Multilevel analysis in R

Emmeke Aarts and Beth Grandfield

Packages used in this script:

- lme4
- lattice
- lmerTest
- ggplot2

The following steps will estimate the models with the programming language R. In R, there are two packages available to do multilevel analysis: `lme4` and `nlme`. We will work with `lme4`. There are several differences, a selection:

- the package `lme4` is likely to be faster and more memory-efficient compared to nlme (as `lme4` uses modern, efficient linear algebra methods as implemented in the Eigen package, and uses reference classes to avoid undue copying of large objects).
- `lme4` includes generalized linear mixed model (GLMM) capabilities, via the `glmer` function. This allows to fit models with a Poisson or Binomial distribution.
- `nlme` provides p-values for the fixed effects, while `lme4` does not (because of theoretical objections against the p-value). As such, you will have to test the significance of subsequent models to get this information, which is easily done with the function `anova()`.

Note that if you switch between the packages, the manner in which you specify the model (that is, operationalize the random effects) differs between the packages! Have a look at the help file of the package `nlme` for more information.

Exercise 1

Form groups of 3 and complete the exercise during lab. Each group will hand in (upload to blackboard) the completed excel file by the end of lab on Monday.

In this exercise, you will be introduced to R for performing multilevel analysis (and if you want SPSS).

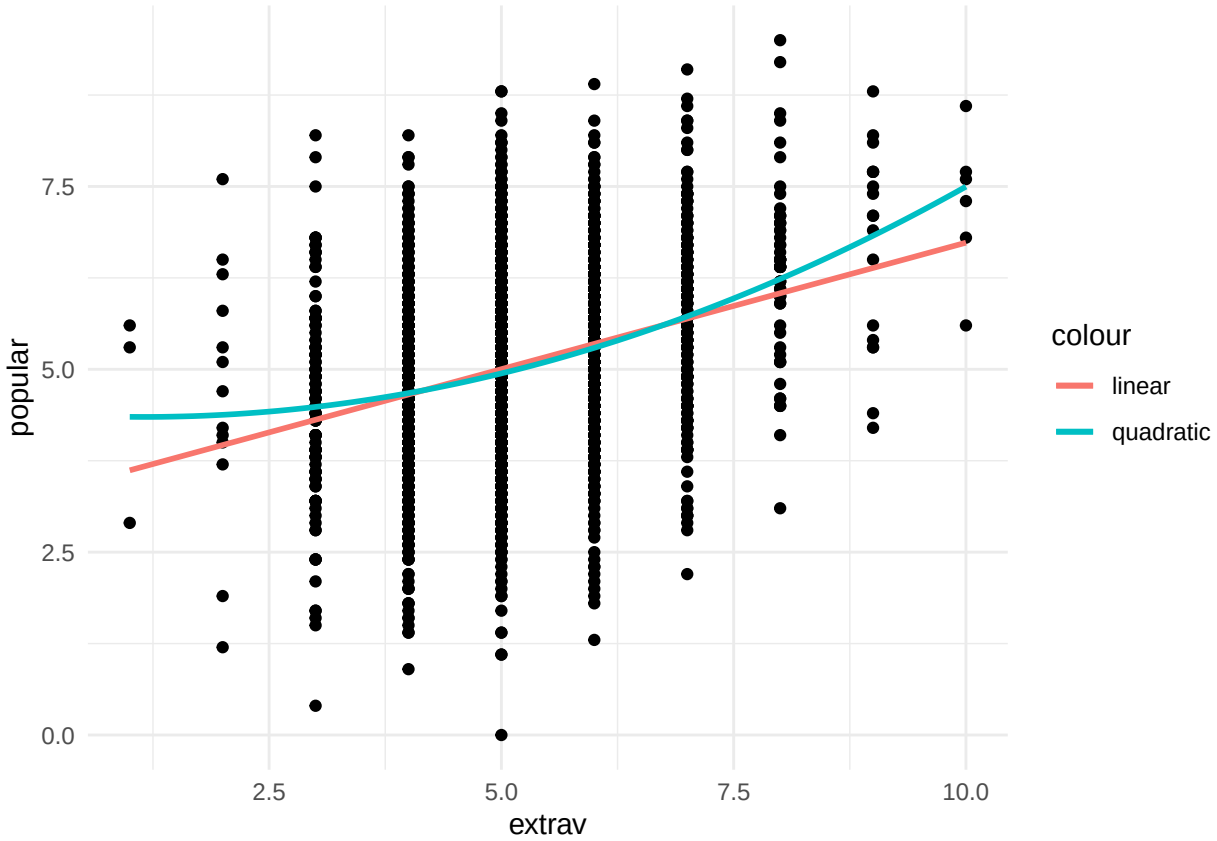
The file popular.csv holds the data for the popularity example. The file popular.sav can be opened in SPSS if interested but we will be reading the .csv data directly into R.

