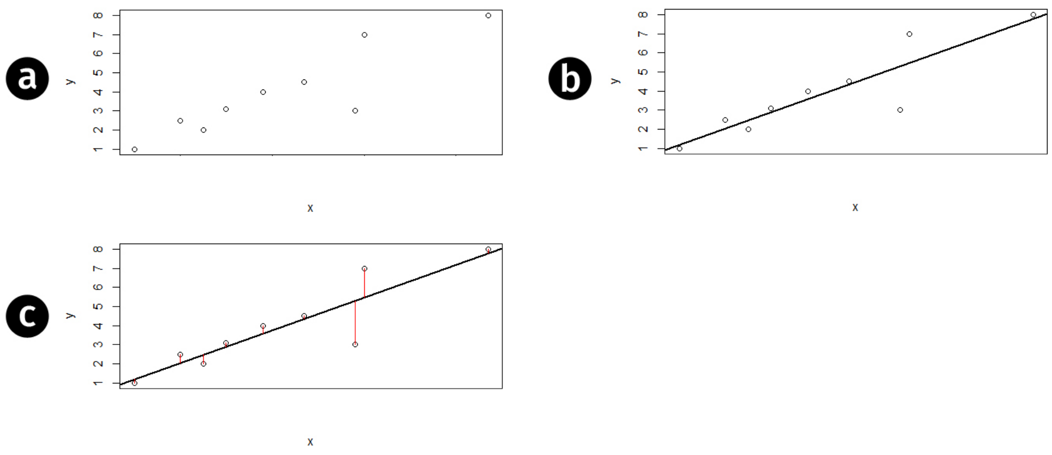
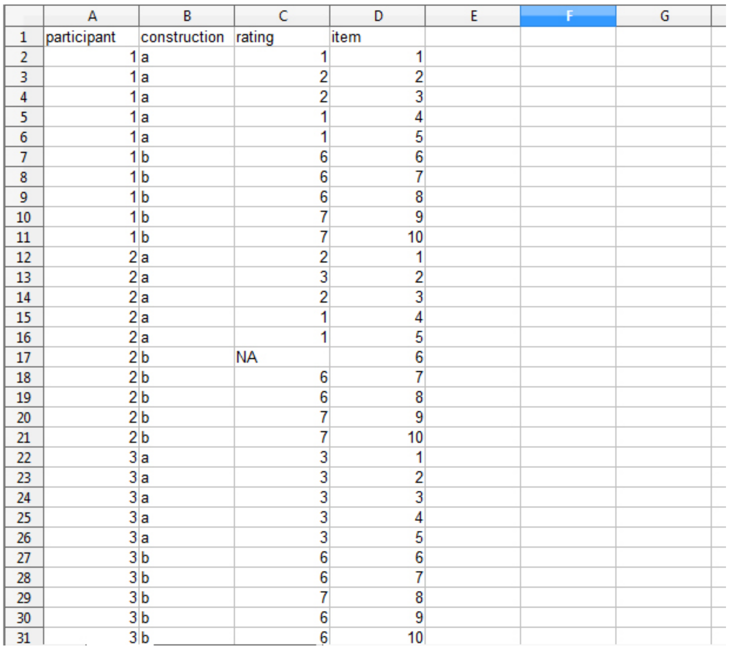


Figure 4: Residuals.

Thus, a residual is the distance between an observed and a predicted value. Remember that we are dealing with line fitting when doing our models. Why am I telling you this? Each statistical method is built upon assumptions. The assumptions we are dealing with here mainly have to do with the distribution of the residuals (and not with the distribution of the data itself). We will talk about this later.

6. Building a Model

Suppose 5 participants have rated 10 items. The research question was if two constructions A and B differ from one another. 5 items represent construction A and 5 items represent construction B. Your data will come from a .csv file that looks like the one in Figure 5.



	A	B	C	D	E	F	G
1	participant	construction	rating	item			
2	1	a	1	1			
3	1	a	2	2			
4	1	a	2	3			
5	1	a	1	4			
6	1	a	1	5			
7	1	b	6	6			
8	1	b	6	7			
9	1	b	6	8			
10	1	b	7	9			
11	1	b	7	10			
12	2	a	2	1			
13	2	a	3	2			
14	2	a	2	3			
15	2	a	1	4			
16	2	a	1	5			
17	2	b	NA	6			
18	2	b	6	7			
19	2	b	6	8			
20	2	b	7	9			
21	2	b	7	10			
22	3	a	3	1			
23	3	a	3	2			
24	3	a	3	3			
25	3	a	3	4			
26	3	a	3	5			
27	3	b	6	6			
28	3	b	6	7			
29	3	b	7	8			
30	3	b	6	9			
31	3	b	6	10			

Figure 5: A simple .csv file in OpenOffice.

The structure of this file is pretty much straightforward: There is a column for participants and each participant has a number. There is a column specifying which item belongs to which construction and there is a column for the ratings and a column for each item. Note that there is one rating missing. You don't need to worry about missing items. Simply label them "NA" in your .csv file. The mixed model can handle that.

Our goal is a model with both, fixed and random effects. That's why it is called a mixed model. The general formula looks like this:

$$y = X\beta + Zu + \varepsilon \tag{5}$$

We do not need to go into the formula. I just want to make two brief notes. The first note is about ε. This is the error term. You do not measure deterministic systems. Thus, there will always be some variation from a lot of different sources. That's why there is an error term in the formula. The second note is that this formula looks pretty much the same as the simple formula for linear models we have already seen. The reason for this is that both models contain an intercept and a slope.

Now we are going to analyze our data by building the model. We need to talk to R.

Our goal is to predict the ratings based on the construction type. In other words: We believe that the ratings of construction A will differ from the ratings of construction B. We should thus be able to predict the ratings when looking at the construction type. In R language, this can be written as

```
rating ~ construction
```

We can read this term as “rating predicted by construction” or “rating as a function of construction”. Before we can start to build our model we need two things. The first thing is the .csv file and the second thing is a package called “lme4” (Bates, Maechler & Bolker 2019). Open RStudio and import your data. You can import the data by typing in:

```
data = read.csv("http://www.fabianbross.de/tutorialdata.csv")
```

Note that I already gave the dataset a name. It’s now called ‘data’. Alternatively, you can download the file and go to RStudio. You can import the file by clicking on “Import Dataset” (in upper right corner). The data is completely made-up. However, it will show you how you need to organize your data. You can take a look at the file by typing in “data” or by just looking at the first few rows with “head(data)”. The result of the latter command looks like this:

```
 participant item rating construction dialect zrating
1     a1  1   7       a         dialecta −1.9668302
2     a1  2   4       a         dialecta −0.2809757
3     a1  3   6       a         dialecta −1.4048787
4     a1  4   2       a         dialecta 0.8429272
5     a1  5   2       a         dialecta 0.8429272
6     a1  6   2       b         dialecta 0.8429272
```

