Notes for MATH 6627: Statistical Consulting Practicum

Kelly Ramsay

2024-01-23

Table of contents

Preface

- These notes are to be used for MATH 6627: Practicum in Statistical Consulting
- Some references used are:
- Lastly, see below:

```
print("Make sure you install R!")
```

```
[1] "Make sure you install R!"
```

1 Mixed Models

1.1 Clustered data

1.1.1 Review of multiple linear regression

Recall the multiple linear regression model:

For the model $Y|X = X\beta + \epsilon$, we have

- $Y \in \mathbb{R}^{n\times 1}$ is the response variable (a continuous random vector)
- $X \in \mathbb{R}^{n\times p}$ is the covariate matrix (Note that the first column is often 1_n – the column vector of ones)
- $X_i \in \mathbb{R}^{p\times 1}$ is the i^{th} observed explanatory variable ($i = 1, ..., n$) (not a random variable, in the sense that we condition on it)
- $\beta \in \mathbb{R}^{p\times 1}$ is the coefficient vector
- $\epsilon \in \mathbb{R}^n$ is the random error (continuous random variable)

The key assumptions of the (normal) MLR are that

- ϵ is multivariate normally distributed
- $E(\epsilon) = 0$
- $Cov(\epsilon) = \sigma^2 I_n$
- $Y|X = X\beta + \epsilon$

As such, it is critical that when applying MLR models, the observations are *independent*. However, there are many, many problems where the data contains dependent observations. If we have data that can be split into mutually independent clusters, then we call this *clustered data*.

1.1.2 An example

Consider the following simple example. Suppose a study wishes to prove/disprove the following: **Does Ozempic cause sustained weight loss over time?**

What type of data would we need to answer this question? We might start with the question: Can we collect data that would allow us to answer this question with a MLR model?

Could we:

- Take a sample of individuals on Ozempic and measure their weight? – **No**. How do we determine if their weight has decreased since starting it?
- Take a sample of individuals both on and not on Ozempic at a point in time, and compare their weights? – **No**. How can we rule out the fact that these are different populations?

It seems that this question could not be reliably answered using the above suggested methods. We would **need** to be able to follow individuals, starting when they begin Ozempic, recording their weights, and continue following them for a period of time. We might have data that looks like:

Month 1	Month 2	Month 3	Month 4	...
360	355	350	340	...
225	222	224	225	...
288	270	253	260	...

We could simply compare the weights in month 1 to the last month measured, and apply a one-sample t-test. What if the patients lose weight in the first 6 months and then gain it back? This would not be captured by such a model. We could run one t-test for each month, but of course then the type-1 error would be very large.

It is better to model the weights of patients on Ozempic over time. Inspired by the Normal MLR model, we might posit that patient is weight at time j is governed by the following equation:

$$Y_{ij} = \beta_0 + \beta_1 t_j + \epsilon_{ij}.$$

with $\epsilon_{ij} \sim \mathcal{N}(0, \sigma^2)$. Now, if we want to apply the Normal MLR model, we would need to assume $\epsilon_{ij} \perp \epsilon_{\ell k}$ when $ij \neq \ell k$. Is this reasonable? This would implies that $Y_{ij} \perp Y_{ij+1}$, i.e., a patients weight in month j is independent of their weight in month $j+1$. This is, of course unreasonable. However, it would be reasonable to assume that $Y_{ij} \perp Y_{\ell k}$ when $\ell \neq i$. This is an example of **clustered data**. Here the clusters are the patients.

- It is safe to assume that a patient's weight at a given time is unrelated to another patient's weight at any given time

- However, a patients weight a given time is related to their past and future weights; the within patient weights are dependent.

Therefore, a better assumption might be that the

$$Y_{ij} = \beta_{0i} + \beta_1 t_j + \epsilon_{ij}.$$

where β_{0i} are now **random variables,** where there exists one per patient. The coefficients β_{0i} contain the dependence of the weight within individuals, and allow us to model the random errors as independent: $Cov(\epsilon) = \sigma^2 I$. This is only one way to model the within-patient dependence between patients.

Example 1.1. Testing this theory…

Simplify the correlation between Y_{ik} and Y_{ij} in this model, and in the Normal MLR (the Normal MLR is the MLR, with the added assumption that the errors are normally distributed. Compare the results.

> Solution:
> Done in lecture.

The previous example is a longitudinal study, which is a sub-type of the more general clustered data. A longitudinal study is a research study in which subjects are followed over time. Typically this involves repeated measurements of the same variables. Longitudinal studies differ from cross-sectional studies and time series studies. Cross-sectional studies have no clusters, and time series follow one ore more variates over time. The random errors in a time series may be correlated.

Longitudinal studies are useful for:

- detecting changes in outcomes, both at the population and individual level,
- assessing **Longitudinal effects,** as compared to cohort effects/cross sectional effects,
- understanding different sources of variation, e.g., between- and within-subject variation.
- detecting **time effects**, both directly and as interactions with other relevant factors.

One example of longitudinal data, which we will see later is the TLC trial data:

```
##########################################################
TLC <- read.csv("data/TLC.csv",stringsAsFactors = T)
head(TLC)
```

```
  ID Treatment   W0   W1   W4   W6
1  1         P 30.8 26.9 25.8 23.8
2  2         A 26.5 14.8 19.5 21.0
3  3         A 25.8 23.0 19.1 23.2
4  4         P 24.7 24.5 22.0 22.5
5  5         A 20.4  2.8  3.2  9.4
6  6         A 20.4  5.4  4.5 11.9
```

As mentioned, longitudinal data is one example of clustered data. Clustered data refers to data that can be divided into clusters, such that data within a given cluster are correlated. For longitudinal observations, observations taken from the same subject at different time points are correlated because they belong to the same subject. In general, real world data have a complex dependence structure - can often be fit into this clustered framework.

Another example of clustered data is hierarchical data. These data have clusters within clusters. You could make a super dated inception joke here. For instance, if we wanted to assess how a new way of teaching p-values affects statistical literacy, we could sample universities, then professors, then classes. Here, assuming professors teach multiple sections, we could safely assume that the effect of this new teaching method may differ by university, by professor, and by class. In other words, observations within these groups would be correlated.

1.1.3 Additional challenges with clustered data

Often, clustered data are accompanied by other, additional challenges.

- Missing data or dropouts
- Measurement errors
- Censoring
- Outliers

The below examples are adapted from Wu (2019) .

Example 1.2. Blood pressure

A researcher wishes to evaluate a treatment for reducing high blood pressure. Blood pressures of each subject in the study are measured before and after the treatment. The researcher is also interested in how blood pressures of the subjects change over time after the treatment, so blood pressure is also measured after treatment once a month for 5 months. What are some potential challenges associated with this data? One answer: The data may contain missing values, e.g., drop out. Blood pressure has measurement error – often repeatedly measured.

Example 1.3. Mental distress

Investigate changes in subjects' mental distress over time in a treatment group and a control group. Mental distress in 239 subjects were measured at baseline, 4, 12, 24, and 60 months, based on their answers to questionnaires. Subjects randomly assigned into two groups: a treatment and a control group. The Global Severity Index (GSI) is used to measure subjects' distress levels. Other variables such as education, annual income, depression, anxiety, etc. were collected

Variable	Mean	Standard Deviation
GSI score of subjects (0 – 10)	1.13	0.72
Education of subjects (in years)	13.70	2.36
Income of subjects (in $10,000)	4.68	1.90
Depression score of subjects (0 – 10)	1.55	0.99
Anxiety score of subjects (0 – 10)	1.23	0.92

Table 1.2 *Missing data rat*

Variable	baseline	3 months
GSI	0.04	0.14
Depression	0.03	0.13
Anxiety	0.03	0.13

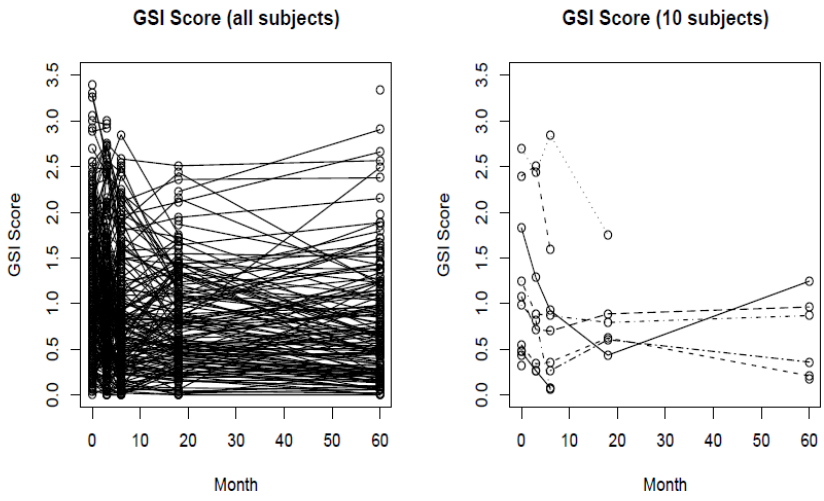


Figure 1.1 *GSI scores over time. Left figure: GSI scores for all subjects. Right figure: GSI scores for 10 randomly selected subjects. The open dots are the observed values.*

What are some potential issues for analysis?

Substantial individual variability, Missing data, Outliers, Measurement error? GSI influenced by short-term emotional state

Example 1.4. AIDS Study

The following is an AIDS study designed to evaluate an anti-HIV treatment. 53 HIV infected patients were treated with an antiviral regimen. Viral load (RNA) was repeatedly quantified on days 0, 2, 7, 10, 14, 21, and 28, and weeks 8, 12, 24, and 48 after initiation of the treatment. Immunologic markers known as CD4 and CD8 cell counts were also measured along with viral load, as well as some other variables. Viral load has a lower detection limit of 100, i.e., viral loads below 100 are not quantifiable.

Table 1.3 *Summary statistics for viral load (RNA), CD4, and CD8 at five selected measurement times*

Variable	Day 2		Day 7		Day 14		Day 28		Day 56	
	Mean	S.D.[b]	Mean	S.D.	Mean	S.D.	Mean	S.D.	Mean	S.D.
RNA[a]	5.00	0.59	4.06	0.81	3.23	0.64	3.02	0.61	2.52	0.74
CD4	203	74	231	89	274	108	284	89	300	94
CD8	961	506	1026	643	1037	545	1086	627	1033	329

Note: a) RNA (viral load) is in $\log_{10}$ scale; b) S.D.: standard deviation

Figure 1.1: AD1

Table 1.4 *Missing data rates for some variables at baseline*

Covariate	Definition	Missing Rate
AGE	age of the patient	0
WEIGHT	weight of the patient	0
LU20	NK activity	37.5%
TNF	plasma tumor necrosis factor	16.7%
APOP	% of cells that are apoptotic	0
CH50	complement CH50	18.75%
BIGG	gp120-binding IgG levels	22.92%
BIGC3	C3 binding to HIV-infected cells	27.08%

Figure 1.2: AD2

Other information about this data is given by:

- "HIV viral dynamic models model viral load trajectories during an anti-HIV treatment"
- "In an HIV viral dynamic model, the relationship between viral load and viral dynamic parameters is often nonlinear, and the viral dynamic parameters often vary substantially across patients."
- Thus, nonlinear mixed effect models

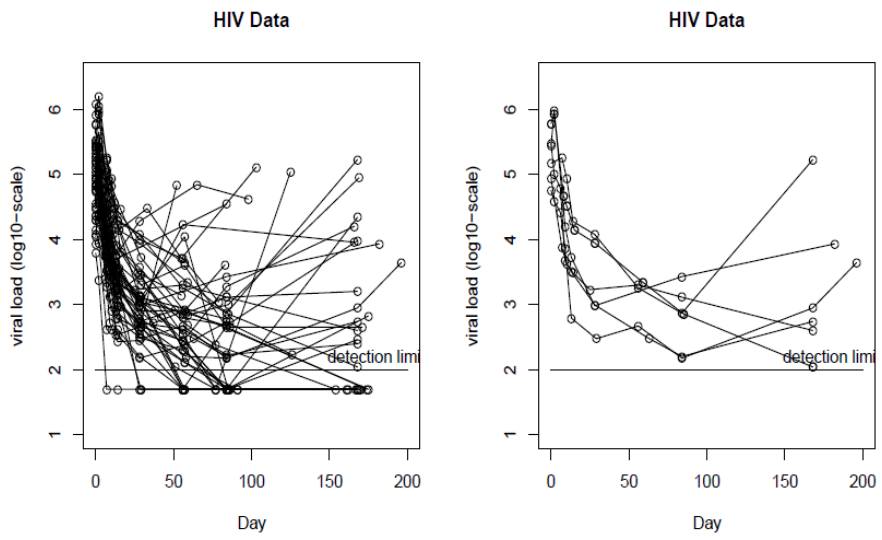


Figure 1.3 *Viral loads trajectories (in log_{10} scale). The open circles are observed values. The viral load detection limit in this study is $log_{10}(100) = 2$. Viral loads below the detection limit are substituted by half the limit.*

Figure 1.3: AD3

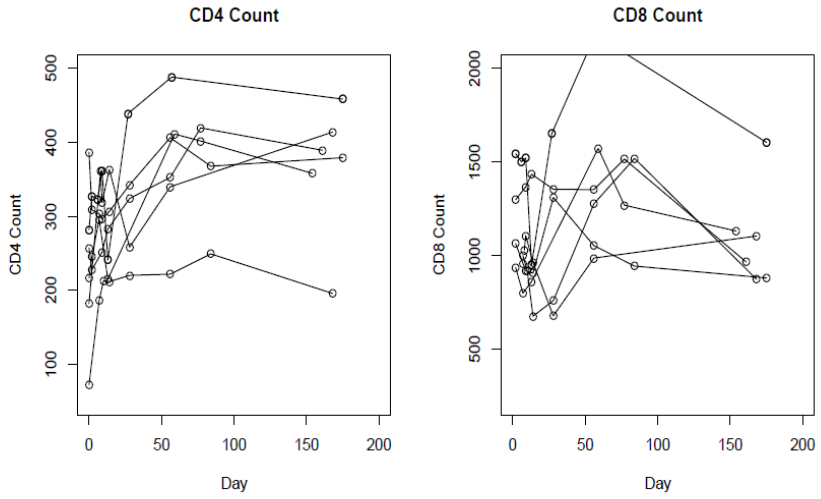


Figure 1.4 *CD4 Counts (left) and CD8 counts (right) of six randomly selected patients.*

Figure 1.4: AD4

- AIDS researchers are also interested in the relationship between viral loads and CD4 counts over time
- "CD4 counts are known to be measured with substantial errors, and patients often drop out because of drug side effects or other problems."

What are some potential issues with this dataset?

- different measurement times across patients
- different numbers of within-individual measurements across patients

- large variation between patients
- large variation in the data within each patient
- some patients dropping out of the study
- some viral loads being censored (i.e., below the limit of detection)
- substantial measurement errors in the data
- complex long-term trajectories
- data being missing at measurement times

Multilevel models: Multilevel models/Hierarchical linear models/Nested data models: Statistical models for "nested clusters". They contain parameters that vary at more than one level. An example could be a model of student performance, where the data are collected from students from multiple classes from multiple schools.

Other model classes: Marginal models/GEE models – Mean and the correlation (covariance) structure are modeled separately. Does not require distributional assumptions (see Chapter 10 Wu (2019)) Transitional models – Within-individual correlation is modeled via Markov structures.

1.1.4 Homework questions

- Write down why clustered data are challenging to analyse?

1.2 Analysing clustered data with mixed models

A **mixed model** is a convenient modelling framework which can be used to model complex dependency structure within a data set. They are an extension of the familiar Normal MLR model, where the independent errors assumption is relaxed. In order to relax that assumption, a new concept is introduced: the **random effect**.

1.2.1 Random effects

In a mixed effect model, the effects of each of the covariates can be split into two categories: fixed and random effects. Deciding on what is a fixed effect and what is a random effect can be difficult, and there is no agreed upon definition: see this post by Andrew Gelman. I will provide some guidance below, but ultimately, there is no binary rule for determining whether one should model an effect as fixed or random. Your model should reflect the assumptions that are reasonable to make about the data at hand, and answer the research questions adequately.

When we model a fixed effect, we only model the average across the whole population. On the other hand, when we model a covariate as a random effect, we are modelling the average effect of that covariate as well as how that effect might vary between clusters. So, if we want a measure of how an effect varies between clusters, one would use a random effect. Random effects can also be used to implicitly introduce a dependency structure within clusters. For instance, in the Ozempic example in Section Section 1.1.2, we saw that the random effect introduced a correlation between the observations coming from the same individual. Furthermore, modelling the intercept as a random effect allows us to estimate the average "regression line" for the population taking Ozempic, as well as how that "regression line" varies from person to person. (This will be made precise later if you missed the lecture.)

Example 1.5. A first example:

A clinical trial is set up to compare a new drug with a standard drug. The drug effect is of interest in the trial. We propose a Normal MLR (or fixed-effects) model with "drug" and "gender" as the two-fixed effects factors. Each has finite levels: "drug" – "new drug" and "standard drug"; "gender" – "female","male", "non-binary". Is there a cluster variable? Should we introduce any random effects?

Example 1.6. Clinical trial:

In a clinical trial, several hospitals in Canada are sampled. In each of the selected hospitals, a new treatment is compared with an existing treatment. Is there a cluster variable? Should we introduce any random effects? What definition(s) do any of these random effects fit?

One way to decide on whether an effect is a fixed or random effect is to ask if the observations for that covariate contain the complete set of levels we are interested in for a given covariate. In Example 1.6, the data can be clustered by hospital. The treatment is a fixed effect, as we have observed the complete set of levels we are interested in for it. On the other hand, we would like our analysis to generalize beyond the selected hospitals, and so we would like to assess how the regression line varies between hospitals. This rule does not work well for continuous covariates, such as height, which may not require a random effect, but also, we cannot observe all levels of this factor.

Example 1.7. Antibiotics:

The efficacy an antibiotic maintains after it has been stored for two years is of scientific interest. Eight batches of the drug are selected at random from a population of available batches. From each batch, we take a sample of size two. The goal of the analysis: Estimate the overall mean concentration. Does the random batch have a significant effect on the variability of the responses?

batch	r1	r2
1	40.00	42.00
2	33.00	34.00
3	46.00	47.00
4	55.00	52.00
5	63.00	59.00
6	35.00	38.00
7	56.00	56.00
8	34.00	29.00

Since the batches are drawn randomly from a larger population, we could model the batch effect as a random effect. Obviously, the within batch observations will be correlated. The data are clustered by batch. Suppose instead that only eight batches exist in the whole world, and we are interested in knowing whether the batch number has an effect on the response. Then, the batch becomes a fixed effect.

Example 1.8. A Tale of Two Thieves (Cabrera and McDougall (2002)):

Recall that the client wanted us to assess the level of active ingredient in their tablets, as well as assess the variability in that can be attributed to the sampling technique.

	METHOD	LOCATION	REPLICATE	ASSAY
1	Intm	1	1	34.38
2	Intm	1	2	34.87
3	Intm	1	3	35.71
4	Intm	2	1	35.31
5	Intm	2	2	37.59
6	Intm	2	3	38.02

Number	methdb	drum	tablet	yb
1	Tablet	1	1	35.77
2	Tablet	1	2	39.44

Number	methdb	drum	tablet	yb
3	Tablet	1	3	36.43
4	Tablet	5	1	35.71
5	Tablet	5	2	37.08
6	Tablet	5	3	36.54

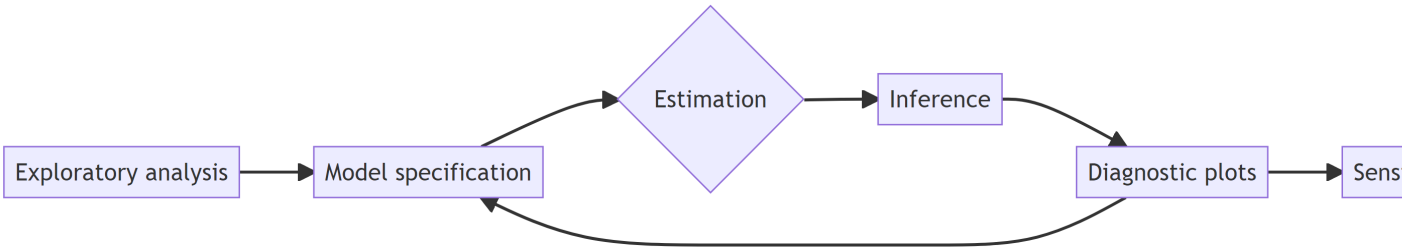
What are the clusters? What might be a random effect?

1.2.2 Homework questions

- Give an example of a study where a mixed effect model would apply, which effects are random and which are fixed?
- Describe the potential differences between a random and fixed effect. Compare and contrast the different definitions of random effects. Which one do you prefer and why?
- How would you determine which definition for random effects a client is using?

1.2.3 Regression and general data modelling review

By the end of this section, we will have covered all steps involved in analyzing data using mixed models:



Example 1.9. How do you do each of these steps in a simple linear regression model?

Let's review. Suppose Y_i are continuous and we want to model $E[Y_i|X_i]$. A linear regression model takes

$$E[Y_i|X_i] = X_i'\beta.$$

We take $\hat{\beta} = (X'X)^{-1}X'Y$, and call these ordinary least squares (OLS) estimators. If $Y_i|X_i \sim N(X_i'\beta, \sigma^2)$, then the OLS estimators are the maximum likelihood estimators.

If we take $Y_i = X_i'\beta + \epsilon_i$, where X_i is non-normal, then the OLS estimators minimize the MSE of any predictor:

$$\phi(\beta) = \frac{1}{n}\sum_{i=1}^{n} ||\beta - E[Y_i|X_i]||^2$$

is minimized at $\hat{\beta}$.

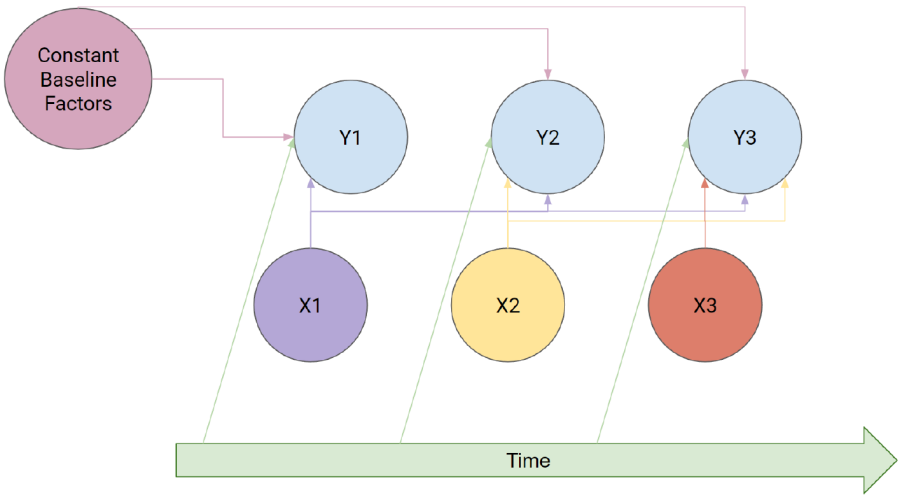
In this case, as discussed in Section Section 1.1.1, we assume that:

1. The conditional mean is linear (in parameters).
2. All values of Y_i have constant variance, denoted σ^2 (conditionally).
3. The Y_i are independent.

Then, one can show that $\hat{\beta}$ is asymptotically normal with $Var(\hat{\beta}) = \sigma^2(X'X)^{-1}$. We then use this fact to construct confidence intervals, hypothesis tests etc. We later analyze the residuals to diagnose any problems with the fit. Overall, linear regression allows us to estimate a functional form for the conditional mean of a continuous outcome. The ordinary least-squares estimators are valid MLE-type estimators when normality is assumed, and are least-squares estimators otherwise. The asymptotic analysis is valid in large samples, regardless of distributional assumptions. We now present equivalents of these results for the mixed model.

1.2.4 Defining the linear mixed model

Back to analyzing clustered data. Let's start with longitudinal data, which could be represented by the following diagram:



In (**goal?**), we see that there are time-varying and constant covariates, as well as the time-varying response. To formlize this, we can write:

- Y_{ij} response of subject i at jth time point for $i \in [n]$ and $j \in [J_i]$.

- X_{ijk} covariate k of subject i at jth time point for $k \in [K]$, $i \in [n]$ and $j \in [J_i]$.
- X_{ij} covariate vector for subject i at jth time point for $i \in [n]$ and $j \in [J_i]$.
- t_{ij} actual time for subject i at time point j for $i \in [n]$ and $j \in [J_i]$.

We can split the covariate matrix into time-varying covariates Z and constant covariates W. We have that $X = [W|Z]$. Each subject has J_i rows in X associated with it. Let X_i be the $J_i \times p$ submatrix corresponding to the covariates for subject i and let Z_i be the $J_i \times m$ submatrix corresponding to the time-varying covariates for subject i.

Now, the goal is to fit a model for $E[Y_{ij}|X_{ij}, t_{ij}]$ with interpretable parameters.

To account for the correlation between subjects, we model the response as a vector Y_i, where

$$Y_i = X_i\alpha + Z_i\beta_i + \epsilon_i,$$

where

- α: Population level effects – constant between subjects ($p \times 1$)
- β_i: Patient-level heterogeneity – varies between subjects ($m \times 1$)
- ϵ_{ij}: Individual measurement variation – varies between measurements (scalars)

Note that in the mixed model, we assume: α is a fixed vector, β_i is randomly drawn for each individual, ϵ_{ij} are also randomly drawn.

We can then assume that $\beta_i \sim N(0, \Sigma_\beta)$, $\epsilon_{ij} \sim N(0, \sigma^2)$ with $\epsilon_{ij} \perp \beta_i$.

Now, let's look at some properties of the model. Conditional on the random effects

$$E[Y_i|\beta_i, X_i] = X_i\alpha + Z_i\beta_i$$

and

$$Cov(Y_i|\beta_i, X_i) = Cov(\epsilon_i|\beta_i, X_i) = \sigma^2 I.$$

Derive these. If we consider the marginal distribution of Y_i we find:

$$E[Y_i|X_i] = X_i\alpha \qquad \text{and} \qquad Cov(Y_i|X_i) = Z_i\Sigma_\beta Z_i' + \sigma^2 I$$

Derive this. Combining these results we find that, under this assumed model,

$$Y_i|X_i \sim N(X_i\alpha, Z_i\Sigma_\beta Z_i' + \sigma^2 I).$$

1.2.5 Special cases of single layer models

We will now cover some special cases of the above model. The most basic mixed model is the random intercept model. Let $\tilde{W}_i$ be the covariate matrix of fixed effects with the intercept column removed. The resulting model is

$$Y_i = \alpha_1 \mathbb{1}_{J_i} + \tilde{W}_i \alpha + \beta_i \mathbb{1}_{J_i} + \epsilon_i,$$

where $\beta_i \sim N(0, \sigma_\beta^2)$ and $\epsilon_i \sim N(0, \sigma^2 I_{J_i})$. For $\ell \neq j$, it follows that

$$Corr(Y_{ij}, Y_{i\ell}) = \frac{\sigma_\beta^2}{\sigma_\beta^2 + \sigma^2}.$$

The variance is constant across time or clusters: $Cov(Y_i|X_i) = (\sigma_\beta^2 + \sigma^2)I$. **Derive these.** Observe that in this model, all subject level regression lines are parallel. This can be used when we suspect that the correlation is constant over time, and only the mean response is thought to vary between clusters.

If instead, we would like the regression lines to vary in general between subjects, we can introduce slopes are random effects. This is the **random intercept and slope model.** Here,

$$Y_i = \alpha_0 \mathbb{1}_{J_i} + \widetilde{W}_i \alpha + \beta_{0i} \mathbb{1}_{J_i} + \alpha_1 t_i + \beta_{1i} t_i + \epsilon_i.$$

What is Z_i here? The within-subject correlation will be time dependent in this model automatically, in this model, we assume that $\beta_i = (\beta_{0i}, \beta_{1i})' \sim N(0, \Sigma_\beta)$, where

$$\Sigma_\beta = \begin{pmatrix} \sigma_{\beta_0}^2 & \sigma_{\beta_0,\beta_1} \\ \sigma_{\beta_0,\beta_1} & \sigma_{\beta_1}^2 \end{pmatrix}.$$

Now, let's understand some of the features of this model. For any $i \in [n]$, we have that

$$Cov(Y_i|X_i) = Z_i \Sigma_\beta Z_i' + \sigma^2 I = (1_{J_i} \; t_i) \Sigma_\beta (1_{J_i} \; t_i)' + \sigma^2 I.$$

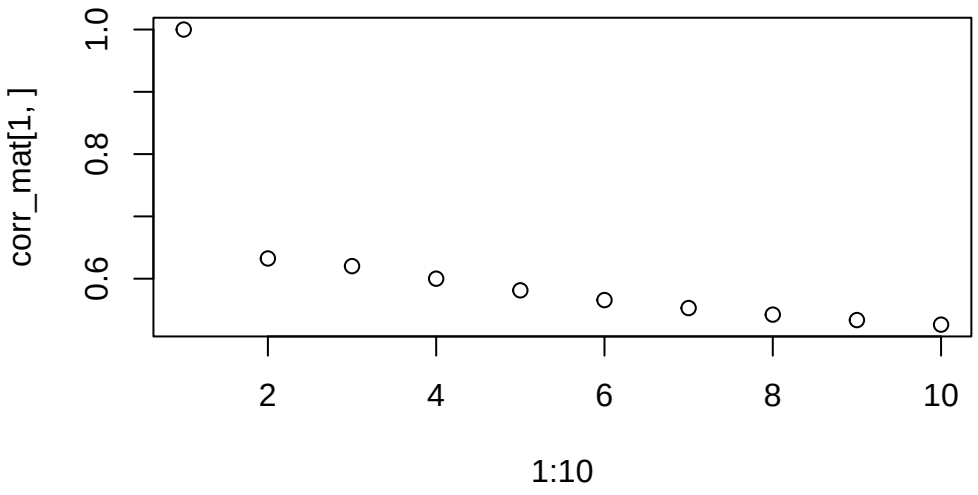
We see that the variance of the response is not constant across time. Further, for $\ell \neq j$:

$$Cov(Y_{ij}, Y_{i\ell}) = \sigma_{\beta_0}^2 + \sigma_{\beta_0,\beta_1}(t_{ij} + t_{i\ell}) + \sigma_{\beta_1}^2 t_{ij} t_{i\ell}.$$

In this model, the correlation between subject responses at different time points is varying.