List of all the variables:

- pupil: pupil identification variable, not needed in the analysis
- class: class identification variable, the linking variable to define the 2-level structure
- student-level: extrav, gender: independent variables
- class-level: texp: independent variable
- popular: continuous outcome variable (student level)

Start with reading in the data, viewing descriptive statistics and scatterplots.

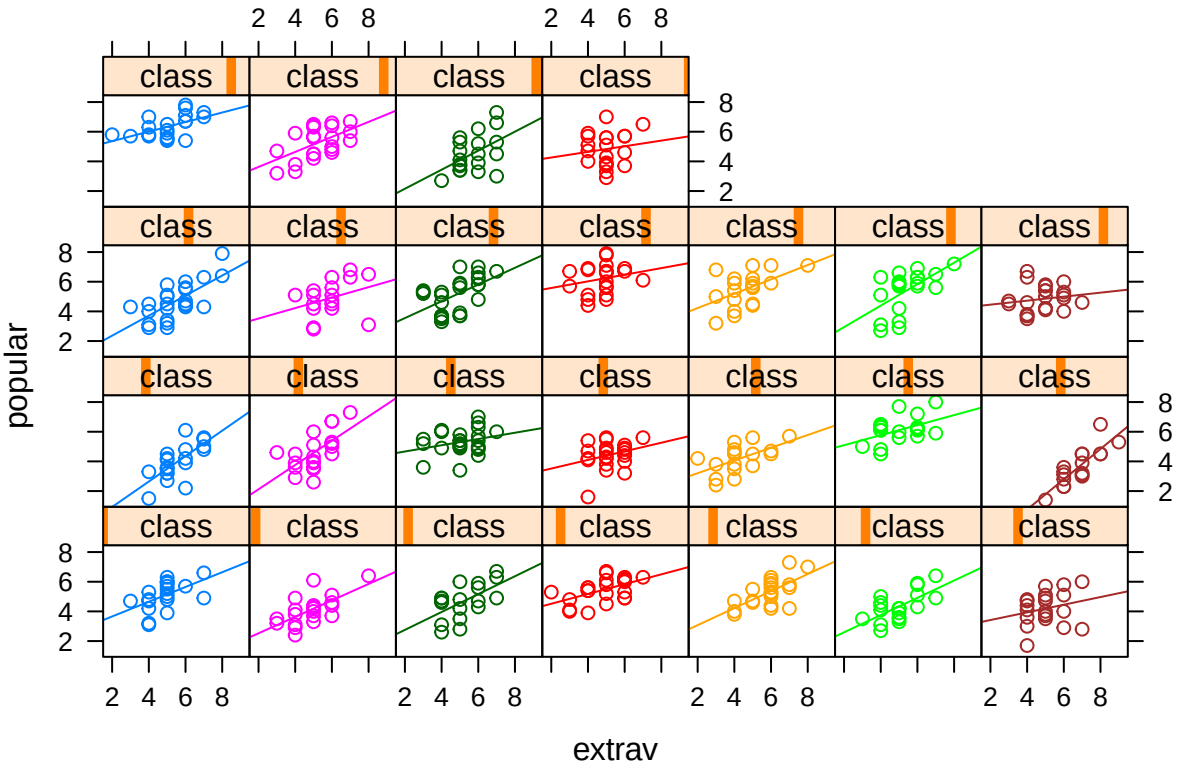
```r
popular <- read.csv(file = "popular.csv")
head(popular)

# We can view some descriptive statistics with summary ()
summary(popular)
# we can also create scatterplots to view linear relations and outliers using ggplot2

library(ggplot2)
ggplot(popular,
aes(x = extrav, y = popular)) +
  geom_point() +
  geom_smooth(method = "lm",
              aes(color = "linear"),
              se = FALSE) +
  geom_smooth(method = "lm",
              formula = y ~ x + I(x^2),
              aes(color = "quadratic"),
              se = FALSE) +
  theme_minimal()
```