We can see that there is a column for the participants. I gave them names like ‘a1’, ‘a2’ and the like. It’s just names. Then there is a column for the items. Each participant rated 10 items from 1 (‘unnatural’) to 7 (‘natural’). Two constructions were tested. These are labeled ‘a’ and ‘b’. Finally, the language under investigation has three dialects (and no more). I called them ‘dialecta’, ‘dialectb’, and ‘dialectc’. Additionally, there is a column called ‘zrating’ containing the *z*-transformed values of the Likert ratings (see part one of the tutorial). We will need this column later. Let’s explore this data set a little bit more. Let’s ignore the fact that there are different dialects of the language for the moment. The first thing we want to know are the mean ratings of construction A and construction B. To get these values we need to explain to R first that we want to take a look at the column 'rating' in the data set called 'data'. To achieve this we write 'data$rating'. Thus, the general format of specifying a specific column in a data set is 'datasetname$columnname'. With 'mean(data$rating)' we would get the mean of all ratings. However, this is not what we want. We want the mean values of the constructions A and B. To do this, we type:

```
mean(data[data$construction=="a",]$rating)
mean(data[data$construction=="b",]$rating)
```

The result you get with these two commands is that the mean rating of construction A is not available ('NA') and that the mean rating of construction B is 4.682759. That the mean of construction A is not available is easy to explain: The reason is that there are some missing values in our data set. To fix this we need to tell R to ignore the missing values. The command for this is 'na.rm=TRUE'. Thus we now write:

```
mean(data[data$construction=="a",]$rating, na.rm=TRUE)
mean(data[data$construction=="b",]$rating, na.rm=TRUE)
```

We can get the standard deviations in a similar way:

```
sd(data[data$construction=="a",]$rating, na.rm=TRUE)
sd(data[data$construction=="b",]$rating, na.rm=TRUE)
```

So what we have now is that construction A received a mean rating of 3.167832 ($SD = 1.723843$) and construction B received a mean rating of 4.682759 ($SD = 1.843545$). When reporting these values it is common to round the numbers, so we would say that the mean rating of construction a was 3.17 ($SD = 1.72$) and the mean rating of construction B 4.68 ($SD = 1.84$). To get a better mental representation of the data let's look at a box plot in my favorite box plot format discussed in part one of the tutorial:

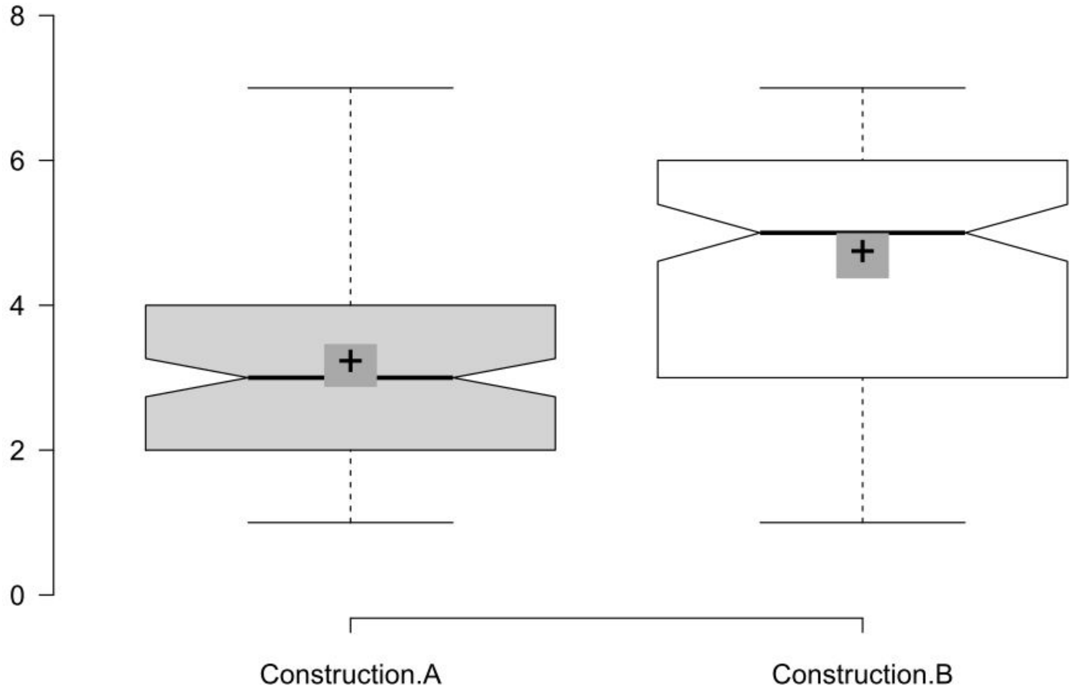


Figure 6: Box plots of the ratings of construction A and construction B. The black crosses indicate the mean ratings, the gray boxes around the crosses indicate the 95 % confidence intervals of the means. The notches indicate the 95 % confidence intervals of the medians (the definition of whisker extend is Tukey).

From visual inspection we expect the two constructions to be significantly different from each other as the confidence intervals do not overlap. Now let's look at how the dialect speakers differ in their judgments. Figure 7 shows six box plots. The gray box plots show the ratings of construction A and the white box plots the ratings of construction B. The first two box plots represent the data from the speakers of dialect A, the third and fourth box plots the ratings of the speakers of dialect B, and the last two box plots are the ratings of the speakers of dialect C.