Use the following code to explore how the correlation between time points changes as the distance between the time points grows.

```r
plot_cor=function(matt){

  Zi=cbind(rep(1,10),1:10)
  # err=s
  cov_mat=Zi%*%matt%*%t(Zi)+diag(rep(1,10))
  cov_mat
  vars=sqrt(diag(cov_mat))
  vars
  corr_mat=apply(cov_mat,1,"*",1/vars)
  corr_mat=apply(corr_mat,1,"*",1/vars)
  corr_mat
  plot(1:10,corr_mat[1,])
}


matt=matrix(c(1,0.4,0.4,1),nrow=2)
matt=matrix(c(1,0.4,0.4,1),nrow=2)
matt=matrix(c(1,0,0,2),nrow=2)
matt=matrix(c(1,0,0,6),nrow=2)
matt=matrix(c(1,-0.9,-0.9,4),nrow=2)
matt=matrix(c(1,0,0,0.5),nrow=2)
plot_cor(matt)
```



1.2.6 Multi-level models

So far we have discussed single level mixed models, which do not admit a nested structure. In this way, there is a nested structure of clusters. For example, if we wanted to assess how a new way of teaching p-values affects statistical literacy, we could sample universities, then professors, then classes. Here, assuming professors teach multiple sections, we could assume that effects differ by university, by professor, and by class. We could write a mixed effect model with 3 levels as

$$\begin{aligned} Y_{ijk} &= X_{ijk}\alpha + Z_{i,jk}\beta_i + Z_{ij,k}\beta_{ij} + Z_{ijk}\beta_{ijk} + \epsilon_{ijk} \\ &\quad i \in [n],\ j \in [J_i],\ k \in [m_{ij}], \\ &\quad \beta_i \sim N(0, \Sigma_1),\ \beta_{ij} \sim N(0, \Sigma_2),\ \beta_{ijk} \sim N(0, \Sigma_3),\ \epsilon_{ijk} \sim N(0, \sigma^2 I). \end{aligned}$$

Note that the "," tells us which columns of the covariate matrix we are concerned with: $Z_{i,jk}$ denotes the covariates nested in the highest level, $Z_{ij,k}$ the second highest and Z_{ijk} the innermost level. For instance, with respect to the above example, $Z_{i,jk}$ would be the university level covariates. (Note that some books count the number of levels by the number of sources of random variation, which is 1+ the level definition used here.) In addition, Pinheiro and Bates (2009) represents Σ_1 in form $\Sigma_1^{-1}/\sigma^2 = \Delta'\Delta$, where Δ is a non-unique relative precision factor. Specifications of the covariance matrices depend on the context. For more details, see Pinheiro and Bates (2009), Gelman and Hill (2006), Wu (2019).

1.2.7 Parameter estimation and inference

Generally, parameters are estimated with either maximum likelihood or restricted maximum likelihood (REML). Recall that this model is parametric, we have assumed normality. Thus, we can write down the likelihood. Let $V_i = Cov(Y_i) = Z_i\Sigma_\beta Z_i' + \sigma^2 I$. Then, the (familiar) asymptotic result for both the MLE and the REML estimates:

$$\hat{\alpha} \dot{\sim} N\left(\alpha\ ,\ \left[\sum_{i=1}^{n} X_i' V_i^{-1} X_i\right]^{-1}\right),$$

where $\dot{\sim}$ denotes asymptotically distributed as. For more details on how to derive the estimates, see Pinheiro and Bates (2009).

Restricted maximum likelihood is used because the MLE biases the variance estimates downward. In REML, we maximize

$$\mathcal{L}(\Sigma_\beta, \sigma^2|y) = \int \mathcal{L}(\alpha, \Sigma_\beta, \sigma^2|y) d\alpha.$$

This constitutes a uniform prior on α. Note that REML estimates are not invariant under reparameterizations of the fixed effects – changing the units of the covariates X_i units changes the estimates. As a result, LRT are not valid for testings significance of fixed effects – the restricted likelihoods cannot be compared to determine significance.

Testing – Fixed effects – MLE: When using the MLEs, we can use likelihood ratio tests to test significance of various parameters. The parameters for the covariances, denoted $\sigma_{\beta_k,\beta_\ell}$, will have some regularity concerns. Suppose we want to test whether a subset of the parameters are 0. Let $k =$# df in alt - # df in null. Recall that Wilks' Theorem gives $-2(\ell_1(\hat{\theta}) - \ell_0(\hat{\theta})) \sim \chi^2_k$, which can be used to conduct the test.

Testing – Random effects: However, this does not apply to random effects. The variance parameters lie on the boundary of the parameter space, and so Wilks' Theorem does not apply! Instead, we can simulate the distribution of the LRT statistic under the null and use the simulated distribution to obtain our critical value. If you are using a software where this is not feasible, then you can use $\frac{1}{2}\chi^2_{\#\ RE\ Null} + \frac{1}{2}\chi^2_{\#\ RE\ ALT}$.

Testing – Fixed effects – REML: REML estimates are not invariant under reparameterizations of the fixed effects – changing the units for the covariates X_i changes the REML estimates. As a result, LRT are not valid for testing the significance of fixed effects – the restricted likelihoods cannot be compared to determine significance. To test the fixed effects, we can use tests conditional on the variance parameters/RE parameters. In this case, we can perform either marginal t-tests – tests which consider adding the parameter to the model with all other covariates, or sequential F-tests – a test that adds the variables sequentially in the order they enter the model.

Confidence intervals – Fixed effects – REML: Both REML and MLE give asymptotic normality of both $\hat{\sigma}$ and fixed effect estimates. This can be used to obtain confidence intervals. For the parameters contained in Σ_β, constructing confidence intervals can be more difficult because Σ_β must be positive definite, which restricts the parameter space. In this case, we transform the parameters so that they are unconstrained, compute the confidence interval, and transform the interval back. See Section 2.4 in "Mixed Models in S and S-plus" for more details Pinheiro and Bates (2009).

1.2.8 Individual effects

One thing we may want to do is produce an estimate of β_i for observation i. One may notice that $E[\beta_i|Y_i] = \Sigma_\beta Z_i' V_i^{-1}(Y_i - X_i\alpha)$. Wait, we either know or have estimates of all of the values on the right-hand side. BLUP: $\hat{\beta}_i = \hat{\Sigma}_\beta Z_i' \hat{V}_i^{-1}(Y_i - X_i\hat{\alpha})$. Fitted values:

$$\hat{Y}_i = X_i\hat{\alpha} + Z_i\hat{\beta}_i.$$

Let's analyze V_i:

$$
\begin{aligned}
V_i &= Z_i \Sigma_\beta Z_i' + \sigma^2 I \\
\implies V_i V_i^{-1} &= Z_i \Sigma_\beta Z_i' V_i^{-1} + \sigma^2 I V_i^{-1} \\
\implies I &= Z_i \Sigma_\beta Z_i' V_i^{-1} + \sigma^2 V_i^{-1}.
\end{aligned}
$$

Now, the same logic gives that

$$
I = Z_i \hat{\Sigma}_\beta Z_i' V_i^{-1} + \hat{\sigma}^2 V_i^{-1}.
$$

We have

$$
\begin{aligned}
\hat{Y}_i &= X_i \hat{\alpha} + Z_i \hat{\beta}_i \\
&= X_i \hat{\alpha} + Z_i (\hat{\Sigma}_\beta Z_i' \hat{V}_i^{-1} (Y_i - X_i \hat{\alpha})) \\
&= (I - \hat{\Sigma}_\beta Z_i' \hat{V}_i^{-1}) X_i \hat{\alpha} + Z_i \hat{\Sigma}_\beta Z_i' \hat{V}_i^{-1} Y_i \\
&= \hat{\sigma}^2 \hat{V}_i^{-1} X_i \hat{\alpha} + (I - \hat{\sigma}^2 \hat{V}_i^{-1}) Y_i \\
&= \hat{\sigma}^2 \hat{V}_i^{-1} X_i \hat{\alpha} + Z_i \hat{\Sigma}_\beta Z_i' V_i^{-1} Y_i.
\end{aligned}
$$

Thus,

$$
\hat{Y}_i = \hat{\sigma}^2 \hat{V}_i^{-1} X_i \hat{\alpha} + Z_i \hat{\Sigma}_\beta Z_i' V_i^{-1} Y_i.
$$

We have that:

- $\hat{\sigma}^2 I$ within subject variation
- $Z_i \hat{\Sigma}_\beta Z_i'$ between subject variation
- Higher within subject variation – more weight to the population average

1.2.9 Exploratory analysis, checking assumptions and sensitivity testing

Exploratory analysis: Prior to setting up our model, we would ideally conduct exploratory analysis. Here, we look for outliers, inconsistencies in the data, and try to ascertain the relationships between the provided variables. This will help inform the model we will choose. It is also helpful to check that the model results approximately mirror what we saw in the EDA, as a sanity check. Some tools you can use in EDA are:

- Descriptive statistics
- xy plots – may have to subsample
- Box plots by cluster variable
- Cross-sectional plots

Diagnostic plots: These are used to check the fit of the model and check the assumptions. In a mixed model, we have independence and normality, and structure assumptions. This involves using graphics we are likely familiar with, such as qqplots. We may use:

- Checking independence between residuals across time – acf (may not be appropriate), variogram
- We have independence and normality, and structure assumptions
- Residuals vs. fitted values
- qqplot of residuals/random effects for normality
- Observed vs. fitted values

Sensitivity testing: In reality, there may be several models/frameworks with assumptions that could fit your data. For example, we may use a nonparametric method, a robust method, inclusion of different effects, use of different statistical tests. One thing you can do after performing a data analysis is to do the analysis under other models that may have been applied, and see if your results change. This helps support the conclusions made, and can reveal additional insights about your dataset. Be careful not to apply models whose assumptions are not reasonable for your data.

1.2.10 Homework questions

- How would you analyse the TLC data discussed last class? What are some statistical tests you might conduct?
- Write down the likelihood function under the random intercept model.

1.3 Case study: Batches of antibiotic and quality control

```
library(nlme)
library(lme4)
```

```
Loading required package: Matrix
```

```
Warning: package 'Matrix' was built under R version 4.2.3
```

```

Attaching package: 'lme4'
```

```
The following object is masked from 'package:nlme':

    lmList
```

1.3.1 Case information:

- After an antibiotic has been stored for two years, it is of scientific interest to know what concentration of active ingredient is.
- Eight batches of the drug are selected at random from a population of available batches.
- From each batch, we take a sample of size two.
- The goal of the analysis: Determine (to estimate) the overall mean concentration. A further question is whether or not the random batch has a significant effect on the variability of the responses.

From 8 batches of antibiotics, 2 samples are drawn.

Batch	1	2	3	4	5	6	7	8
Sample 1	40	33	46	55	63	35	56	34
Sample 2	42	34	47	52	59	38	56	29

```
batch=as.matrix(read.csv('data/batch.csv'))
```

```
Warning in read.table(file = file, header = header, sep = sep, quote = quote, :
incomplete final line found by readTableHeader on 'data/batch.csv'
```

```
batch=t(batch)
batch=unname(batch)
batch=data.frame(cbind(1:8,batch))
names(batch)=c("batch","r1","r2")
batch$r1=as.double(batch$r1)
batch$r2=as.double(batch$r2)
batch
```

```
  batch r1 r2
1     1 40 42
2     2 33 34
3     3 46 47
4     4 55 52
5     5 63 59
6     6 35 38
7     7 56 56
8     8 34 29
```

Overall mean: You can just take the sample mean here – the batches have an equal number of samples in each of the batches.

```
mean(c(batch$r1,batch$r2))
```

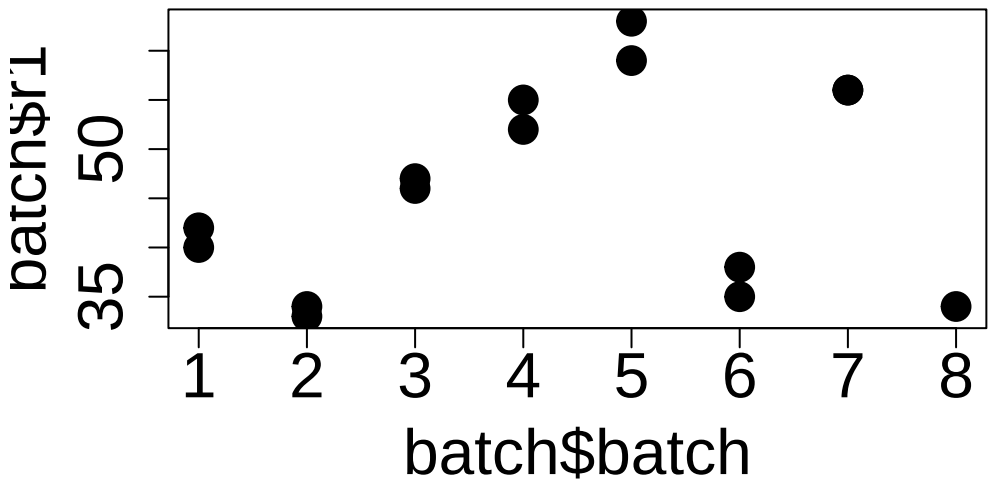
```
[1] 44.9375
```

```
summary(batch)
```

```
     batch            r1              r2       
 Min.   :1.00   Min.   :33.00   Min.   :29.00  
 1st Qu.:2.75   1st Qu.:34.75   1st Qu.:37.00  
 Median :4.50   Median :43.00   Median :44.50  
 Mean   :4.50   Mean   :45.25   Mean   :44.62  
 3rd Qu.:6.25   3rd Qu.:55.25   3rd Qu.:53.00  
 Max.   :8.00   Max.   :63.00   Max.   :59.00  
```

Graphically, the within batch variability is low relative to the between batch.

```
par(cex.lab=2,cex.axis=2,mfrow=c(1,1))
plot(batch$batch,batch$r1,pch=21,bg=1,cex=2)
points(batch$batch,batch$r2,pch=21,bg=1,cex=2)
```




```
cor(batch$r1,batch$r2)
```

```
[1] 0.9672002
```

Okay what about a confidence intervals for the mean? What about whether or not the random batch has a significant effect on the variability of the responses? We need a model for this.

It appears that the within batch mean is not constant.

$$Y_{ij} = \mu + \beta_i + \epsilon_{ij},$$

where for $i \in [8]$ and $j \in [2]$, we have

- Y_{ij}: concentration
- μ: overall mean
- β_i: effect of batch i, this effect is random!
- ϵ_{ij} random error

The assumptions are

- $\beta_i \sim N(0, \sigma_b^2)$ iid
- $\epsilon_{ij} \sim N(0, \sigma^2)$ iid
- β_i is independent of ϵ_{ij}

Under these assumptions $E(Y_{ij}) = \mu$ and $Var(Y_{ij}) = \sigma^2 + \sigma_b^2$.

In addition, we see that we capture the dependence structure: One can check that

- $Cov(Y_{i1}, Y_{i2}) = \sigma_b^2$
- $Cov(Y_{i1}, Y_{i'1}) = 0$

Recall we are interested in whether or not the random batch has a significant effect on the variability of the responses. This means we would like to estimate σ_b and test if it is negligible.

Let's estimate the parameters of this model. The relevant R package for (generalised) linear mixed models in R are `nlme`, `lme4` and `lmerTest`. Let's use REML to estimate our parameters.

```
#We need to reshape this data into long format!
batch_long=reshape(batch,
                   varying=c('r1','r2'),
                   timevar = 'replicate',
                   idvar = 'batch',
                   times=c(1,2),
                   direction = "long",sep = "")

head(batch_long)
```

```
  batch replicate  r
1.1     1         1 40
2.1     2         1 33
3.1     3         1 46
4.1     4         1 55
5.1     5         1 63
6.1     6         1 35
```

```
#using other package
# fit.lme<-lme4::lmer(r ~ 1 | batch, data=batch_long)
# summary(fit.lme)
```

```
rownames(batch_long) <- NULL

#defaults to REML
model_1=lme(
  fixed= r~1,
```

```
  random=  ~ 1 | batch, data=batch_long )
summary(model_1)
```

```
Linear mixed-effects model fit by REML
  Data: batch_long
       AIC      BIC    logLik
  101.0371 103.1613 -47.51855

Random effects:
 Formula: ~1 | batch
        (Intercept) Residual
StdDev:    10.95445 2.015565

Fixed effects:  r ~ 1
              Value Std.Error DF  t-value p-value
(Intercept) 44.9375  3.905623  8 11.50585       0

Standardized Within-Group Residuals:
        Min          Q1         Med          Q3         Max
-1.35131912 -0.56486600  0.09135863  0.51634786  1.12937443

Number of Observations: 16
Number of Groups: 8
```

Okay, between batch variance is huge. Let's test if its non-zero anyways. Recall that for REML estimates, the asymptotic distribution for the LRT is not the same as usual. In this case, under the null hypothesis, $Y_{ij} \sim N(\mu, \sigma^2)$.

Therefore, in order to construct a hypothesis test for σ^2_β, we can do the following:

1. Compute the LRT statistic from the sample, call it $\hat{T}$.
2. Simulate many, say n_{sim}, new samples of the same size from the model $Y_{ij} \sim N(\mu, \sigma^2)$.
3. For each of the n_{sim} samples, compute the LRT statistic: $\tilde{T}_1, \ldots, \tilde{T}_{n_{sim}}$.
4. The (empirical) p-value is then the proportion of $\tilde{T}_1, \ldots, \tilde{T}_{n_{sim}}$ larger than $\hat{T}$.

Thus,

```
#Step 1. Computing T-hat
fit_null<-lm(r ~ 1 , data=batch_long)

observed=lmtest::lrtest(fit_null,model_1)$Chisq[2]; observed
```

```
Warning in modelUpdate(objects[[i - 1]], objects[[i]]): original model was of
class "lm", updated model is of class "lme"
```

```
[1] 25.40237
```

```
  #Step 2.

  #Get the fixed effects
  fe=nlme::fixed.effects(model_1); fe
```

```
(Intercept)
    44.9375
```

```
  #Get the estimated variance of the RE
  sigma_batch_est= nlme::getVarCov(model_1); sigma_batch_est
```

```
Random effects variance covariance matrix
            (Intercept)
(Intercept)         120
  Standard Deviations: 10.954
```

```
  #Get the estimate of sigma
  sigma_est=model_1$sigma; sigma_est
```

```
[1] 2.015565
```

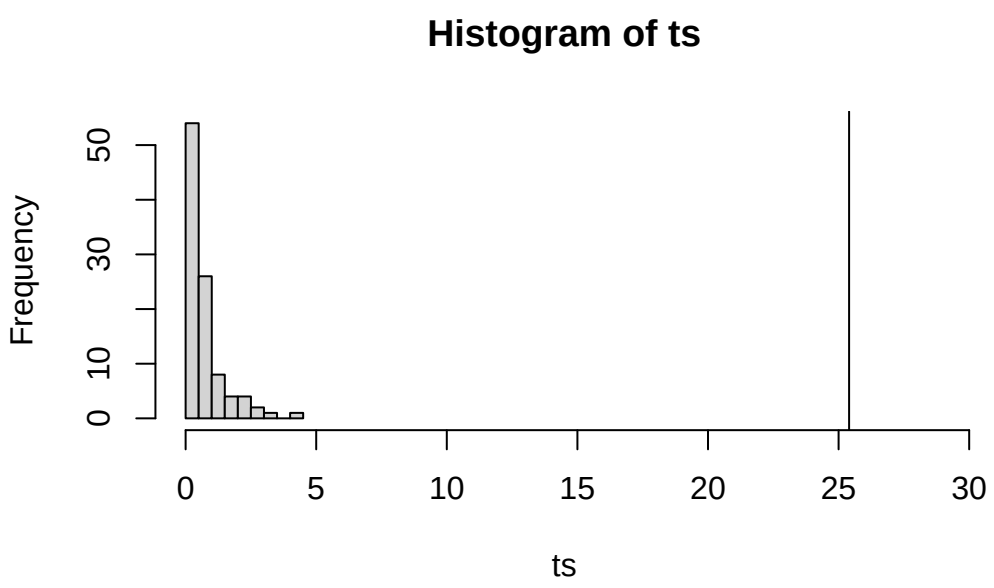
```
  n=nrow(batch_long)
  n_sim=100


  #Step 2
  simulated=t(replicate(n_sim,rnorm(n,fe[1],sigma_est)))

  # n_sim x 16
  dim(simulated)
```

```
[1] 100  16
```

```r
#Step 3
#takes a simulated sample y and computes the LRT for Y
compute_lrt=function(y){

  #create a copy of the dataset
  batch_copy=batch_long

  #replace response with new sample
  batch_copy$r=y

  #replace response with new sampl
  alt=lme(
      fixed= r~1,
      random= r ~ 1 | batch, data=batch_copy )

  null<-lm(r ~ 1 , data=batch_copy)

  test=lmtest::lrtest(null,alt)$Chisq[2]


  return(test)
}

#compute LRT for each simulated sample
ts=suppressWarnings(apply(simulated, 1, compute_lrt))



pvalue=mean(observed<=ts); pvalue
```

```
[1] 0
```

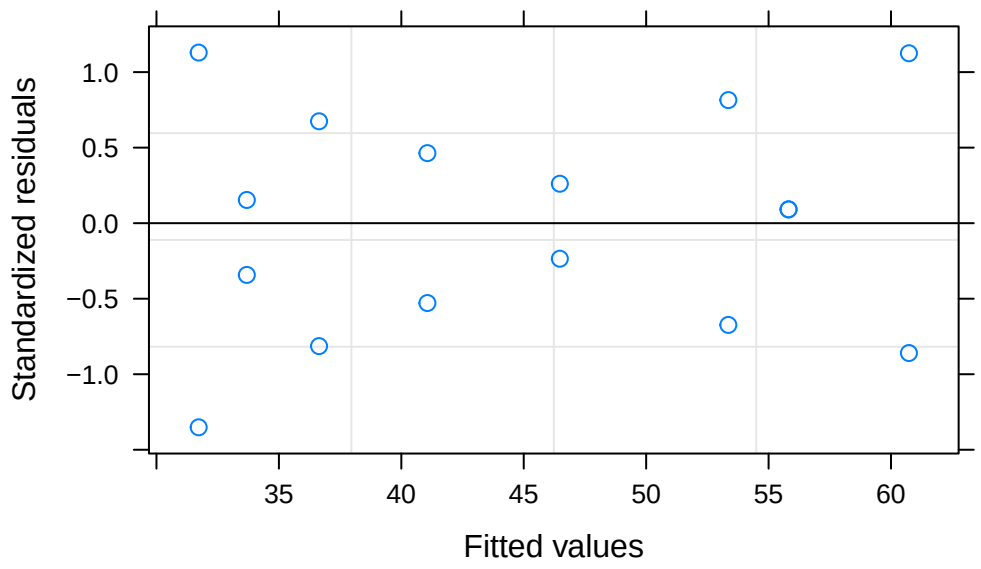
```r
hist(ts,xlim=c(min(ts),29))
abline(v=observed)
```



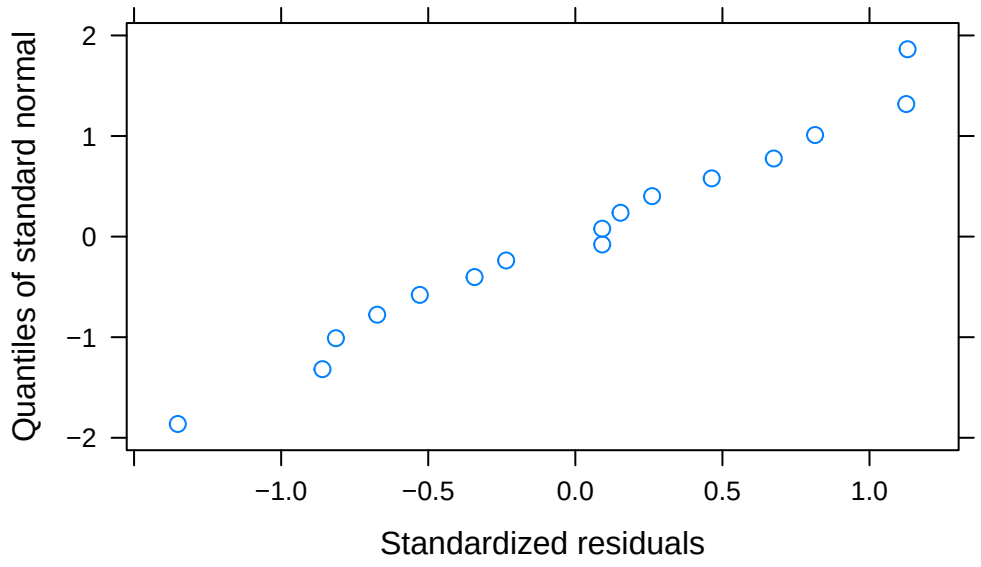

```
observed
```

```
[1] 25.40237
```

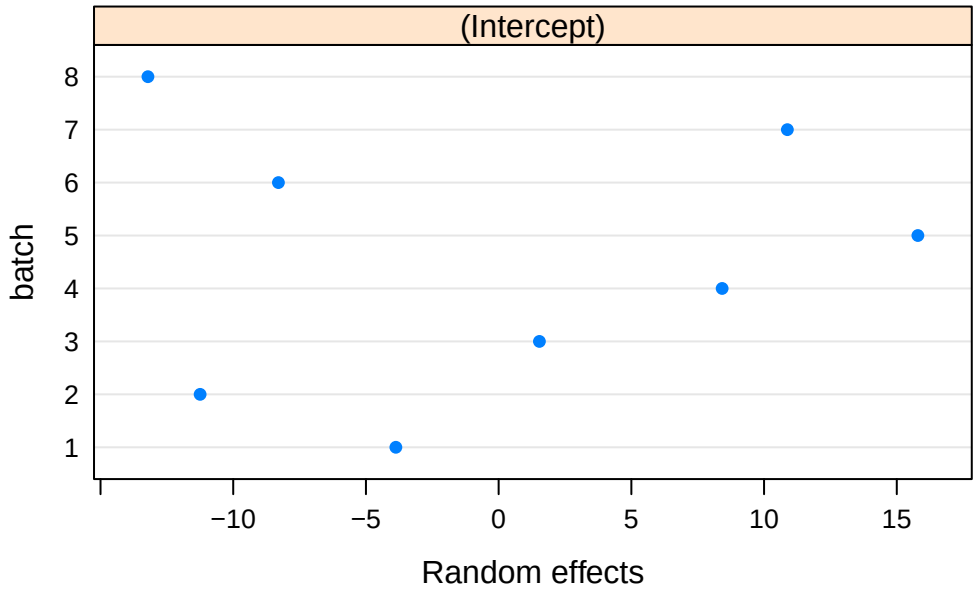
```
plot(model_1)
```



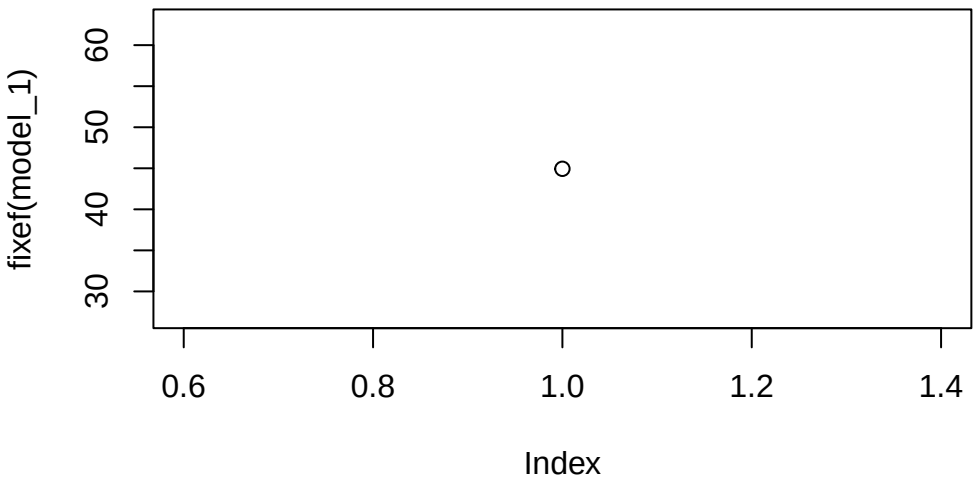

```
qqnorm(model_1, ~ residuals(.,type="pearson"))
```