Note: Do not use abline(), this will only take the first two values of your lm object (so the intercept and slope), and will ignore the quadratic trend and not warn you about this. If you don't want to use ggplot, use something like plot(x, y) with lines(fitted(...)) instead.

Using the package `lattice`, we can also make some cool plots of the observed data, separating by class. For example, the relation between *popularity* and *extraversion*. Note that this plot will become even more insightful if you do not plot it for all classes simultaneously, but by batches of, for example, 25 classes.

```
#install.packages("lattice")
library(lattice)
# xyplot(popular~extrav | class, groups=class, data=popular, type=c('p','r'), auto.key=F)
# for the first 25 classes:
xyplot(popular~extrav | class, groups=class, data=popular[popular$class %in% 1:25,],
       type=c('p','r'), auto.key=F)
```



What do you notice from the plot?

Now we can loading the lme4 package to have a look at the function used to fit multilevel models and start the analysis!

Make sure to fill out the estimated values for the different models in the excel sheet.

Specify the intercept only model

Start with building the basic model, one without and one with a random intercept (note that `lmer()` requires you to specify random effects. Therefore, a model without random effects has to be fitted using the normal `lm()` function).

Compare the results with the models on the slides of lecture 1a. When they are the same, you can go on to the next step. Don't forget to fill in the Excel sheet (also for the next steps).

```
library(lme4)
# ?lmer

model_0 <- lm(popular~1, data = popular)
```

```
model_1 <- lmer(popular~1 + (1|class) , REML = FALSE, data = popular)
summary(model_0)
summary(model_1)
```

A couple of remarks here:

- Note that the random part is given in the equation by the part in between brackets: `(1|class)`. The 1 means that we are only making the intercept random. Hence, we are telling the program that the intercept is conditional (i.e., given) on which class a students belongs to.
- With the specification of `REML = FALSE` we are telling R to use full maximum likelihood (ML) th perform the estimation, instead of restricted maximum likelihood (which is the default).
- Note that for the variance components (i.e., the random effects), the output gives you the value for the variance component (here the intercept variance and residual variance), and the standard deviation. This is NOT the standard error of these variance components, it is simply the square root of the variance components!

Compare the results in the slides. What is your conclusion based on the output? Note that in R, we can easily test if the random intercept is significant by comparing `model_0a` to `model_0b` (note that we can only do this given that we use full maximum likelihood estimation):

```
anova(model_1, model_0)
```

Another option is to use the `lmerTest` package Try it and compare the result to what was provided with the anova function.

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

ranova(model_1)
```

Set up the model with the level 1 explanatory variables

Now let's add the level 1 fixed predictors to the model.

- Add 'extrav' and 'gender' as predictors to the model
- Run the analysis
- Request and review the output

```
model_2 <- lmer(popular~1 + gender + extrav + (1|class) , REML = FALSE, data = popular)
summary(model_2)

# to test if the addition of extrav and gender significantly improves the model, use
# anova(model_2, model_1)
```

Compare the results to the slides. What is your conclusion based on the output?

Set up the model with fixed predictors at level 1 and level 2

Next, we add the predictors at level 2:

- Add 'teacher experience' as a predictor to the model
- Run the analysis
- Request and review the output