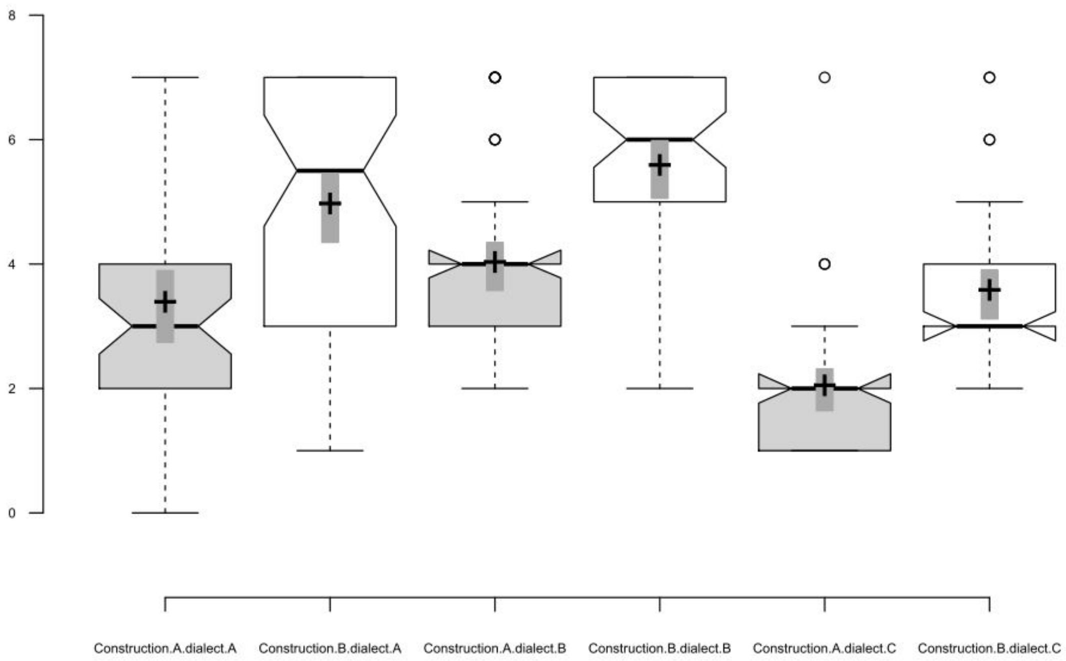


Figure 7: Box plots of the ratings of construction A and construction B by dialects. The black crosses indicate the mean ratings, the gray boxes around the crosses indicate the 95 % confidence intervals of the means. The notches indicate the 95 % confidence intervals of the medians.

What the box plots show is that the same trend can be observed in all dialects, namely that construction B was rated to be more natural than construction A. Nevertheless, the dialects behave quite dissimilar.

Now we come back to our model. Again, let's ignore the fact that there are different dialects for a moment. Remember that we want to predict the ratings by taking into account that different constructions were rated:

```
rating ~ construction
```

To model this we need an additional package:

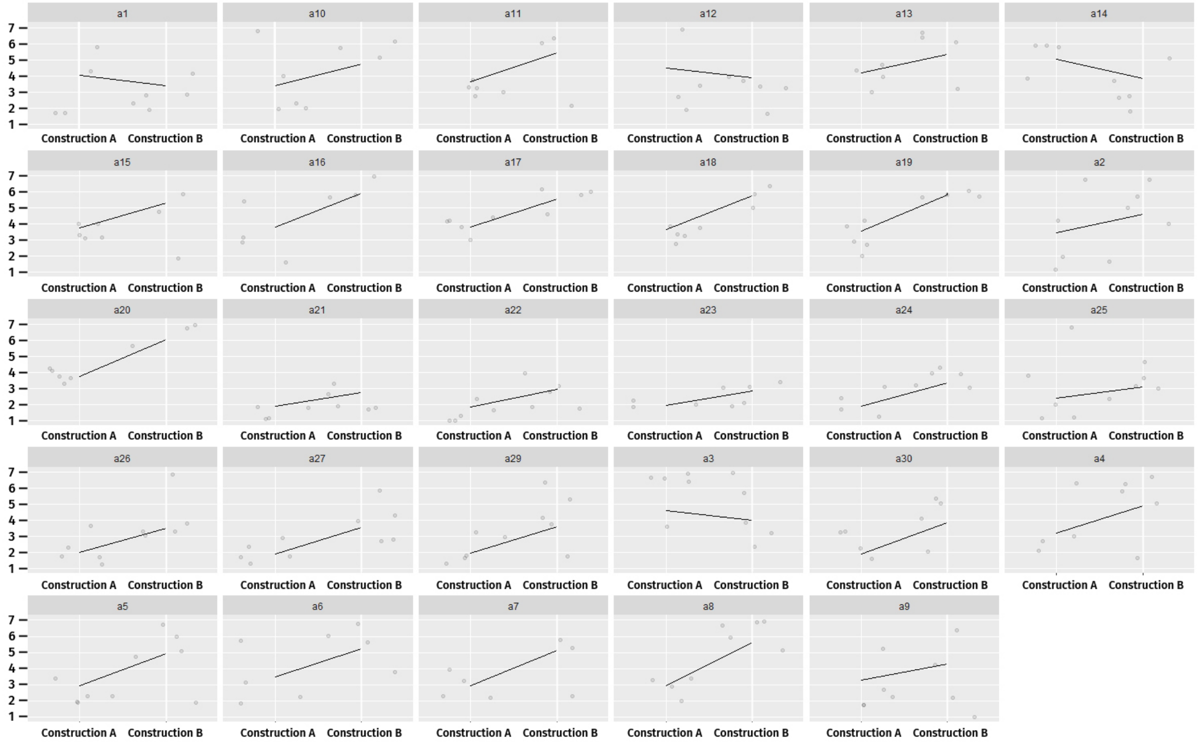
```
install.packages("lme4")
library(lme4)
```

The first line installs the package (don't forget the quotation marks) and the second line loads the package (don't forget not to use quotation marks here). Now, we can build a first model:

```
modelone = lmer(rating ~ construction, data=data)
```

This builds a model and gives it the name ‘modelone’. It is a linear mixed effect model specified by ‘lmer’. This model wants to predict ‘rating’ on the basis of ‘construction’ using the data labeled ‘data’ (note that if your dataset would be called “datacool” this would read “data=datacool”). This, however does not work and the result is an error: ‘No random effects terms specified in formula’. The reason for this error is that we did not build a mixed model as we did only specify a predictor (rating) and a fixed effect (construction), but no random effect.

We already know from our box plots comparing the dialects that the speakers obviously differed in their ratings. We can confirm this by making a scatterplot for each participant. For a better overview I have already fitted a model for each of them:



From this graphical representation we can see the random variation we want to account for. The most noticeable difference between the participants is that they differ in slope. However, let’s ignore that for a minute. The second thing we notice is that the participants differ in their intercepts. For example, participant a1 in the upper left corner has an intercept of 4 while participant a14 in the upper right corner has an intercept of 5. Let’s account for this in our model. Thus, let’s add the missing random part to our model. To be more precise, we tell the model that each participant may have its own intercept:

```
modeltwo = lmer(rating ~ construction + (1|participant),
   data=data)
```

What is different compared to the first model is that we added '(1|participant)'. The number 1 stands for the intercept. Thus the term means that each participant is allowed to have its own intercept. Think again about why we are doing this: It may be that some participants may rate your items generally worse or better than other participants. Adding random intercepts for participants accounts for this variation. Before we take a look at the output of this model let's think more about potential sources of variation.