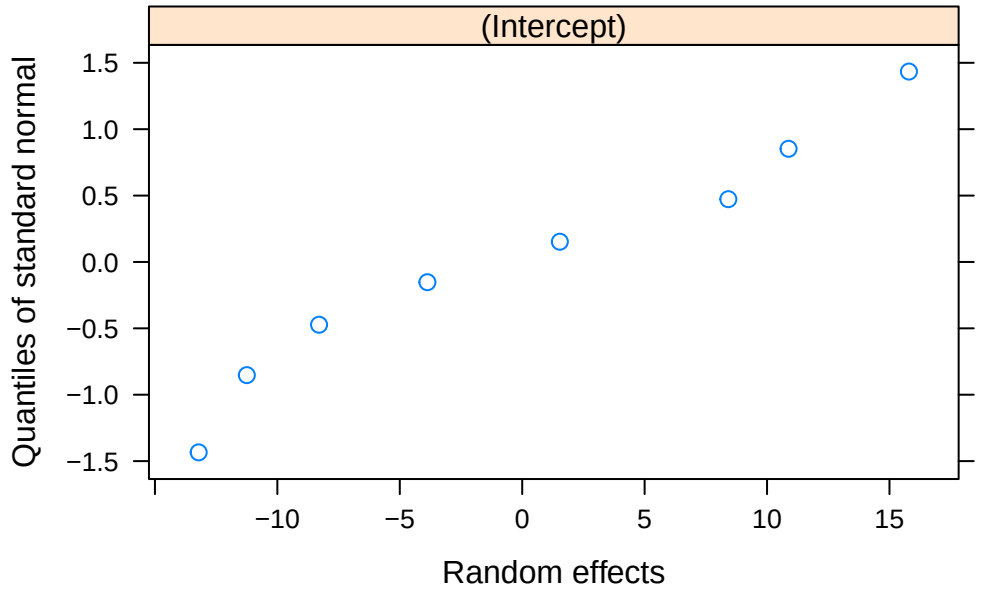
```
plot(ranef(model_1))
```



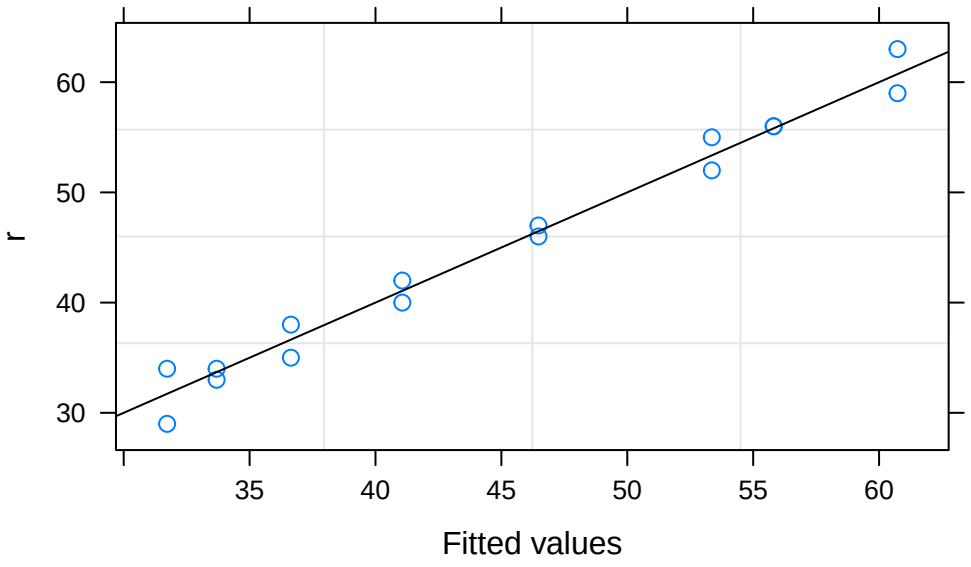

```
plot(fixef(model_1))
```




```
qqnorm(model_1, ~ ranef(.))
```



```
plot(model_1, r ~ fitted(.),abline=c(0,1))
```




```
intervals(model_1)
```

```
Approximate 95% confidence intervals

 Fixed effects:
               lower    est.    upper
(Intercept) 35.93112 44.9375 53.94388

 Random Effects:
  Level: batch
                    lower     est.    upper
sd((Intercept)) 6.430117 10.95445 18.66216

 Within-group standard error:
   lower     est.    upper
1.234790 2.015565 3.290036
```

Results summary:

- Quick summary: the mean is estimated to be 45, and we saw significant variation between batches. The batch mean concentration has standard deviation 11.
- More on the mean: The mean concentration is estimated to be 45, at least, we estimate that the mean concentration is not below 36 and does not exceed 54.
- Batch variability: The concentration varies significantly between batches. The standard deviation of the mean batch concentration is estimated to be 12, ranging from (6,19). This means we estimate that roughly 68% of the batches have a mean concentration within 11 units of the overall mean (estimated to be 45) and 95% are within 22 units of the overall mean.

1.4 Case study: Air Pollution

```
library(nlme)
library(lme4)
```

1.4.1 Case information:

- Six Cities Air Pollution Data – Data on lung growth along with assorted patient information.
- How much of lung size do age and height explain?

Data Columns:

- id: Patient ID
- ht: Patient height at the corresponding visit
- age: Patient age
- baseht: Patient height at the first visit
- baseage: Patient age at the first visit
- logfev1: The log of FEV1 measurement (outcome based on lung function)

Data Info:

- https://content.sph.harvard.edu/fitzmaur/ala2e/
- Applied LDA: Garrett Fitzmaurice, Nan Laird & James Ware
- Dockery, D.W., Berkey, C.S., Ware, J.H., Speizer, F.E. and Ferris, B.G. (1983). Distribution of FVC and FEV1 in children 6 to 11 years old. American Review of Respiratory Disease, 128, 405-412.

1.4.2 EDA:

```
air_pollution <- read.csv("data/air_pollution.csv")
```

Let's explore the data.

```
par(cex.lab=2,cex.axis=2,mfrow=c(1,1))

head(air_pollution)
```

```
  id   ht     age baseht baseage logfev1
1  1 1.20  9.3415    1.2  9.3415 0.21511
2  1 1.28 10.3929    1.2  9.3415 0.37156
3  1 1.33 11.4524    1.2  9.3415 0.48858
4  1 1.42 12.4600    1.2  9.3415 0.75142
5  1 1.48 13.4182    1.2  9.3415 0.83291
6  1 1.50 15.4743    1.2  9.3415 0.89200
```

```
summary(air_pollution)
```

```
       id              ht             age             baseht     
 Min.   :  1.0   Min.   :1.110   Min.   : 6.434   Min.   :1.110  
 1st Qu.: 69.0   1st Qu.:1.370   1st Qu.: 9.719   1st Qu.:1.220  
 Median :129.0   Median :1.540   Median :12.597   Median :1.260  
 Mean   :135.7   Mean   :1.498   Mean   :12.568   Mean   :1.276  
 3rd Qu.:199.0   3rd Qu.:1.620   3rd Qu.:15.368   3rd Qu.:1.320  
 Max.   :300.0   Max.   :1.790   Max.   :18.691   Max.   :1.720  
    baseage          logfev1       
 Min.   : 6.434   Min.   :-0.04082  
 1st Qu.: 7.135   1st Qu.: 0.54812  
 Median : 7.781   Median : 0.86710  
 Mean   : 8.030   Mean   : 0.81600  
 3rd Qu.: 8.449   3rd Qu.: 1.09861  
 Max.   :14.067   Max.   : 1.59534  
```

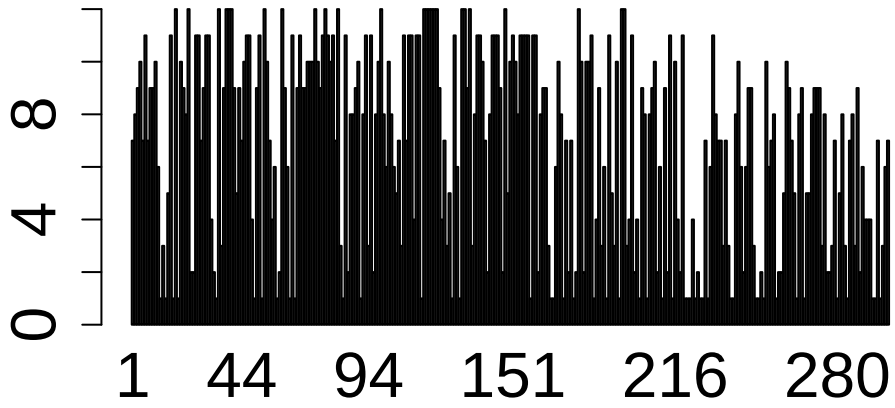
```
#Check for missing values
colSums(is.na(air_pollution))
```

```
     id      ht     age  baseht baseage logfev1 
      0       0       0       0       0       0 
```

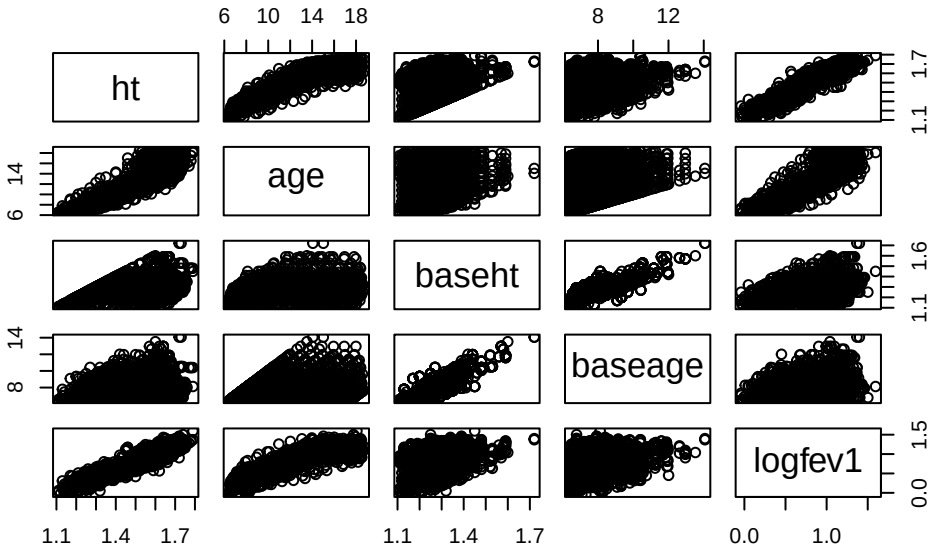
```r
unique(air_pollution$baseht)
```

```
 [1] 1.20 1.13 1.18 1.15 1.11 1.24 1.27 1.17 1.32 1.26 1.25 1.19 1.21 1.23 1.22
[16] 1.30 1.37 1.41 1.14 1.29 1.31 1.28 1.36 1.33 1.38 1.12 1.35 1.34 1.45 1.39
[31] 1.16 1.58 1.60 1.40 1.42 1.72 1.46 1.48 1.52 1.49 1.56 1.53 1.43 1.44 1.59
[46] 1.57
```

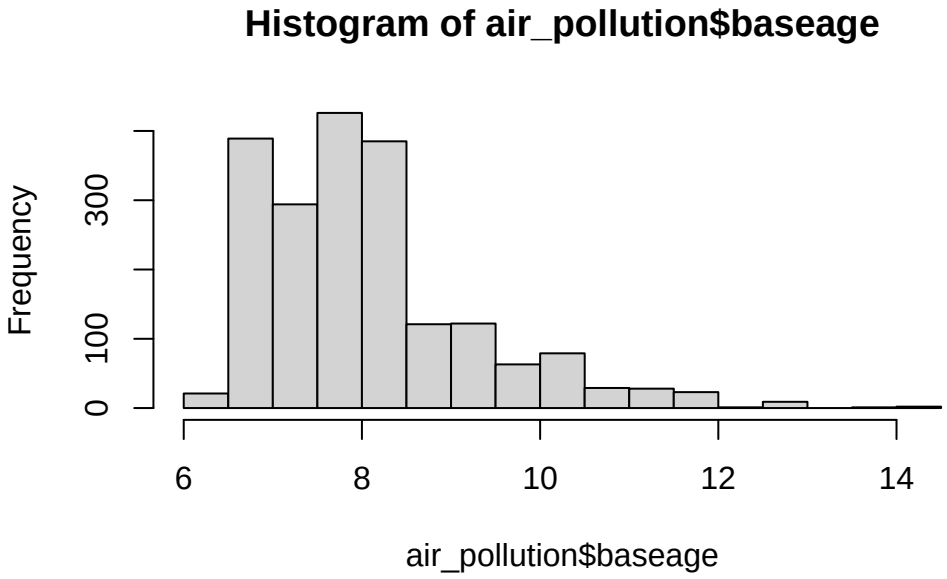
```r
# Is it balanced?
n=length(unique(air_pollution$id))
# typeof(air_pollution$id)
barplot(table(as.factor(air_pollution$id)))
```



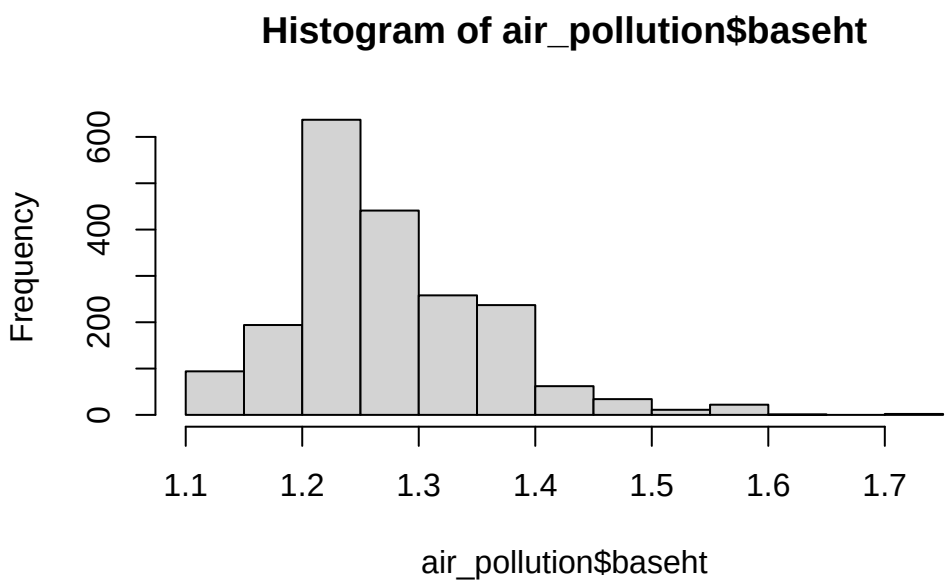

```r
plot(air_pollution[,-1])
```



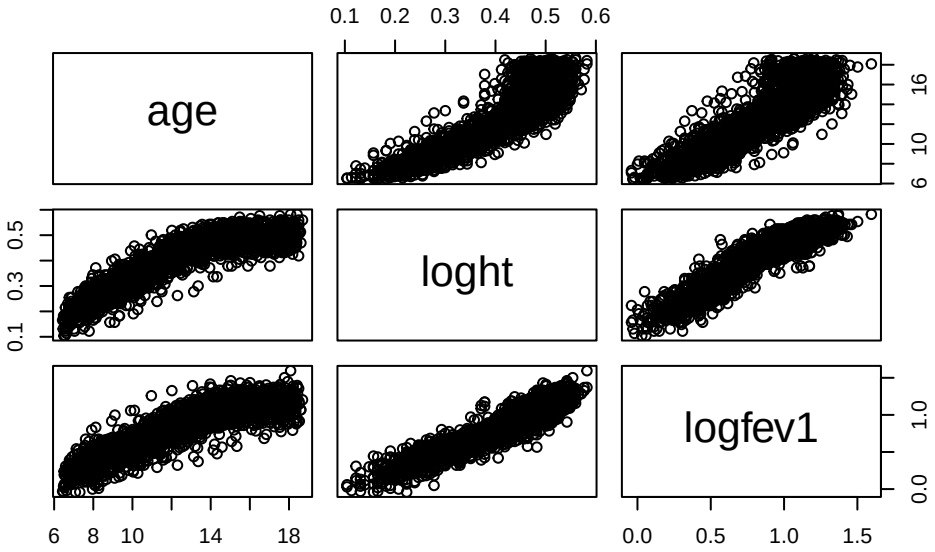

```
hist(air_pollution$baseage)
```



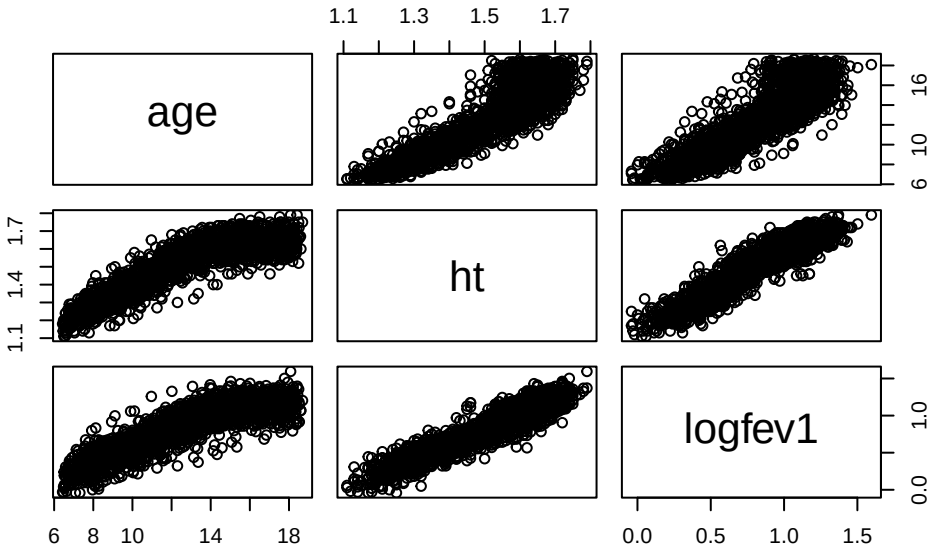
```r
hist(air_pollution$baseht)
```



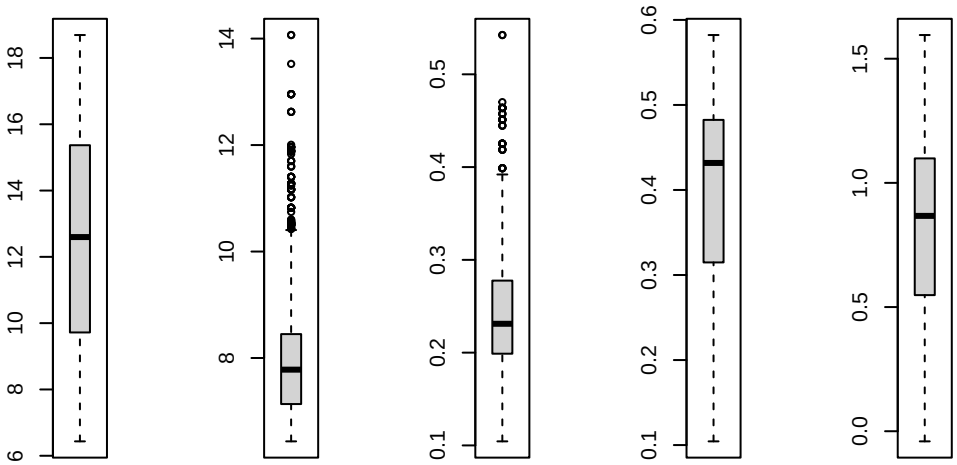

```r
# hist(air_pollution$age)
# hist(air_pollution$ht)

#Hint: height has been shown to be linearly associated with logfev1 on log scale
air_pollution$loght=log(air_pollution$ht)
air_pollution$logbht=log(air_pollution$baseht)
plot(air_pollution[,c('age','loght','logfev1')])
```



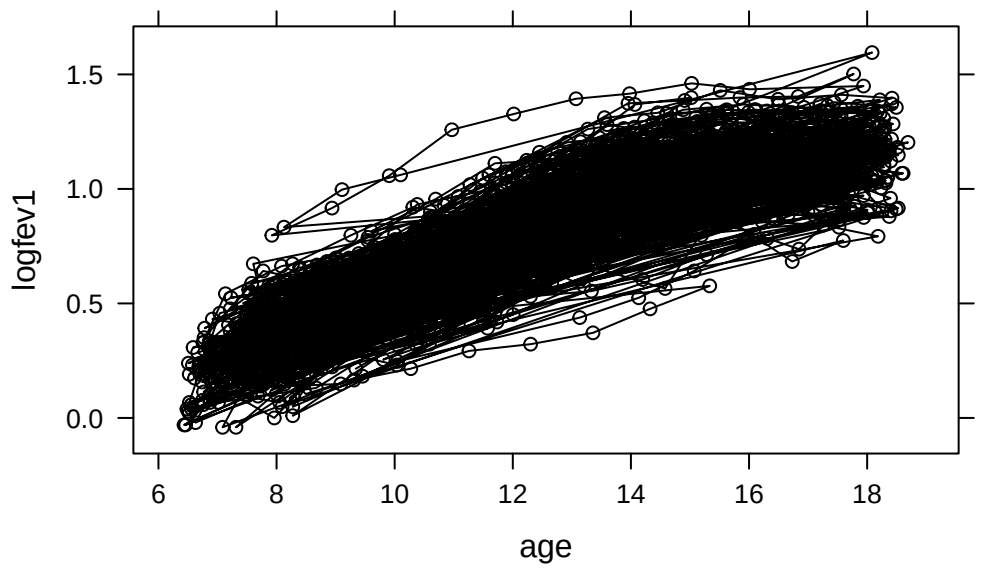

```
plot(air_pollution[,c('age','ht','logfev1')])
```



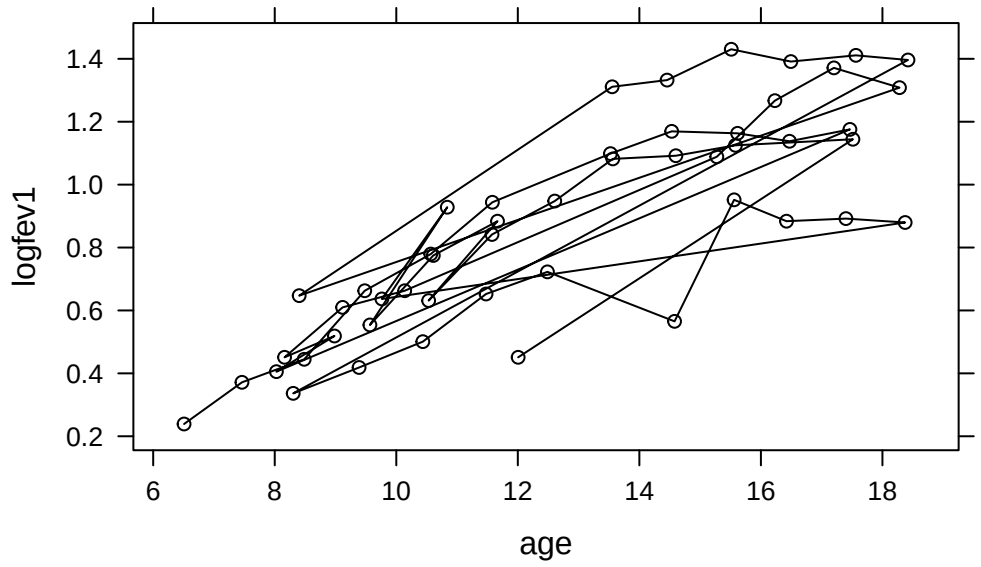
```
par(mfrow=c(1,5))
bx=apply(air_pollution[,c('age','baseage','logbht','loght','logfev1')],2,boxplot)
```



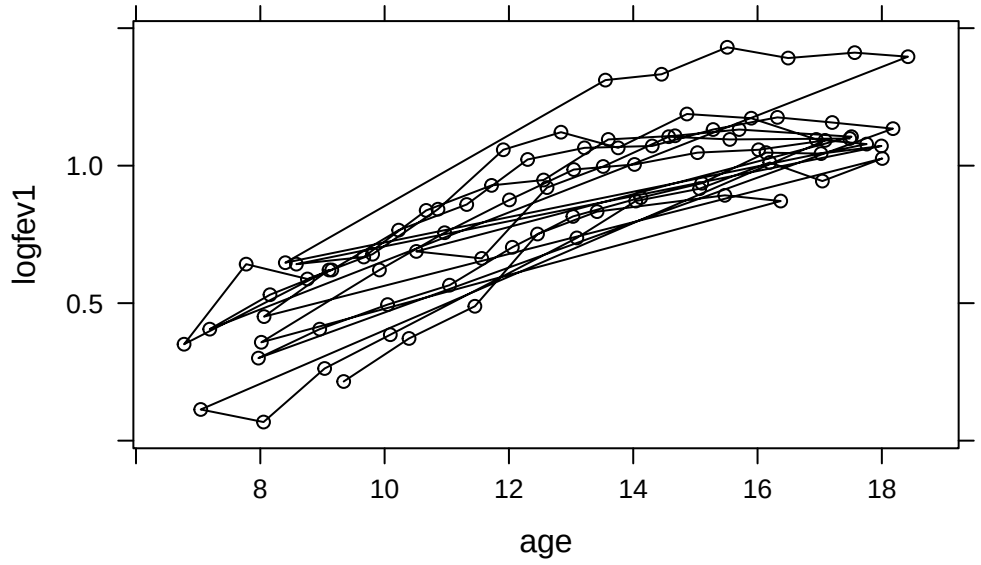

```
lattice::xyplot(logfev1~age, air_pollution, col = 'black', type = c('l', 'p'))
```



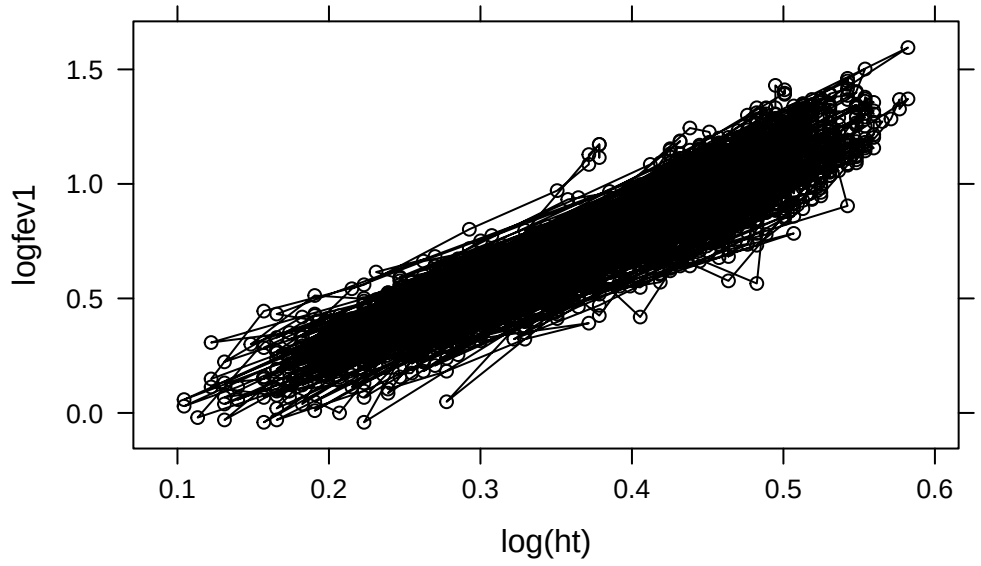

```
lattice::xyplot(logfev1~age, air_pollution[ air_pollution$id %in% sample(1:n,10),], col =
```



```
lattice::xyplot(logfev1~age, air_pollution[ air_pollution$id %in% sample(1:n,10),], col =
```




```
lattice::xyplot(logfev1~log(ht), air_pollution, col = 'black', type = c('l', 'p'))
```



What can we conclude from this exploratory analysis?

- 300 participants
- No missing values
- Not balanced
- Starting age ranges widely
- Somewhat linear relationships with the response, but not quite.
- Log scale might be better for height
- Heavy tail in base age, and base height.
- Parallel regression line between response and age

1.4.3 Specifying

- Let i index the individuals and j index the jth age recorded for a given individual.

$$Y_{ij} = X_{ij}^\top \alpha + Z_{ij}^\top \beta_i + \epsilon_{ij},$$

where for $i \in [299]$ and $j \in [J_i]$.

Here,

- $Z_{ij}^\top = (1, age)$, $X_{ij}^\top = (1, age, loght...)$
- Each observation has a random intercept and slope, which depends on their age.