```
model_3 <- lmer(popular~1 + gender + extrav + texp + (1|class) , REML = FALSE,
                data = popular)
summary(model_3)

# again, to evaluate the significance of adding texp to the model:
# anova(model_3, model_2)
```

Compare the results to the slides. What is your conclusion based on the output?

Set up the random coefficients model

Now we add the random slope for *extraversion* (not for *gender*, since we all ready know it is not significant). We do this by replacing the intercept (i.e., the 1) in the random part of the equation by `extrav`:

- Add the code allowing 'extrav' to have a random slope
- Run the analysis
- Request and review the output

```
model_4 <- lmer(popular~1 + gender + extrav + texp + (extrav|class), REML = FALSE,
                data = popular)
summary(model_4)

# to evaluate the significance of adding a random slope for extrav:
# anova(model_4, model_3)
```

Compare the results to the slides. What is your conclusion based on the output?

The function `lmer` by default fits a covariance between the intercept and slope variance, although it only gives you the standardized metric of the covariance: the correlation.

- If you want to obtain the value of the covariance (so unstandardized), you have to take a bit of a detour. First, you have to extract and save the variance correlation matrix using `vc <- VarCorr(model_3)`. Next, you have to extract the lower triangle of this matrix, using `as.data.frame(vc, order="lower.tri")`.
- If you want to fit a model with random intercept and slope where the intercept and slope are uncorrelated (not necessarily a good idea), you can use the formula `(1|class) + (0 + extraversion|class)`, the 0 in the second term suppresses the intercept.The last part also works if you would want to fit a model without a random intercept, but with a random slope.
- also nice, you can obtain the random effect for each class for both the intercept and slope of extraversion by using `ranef(model_3,condVar=TRUE)`.

Set up the model with cross-level interaction

To finish off, lets fit the model that includes the cross level interaction between *extraversion* and *teacher experience*, to see if we can explain any of the slope variance.

- Add the code allowing 'teacher experience' to have a cross-level interaction

- Run the analysis
- Request and review the output

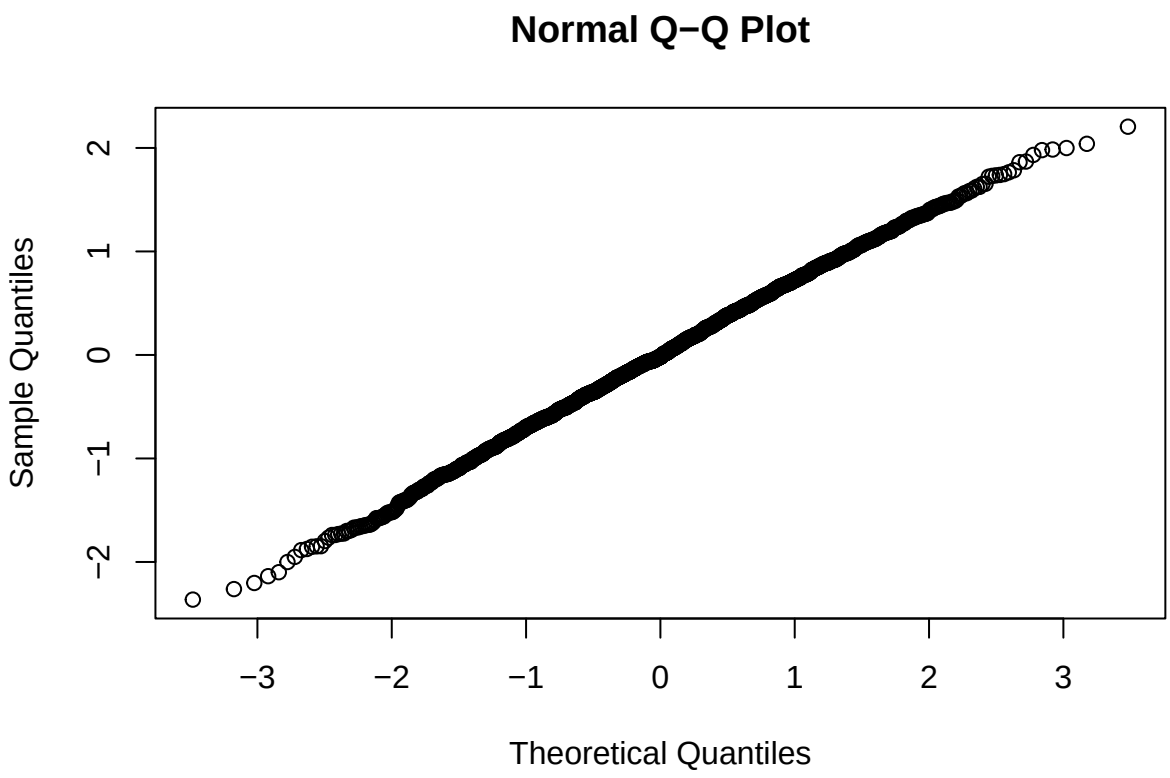
```
model_5 <- lmer(popular~1 + gender + extrav + texp + extrav*texp + (extrav|class),
                REML = FALSE, data = popular)
summary(model_5)