Participants will surely vary in their ratings. As the levels of the participants do not exceed the levels of the population (there are more speakers than we tested) we will regard them as random factors. However, the same is true for our items. We constructed only 5 example sentences per construction although there is an infinitely large number of sentences that could be constructed using an incredibly large amount of lexical items. Additionally, it may be that some items in general receive better ratings than others and some items may be judged to be worse. Thus, there is variation concerning the intercepts of the items and we want to account for this variation too. So let's add our items into the random effect part of our model:

```
modelthree = lmer(rating ~ construction + (1|participant) +
   (1|item), data=data)
```

Again, the term '(1|item)' tells the model that the items are allowed to vary in their intercepts. Note that you may receive a warning message if you compile the code above: 'singular fit'. Pretty much simplified this means that the variance of one effect (or a linear combination of variances of effects) is zero or close to zero.[1] As this is only an example we can ignore the warning here.

Now, lets look at the output. If you type:

```
summary(modelthree)
```

you will get:

```
Linear mixed model fit by REML ['lmerMod']
Formula: rating ~ construction + (1 | participant) + (1 | item)
  Data: data
```

[1] When a model is singular this also means that the standard deviations of the estimates cannot be derived.

```
REML criterion at convergence: 1116.7

Scaled residuals:
   Min     1Q  Median   3Q    Max
−2.48719 −0.59148 −0.09075 0.54508 2.80007

Random effects:
 Groups   Name     Variance Std.Dev.
 participant (Intercept) 0.7606 0.8721
 item     (Intercept) 0.0000 0.0000
 Residual          2.4446 1.5635
Number of obs: 288, groups: participant, 29; item, 10

Fixed effects:
          Estimate Std. Error t value
(Intercept) 3.1729 0.2082 15.241
constructionb 1.5099 0.1843 8.192

Correlation of Fixed Effects:
         (Intr)
constructnb −0.446
convergence code: 0
singular fit
```

First, the summary tells you that you produced a mixed model and reminds you of your formula. There is a lot of stuff we will ignore. The two things we are interested in are the random effects and the fixed effects part. Let's stick with the fixed effect part first. There are three columns for the fixed effects: a column for the estimate, a column for the standard error and a column for the *t*-value. The estimate for the intercept is 3.1729. This is simply the mean of construction A (or approximately the mean). The mixed model displays the intercept of the fixed effect that comes first in the alphabet. As our constructions were labeled 'a' and 'b' in the data set we see the mean of construction A here. The more complicated the model gets the harder the intercept value will be to interpret.

Remember that the mean we calculated for construction B was 4.6828. This number does not show up in the summary at all. However, we don't need this information. Remember that we are dealing with lines here. And we describe lines in terms of intercepts and slopes. Let's look at the next number, the estimate of what is mysteriously called 'constructionb': 1.5099. This is the slope. We can easily calculate it by subtracting the mean value of construction B from the mean value of construction A: $4.6828 - 3.1729 = 1.5099$.

Side note: Each participant and each item is allowed to have a different intercept (but the same slope) in our model. You can take a look at all these intercepts by typing: 'coef(modelthree)'.

Now let's turn to the random effects part. Here, variance and standard deviations for participants and items are reported. Actually, there is no variance for the items which is due to the fact that we took the items in our model into account. Similarly, the participants do not vary much.

Before we take the dialects into account let's look at some *p*-values (because I know you want them). The most simple version: First, load the lmerTest package (Kuznetsova, Brockhoff & Christensen 2019):

```
install.packages("lmerTest")
library(lmerTest)
```

Now, run the model again:

```
modelthree = lmer(rating ~ construction + (1|participant) +
   (1|item), data=data)
summary(modelthree)
```

The only thing that has changed in the output is that we have *p*-values now:

```
Fixed effects:
          Estimate Std. Error df t value Pr(>|t|)
(Intercept) 3.1729 0.2082 43.4249 15.241 < 2e-16 ***
constructionb 1.5099 0.1843 258.1645 8.192 1.2e-14 ***
---
Signif. codes: 0 ''*** 0.001 ''** 0.01 ''* 0.05 ''. 0.1 ''1
```

In short: The intercept is significantly different from 0 with $p = 2e - 16$ and construction B is significantly different from construction A with $p = 1.2e - 14$. We could report:

> Construction A received a mean rating of 3.17 ($SD = 1.72$) and construction B a mean rating of 4.68 ($SD = 1.84$). See Figure XY (error bars indicate 95 % confidence intervals). A mixed-effects model was constructed in R (R Core Team 2015) using the lme4 package (Bates, Maechler & Bolker 2019) and lmerTest (Kuznetsova, Brockhoff & Christensen 2019) to obtain p-values. The model contained construction type as a fixed effect (i. e., construction A versus construction B). Random intercepts for participants and items were added. We predicted participant's ratings as a function of construction type. The full model translates to: lmer(rating ~ construction + (1|participant) + (1|item), data=data). Construction B was rated more acceptable compared to construction B (fixed effect intercept estimate: $\beta_0 = 3.1729$ (SE $= 0.2082$); fixed effect slope estimate $\beta_1 = 1.5099$ (SE $= 0.1843$); $p < 0.001$).

Note that there is no standardized way of reporting mixed models (yet). You should describe your model as precisely as possible (what were the fixed effects and what were the random effects and what did they look like) so that others can replicate it. I'll give you more tips on reporting later. A great tip is to look at what happens graphically by exploring:

```
modelfoura <-step(modelthree)
plot(modelfoura)
```

Note that the error bars in the plot indicate 95 %-confidence intervals.

7. Taking the Dialects into Account

Now, let's account for the fact that the language has three dialects. We'll add the dialects as a fixed effect as the levels we looked at exceed the levels of dialects (as I said, the language only has three dialects). This is pretty straightforward:

```
modelfour = lmer(rating ~ construction + dialect +
   (1|participant) + (1|item), data=data)
summary(modelfour)
```