1.4.4 Estimation

Let's estimate the parameters of this model. Let's use REML to estimate our parameters.

```r
#defaults to REML
model <- nlme::lme(
    fixed = logfev1 ~ age + log(ht) + baseage + log(baseht) ,
    random =~age|id,
    correlation = NULL, # Defaults to sigma^2 I
    method = 'REML',
    data = air_pollution
)
summary(model)
```

```
Linear mixed-effects model fit by REML
  Data: air_pollution
        AIC       BIC   logLik
  -4549.882 -4499.528 2283.941

Random effects:
 Formula: ~age | id
 Structure: General positive-definite, Log-Cholesky parametrization
            StdDev      Corr
(Intercept) 0.110485541 (Intr)
age         0.007078381 -0.553
Residual    0.060237881

Fixed effects:  logfev1 ~ age + log(ht) + baseage + log(baseht)
                 Value  Std.Error   DF  t-value p-value
(Intercept) -0.2883233 0.03871675 1692 -7.44699  0.0000
age          0.0235286 0.00139534 1692 16.86231  0.0000
log(ht)      2.2371984 0.04353724 1692 51.38585  0.0000
baseage     -0.0165088 0.00745785  296 -2.21362  0.0276
log(baseht)  0.2182148 0.14552087  296  1.49954  0.1348
 Correlation:
            (Intr) age    lg(ht) baseag
age          0.023
log(ht)     -0.077 -0.875
baseage     -0.822 -0.184  0.180
log(baseht)  0.370  0.239 -0.275 -0.815

Standardized Within-Group Residuals:
```

```
        Min          Q1         Med          Q3         Max
-6.45672792 -0.52534885  0.05351814  0.60114614  2.76671671

Number of Observations: 1993
Number of Groups: 299
```

1.4.5 Testing

Do we need the baseline fixed effects? Did the height and age they entered the study at effect the outcome?

```
library(nlme)
#marginal tests:
summary(model)
```

```
Linear mixed-effects model fit by REML
  Data: air_pollution
        AIC       BIC   logLik
  -4549.882 -4499.528 2283.941

Random effects:
 Formula: ~age | id
 Structure: General positive-definite, Log-Cholesky parametrization
            StdDev      Corr
(Intercept) 0.110485541 (Intr)
age         0.007078381 -0.553
Residual    0.060237881

Fixed effects:  logfev1 ~ age + log(ht) + baseage + log(baseht)
                 Value  Std.Error   DF  t-value p-value
(Intercept) -0.2883233 0.03871675 1692 -7.44699  0.0000
age          0.0235286 0.00139534 1692 16.86231  0.0000
log(ht)      2.2371984 0.04353724 1692 51.38585  0.0000
baseage     -0.0165088 0.00745785  296 -2.21362  0.0276
log(baseht)  0.2182148 0.14552087  296  1.49954  0.1348
 Correlation:
            (Intr) age    lg(ht) baseag
age          0.023
log(ht)     -0.077 -0.875
baseage     -0.822 -0.184  0.180
log(baseht)  0.370  0.239 -0.275 -0.815
```

```
Standardized Within-Group Residuals:
        Min          Q1         Med          Q3         Max
-6.45672792 -0.52534885  0.05351814  0.60114614  2.76671671

Number of Observations: 1993
Number of Groups: 299
```

```
#sequential tests:
anova(model)
```

```
            numDF denDF   F-value p-value
(Intercept)     1  1692 11406.811  <.0001
age             1  1692 16605.209  <.0001
log(ht)         1  1692  2905.336  <.0001
baseage         1   296     2.929  0.0881
log(baseht)     1   296     2.249  0.1348
```

```
#based on the above, lets remove the base effects
model=update(model,fixed= logfev1 ~ age+log(ht))
summary(model)
```

```
Linear mixed-effects model fit by REML
  Data: air_pollution
        AIC       BIC   logLik
  -4559.789 -4520.618 2286.895

Random effects:
 Formula: ~age | id
 Structure: General positive-definite, Log-Cholesky parametrization
            StdDev     Corr
(Intercept) 0.11107952 (Intr)
age         0.00706045 -0.552
Residual    0.06025488

Fixed effects:  logfev1 ~ age + log(ht)
                 Value  Std.Error   DF   t-value p-value
(Intercept) -0.3693653 0.00916684 1692 -40.29366       0
age          0.0230800 0.00135469 1692  17.03715       0
log(ht)      2.2493934 0.04176179 1692  53.86247       0
```

```
 Correlation:
        (Intr) age
age     -0.208
log(ht) -0.190 -0.869

Standardized Within-Group Residuals:
        Min          Q1         Med          Q3         Max
-6.44851637 -0.51754969  0.05387027  0.59382767  2.78517863

Number of Observations: 1993
Number of Groups: 299
```

Can the random slope be dropped?

```
model_null=update(model,random=logfev1 ~ 1|id)
summary(model_null)
```

```
Linear mixed-effects model fit by REML
  Data: air_pollution
        AIC       BIC   logLik
  -4489.806 -4461.827 2249.903

Random effects:
 Formula: logfev1 ~ 1 | id
        (Intercept)   Residual
StdDev:  0.09608834 0.06429074

Fixed effects:  logfev1 ~ age + log(ht)
                 Value  Std.Error   DF   t-value p-value
(Intercept) -0.3645554 0.00833614 1692 -43.73192       0
age          0.0237689 0.00126198 1692  18.83462       0
log(ht)      2.2162876 0.04195216 1692  52.82893       0
 Correlation:
        (Intr) age
age     -0.048
log(ht) -0.218 -0.928

Standardized Within-Group Residuals:
        Min          Q1         Med          Q3         Max
-6.00485051 -0.54072380  0.06768753  0.61714951  2.86626992

Number of Observations: 1993
```

```
Number of Groups: 299
```

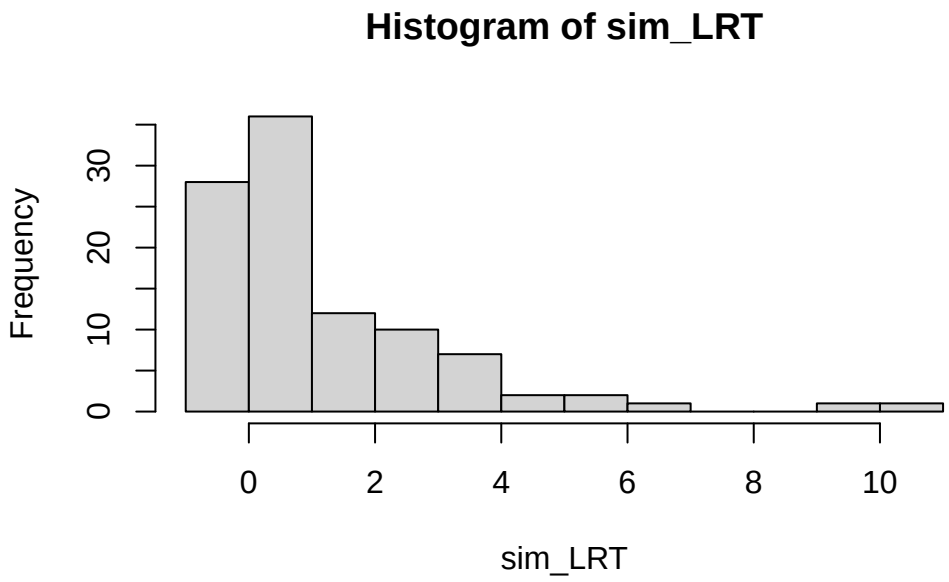
```
LRT=anova(model_null,model)$L[2]; LRT
```

```
[1] 73.98299
```

```
simmed=nlme::simulate.lme(model_null,nsim=100,m2=model)
sim_LRT=-2*(simmed$null$REML[,2]-simmed$alt$REML[,2])

pval=mean(sim_LRT>=LRT); print(pval)
```

```
[1] 0
```

```
hist(sim_LRT)
abline(v=LRT)
```

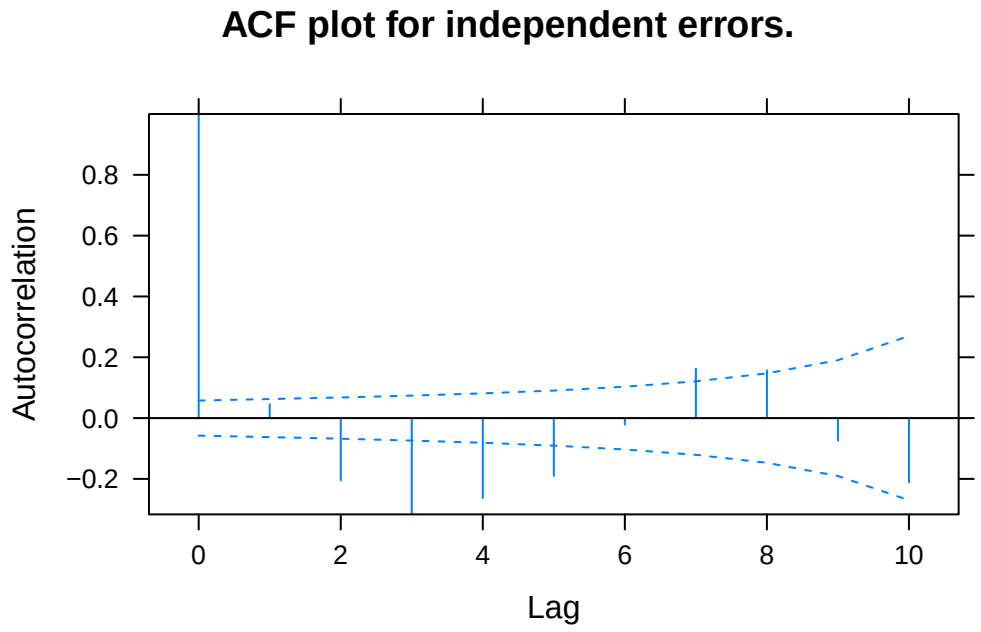



```
anova(model_null,model)
```

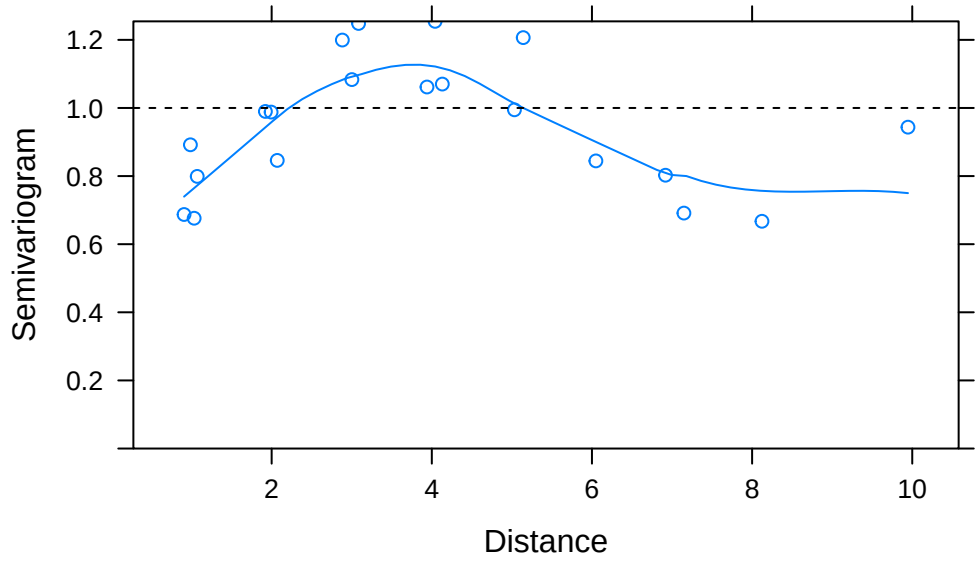
```
           Model df       AIC       BIC   logLik   Test  L.Ratio p-value
model_null     1  5 -4489.806 -4461.827 2249.903
model          2  7 -4559.789 -4520.618 2286.894 1 vs 2 73.98299  <.0001
```

1.4.6 Diagnostics

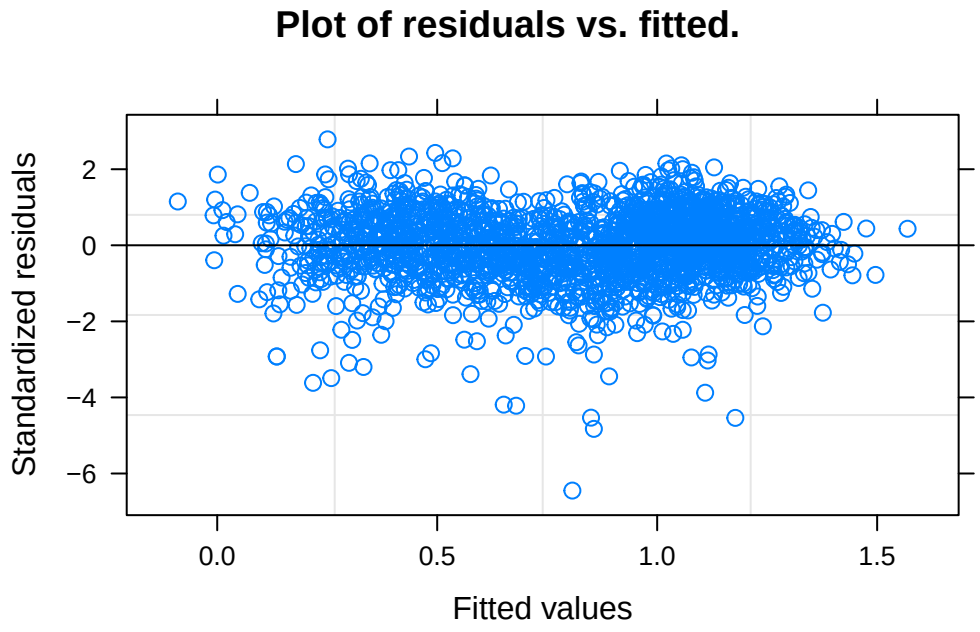
```
## ACF - Checks that errors are independent
plot(ACF(model), alpha = 0.01, main = "ACF plot for independent errors.") # This looks pro
```



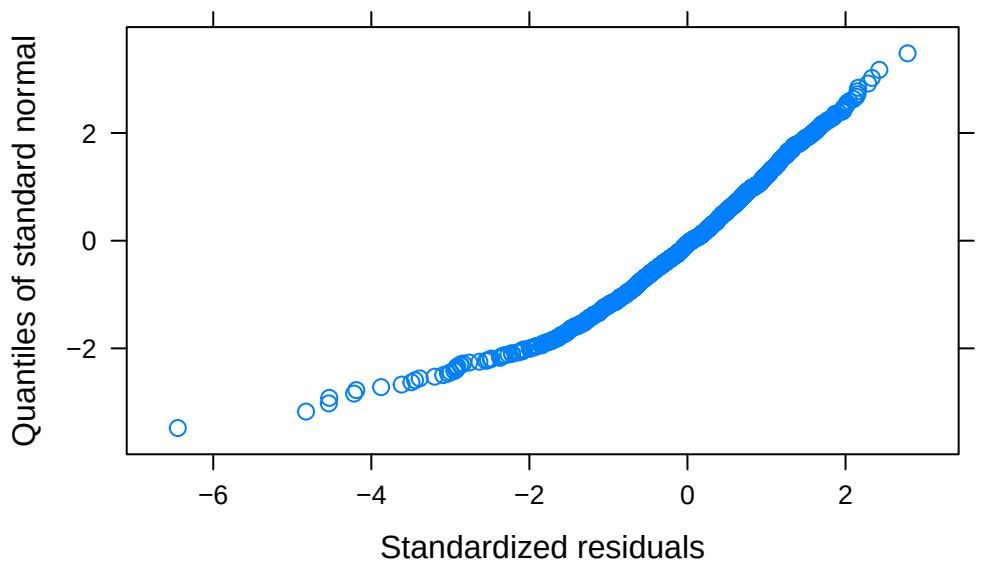

```
# This may not be the best way to check the model fit though, since there are not evenly s

# Instead we can use a 'semi-Variogram'.
# This should fluctuate randomly around 1
vg <- Variogram(model, form = ~age|id, resType = "pearson")
plot(vg, sigma=1) ## Looks okay, honestly, not the best.
```



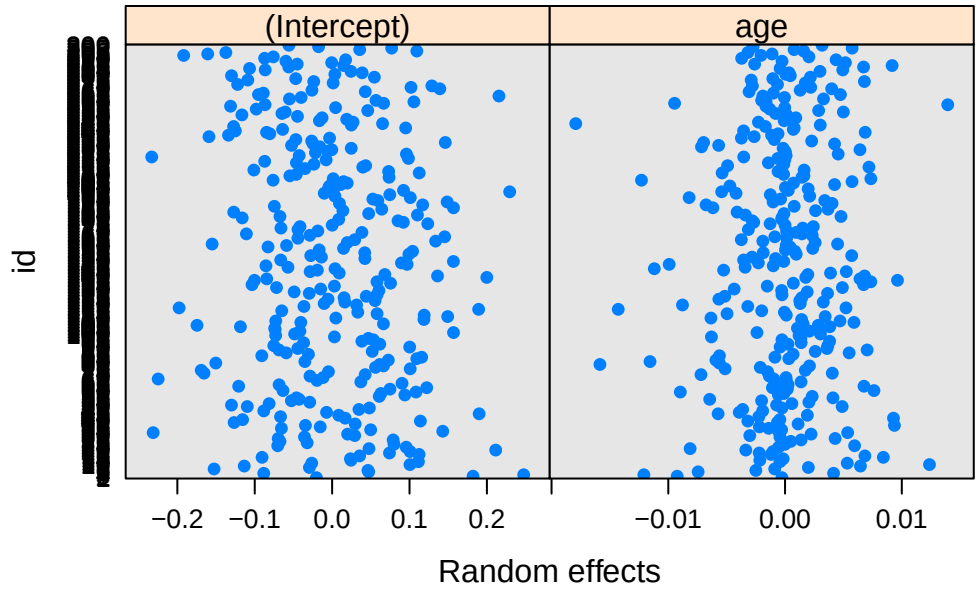

```r
# Residuals vs. Fitted (no patterns)
plot(model, main = "Plot of residuals vs. fitted.")
```



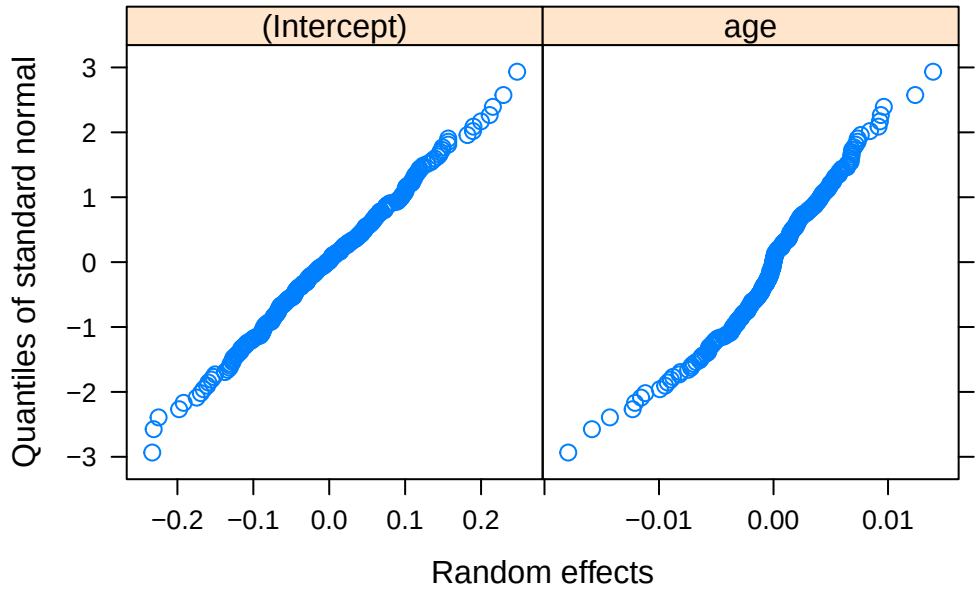
```r
# QQPlot for normality of errors
qqnorm(model, ~ residuals(., type="pearson")) # Some issues... probably
```



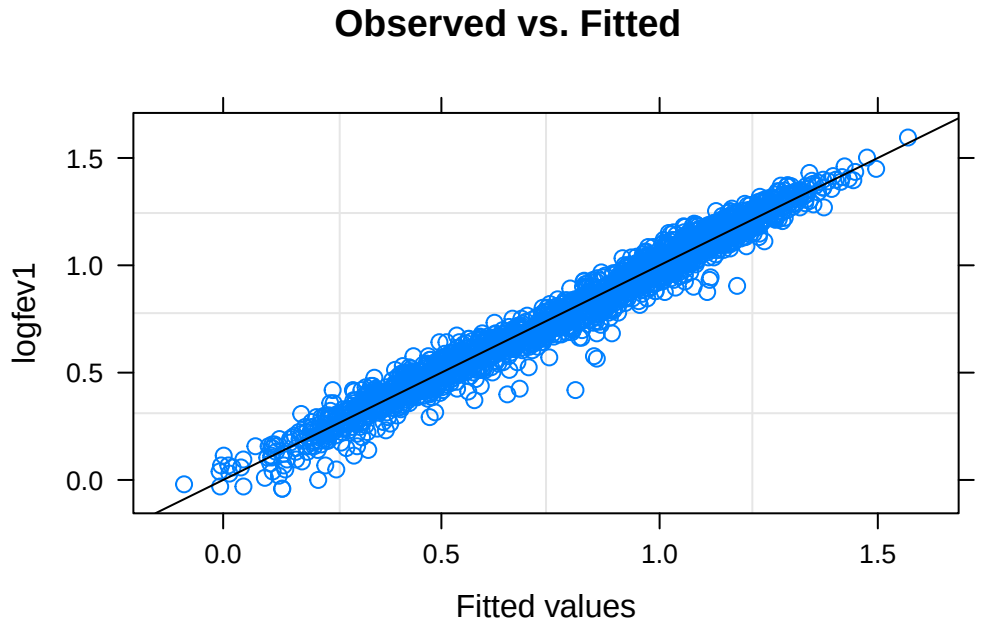

```r
# Plots for the Predicted (BLUPs)
plot(ranef(model))
```



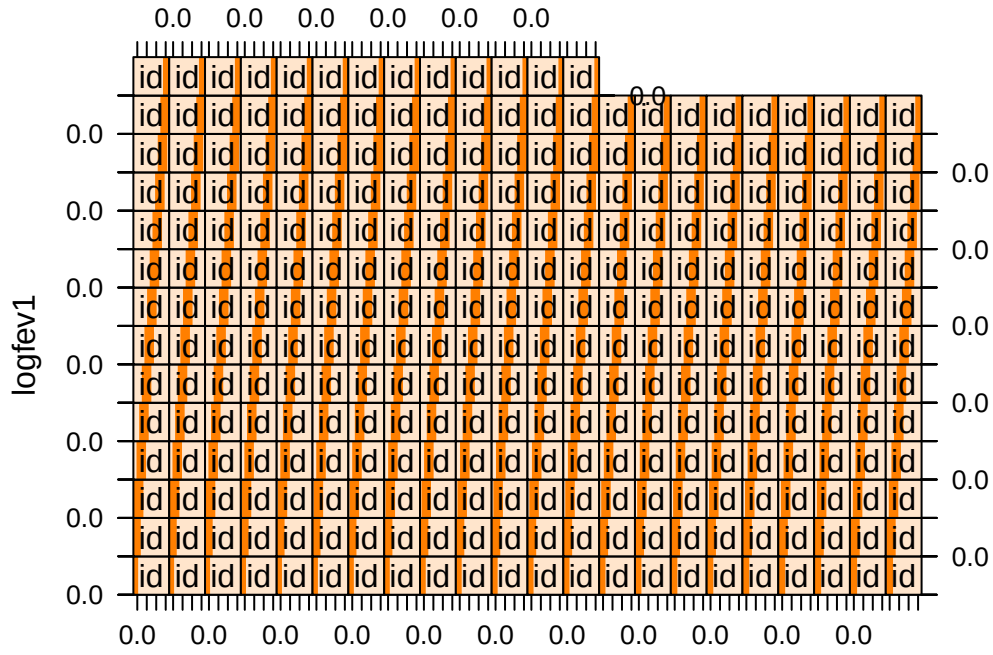

```
qqnorm(model, ~ranef(.)) # These look okay!
```



```r
# Observed vs. Fitted
plot(model, logfev1 ~ fitted(.), abline = c(0,1), main = "Observed vs. Fitted")
```




```r
plot(model, logfev1 ~ fitted(.)|id, abline = c(0,1), main = "Observed vs. Fitted (By Subje
```




```
# Could also look (e.g.) by treatment, if it existed!

### Intervals
intervals(model)
```

```
Approximate 95% confidence intervals

 Fixed effects:
                 lower        est.       upper
(Intercept) -0.3873448 -0.36936532 -0.35138579
age          0.0204230  0.02308004  0.02573708
log(ht)      2.1674832  2.24939340  2.33130360

 Random Effects:
  Level: id
                              lower        est.        upper
sd((Intercept))        0.095315000  0.11107952  0.129451390
sd(age)                0.005828914  0.00706045  0.008552184
cor((Intercept),age) -0.685485568 -0.55220425 -0.383114275

 Within-group standard error:
     lower       est.      upper
```

```
0.05812314 0.06025488 0.06246480
```

```
intervals(model)
```

```
Approximate 95% confidence intervals

 Fixed effects:
                lower        est.       upper
(Intercept) -0.3873448 -0.36936532 -0.35138579
age          0.0204230  0.02308004  0.02573708
log(ht)      2.1674832  2.24939340  2.33130360

 Random Effects:
  Level: id
                            lower        est.        upper
sd((Intercept))       0.095315000  0.11107952  0.129451390
sd(age)               0.005828914  0.00706045  0.008552184
cor((Intercept),age) -0.685485568 -0.55220425 -0.383114275

 Within-group standard error:
     lower       est.      upper
0.05812314 0.06025488 0.06246480
```

```
### Predictions
new_data <- data.frame(id = c(1, 25, 25, 25),
                       age = c(18, 18, 18, 18),
                       ht = c(1.54, 1.85, 1.7, 2),
                       baseht = c(1.2, 1.32, 1.32, 1.32),
                       baseage = c(9.3415, 8.0274, 8.0274, 8.0274),
                       loght=log(c(1.54, 1.85, 1.7, 2)))

# level specifies whether at the population [0] or subject [1] level
predict(model, newdata = new_data, level = c(0,1))
```

```
  id predict.fixed predict.id
1  1      1.017324  0.9922821
2 25      1.429870  1.3972195
3 25      1.239667  1.2070167
4 25      1.605236  1.5725857
```

1.4.7 Results summary:

- Age explains a significant amount of the variability of lung size - a one year increase in age is roughly equivalent to a `exp(0.023)` in lung size (fev1)
- Height also explains lung size, we see that for every 10 cm, we have that lungs are `exp(2.25*log(0.1))` (fev1) bigger
- Population average lung size is `exp(-0.308090040)`
- There is some evidence that the time at which the subject entered the study was predictive of their lung size. Investigate!
- Seems like lungs stop growing at 16 - may be no need to study after that age?

1.5 Case study: Tale of two thieves, see Cabrera and McDougall (2002)

```
library(nlme)
library(lme4)
library(ggplot2)
```

```
Warning: package 'ggplot2' was built under R version 4.2.3
```

1.5.1 Case information:

- Concentration Data – Data on concentration of active ingredient in tablets and samples from blender
- Recall: "Our main concern is that the amount of active ingredient is consistent in the X tablets. We would like you to analyse both samples to determine how much the active ingredient differs from tablet to tablet. We also want to compare the quality of the samples retrieved by the thieves to determine Which one is better?"

Suppose we receive the following documentation:

Prescription and over-the-counter drugs contain a mixture of both active and inactive ingredients, with the dosage determined by the amount of active ingredient in each tablet. Making sure the tablets contain the correct dosage is an important problem in the drug manufacturing industry and in this case study, we consider an experiment conducted by a pharmaceutical company to investigate sampling variability and bias associated with the manufacture of a certain type of tablet.

1.5.2 Outline of the Problem

Tablet Manufacture The tablets were manufactured by mixing the active and inactive ingredients in a "V-blender," so-named because it looks like a large V. (See Figure 8.1.) Mixing was achieved by rotating the V-blender in the vertical direction. After the mixture was thoroughly blended, the powder was discharged from the bottom of the V-blender and compressed into tablet form.

Uniform Content: The most important requirement of this manufacturing process was that the tablets have uniform content. That is, the correct amount of active ingredient must be present in each tablet. The content uniformity of the mixture within the V-blender will need to be assessed. Thief Sampling A "thief" instrument was used to obtain samples from different locations within the V-blender. This was essentially a long pole with a closed scoop at one end, which was plunged into the powder mixture by a mechanical device. At the appropriate depth for a given location, the scoop was opened and a sample collected. Considerable force was needed to insert a thief into the powder mixture and it was of interest to compare two types of thieves.

- The Unit Dose thief collects three individual unit dose samples at each location.
- The Intermediate Dose thief collects one large sample which is itself sampled to give three unit dose samples.

1.5.3 Experiment Procedure

The objective of this experiment was to study bias and variability differences between the two thieves and to compare the thief-sampled results with those of the tablets. The experiment was implemented as follows.

1. Blend the mixture in the V-blender for 20 minutes.
2. Tie the thieves together and use them to obtain samples from six locations within the V-blender. A schematic of the V-blender and sampling locations was shown previously.
3. Discharge the powder from the V-blender and compress it to form tablets. Load tablets into 30 drums.
4. Select 10 drums and sample three tablets from each of these drums.
5. Assay all samples to determine the amount of active ingredient in each sample. The specified assay value is: 35 mg/100 mg.

The locations shown in the blender represented the "desired" sampling positions for the thieves. In the actual experiment, these "fixed" positions were subject to a certain amount of variability. The samples collected by the thieves can be regarded as random within each location.

In the Tablet experiment, the order in which the drums were filled was recorded and this information was incorporated into the random selection procedure. Specifically, one drum was

randomly selected from each triple sequence: {1, 2, 3} { 4, 5, 6} . . . {28, 29, 30}. The factor DRUM could therefore be used to test for a "time" effect in the Tablet data.