# to evaluate the significance of adding the cross level interaction:
# anova(model_5, model_4)
```

Compare the results to the slides. What is your conclusion based on the output?

Examine assumptions on residuals

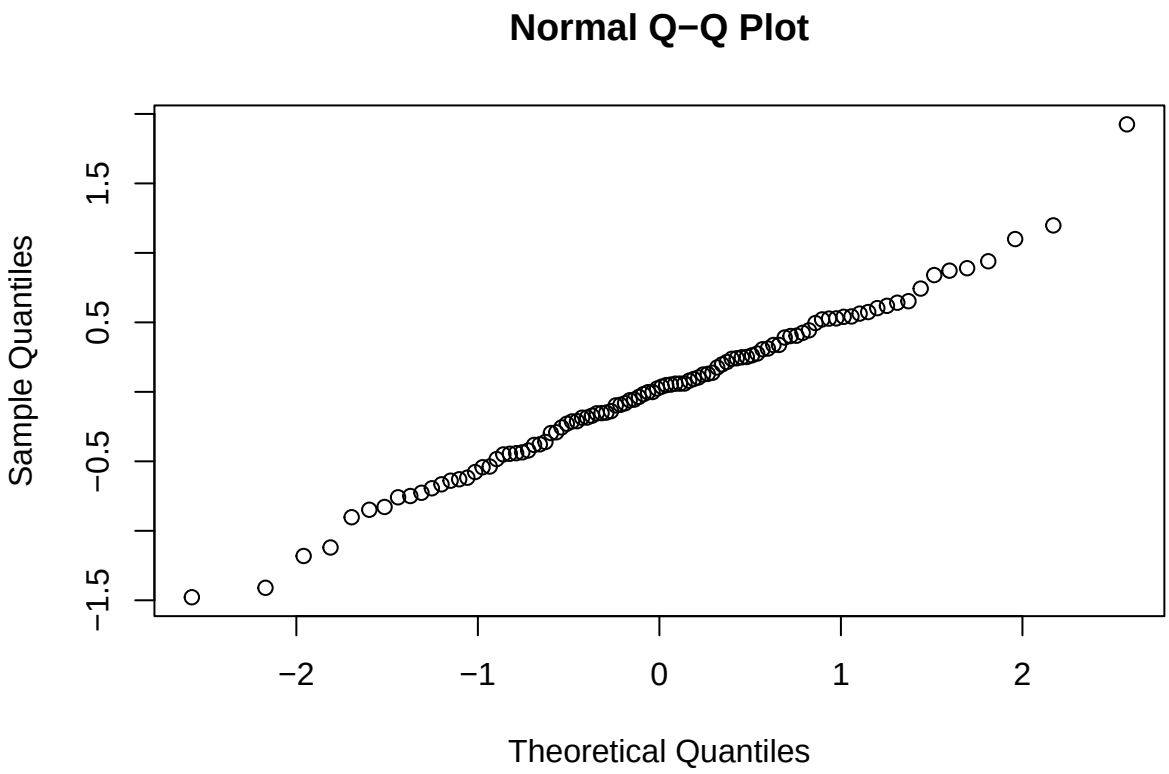
Use `qqnorm()` on the object containing the final ML model for level 1 residuals