The result looks like this:

```
Linear mixed model fit by REML. t-tests use Satterthwaite's
   method [lmerModLmerTest
]
Formula: rating ~ construction + dialect + (1 | participant) +
   (1 | item)
```

```
  Data: data

REML criterion at convergence: 1084.6

Scaled residuals:
   Min     1Q  Median    3Q    Max
−2.51995 −0.65785 −0.03098 0.58802 3.13754

Random effects:
 Groups   Name     Variance Std.Dev.
 participant (Intercept) 0.07392 0.2719
 item     (Intercept) 0.00000 0.0000
 Residual         2.44411 1.5634
Number of obs: 288, groups: participant, 29; item, 10

Fixed effects:
           Estimate Std. Error df t value Pr(>|t|)
(Intercept)  3.4241  0.2029 42.7915 16.872 < 2e−16 ***
constructionb 1.5107 0.1843 258.4005 8.198 1.15e−14 ***
dialectdialectb 0.5605 0.2533 26.2675 2.213 0.0358 *
dialectdialectc −1.4351 0.2602 26.2469 −5.515 8.42e−06 ***
−−−
Signif. codes: 0 ‘’*** 0.001 ‘’** 0.01 ‘’* 0.05 ‘’. 0.1 ‘’1

Correlation of Fixed Effects:
        (Intr) cnstrc dlctdlctb
constructnb −0.463
dialctdlctb −0.633 0.008
dialctdlctc −0.616 0.007 0.491
convergence code: 0
singular fit
```

We want two things now: We want to know if our model (modelfour) which predicted ratings on the basis of construction type and dialects accounts for the variance in the observed ratings better than the model taking only the construction type into account. The second thing you want is the information you need to report your results. We can easily get both at the same time by using the psycho package (Makowski 2019).

```
install.packages("psycho")
library(psycho)
```

```
outputmodelthree <- analyze(modelthree)
print(outputmodelthree)

outputmodelfour <- analyze(modelfour)
print(outputmodelfour)
```

The output is amazing! You get all the information you need:

```
The overall model predicting rating (formula = rating ~
   construction + (1 | participant) + (1 | item)) has an total
   explanatory power (conditional R2) of 35.28%, in which the
   fixed effects explain 15.14% of the variance (marginal R2).
   The model's intercept is at 3.17 (SE = 0.21, 95% CI [2.76,
   3.58]). Within this model:
  - The effect of constructionb is significant (beta = 1.51, SE
     = 0.18, 95% CI [1.14, 1.88], t(258) = 8.19, p < .001) and
     can be considered as medium (std. beta = 0.78, std. SE =
     0.095).
```

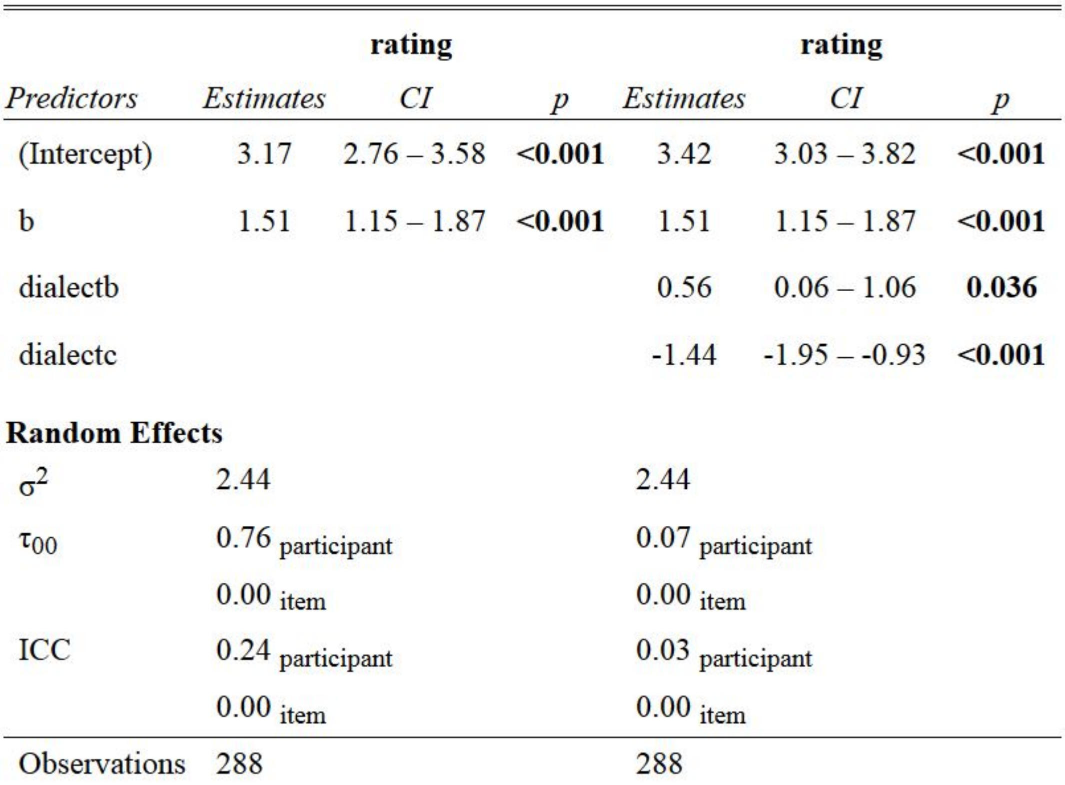
And:

```
The overall model predicting rating (formula = rating ~
   construction + dialect + (1 | participant) + (1 | item)) has
   an total explanatory power (conditional R2) of 35.41%, in
   which the fixed effects explain 33.45% of the variance
   (marginal R2). The models intercept is at 3.42 (SE = 0.20,
   95% CI [3.04, 3.81]). Within this model:
  - The effect of constructionb is significant (beta = 1.51, SE
     = 0.18, 95% CI [1.15, 1.87], t(258) = 8.20, p < .001) and
     can be considered as medium (std. beta = 0.78, std. SE =
     0.095).
  - The effect of dialectdialectb is significant (beta = 0.56,
     SE = 0.25, 95% CI [0.075, 1.05], t(26) = 2.21, p < .05)
     and can be considered as small (std. beta = 0.29, std. SE
     = 0.13).
  - The effect of dialectdialectc is significant (beta = -1.44,
     SE = 0.26, 95% CI [-1.93, -0.94], t(26) = -5.51, p < .001)
     and can be considered as medium (std. beta = -0.74, std.
     SE = 0.13).
```

We not only see that the model taking the dialect into consideration explains more of the variation but we also get a lot of other useful information! Another great tip for publishing your data is the package sjPlot (Lüdecke 2018) which can produce great tables:

```
install.packages("sjPlot")
library(sjPlot)
tab_model(modelthree, modelfour)
```

This produces a table (unfortunately no LaTeX support):



		rating			**rating**	
Predictors	*Estimates*	*CI*	*p*	*Estimates*	*CI*	*p*
(Intercept)	3.17	2.76 – 3.58	**<0.001**	3.42	3.03 – 3.82	**<0.001**
b	1.51	1.15 – 1.87	**<0.001**	1.51	1.15 – 1.87	**<0.001**
dialectb				0.56	0.06 – 1.06	**0.036**
dialectc				-1.44	-1.95 – -0.93	**<0.001**
Random Effects						
σ^2	2.44			2.44		
τ_{00}	0.76 $_{\text{participant}}$			0.07 $_{\text{participant}}$		
	0.00 $_{\text{item}}$			0.00 $_{\text{item}}$		
ICC	0.24 $_{\text{participant}}$			0.03 $_{\text{participant}}$		
	0.00 $_{\text{item}}$			0.00 $_{\text{item}}$		
Observations	288			288		

There are only two main topics left I want to talk about in this tutorial. The first thing is that we still haven't implemented random slopes into our model and the second thing is that we can also build an ordinal model. In the following we will fit a model with random intercepts and random slopes with the ordinal package. For this we are going to use the clmm function instead of the lmer function. Fortunately, both have the same syntax. Before doing this I will make a short side note on the assumptions underlying linear mixed models and why you may want to fit an ordinal model.