Data Columns:

- method
- location
- replicate
- assay/yb
- drum

Data Info: see 8.1 in Cabrera and McDougall (2002)

```
thief=read.csv('data/thief.csv')
tablet=read.csv('data/tablet.csv')
```

1.5.4 EDA

Let's explore the data

```
par(cex.lab=2,cex.axis=2,mfrow=c(1,1))

head(tablet)
```

```
  methdb drum tablet    yb
1 Tablet    1      1 35.77
2 Tablet    1      2 39.44
3 Tablet    1      3 36.43
4 Tablet    5      1 35.71
5 Tablet    5      2 37.08
6 Tablet    5      3 36.54
```

```
summary(tablet)
```

```
   methdb               drum          tablet        yb
 Length:30          Min.   : 1.0   Min.   :1   Min.   :33.09
 Class :character   1st Qu.: 7.0   1st Qu.:1   1st Qu.:35.10
 Mode  :character   Median :15.5   Median :2   Median :35.69
                    Mean   :14.9   Mean   :2   Mean   :35.79
                    3rd Qu.:22.0   3rd Qu.:3   3rd Qu.:36.52
                    Max.   :28.0   Max.   :3   Max.   :39.44
```

```
head(thief)
```

```
  METHOD LOCATION REPLICATE ASSAY
1   Intm        1         1 34.38
2   Intm        1         2 34.87
3   Intm        1         3 35.71
4   Intm        2         1 35.31
5   Intm        2         2 37.59
6   Intm        2         3 38.02
```

```
summary(thief)
```

```
    METHOD             LOCATION     REPLICATE     ASSAY      
 Length:36          Min.   :1.0   Min.   :1   Min.   :32.77  
 Class :character   1st Qu.:2.0   1st Qu.:1   1st Qu.:35.39  
 Mode  :character   Median :3.5   Median :2   Median :36.67  
                    Mean   :3.5   Mean   :2   Mean   :36.65  
                    3rd Qu.:5.0   3rd Qu.:3   3rd Qu.:37.88  
                    Max.   :6.0   Max.   :3   Max.   :39.80  
```

```
#Check for missing values
colSums(is.na(thief))
```

```
   METHOD  LOCATION REPLICATE     ASSAY 
        0         0         0         0 
```

```
colSums(is.na(tablet))
```

```
methdb   drum tablet     yb 
     0      0      0      0 
```

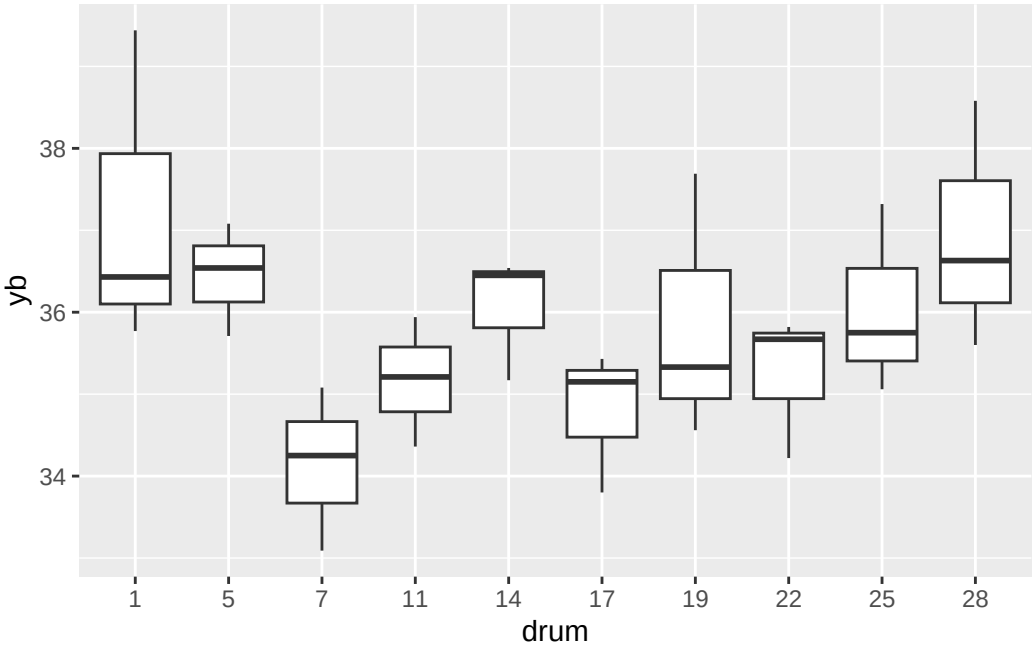
```
unique(tablet$methdb)
```

```
[1] "Tablet"
```

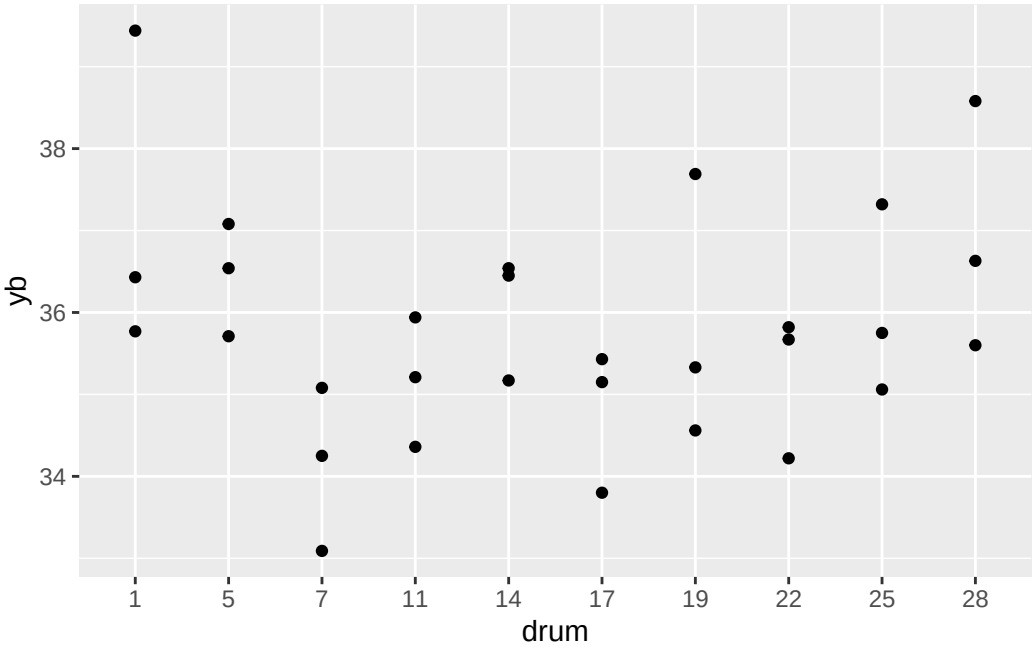
```
tablet$drum=as.factor(tablet$drum); tablet$drum
```

```
 [1] 1  1  1  5  5  5  7  7  7  11 11 11 14 14 14 17 17 17 19 19 19 22 22 22 25
[26] 25 25 28 28 28
Levels: 1 5 7 11 14 17 19 22 25 28
```

```
e <- ggplot2::ggplot(tablet, ggplot2::aes(x = drum, y=yb)) + ggplot2::geom_boxplot()
e
```

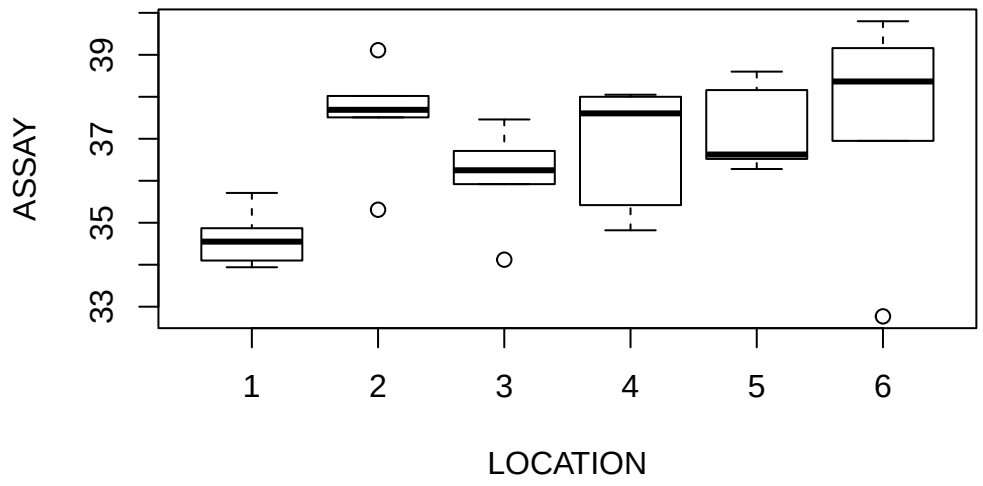



```
e <- ggplot2::ggplot(tablet, ggplot2::aes(x = drum, y=yb)) + ggplot2::geom_point()
e
```

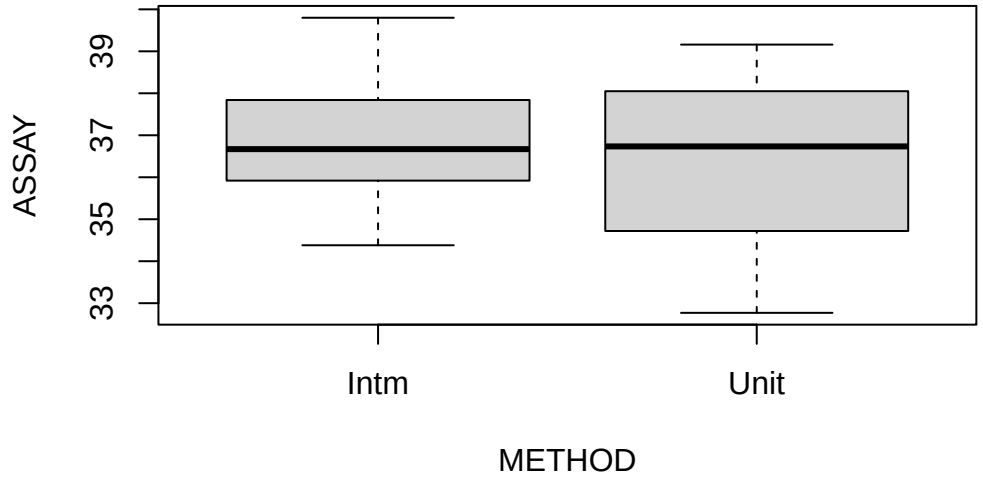



```
boxplot(ASSAY ~ LOCATION, col=as.numeric(thief$METHOD),data=thief)
```

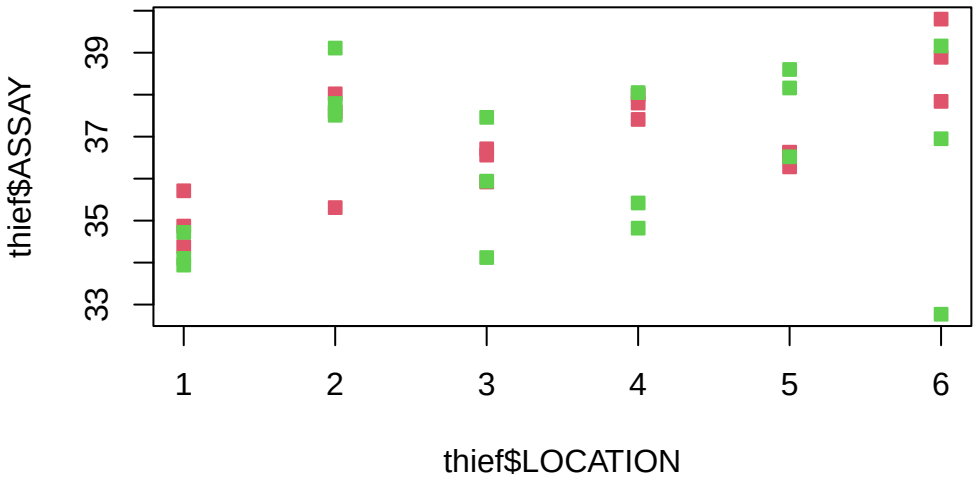
```
Warning in boxplot.default(split(mf[[response]], mf[-response], drop = drop, :
NAs introduced by coercion
```




```
boxplot(ASSAY ~ METHOD,data=thief)
```



```
color=as.integer(as.factor(thief$METHOD))+1
plot(thief$LOCATION ,thief$ASSAY,col=color,pch=22,bg=color)
```



What can we conclude from this exploratory analysis?

1.5.5 Specification

Let's tackle the first question: how much does the active ingredient in the tablets vary? Write down the model fit below.

```
names(tablet)[4]="con"
model=lme(
  fixed= con ~1,
  random= con ~ 1 | drum, data=tablet )
summary(model)
```

```
Linear mixed-effects model fit by REML
  Data: tablet
      AIC      BIC  logLik
  108.092 112.1939 -51.046
```

```
Random effects:
 Formula: con ~ 1 | drum
        (Intercept) Residual
StdDev:   0.6673375 1.197821

Fixed effects:  con ~ 1
             Value Std.Error DF  t-value p-value
(Intercept) 35.789 0.3039075 20 117.7628       0

Standardized Within-Group Residuals:
        Min          Q1         Med          Q3         Max
-1.58945875 -0.62071916 -0.08544433  0.37232202  2.47467411

Number of Observations: 30
Number of Groups: 10
```

Okay, the "between drum standard deviation" is half the residual standard deviation. Let's test if its non-zero. Recall that for REML estimates, the asymptotic distribution for the LRT is not the same as usual. In this case, under the null hypothesis, $Y_{ij} \sim N(\mu, \sigma^2)$. Thus,

```
fit_null<-lm(con ~ 1 , data=tablet)
observed=lmtest::lrtest(fit_null,model)$Chisq[2]
```

```
Warning in modelUpdate(objects[[i - 1]], objects[[i]]): original model was of
class "lm", updated model is of class "lme"
```

```
fe=nlme::fixed.effects(model); fe
```

```
(Intercept)
     35.789
```

```
sigma_drum_est= nlme::getVarCov(model); sigma_drum_est
```

```
Random effects variance covariance matrix
            (Intercept)
(Intercept)     0.44534
  Standard Deviations: 0.66734
```

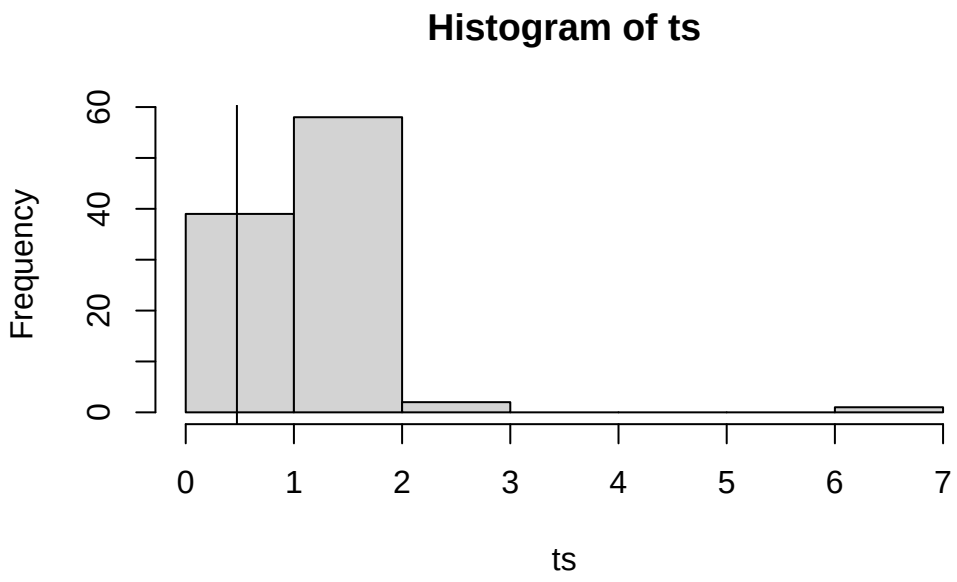
```r
sigma_est=model$sigma; sigma_est
```

```
[1] 1.197821
```

```r
n=nrow(tablet)
n_sim=100

simulated=replicate(n,rnorm(n_sim,fe[1],sigma_est))

# n_sim x n
# dim(simulated)

compute_lrt=function(y){
  tablet_copy=tablet
  tablet_copy$con=y

  alt=lme(
      fixed= con~1,
      random= con ~ 1 | drum, data=tablet_copy )

  null<-lm(con ~ 1 , data=tablet_copy)

  test=lmtest::lrtest(null,alt)$Chisq[2]
  return(test)
}

ts=suppressWarnings(apply(simulated, 1, compute_lrt))


# ts

pvalue=mean(observed<=ts); pvalue
```

```
[1] 0.87
```

```r
hist(ts)
abline(v=observed); observed
```




```
[1] 0.4731465
```

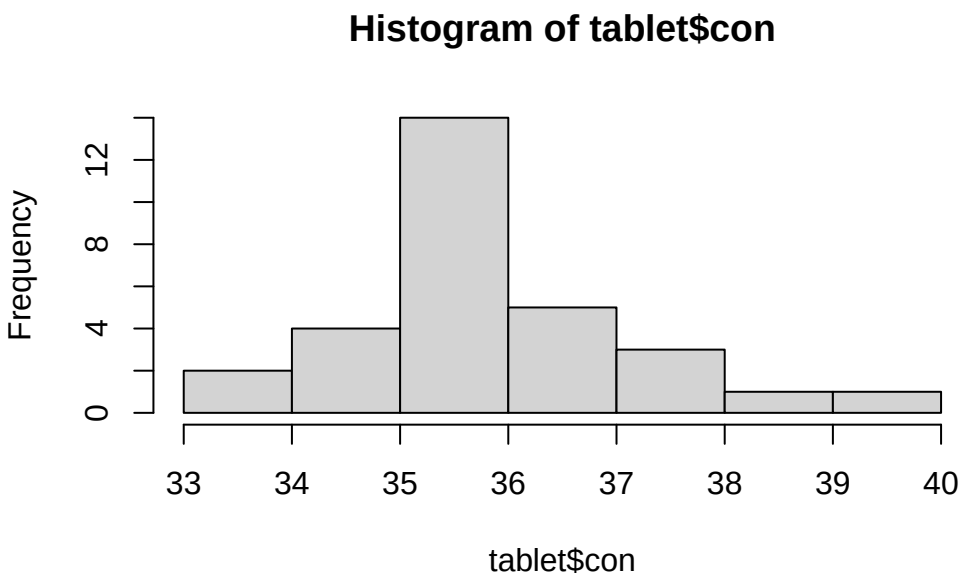
```
  1-pchisq(observed,1)/2-(observed>0)/2
```

```
[1] 0.2457716
```

```
  lmtest::lrtest(fit_null,model)
```

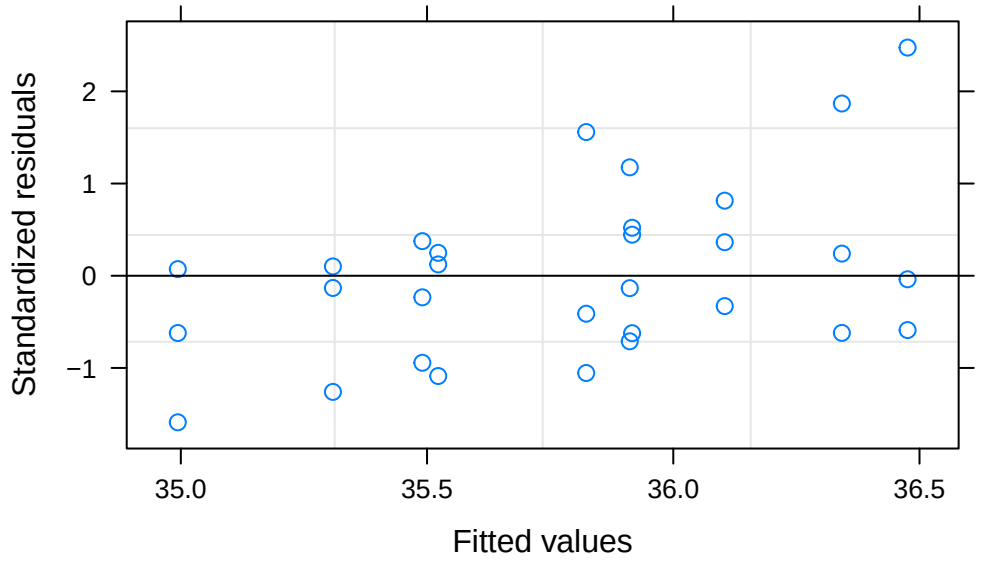
```
Warning in modelUpdate(objects[[i - 1]], objects[[i]]): original model was of
class "lm", updated model is of class "lme"

Likelihood ratio test

Model 1: con ~ 1
Model 2: con ~ 1
  #Df  LogLik Df  Chisq Pr(>Chisq)
1   2 -51.283
2   3 -51.046  1 0.4731     0.4915
```

```
hist(tablet$con)
```

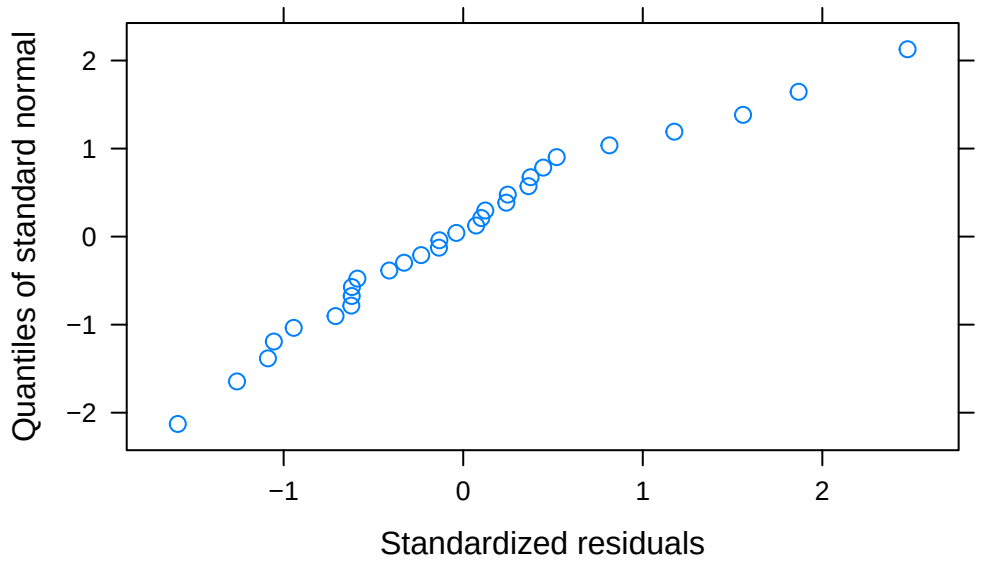


1.5.6 Diagnostics

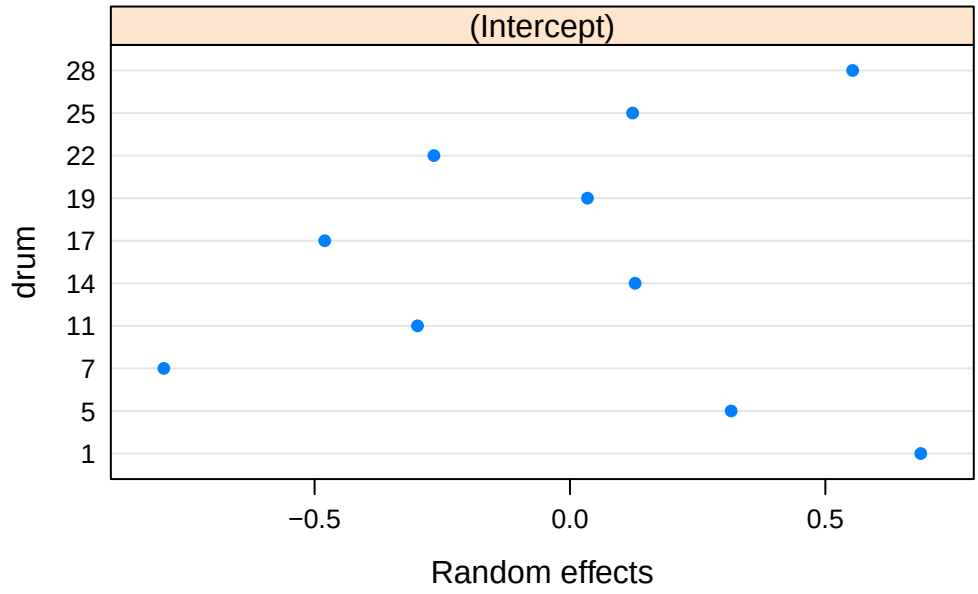
```
plot(model)
```



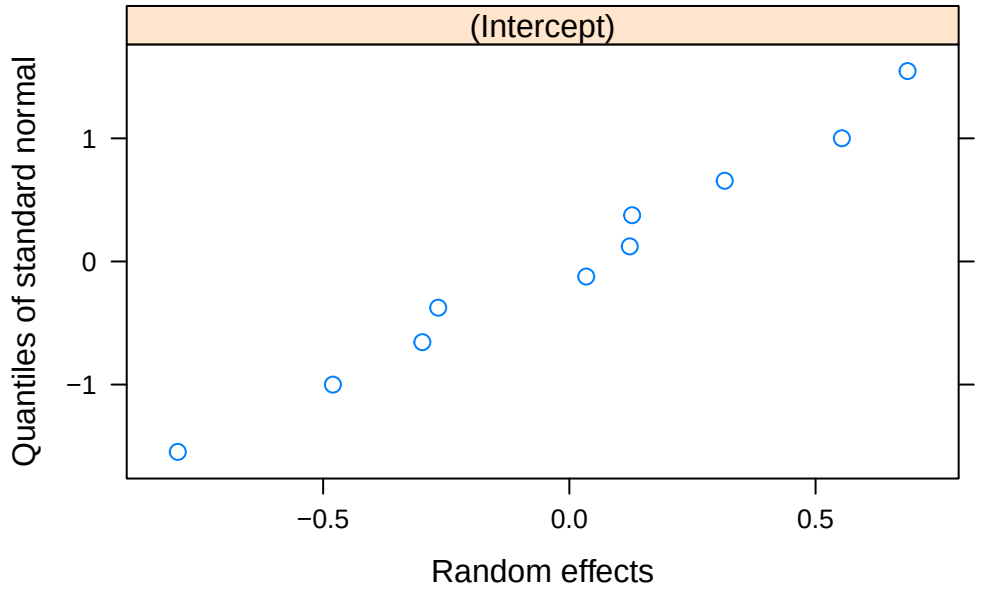

```
qqnorm(model, ~ residuals(.,type="pearson"))
```



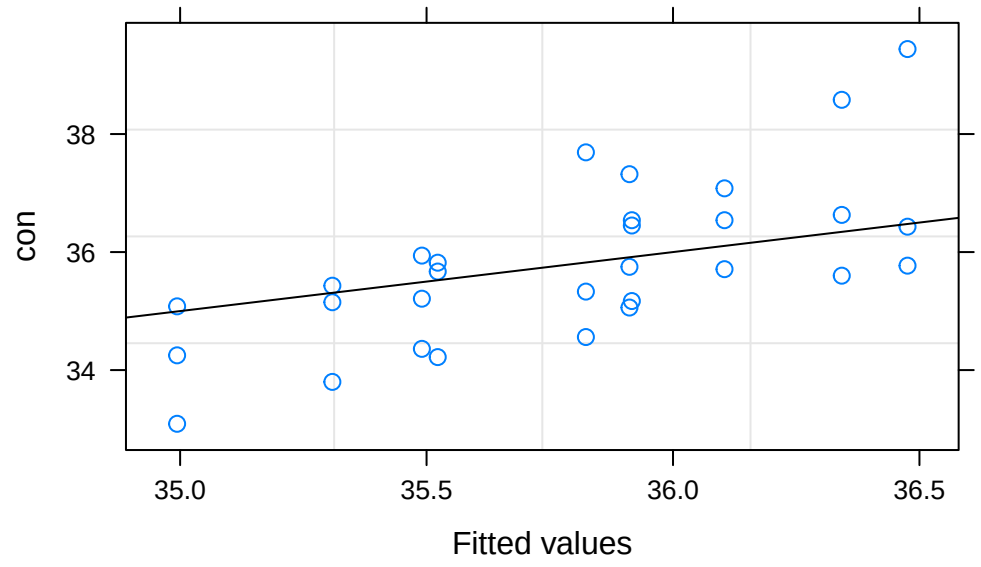
```
plot(ranef(model))
```




```
qqnorm(model, ~ ranef(.))
```




```
plot(model, con ~ fitted(.),abline=c(0,1))
```

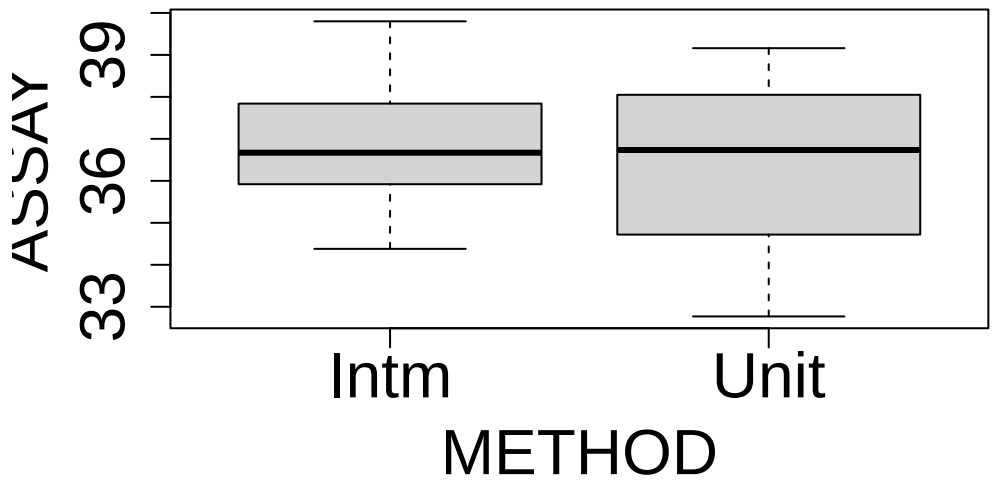


```r
fit_null<-lm(con ~ 1 , data=tablet)


confint(fit_null)
```

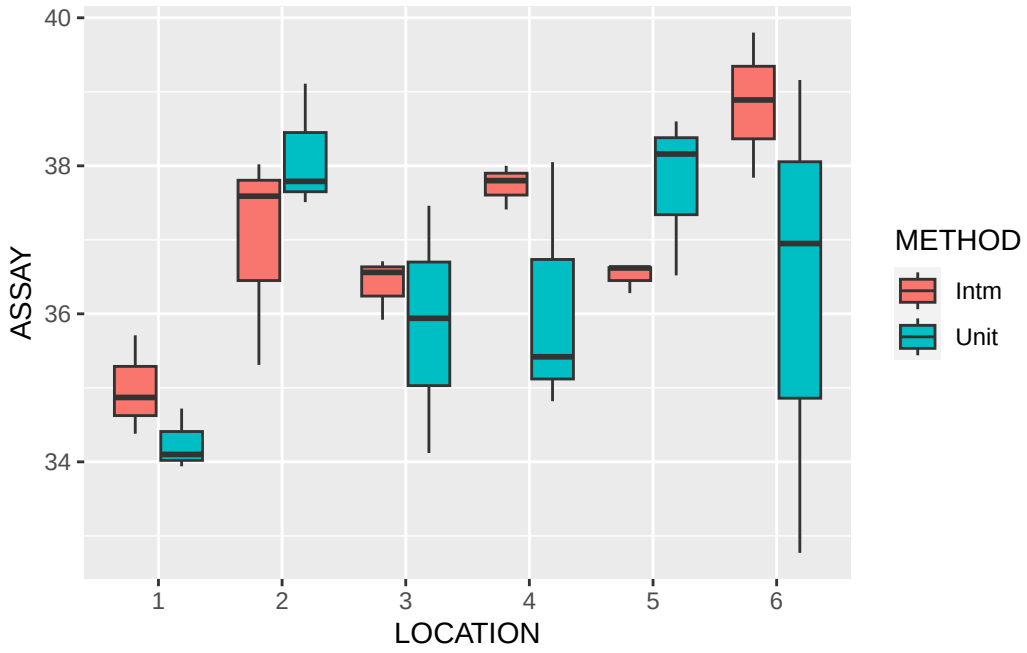
```
                2.5 %   97.5 %
(Intercept) 35.28119 36.29681
```

Let's tackle the next question: which sampling method is better? – which sampling method has a lower variability?