```
qqnorm(residuals(model_4))
```

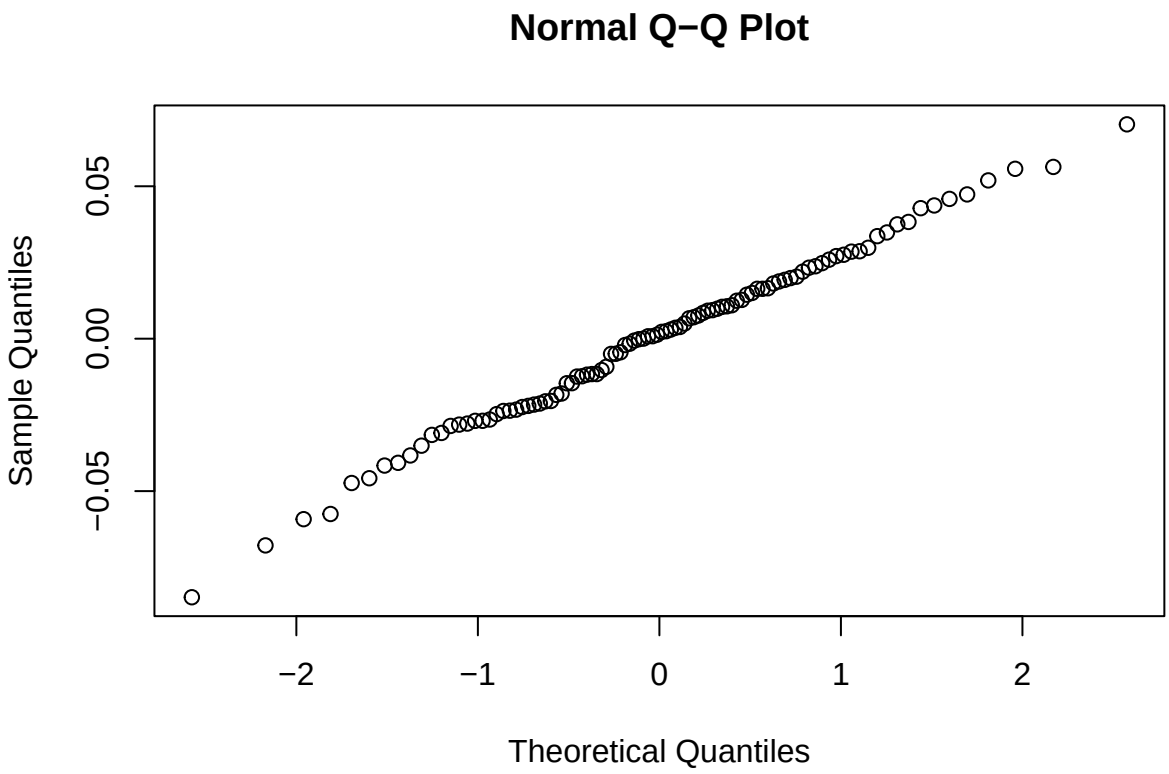


Extract the cluster specific deviances to the intercept and slope using `ranef()` and use `qqnorm()` on the resulting object.

```
# intercept variance
qqnorm(ranef(model_5)$class[,1])
```




```
# slope variance
qqnorm(ranef(model_5)$class[,2])
```



Finally, Hand in (upload) the completed Excel sheet to Blackboard for your group. Include the first name of all group members to the file name in CamelCase.

Additionally: The interested reader can find how to run a Multilevel analysis in SPSS. This part can be found in a separate pdf document on Blackboard