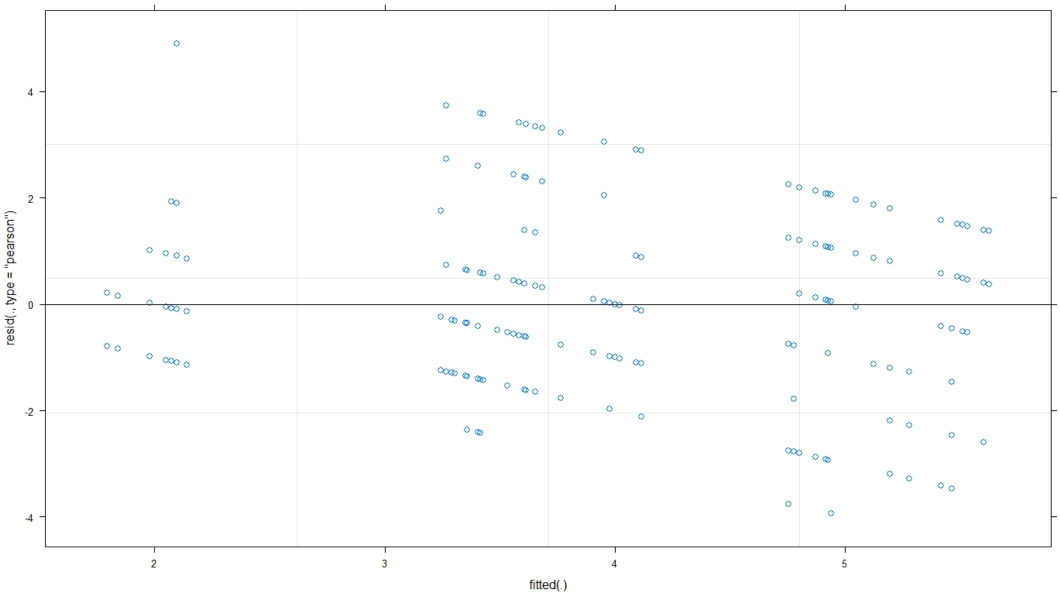
8. Side Note: Assumptions

There are several assumptions underlying mixed models. I already noted that independence is not an assumption so we are safe with our repeated-measure design. I will not go into much details here, but refer to Winter (2013) on what the assumptions are and how to test them. I will only make one brief remark on the linearity assumption and on homoscedasticity.

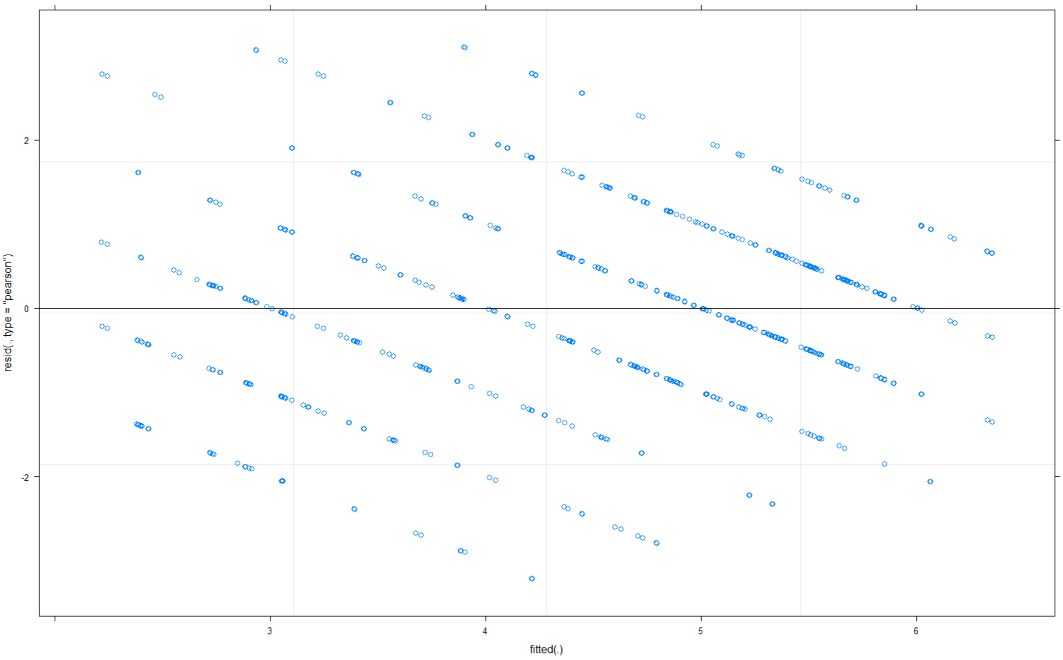
The residuals of your model should be linearly distributed (linearity assumption) and the variance of the residuals should be equally distributed (so the variance should not, for example, get greater with larger x-values) (homoscedasticity assumption). Both assumption can be assessed by visual inspection. You can test this yourself:

```
plot(model)
```

What you should not see is some kind of pattern (e. g., a curve), but more or less randomness (a blob or a cloud). For our made-up data, the plot does not look so bad, but there seems to be an emerging patter, namely stripes:



However, there should be no pattern and no strings nor stripes. The following plot comes from a subset of some actual rating data. Here we can clearly see the stripes. Each stripe indicates one of the seven scales from the Likert items.



This result indicates that an ordinal regression model is a more suited tool for the analysis, although I would not be too concerned with violating the two assumptions mentioned. If you want to be more conservative you can use an ordinal model.