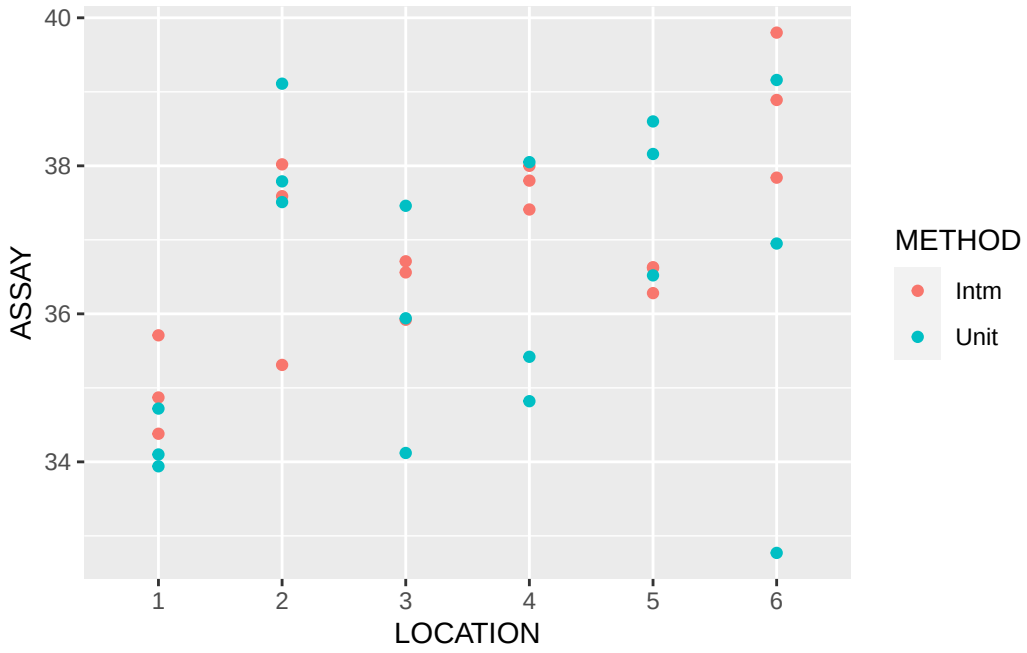
```r
library(ggplot2)
par(cex.lab=2,cex.axis=2,mfrow=c(1,1))
boxplot(ASSAY~METHOD,data=thief)
```




```r
thief$LOCATION=as.factor(thief$LOCATION)
e <- ggplot(thief, aes(x = LOCATION, y=ASSAY,fill=METHOD)) + geom_boxplot()
e
```



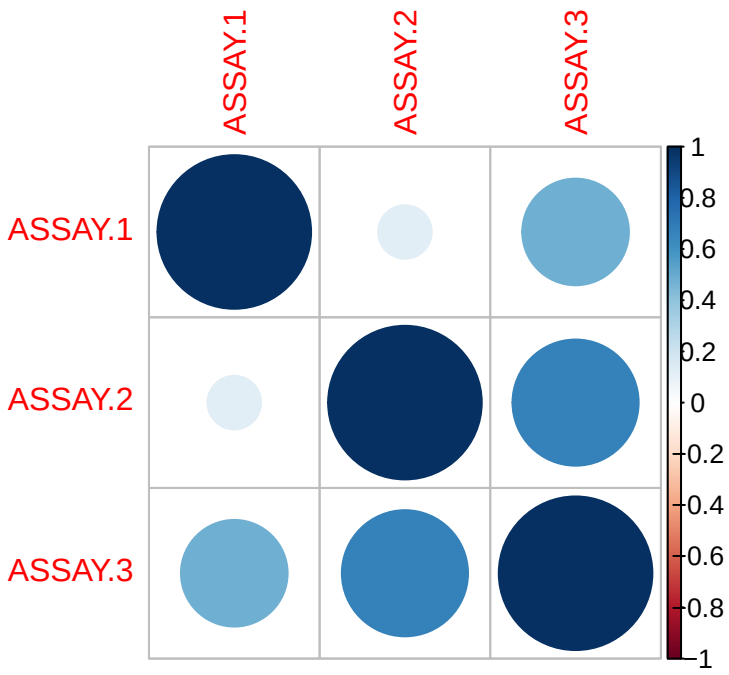

```
thief$LOCATION=as.factor(thief$LOCATION)
thief$METHOD=as.factor(thief$METHOD)
e <- ggplot(thief, aes(x = LOCATION, y=ASSAY, color=METHOD)) + geom_point()
e
```



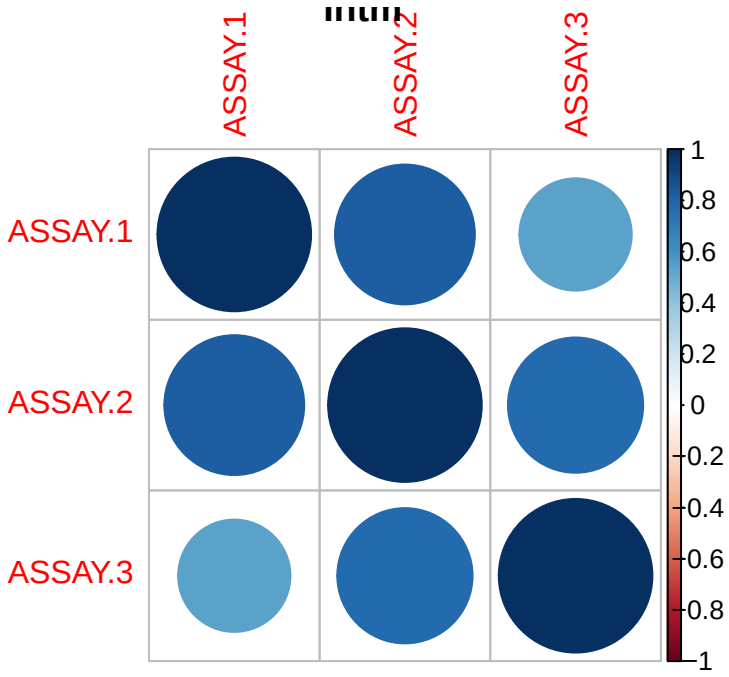

```
# boxplot(ASSAY ~ LOCATION, col=as.numeric(thief$METHOD),data=thief)
thief_wide=reshape(thief[,1:4],timevar='REPLICATE', idvar=c('LOCATION','METHOD'),  directi
head(thief_wide)
```

```
   METHOD LOCATION ASSAY.1 ASSAY.2 ASSAY.3
1    Intm        1   34.38   34.87   35.71
4    Intm        2   35.31   37.59   38.02
7    Intm        3   36.71   36.56   35.92
10   Intm        4   37.80   37.41   38.00
13   Intm        5   36.28   36.63   36.62
16   Intm        6   38.89   39.80   37.84
```

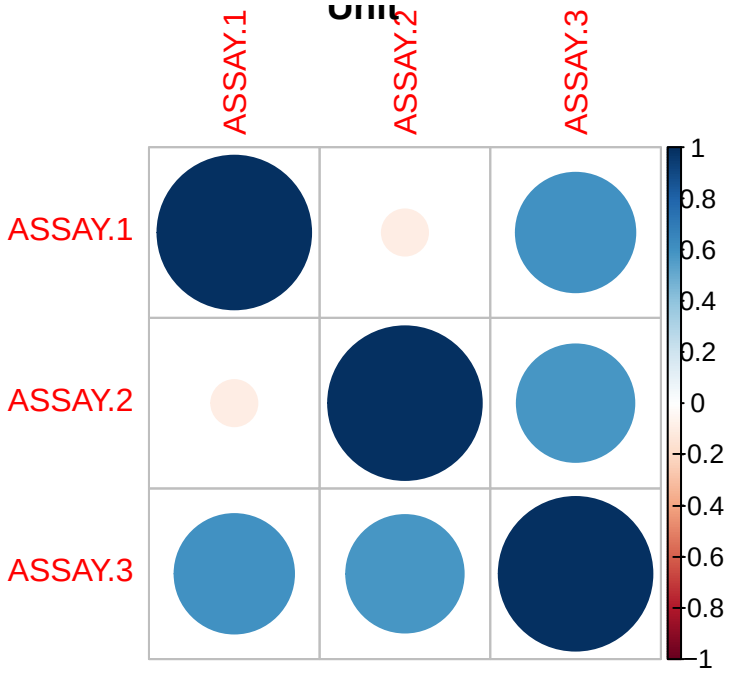
```
corrplot::corrplot(cor(thief_wide[,3:5]))
```




```
corrplot::corrplot(cor(thief_wide[1:6,3:5]),main="Intm")
```



```
corrplot::corrplot(cor(thief_wide[7:12,3:5]),main="Unit")
```



- We have 6 observations per location, three for each sampling type
- Each location could have its own mean concentration – see EDA $\mu + \beta_i$
- We expect that the assays within each sampling type will vary around their location means - β_i
- Nested within the locations is the replicates , 3 per sampling method
- We can capture the variability of sampling at each location via the random effect location.
- Let i be the location number, j be the sampling type and k be the replicate number

$$Y_{ijk} = \mu + \beta_i + \alpha_j + \epsilon_{ijk}.$$

We can also write this as follows:

- Let i be the location number and j be the sampling type
- Let $W_{ij} = (1, j-1)$
- Let $Z_{ij} = 1$ with $\beta_i \sim N(0, \Sigma_\beta)$.

$$Y_{ij} = W_{ij}\alpha + Z_{ij}\beta_i + \epsilon_{ij}.$$

```r
thief=thief[,-5]
thief$REPLICATE=as.factor(thief$REPLICATE)
thief$LOCATION=as.factor(thief$LOCATION)
thief$METHOD=as.factor(thief$METHOD)
print(thief$METHOD)
```

```
 [1] Intm Intm Intm Intm Intm Intm Intm Intm Intm Intm Intm Intm Intm Intm Intm
[16] Intm Intm Intm Unit Unit Unit Unit Unit Unit Unit Unit Unit Unit Unit Unit
[31] Unit Unit Unit Unit Unit Unit
Levels: Intm Unit
```

```r
#Unit is True
thief$ASSAY=as.numeric(thief$ASSAY)
thief
```

```
   METHOD LOCATION REPLICATE ASSAY
1    Intm        1         1 34.38
2    Intm        1         2 34.87
3    Intm        1         3 35.71
4    Intm        2         1 35.31
5    Intm        2         2 37.59
6    Intm        2         3 38.02
7    Intm        3         1 36.71
8    Intm        3         2 36.56
9    Intm        3         3 35.92
10   Intm        4         1 37.80
11   Intm        4         2 37.41
12   Intm        4         3 38.00
13   Intm        5         1 36.28
14   Intm        5         2 36.63
15   Intm        5         3 36.62
16   Intm        6         1 38.89
17   Intm        6         2 39.80
18   Intm        6         3 37.84
19   Unit        1         1 33.94
20   Unit        1         2 34.72
21   Unit        1         3 34.10
22   Unit        2         1 39.11
23   Unit        2         2 37.51
24   Unit        2         3 37.79
```

```
25   Unit        3         1 37.46
26   Unit        3         2 34.12
27   Unit        3         3 35.94
28   Unit        4         1 38.05
29   Unit        4         2 34.82
30   Unit        4         3 35.42
31   Unit        5         1 36.52
32   Unit        5         2 38.60
33   Unit        5         3 38.16
34   Unit        6         1 39.16
35   Unit        6         2 32.77
36   Unit        6         3 36.95
```

```
model=lme(
  fixed= ASSAY ~ METHOD ,
  random=  ~ 1|LOCATION, data=thief)
summary(model)
```

```
Linear mixed-effects model fit by REML
  Data: thief
       AIC      BIC   logLik
  142.2532 148.3586 -67.1266

Random effects:
 Formula: ~1 | LOCATION
        (Intercept) Residual
StdDev:   0.9596085 1.456579

Fixed effects:  ASSAY ~ METHOD
               Value Std.Error DF  t-value p-value
(Intercept) 36.90778 0.5209056 29 70.85310  0.0000
METHODUnit  -0.51111 0.4855263 29 -1.05269  0.3012
 Correlation:
           (Intr)
METHODUnit -0.466

Standardized Within-Group Residuals:
        Min          Q1         Med          Q3         Max
-2.94429462 -0.46995636  0.02331003  0.53623214  1.53118448

Number of Observations: 36
Number of Groups: 6
```

```
  print(ranef(model))
```

```
  (Intercept)
1  -1.4683710
2   0.6522971
3  -0.3857585
4   0.1910729
5   0.3488284
6   0.6619311
```

```
  #Step 1. Computing T-hat
  fit_null<-lm(ASSAY ~ 1 , data=thief)

  observed=lmtest::lrtest(fit_null,model)$Chisq[2]; observed
```

```
Warning in modelUpdate(objects[[i - 1]], objects[[i]]): original model was of
class "lm", updated model is of class "lme"
```

```
[1] 5.442094
```

```
  #Step 2.

  #Get the fixed effects
  fe=nlme::fixed.effects(model); fe
```

```
(Intercept)  METHODUnit
 36.9077778  -0.5111111
```

```
  #Get the estimated variance of the RE
  sigma_b_est= nlme::getVarCov(model); sigma_b_est
```

```
Random effects variance covariance matrix
            (Intercept)
(Intercept)     0.92085
  Standard Deviations: 0.95961
```

```r
#Get the estimate of sigma
sigma_est=model$sigma; sigma_est
```

```
[1] 1.456579
```

```r
n=nrow(thief)
n_sim=100


#Step 2
simulated=t(replicate(n_sim,rnorm(n,fe[1],sigma_est))); dim(simulated)
```

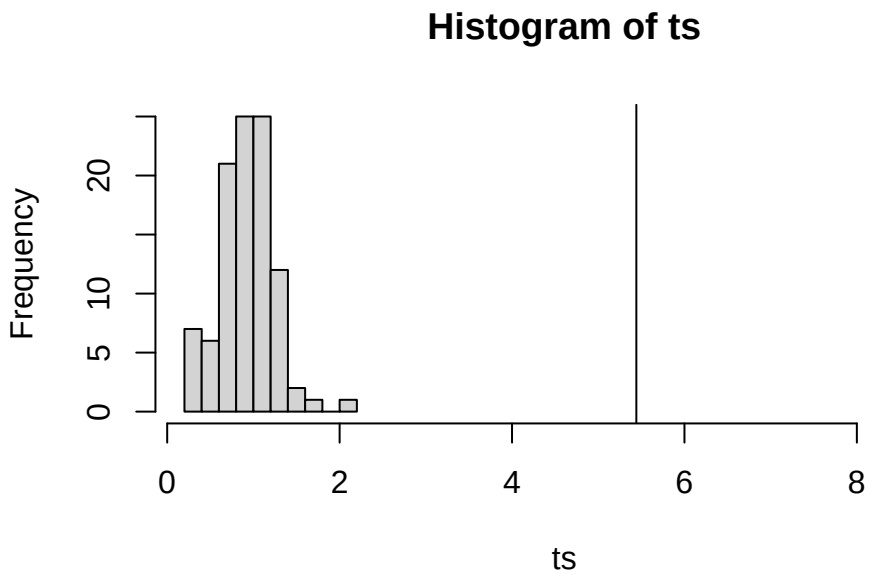
```
[1] 100  36
```

```r
#Step 3
#takes a simulated sample y and computes the LRT for Y
compute_lrt=function(y){

  #create a copy of the dataset
  t_copy=thief

  #replace response with new sample
  t_copy$ASSAY=y

  #replace response with new sampl
  alt=lme(
      fixed= ASSAY~1,
      random=ASSAY ~ 1 | LOCATION, data=t_copy )

  null<-lm(ASSAY ~ 1 , data=t_copy)

  test=lmtest::lrtest(null,alt)$Chisq[2]


  return(test)
}

#compute LRT for each simulated sample
ts=suppressWarnings(apply(simulated, 1, compute_lrt))
```

```
pvalue=mean(observed<=ts); pvalue
```

```
[1] 0
```

```
hist(ts,xlim=c(min(ts),9))
abline(v=observed)
```




```
observed
```

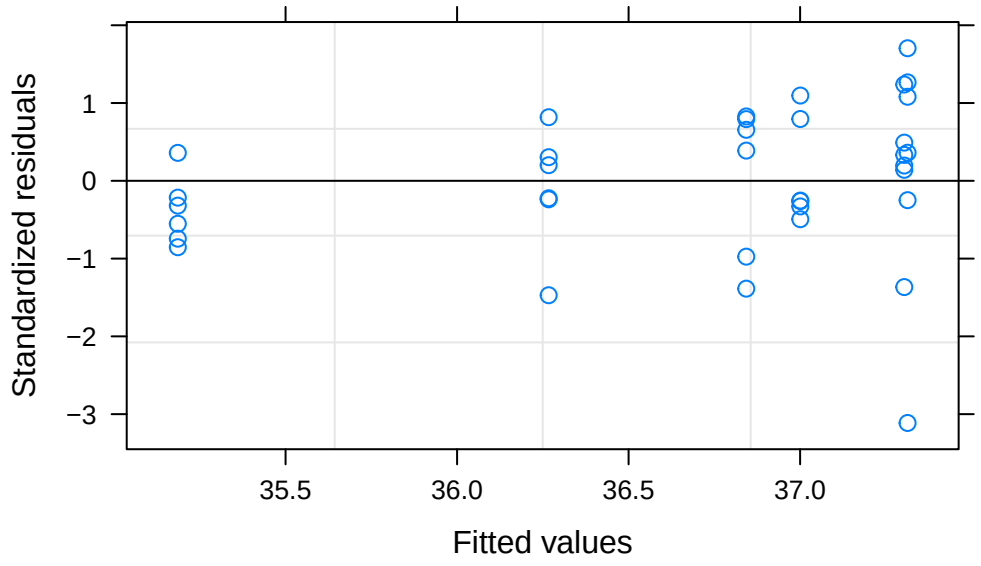
```
[1] 5.442094
```

So then, the location effect introduces significant variability. We should recommend that they continue to sample multiple locations. The unit dose thief seems to estimate lower values of concentration, though this is erased by the standard error. Now, in general, we can see that this method is more variable, and potentially tends to underestimate the active ingredient. This fact leads me to recommend the intermediate sampling method.

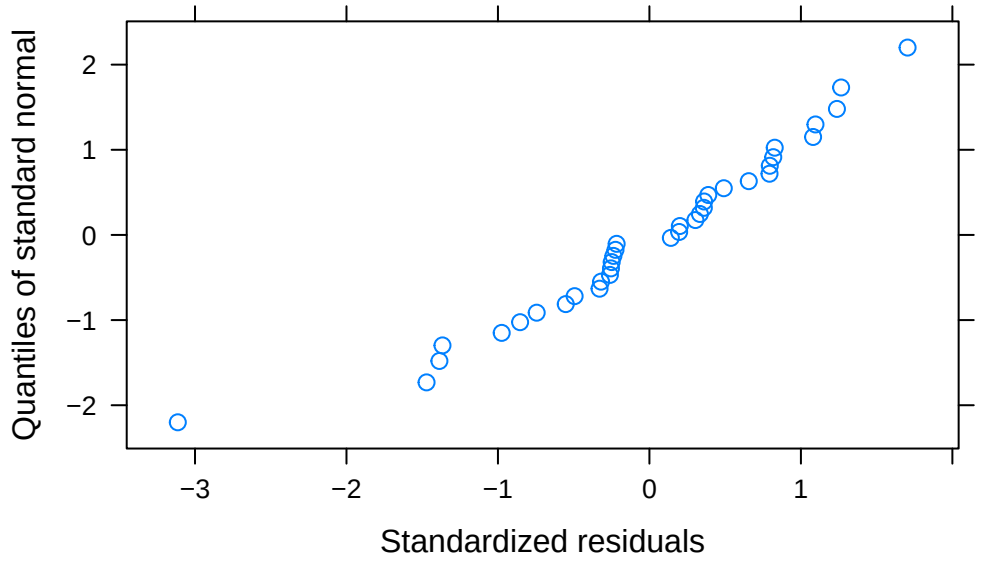
- Are the assay values generally well behaved?

- Is there any evidence of a location effect
- Are the thief-sampled values comparable to the tablet values?
- Do the tablet data show any drum or time effect?

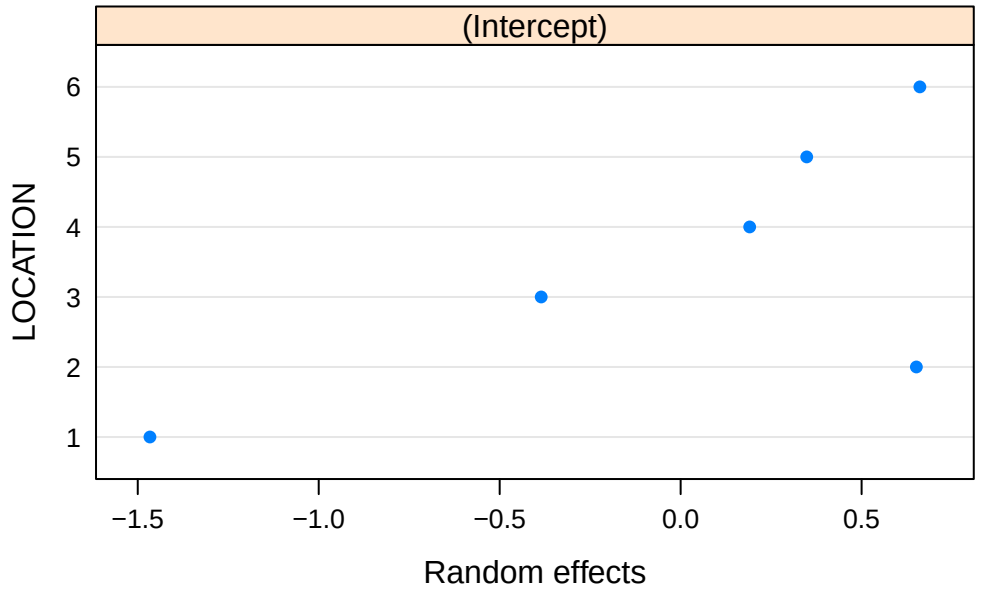
```
model=lme(
      fixed= ASSAY~1,
      random= ASSAY ~ 1 | LOCATION , data=thief)


plot(model)
```



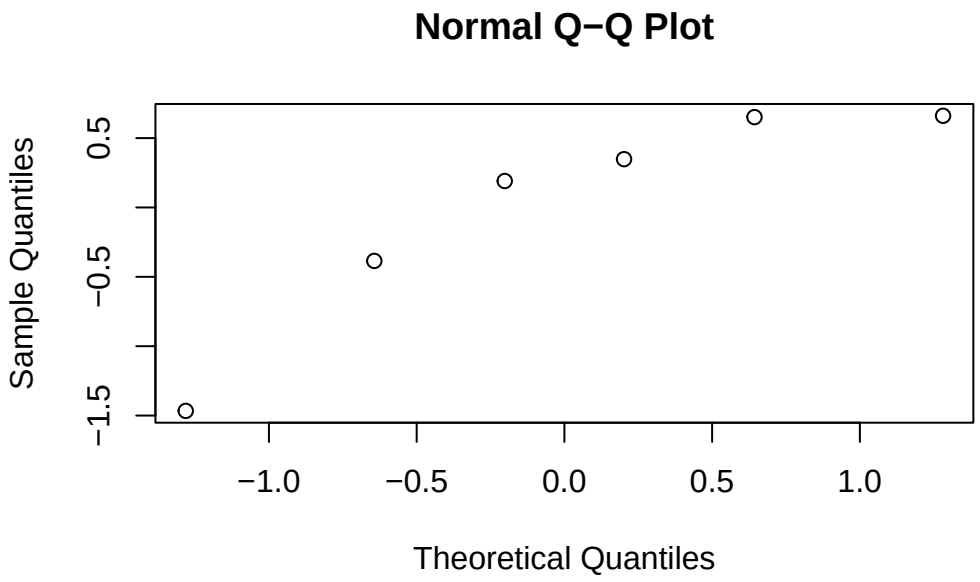

```
qqnorm(model, ~ residuals(.,type="pearson"))
```



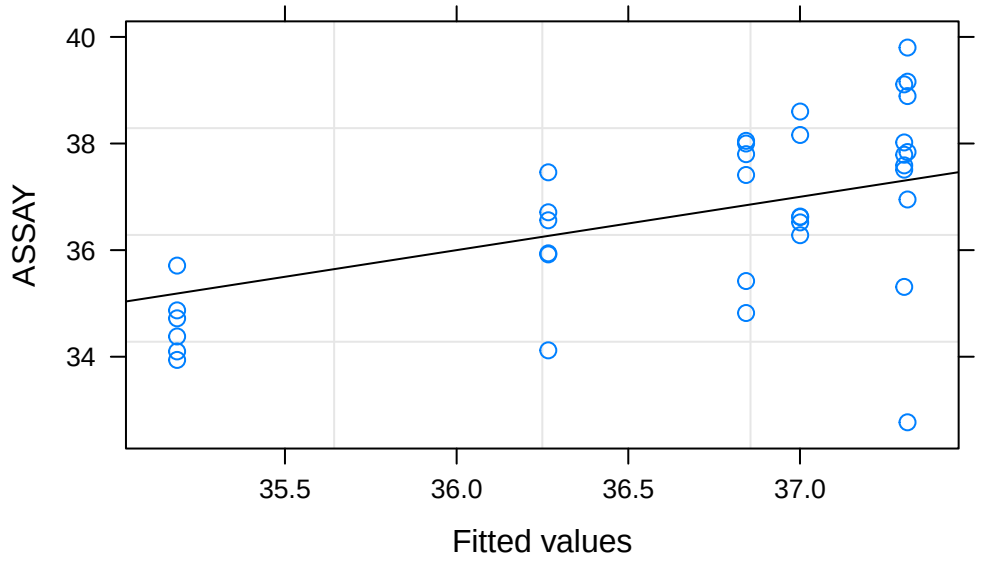

```
plot(ranef(model))
```



```
qqnorm(ranef(model)[,1])
```




```
plot(model, ASSAY ~ fitted(.),abline=c(0,1))
```




```
# fit_null<-lm(ASSAY ~ 1 , data=thief)
# confint(fit_null)
# fit_null
```