9. Building an Ordinal Model

The last thing we want to do is to add random slopes. Remember that we want to account for the fact the participants not only differ in their intercepts but also in their slopes. The same is true for the items. Actually, adding random slopes for subjects and items is no big deal (and this works the same with lmer and clmm).[2] First, we load the ordinal package (Christensen 2018):

```
install.packages("ordinal")
library(ordinal)
```

Then, we fit our model with random slopes and random intercepts:

```
modelfive = clmm(rating ~ construction + dialect + (1 +
   construction|participant) + (1 + construction|item),
   data=data)
```

This, however, will result in an error message: ‘Error in getY(fullmf) : response needs to be a factor’. As we are trying to fit an ordinal model, the model expects the response, in

[2] By the way, clmm stands for ‘cumulative link mixed model’

our case ‘rating’, to be ordinal. To convert the responses into the ordinal level, we have to tell R that we want the responses to be ordered factors (i. e., ordinal):

```
data$rating <- factor(data$rating, ordered=TRUE)
```

Now, we can type again (this takes a bit):

```
modelfive = clmm(rating ~ construction + dialect + (1 +
   construction|participant) + (1 + construction|item),
   data=data)
```

The random factors now look like “(1 + factora|factorb)”. Again 1 stands for the intercept (that is allowed to vary) and the vertical bar indicates that participants are allowed to differ in their ratings of the construction regarding their slope and the same is done for the items. Wow! We created a model with random slopes and random intercepts! Now let’s look at what “summary(model)” tells us. Fortunately, the output looks very similar to what we have seen already:

```
Cumulative Link Mixed Model fitted with the Laplace
   approximation

formula: rating ~ construction + dialect + (1 + construction |
   participant) +
   (1 + construction | item)
data: data

 link threshold nobs logLik AIC niter max.grad cond.H
 logit flexible 288 -469.22 968.44 1418(4257) 6.66e-04 6.9e+05

Random effects:
 Groups   Name      Variance Std.Dev. Corr
 participant (Intercept) 3.001e-01 5.478e-01
          constructionb 1.397e+00 1.182e+00 -0.814
 item     (Intercept) 2.245e-11 4.738e-06
          constructionb 4.626e-02 2.151e-01 -0.297
Number of groups: participant 29, item 10

Coefficients:
           Estimate Std. Error z value Pr(>|z|)
constructionb 2.0227 0.3442 5.876 4.20e-09 ***
```

```
dialectdialectb 0.8213 0.3045 2.697 0.00699 **
dialectdialectc −1.8825 0.3388 −5.556 2.76e−08 ***
−−−
Signif. codes: 0 ‘’*** 0.001 ‘’** 0.01 ‘’* 0.05 ‘’. 0.1 ‘’1

Threshold coefficients:
   Estimate Std. Error z value
1|2 −2.8429 0.3662 −7.762
2|3 −0.5723 0.2789 −2.052
3|4  0.6839  0.2811 2.433
4|5  1.6782  0.2980 5.632
5|6  2.1465  0.3111 6.900
6|7  3.2306  0.3479 9.285
(2 observations deleted due to missingness)
```

As with every statistical method, clmm does make assumptions about your data. The most important assumption is the assumption of (partial) proportional odds in our case. The assumption of proportional odds is that the effect of the predictors (in our case construction and dialect) are constant for each increase in the level of the response (in our case ratings). In line-thinking terms, this assumption is also sometimes called the assumption of equal slopes. In R we can test the proportional odds assumption with the function ‘nominal_test()’ (and with ‘scale_test’). This function performs a likelihood ratio test. The hypothesis under test is that relaxing the proportional odds assumption will not improve the fit of our model. Practically, this means that we do not want significant *p*-values from this test. If the proportional odds assumption fails, the results of the model will not be reliable. Unfortunately, at the time of writing this tutorial, the function ‘nominal_ test()’ only works for clm and not for clmm. The problem with this is that with clm no random effect structure is allowed. So we could try fitting a model without random effect:

```
modelfive.clm = clm(rating ~ construction + dialect, data=data)
nominal_test(modelfive.clm)
```

As you will see, this produces significant results. However, the random effect structure is missing. Let’s hope that nominal_ test() will be available for clmm soon (this is planned so try if it’s already available!). Nevertheless, let’s proceed and let’s apply the scale test:

```
scale_test(modelfive.clm)
```

This produces:

```
Tests of scale effects

formula: rating ~ construction + dialect
         Df logLik AIC   LRT Pr(>Chi)
<none>      -473.80 965.59
construction 1 -473.80 967.59 0.0014 0.969770
dialect   2 -466.36 954.71 14.8809 0.000587 ***
---
Signif. codes: 0 ‘’*** 0.001 ‘’** 0.01 ‘’* 0.05 ‘’. 0.1 ‘’1
```

Again, we get a statistically significant result so we have scale effects in our model. But what are scale effects? It means that participants used the scale differently. Some participants may, for example, avoid the extreme values of the scale (so they do not rate items with 1 or 7). Others, in contrast, use the whole scale. One way out is to *z*-transform the ratings as discussed in part one of the tutorial. The *z*-scored ratings are already in our data set. The column's name is 'zrating'. Let's convert it into a factor:

```
data$zrating <- factor(data$zrating, ordered=TRUE)
```

Now, let's overwrite our clm model and test for scale effects again (this may take a while so grab a coffee):

```
data$zrating <- factor(data$zrating, ordered=TRUE)
modelfive.clm = clm(zrating ~ construction + dialect, data=data)
scale_test(modelfive.clm)
```

Indeed, there are no scale effects anymore:

```
Tests of scale effects

formula: zrating ~ construction + dialect
         Df logLik AIC   LRT Pr(>Chi)
<none>      -1363.9 3029.7
construction 1 -1363.8 3031.7 0.00611 0.9377
dialect   2 -1363.0 3032.0 1.67300 0.4332
```

However, the test for the proportional odds is inconclusive as it does not produce any *p*-values:

```
nominal_test(modelfive.clm)

Tests of nominal effects

formula: zrating ~ construction + dialect
         Df logLik AIC LRT Pr(>Chi)
<none>      −1363.9 3029.7
construction
dialect
```

As this is only an example we'll just pretend that the proportional odds assumption is met and proceed. Let's fit our model again, now with the *z*-scored ratings::

```
modelfive = clmm(zrating ~ construction + dialect + (1 +
   construction|participant) + (1 + construction|item),
   data=data)
```

Now, the output tells us that it is only the construction that is significant.

```
Cumulative Link Mixed Model fitted with the Laplace
   approximation

formula: zrating ~ construction + dialect + (1 + construction |
   participant) +
   (1 + construction | item)
data:  data

 link threshold nobs logLik AIC niter max.grad cond.H
 logit flexible 288 −1357.23 3028.46 74648(223944) 1.56e+00
    1.3e+05

Random effects:
 Groups   Name      Variance Std.Dev. Corr
 participant (Intercept) 0.32624 0.5712
         constructionb 1.33559 1.1557 −1.000
 item    (Intercept) 0.00000 0.0000
         constructionb 0.05974 0.2444 NaN
Number of groups: participant 29, item 10
```

```
Coefficients:
           Estimate Std. Error z value Pr(>|z|)
constructionb -1.97257 0.37386 -5.276 1.32e-07 ***
dialectdialectb -0.04135 0.26505 -0.156 0.876
dialectdialectc -0.03153 0.27866 -0.113 0.910
---
Signif. codes: 0 ‘’*** 0.001 ‘’** 0.01 ‘’* 0.05 ‘’. 0.1 ‘’1
```

If we would deal with a linear mixed effects model we would now conduct an ANOVA. As we are using an ordinal model we will conduct an analysis of deviance:

```
#Install and load required packages
install.packages("RVAideMemoire")
library(RVAideMemoire)
install.packages("car")
library(car)

Anova(modelfive, type="II")
```

The Anova function (with a capital A) will tell you:

```
Analysis of Deviance Table (Type II tests)

Response: zrating
         LR Chisq Df Pr(>Chisq)
construction 38.082 1 6.782e-10 ***
dialect     0.817 2   0.6648
---
Signif. codes: 0 ‘’*** 0.001 ‘’** 0.01 ‘’* 0.05 ‘’. 0.1 ‘’1
```

Thus, the main effect of construction is significant while the main effect of dialect is not. As we will see this is in line with further tests. We can now compare our full model (i. e., modelfive) with a reduced model only taking the construction type into account. Let’s call this model ‘modelfivelight’:

```
modelfivelight = clmm(zrating ~ construction + (1 +
   construction|participant) + (1 + construction|item),
   data=data)
```

Using the rcompanion package we can now compare our two models:

```
install.packages("rcompanion")
library(rcompanion)

nagelkerke(fit = modelfive, null = modelfivelight)
```

We could do the same thing with lmer, but we would not use the nagelkerke function, but the anova function (see Winter 2013). The output of this operation tells us, again, that construction type makes a difference, but that dialect seems to play no role (with $p = 0.66476$):

```
Model: "clmm, zrating ~ construction + dialect + (1 +
   construction | participant) + (1 + construction | item),
   data"
Null: "clmm, zrating ~ construction + (1 + construction |
   participant) + (1 + construction | item), data"

$Pseudo.R.squared.for.model.vs.null
                    Pseudo.R.squared
McFadden                0.000300769
Cox and Snell (ML)      0.002831650
Nagelkerke (Cragg and Uhler) 0.002831880

$Likelihood.ratio.test
 Df.diff LogLik.diff Chisq p.value
    -2  -0.40834 0.81667 0.66476

$Number.of.observations

Model: 288
Null: 288
```

What is more interesting in our case is the opposite. We can compare our full model with a model that only takes the dialect into account:

```
modelonlydialect = clmm(rating ~ dialect + (1 +
   construction|participant) + (1 + construction|item),
   data=data)
nagelkerke(fit = modelonlydialect, null = modelfive)
```

The output produced is something we could report:

```
Model: "clmm, rating ~ dialect + (1 + construction |
   participant) + (1 + construction | item), data"
Null: "clmm, zrating ~ construction + dialect + (1 +
   construction | participant) + (1 + construction | item),
   data"

$Pseudo.R.squared.for.model.vs.null
                             Pseudo.R.squared
McFadden                             0.648485
Cox and Snell (ML)                   0.997784
Nagelkerke (Cragg and Uhler) 0.997865

$Likelihood.ratio.test
 Df.diff LogLik.diff Chisq p.value
     143   −880.14 1760.3 2.4102e−276
```

The pseudo R-square is a relative measure. While the R-square tells you how well the model explains the data, the pseudo R-square describes the fit of a model relative to another model. Similar to the R-square, pseudo R-square take values between 0 and 1 with higher values indicating a better model fit. In this case we see that the model taking the construction type into account explains the data very well in that it accounts for 64 % of the variation in the data (McFadden) compared to the model not taking construction into account. That's it!

References

Baayen, H. (2012). Analyzing linguistic data. A practical introduction to statistics using R. Cambridge: Cambridge University Press.

Bates, D. M., Maechler, M. & Bolker, B. (2019). Linear Mixed-Effects Models using 'Eigen' and S4. R package version 1.1-20.

Christensen, R. H. B. (2018). ordinal - Regression Models for Ordinal Data. R package version 2018.8-25. http://www.cran.r-project.org/package=ordinal/.

Clark, H. H. (1973). The Language-as-Fixed-Effect Fallacy: A Critique of Language Statistics in Psychological Research. Journal of Verbal Learning and Verbal Behaviour, 12, 335–359.

Cunnings, I. (2012). An overview of mixed-effects statistical models for second language researchers. Second Language Research, 28(3), 369–382.

Gibson, E. Piantadosi, S. & Fedorenko, K. (2011). Using Mechanical Turk to obtain and analyze English acceptability judgments. Language and Linguistics Compass, 5(8), 509–524.

Kizach, J. (2014). Analyzing Likert-scale data with mixed-effects linear models: a simulation study. Poster session presented at Linguistic Evidence 2014, Tübingen, Germany.

Kuznetsova, A., Brockhoff, P. B. & Christensen, R. H. B. (2019). Package 'lmerTest'. R package version 3.1-0. https://cran.r-project.org/web/packages/lmerTest/index.html.

Lüdecke, D. (2018). Package 'sjPlot'. Data visualization for statistics social science. R package version 2.6.2. https://cran.r-project.org/web/packages/sjPlot/sjPlot.pdf.

Makowski, D. (2019). Package 'psycho'. Efficient and publishing-oriented workflow for psychological science. R package version 0.4.0. https://cran.r-project.org/web/packages/psycho/psycho.pdf.

Mangiafico, S. S. (2016). Summary and Analysis of Extension Program Evaluation in R, version 1.15.0. www.rcompanion.org/handbook/.

Norman, G. (2010). Likert scales, levels of measurement and the "laws" of statistics. Advances in Health Sciences Education, 15, 625–632.

R Core Team (2015). R: A Language and Environment for Statistical Computing. Vienna, Austria: R Foundation for Statistical Computing. https://www.R-project.org.

Winter, B. (2013). Linear models and linear mixed effects models in R with linguistic applications. Online: http://arxiv.org/abs/1308.5499.

Zuur, A. Ienso, E. N., Walker, N. J., Saveliev, A. A. & Smith G. M. (2009). Mixed Effect Models and Extensions in Ecology with R. New York: Springer.