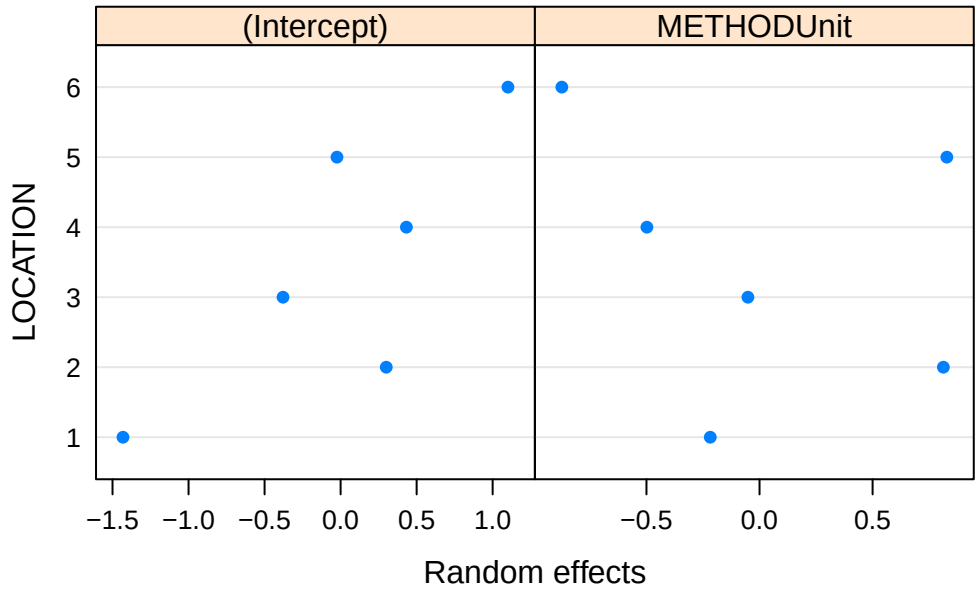
We can also fit a random effect for each of the methods and locations:

```
model=lme(
     fixed= ASSAY~METHOD,
     random= ASSAY ~ METHOD | LOCATION , data=thief)
summary(model)
```

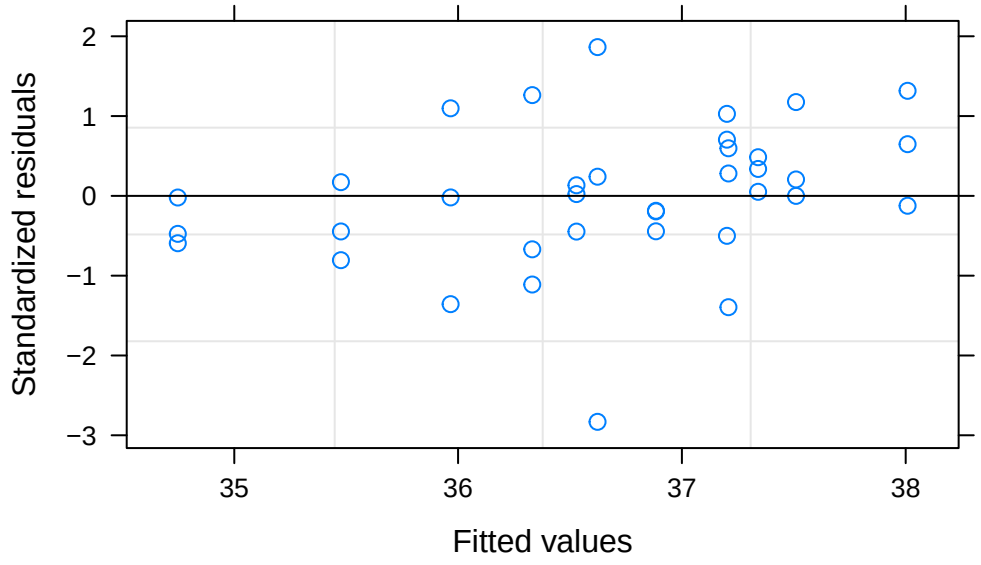
```
Linear mixed-effects model fit by REML
  Data: thief
       AIC      BIC    logLik
  145.3404 154.4986 -66.67022

Random effects:
 Formula: ASSAY ~ METHOD | LOCATION
 Structure: General positive-definite, Log-Cholesky parametrization
            StdDev   Corr
(Intercept) 1.045122 (Intr)
METHODUnit  1.021304 -0.363
```

```
Residual    1.360833

Fixed effects:  ASSAY ~ METHOD
               Value Std.Error DF  t-value p-value
(Intercept) 36.90778 0.5337867 29 69.14331  0.0000
METHODUnit  -0.51111 0.6161222 29 -0.82956  0.4136
 Correlation:
           (Intr)
METHODUnit -0.509

Standardized Within-Group Residuals:
       Min         Q1        Med         Q3        Max
-2.8314642 -0.4546280  0.0113704  0.5126063  1.8641882

Number of Observations: 36
Number of Groups: 6
```

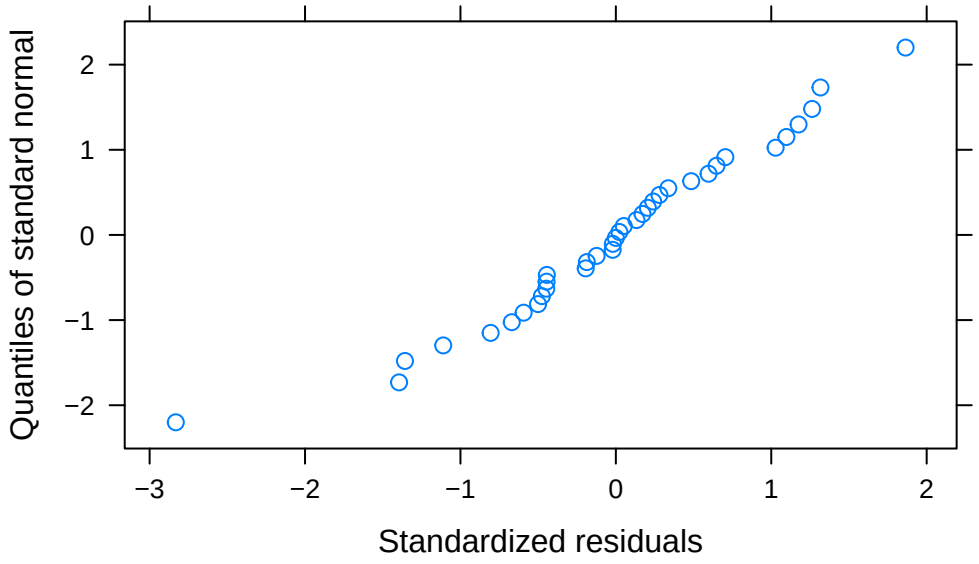
```r
# We can see that the variability coming from each source is somewhat large.


plot(ranef(model))
```



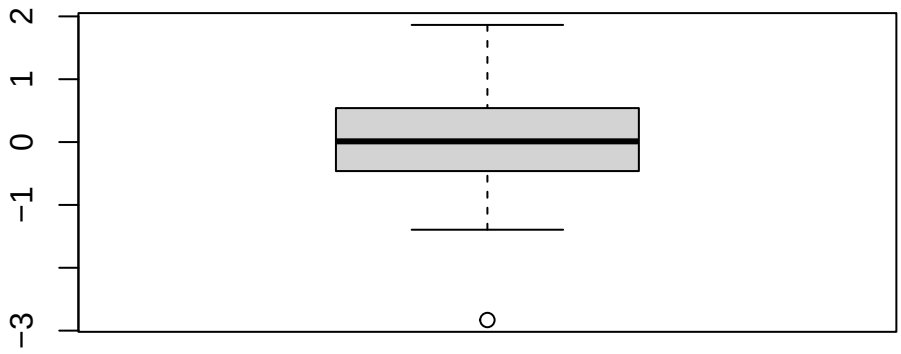
```r
plot(model)
```




```r
qqnorm(model, ~residuals(., type = "normalized"))
```




```
boxplot(residuals(model, type = "normalized"))
```



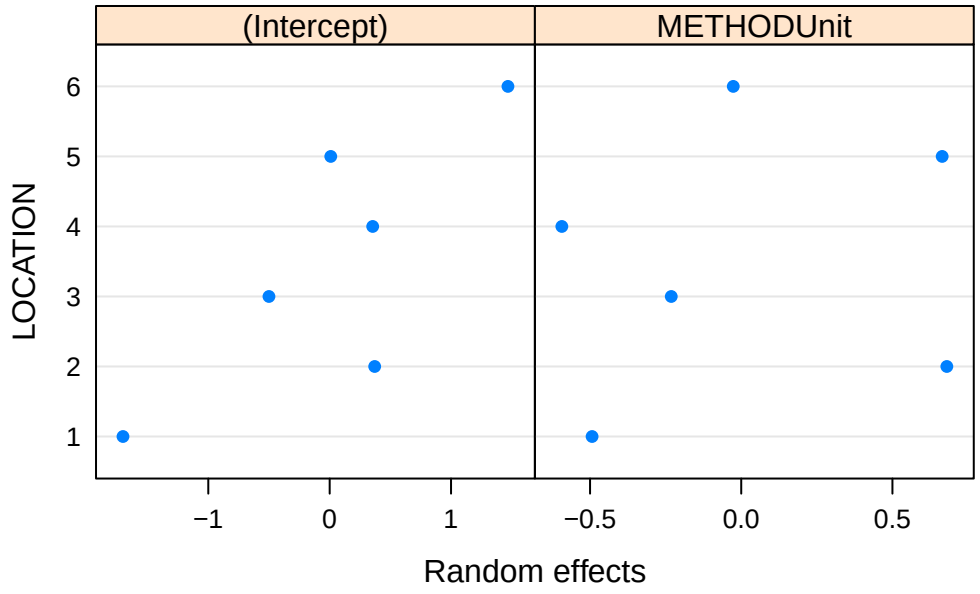
```
large=which.max(abs(residuals(model, type = "normalized"))); large
```

```
 6
35
```

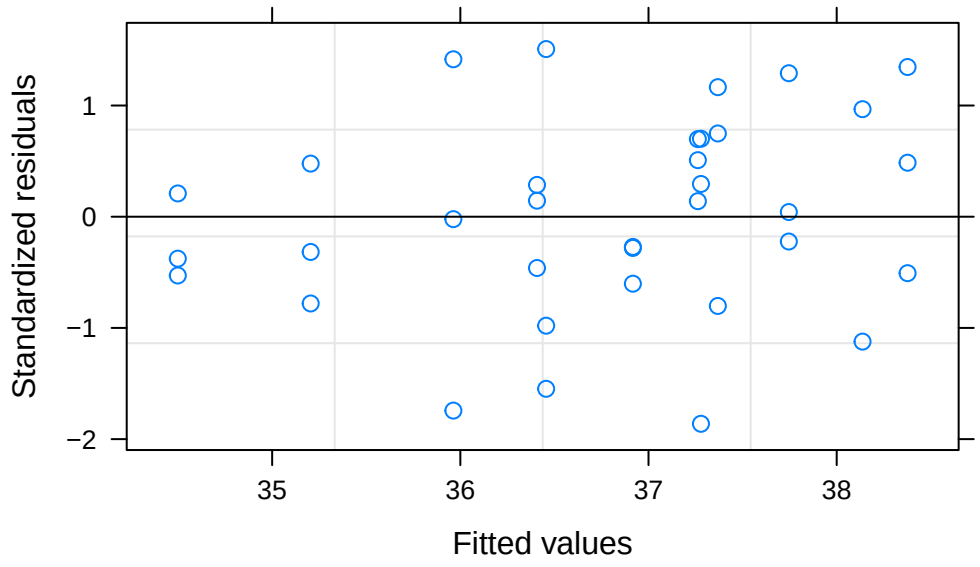
```
model=lme(
     fixed= ASSAY~METHOD,
     random= ASSAY ~ METHOD | LOCATION , data=thief[-large,])
summary(model)
```

```
Linear mixed-effects model fit by REML
  Data: thief[-large, ]
       AIC      BIC   logLik
  128.7654 137.7445 -58.3827

Random effects:
 Formula: ASSAY ~ METHOD | LOCATION
 Structure: General positive-definite, Log-Cholesky parametrization
            StdDev    Corr
(Intercept) 1.1562644 (Intr)
METHODUnit  0.7965863 0.066
Residual    1.0572951

Fixed effects:  ASSAY ~ METHOD
               Value Std.Error DF  t-value p-value
(Intercept) 36.90778 0.5337871 28 69.14326  0.0000
METHODUnit  -0.21293 0.4843533 28 -0.43961  0.6636
 Correlation:
           (Intr)
METHODUnit -0.201

Standardized Within-Group Residuals:
       Min         Q1        Med         Q3        Max
-1.8620371 -0.5179546  0.0419608  0.6032329  1.5075857

Number of Observations: 35
Number of Groups: 6
```

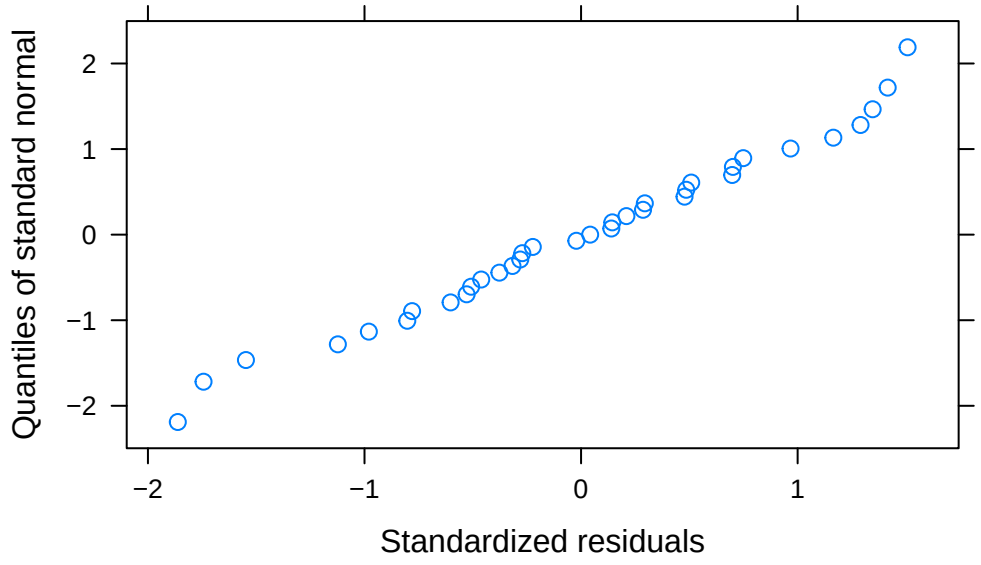
```
plot(ranef(model))
```


(Intercept)
METHODUnit
LOCATION
6
5
4
3
2
1
−1
0
1
−0.5
0.0
0.5
Random effects

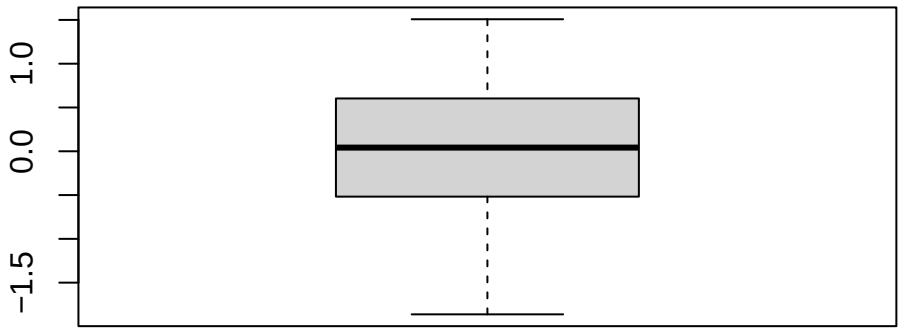
```
plot(model)
```


Standardized residuals
1
0
−1
−2
35
36
37
38
Fitted values

```
qqnorm(model, ~residuals(., type = "normalized"))
```




```
boxplot(residuals(model, type = "normalized"))
```



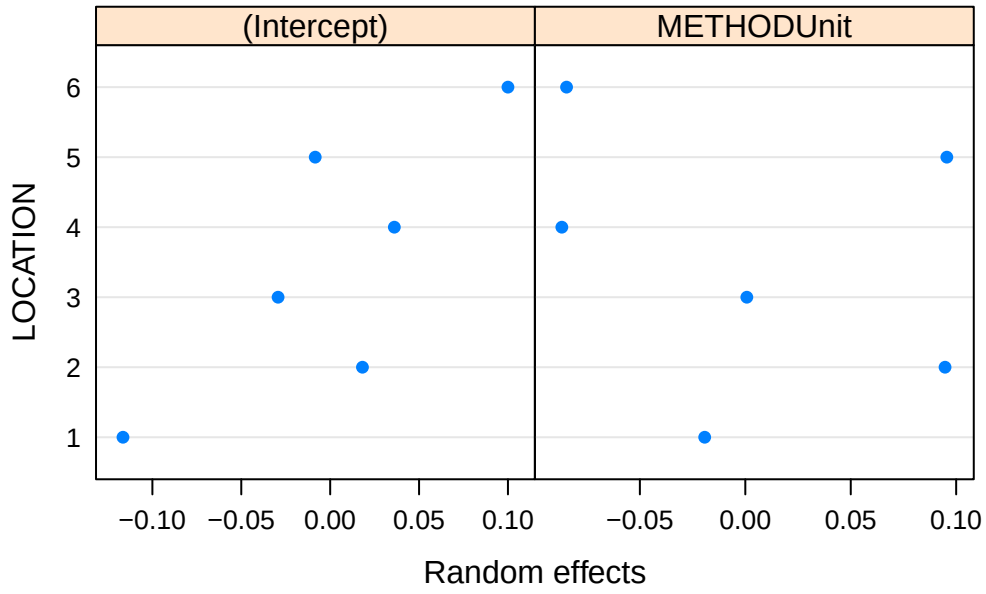

```
large=which.max(residuals(model, type = "normalized"))
```

```
model=lme(
     fixed= sqrt(ASSAY)~METHOD,
     random= sqrt(ASSAY) ~ METHOD | LOCATION , data=thief[-large,])
summary(model)
```

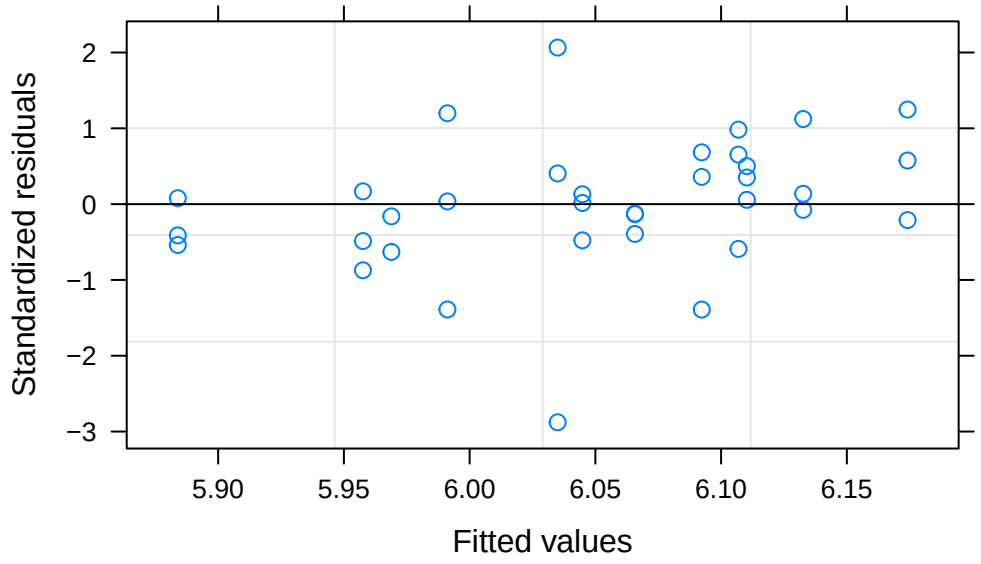
```
Linear mixed-effects model fit by REML
  Data: thief[-large, ]
        AIC       BIC   logLik
  -23.94854 -14.96949 17.97427

Random effects:
 Formula: ASSAY ~ METHOD | LOCATION
 Structure: General positive-definite, Log-Cholesky parametrization
            StdDev     Corr
(Intercept) 0.08767712 (Intr)
METHODUnit  0.10625944 -0.405
Residual    0.10789056

Fixed effects:  sqrt(ASSAY) ~ METHOD
```

```
                Value  Std.Error DF   t-value p-value
(Intercept)  6.074146 0.04390786 28 138.33847  0.0000
METHODUnit  -0.054387 0.05677613 28  -0.95792  0.3463
 Correlation:
            (Intr)
METHODUnit -0.511

Standardized Within-Group Residuals:
        Min          Q1         Med          Q3         Max
-2.87783191 -0.44462638  0.03546945  0.45330842  2.06501422

Number of Observations: 35
Number of Groups: 6
```

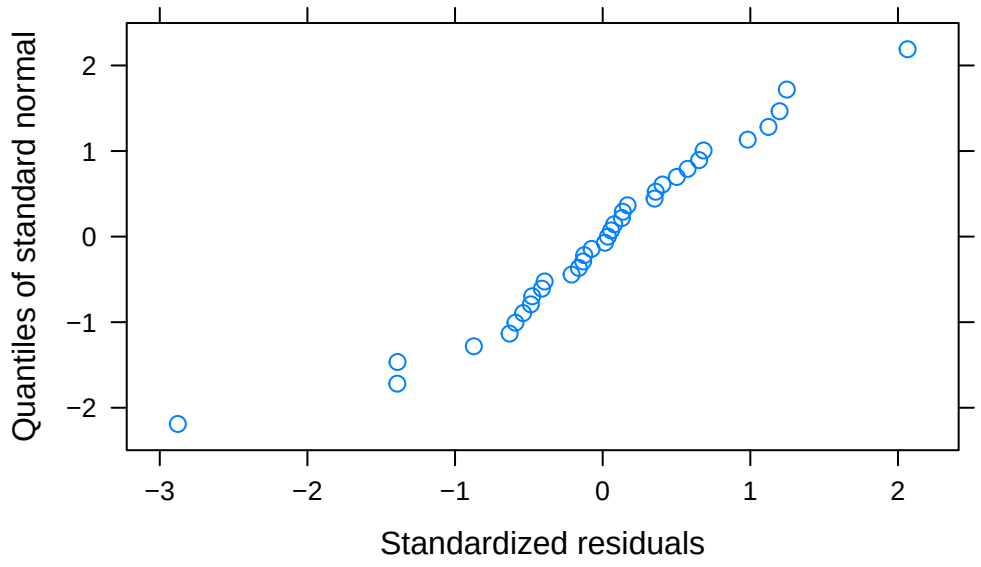
```
plot(ranef(model))
```



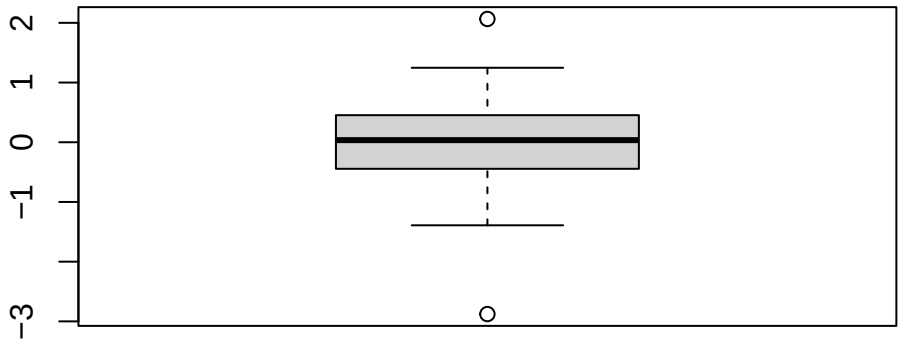

```
plot(model)
```




```
qqnorm(model, ~residuals(., type = "normalized"))
```



```
boxplot(residuals(model, type = "normalized"))
```




```
large=which.max(residuals(model, type = "normalized"))
```

```
thief$CEN=abs(thief$ASSAY-mean(thief$ASSAY))
model=lme(
  fixed= CEN ~ METHOD ,
  random=  ~ 1|LOCATION, data=thief)
summary(model)
```

```
Linear mixed-effects model fit by REML
  Data: thief
       AIC      BIC    logLik
  99.44747 105.5529 -45.72373

Random effects:
 Formula: ~1 | LOCATION
        (Intercept)  Residual
StdDev:   0.5372663 0.7715252
```

```
Fixed effects:  CEN ~ METHOD
                Value Std.Error DF  t-value p-value
(Intercept) 1.0988889 0.2849187 29 3.856850  0.0006
METHODUnit  0.5922222 0.2571751 29 2.302798  0.0287
 Correlation:
           (Intr)
METHODUnit -0.451

Standardized Within-Group Residuals:
        Min          Q1         Med          Q3         Max
-2.59227946 -0.67605167 -0.04425782  0.47231782  2.05364082

Number of Observations: 36
Number of Groups: 6
```

```
print(ranef(model))
```

```
  (Intercept)
1  0.47423228
2 -0.03335199
3 -0.42613374
4 -0.08117489
5 -0.54024717
6  0.60667552
```

```
hist(thief$CEN)
```


```
# Non-normal (of course... maybe use simulation)
```

Results summary:

- Tablet mean concentration was estimated to be 35.8 with CI (35.3, 36.3)
- Active ingredient was uniform across blender and sampling types
- Tablets in the drums on the end points seem to have a higher active ingredient
- The blender has higher concentrations of active ingredient 36.65 with CI (36.0743, 37.23015)
- The location has significant variance (~1 mg)
- Sampling methods are equivalent in mean, but UNIT seems to have higher variance. Possibly produce smaller estimates of the concentration, but this is not statistically significant

1.6 Case study: Treatment of Lead-Exposed Children

Does treatment A (chelation treatment with succimer) affect the levels of lead in the blood of lead-exposed children?

Data source or Data source

Description:

The Treatment of Lead-Exposed Children (TLC) trial was a placebo-controlled,randomized study of succimer (a chelating agent) in children with blood lead levels of 20-44 micrograms/dL. These data consist of four repeated measurements of blood lead levels obtained at baseline (or week 0), week 1, week 4, and week 6 on 100 children who were randomly assigned to chelation treatment with succimer or placebo.

Data Column Descriptions:

- ID: Subject ID Number
- Treatment: Which treatment group (P=Placebo; A=Succimenr)
- W0, W1, W4, W6: Blood-lead levels in micrograms per deciliter at Weeks 0, 1, 4, and 6

```
##############################################################################
TLC <- read.csv("data/TLC.csv",stringsAsFactors = T)
head(TLC)
```

```
  ID Treatment   W0   W1   W4   W6
1  1         P 30.8 26.9 25.8 23.8
2  2         A 26.5 14.8 19.5 21.0
3  3         A 25.8 23.0 19.1 23.2
4  4         P 24.7 24.5 22.0 22.5
5  5         A 20.4  2.8  3.2  9.4
6  6         A 20.4  5.4  4.5 11.9
```

```
dim(TLC)
```

```
[1] 100   6
```

```
TLC$ID=as.factor(TLC$ID)
TLC$ID=as.factor(TLC$ID)
```

1.6.1 Modelling:

- Recall that the goal is to answer: "Does treatment A affect the levels of lead in the blood of lead-exposed children?"
- How are treatment group and time related to lead levels?
- Given treatment and time, what do we expect the blood levels to be $E(W|Treatment, Time)$?

Putting this data into our notation gives:

- $n = 100$, $J = 4$, $K = 1$

- Y_{ij} is the lead level of individual i at time j
- X_{ij} is treatment indicator of individual i at time j
- $t_j \in \{0, 1, 4, 6\}$

Let's explore this data a bit. Notice that the data is in "wide format". To convert to a between long and wide format use `reshape()`

```
head(TLC)
```

```
  ID Treatment   W0   W1   W4   W6
1  1         P 30.8 26.9 25.8 23.8
2  2         A 26.5 14.8 19.5 21.0
3  3         A 25.8 23.0 19.1 23.2
4  4         P 24.7 24.5 22.0 22.5
5  5         A 20.4  2.8  3.2  9.4
6  6         A 20.4  5.4  4.5 11.9
```

```
# Convert from "wide" to "long" and back again, using reshape.
# If you're interested, you can also use `pivot_wider` and `pivot_longer` from the tidyver
#   (If that doesn't mean anything to you, feel free to ignore it!)
TLC_long <- reshape(data = TLC,
                    varying = c("W0", "W1", "W4", "W6"),
                    timevar = "week",
                    idvar = "ID",
                    times = c(0, 1, 4, 6),
                    direction = "long",
                    sep = "")
head(TLC_long)
```

```
    ID Treatment week    W
1.0  1         P    0 30.8
2.0  2         A    0 26.5
3.0  3         A    0 25.8
4.0  4         P    0 24.7
5.0  5         A    0 20.4
6.0  6         A    0 20.4
```

```
TLC_wide <- reshape(data = TLC_long,
                    timevar = "week",
                    v.names = "W",
```

```
                    idvar = "ID",
                    times = c(0, 1, 4, 6),
                    direction = "wide",
                    sep = "")

head(TLC_wide)
```

```
    ID Treatment   W0   W1   W4   W6
1.0  1         P 30.8 26.9 25.8 23.8
2.0  2         A 26.5 14.8 19.5 21.0
3.0  3         A 25.8 23.0 19.1 23.2
4.0  4         P 24.7 24.5 22.0 22.5
5.0  5         A 20.4  2.8  3.2  9.4
6.0  6         A 20.4  5.4  4.5 11.9
```

```
#balanced
table(TLC_long$ID)
```

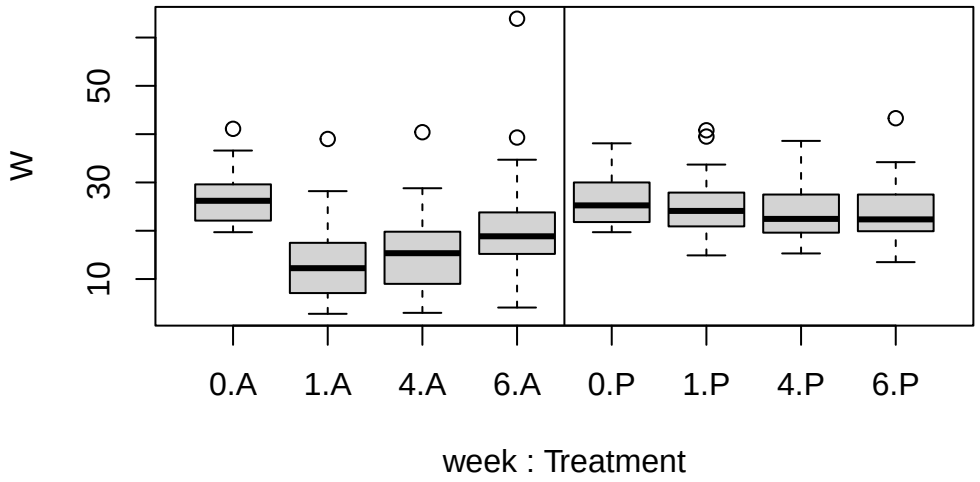
```
  1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20
  4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4
 21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40
  4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4
 41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60
  4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4
 61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80
  4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4
 81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99 100
  4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4
```

```
#Check for missing values
colSums(is.na(TLC_long))
```

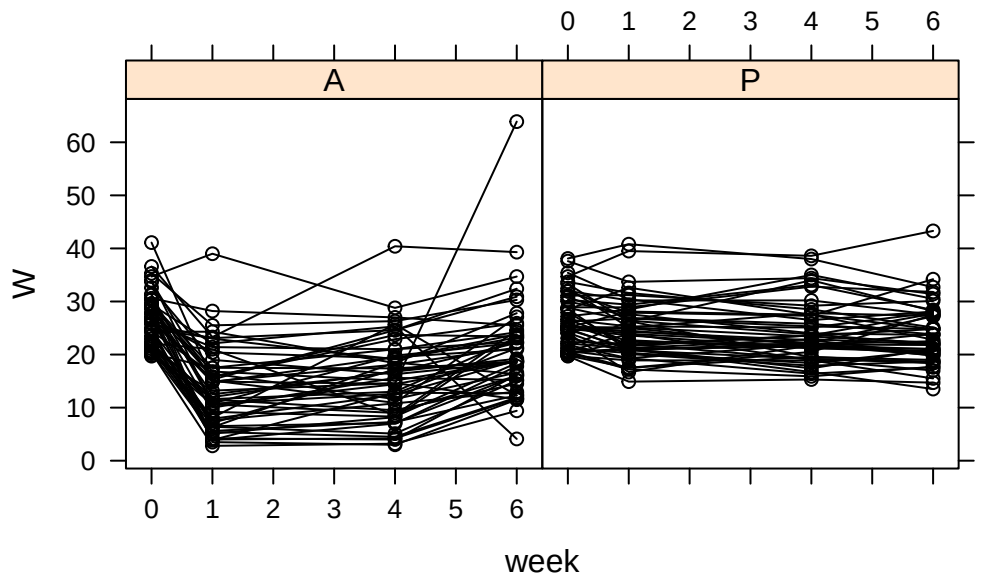
```
      ID Treatment        week           W
       0         0           0           0
```

```
# Create a Basic Boxplot to get a Sense of the Data
boxplot(W ~ week + Treatment, data = TLC_long)
```

```r
abline(v=4.5) # Abline v=... draws a vertical line at 4.5
```

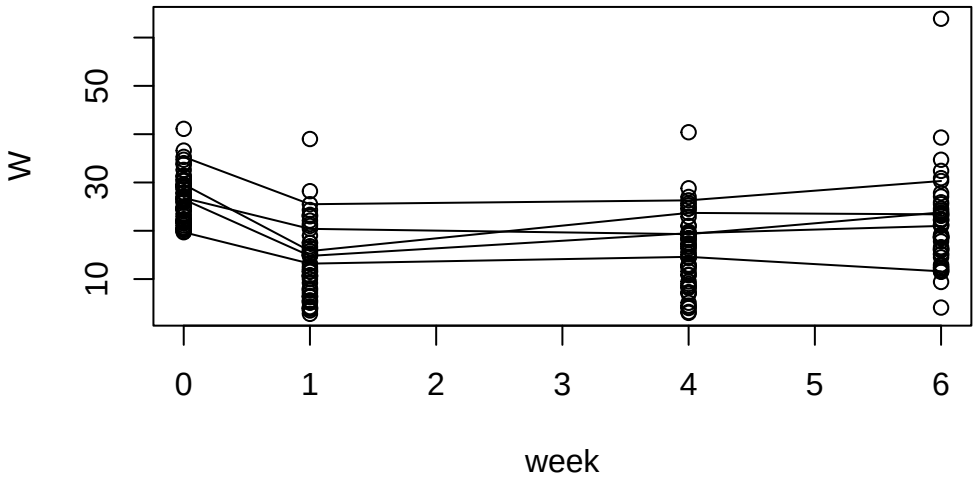



```r
# Start with an xyplot
# This requires the package 'lattice'
# You can install using: install.packages("lattice")

lattice::xyplot(W ~ week | Treatment,
                data = TLC_long,
                groups = ID,
                col = 'black',
                type = c('l', 'p'))
```

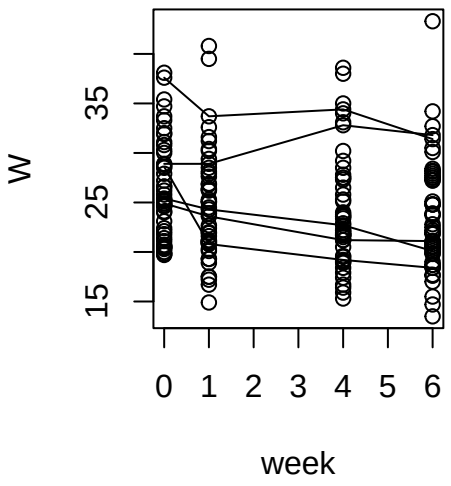



```
# The plot is a mess, as-is, so instead we can subset!
plot_num <- 5 # Select a fixed number

# This is Just Randomly Sampling from Each Group
random_samples_P <- sample(unique(TLC_long$ID[which(TLC_long$Treatment == 'P')]),
                           size = plot_num,
                           replace = FALSE)
random_samples_A <- sample(unique(TLC_long$ID[which(TLC_long$Treatment == 'A')]),
                             size = plot_num,
                             replace = FALSE)




## Actually Draw the Plots
plot(W ~ week, data = TLC_long, subset = (Treatment == 'A'))
for (rid in random_samples_A){
  lines(W ~ week,
        data = TLC_long,
        subset = (ID==rid),
        type = 'l')
```

```
}
```

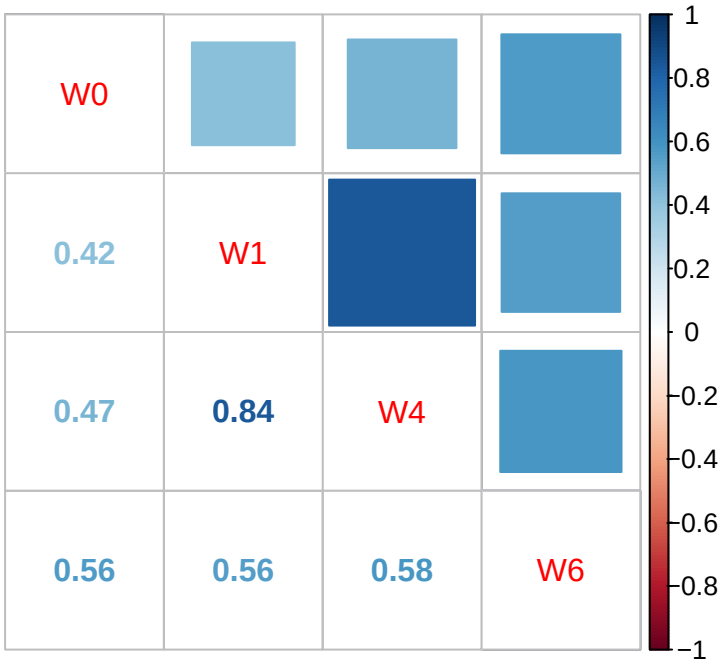



```
# Repeat it for Placebo

par(mfrow=c(1,2))
plot(W ~ week, data = TLC_long, subset = (Treatment == 'P'))
for (rid in random_samples_P){
  # Loop through the Random Points and Draw the Corresponding Lines
  lines(W ~ week,
        data = TLC_long,
        subset = (ID==rid),
        type = 'l')
}
```



Correlation plot

```
# This is a basic correlation plot
# It requires the 'corrplot' library, which can be installed with
# install.packages("corrplot")
corrplot::corrplot.mixed(cor(TLC_wide[c("W0","W1","W4","W6")]),
                         lower = 'number',
                         upper = 'square')
```

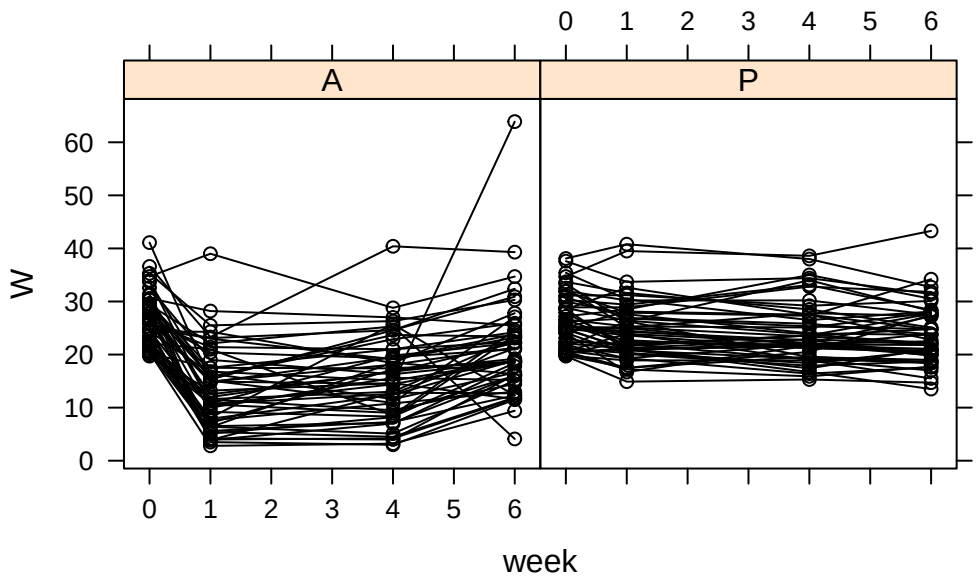


What can we conclude from the EDA? What model could we propose in this case?

1.6.2 Longitudinal Data as a mixed effects model

- Looking at the XY plots, we see that the individual means seem to vary.
- This means that each individual is likely to have a different mean
- The treatment effect should be modelled as a fixed effect
- How to model time effect?

```
lattice::xyplot(W ~ week | Treatment,
                data = TLC_long,
                groups = ID,
                col = 'black',
                type = c('l', 'p'))
```



What are some ways we can model the time effect here?

Let's fit a linear mixed effects model.

```
# head(TLC_long
# head(TLC_long[order(TLC_long$ID),])
typeof(TLC_long)
```

```
[1] "list"
```

```
#REML
#treat the time points as factors
TLC_long$week=as.factor(TLC_long$week)
# head(TLC_long)
model <- nlme::lme(fixed= W ~ 1 +Treatment+week+Treatment*week,
                   random= ~1|ID, data = TLC_long) #to run the model

summary(model)
```

```
Linear mixed-effects model fit by REML
  Data: TLC_long
```

```
       AIC      BIC    logLik
  2480.621 2520.334 -1230.311

Random effects:
 Formula: ~1 | ID
        (Intercept) Residual
StdDev:    5.112717 4.214287

Fixed effects:  W ~ 1 + Treatment + week + Treatment * week
                   Value Std.Error  DF    t-value p-value
(Intercept)       26.540 0.9370175 294  28.323912  0.0000
TreatmentP        -0.268 1.3251428  98  -0.202242  0.8401
week1            -13.018 0.8428574 294 -15.445080  0.0000
week4            -11.026 0.8428574 294 -13.081691  0.0000
week6             -5.778 0.8428574 294  -6.855252  0.0000
TreatmentP:week1  11.406 1.1919804 294   9.568950  0.0000
TreatmentP:week4   8.824 1.1919804 294   7.402807  0.0000
TreatmentP:week6   3.152 1.1919804 294   2.644339  0.0086
 Correlation:
                  (Intr) TrtmnP week1  week4  week6  TrtP:1 TrtP:4
TreatmentP        -0.707
week1             -0.450  0.318
week4             -0.450  0.318  0.500
week6             -0.450  0.318  0.500  0.500
TreatmentP:week1   0.318 -0.450 -0.707 -0.354 -0.354
TreatmentP:week4   0.318 -0.450 -0.354 -0.707 -0.354  0.500
TreatmentP:week6   0.318 -0.450 -0.354 -0.354 -0.707  0.500  0.500

Standardized Within-Group Residuals:
        Min          Q1         Med          Q3         Max
-4.18502705 -0.46501691 -0.04732964  0.36499878  7.66712566

Number of Observations: 400
Number of Groups: 100
```

```
nlme::intervals(model)
```

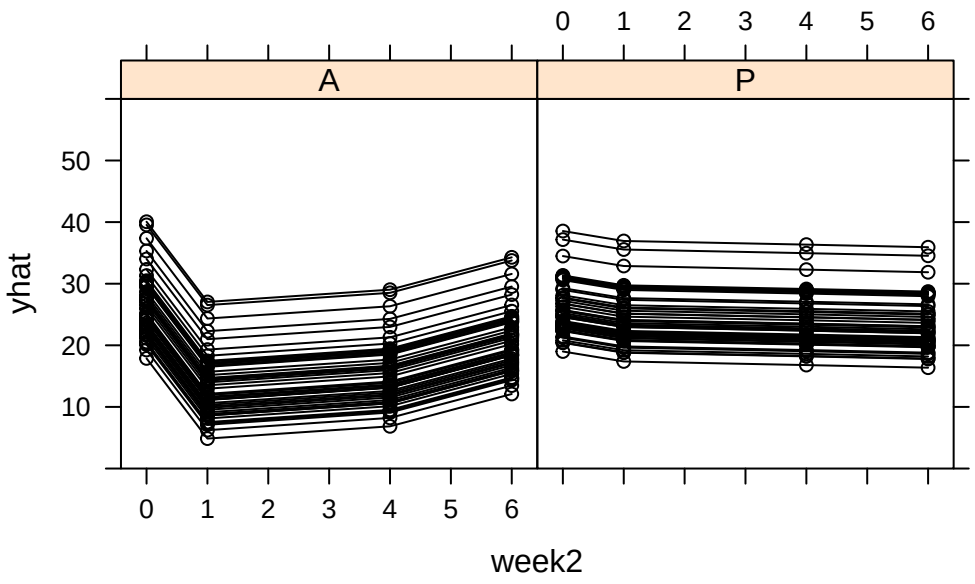
```
Approximate 95% confidence intervals

 Fixed effects:
                      lower    est.      upper
```

```
(Intercept)       24.6958881  26.540  28.384112
TreatmentP        -2.8977028  -0.268   2.361703
week1            -14.6767987 -13.018 -11.359201
week4            -12.6847987 -11.026  -9.367201
week6             -7.4367987  -5.778  -4.119201
TreatmentP:week1   9.0601043  11.406  13.751896
TreatmentP:week4   6.4781043   8.824  11.169896
TreatmentP:week6   0.8061043   3.152   5.497896

 Random Effects:
  Level: ID
                  lower     est.    upper
sd((Intercept)) 4.33785 5.112717 6.025997

 Within-group standard error:
   lower     est.    upper
3.887059 4.214287 4.569062
```

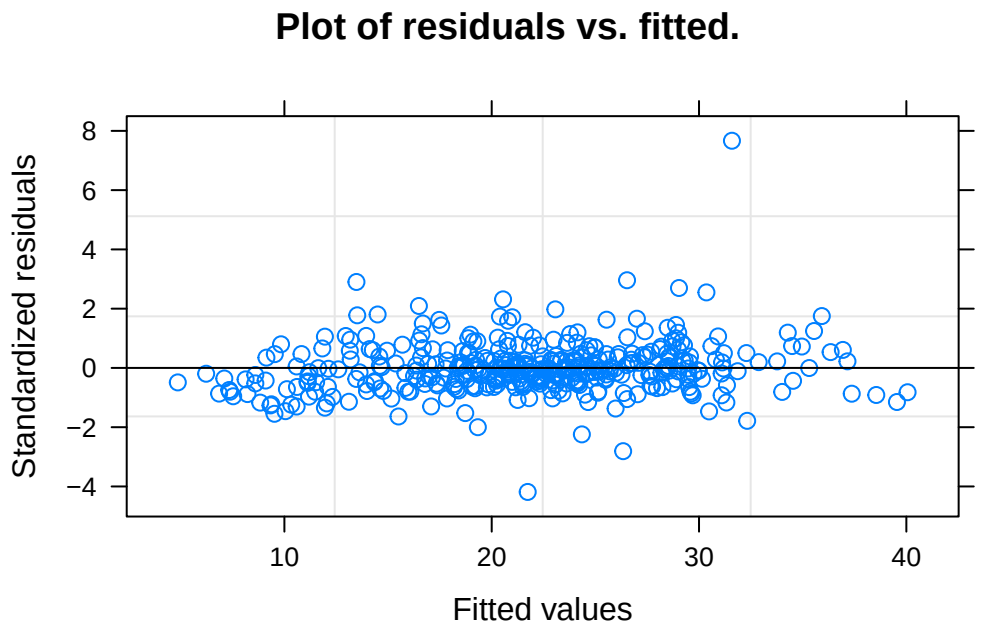
What do we notice here?

- We see that the treatment effect is not significant in this model, but the interaction terms are.

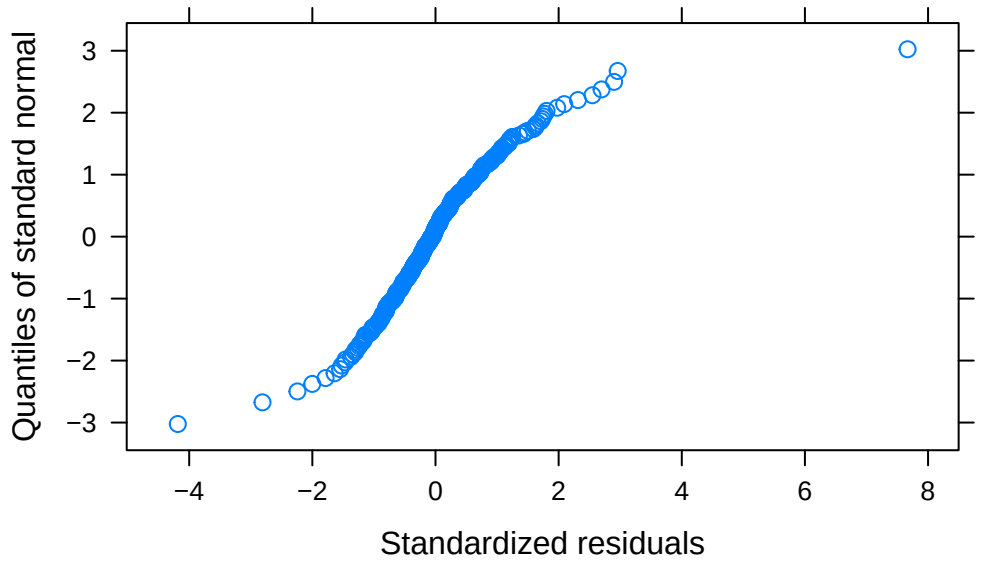
```
#we can plot the xy plot of the fitted values


yhat=predict(model,newdata = TLC_long[,-4],level=0:1)
TLC_long_2=TLC_long
TLC_long_2$yhat=yhat[,3]
TLC_long_2$week2=as.numeric(as.character(TLC_long_2$week))

lattice::xyplot(yhat ~ week2|Treatment,data=TLC_long_2  , groups = ID,
                col = 'black',
                type = c('l', 'p'),ylim=c(0,60))
```



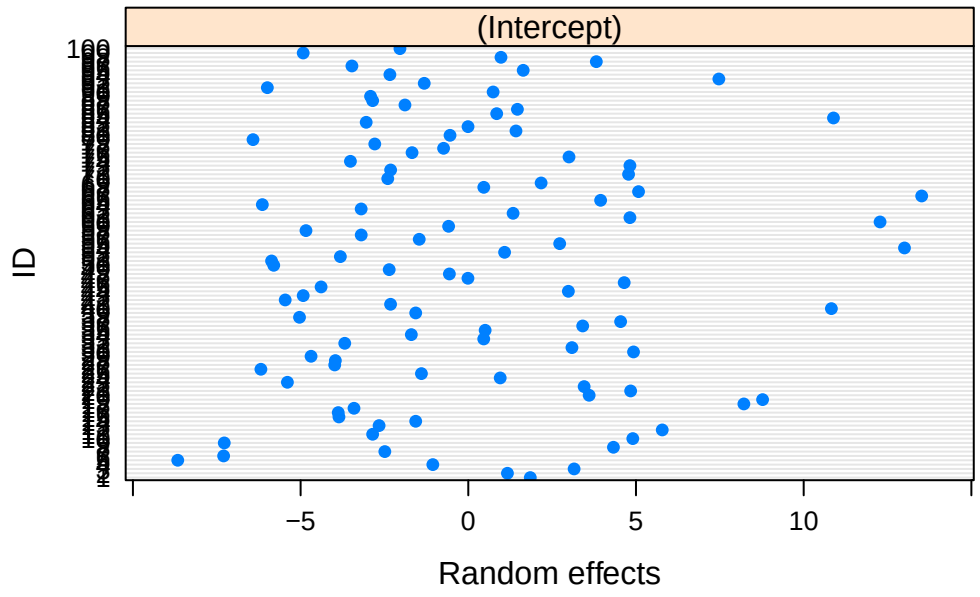

```
# residuals over time?

# Residuals vs. Fitted (no patterns)
plot(model, main = "Plot of residuals vs. fitted.")
```



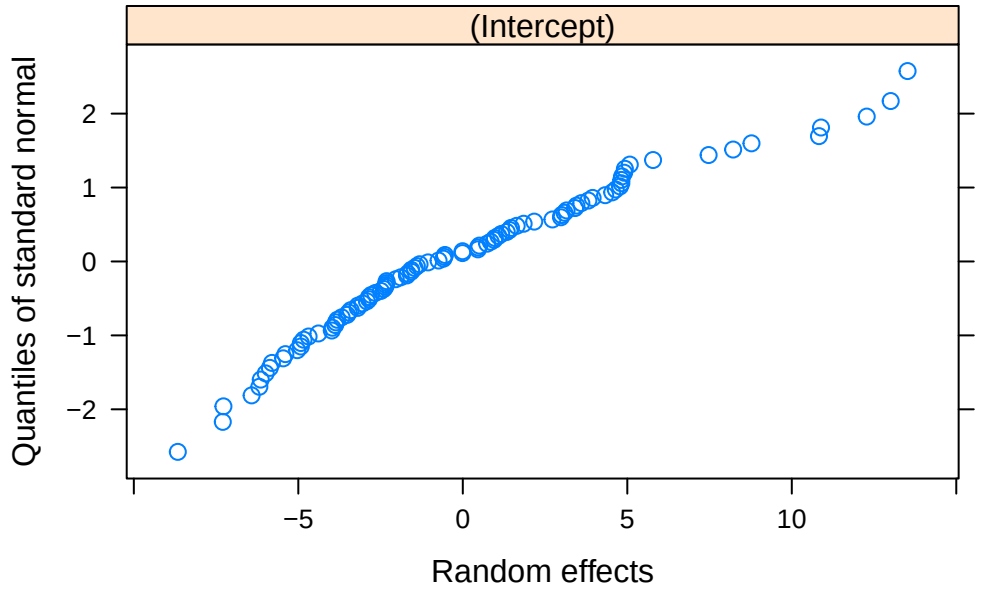

```
# QQPlot for normality of errors
qqnorm(model, ~ residuals(., type="pearson")) # Some issues... probably
```



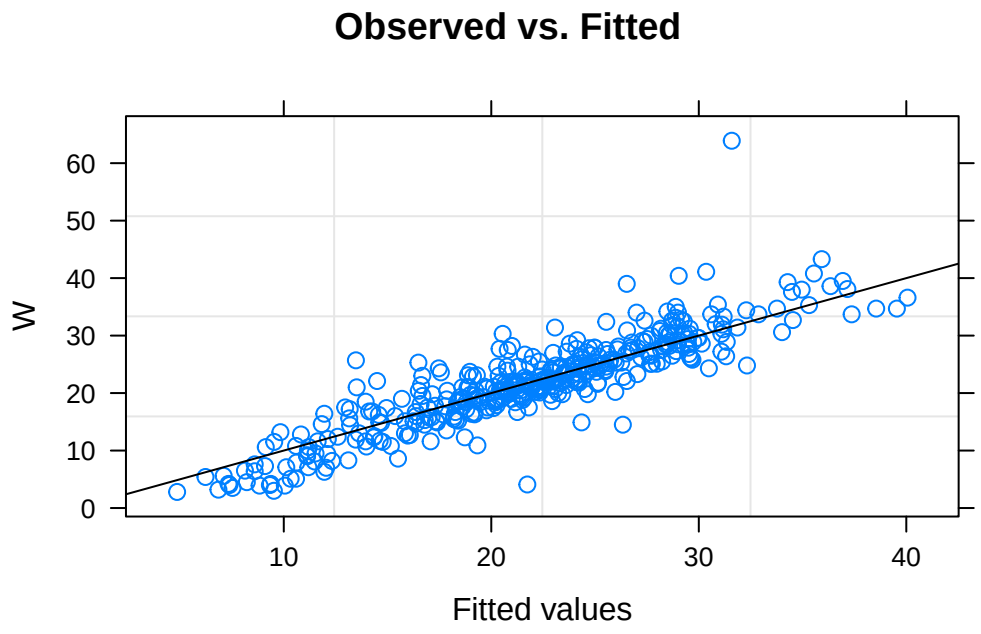
```
# Plots for the Predicted (BLUPs)
plot(nlme::ranef(model))
```



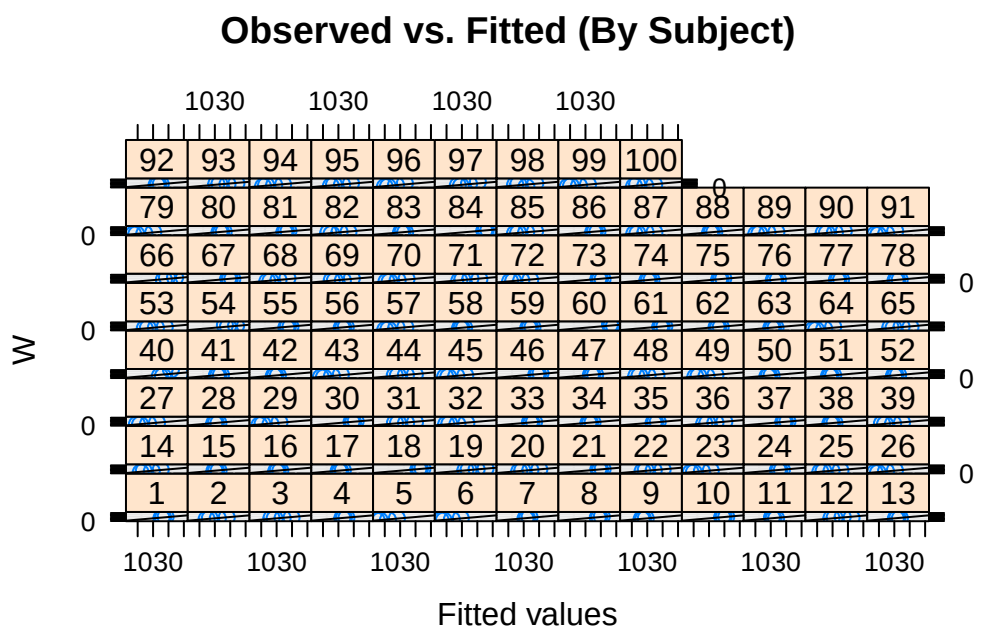

```
qqnorm(model, ~ranef(.)) # These look okay!
```




```
# model$residuals


# Observed vs. Fitted
plot(model, W ~ fitted(.), abline = c(0,1), main = "Observed vs. Fitted")
```




```
plot(model, W~ fitted(.)|ID, abline = c(0,1), main = "Observed vs. Fitted (By Subject)")
```

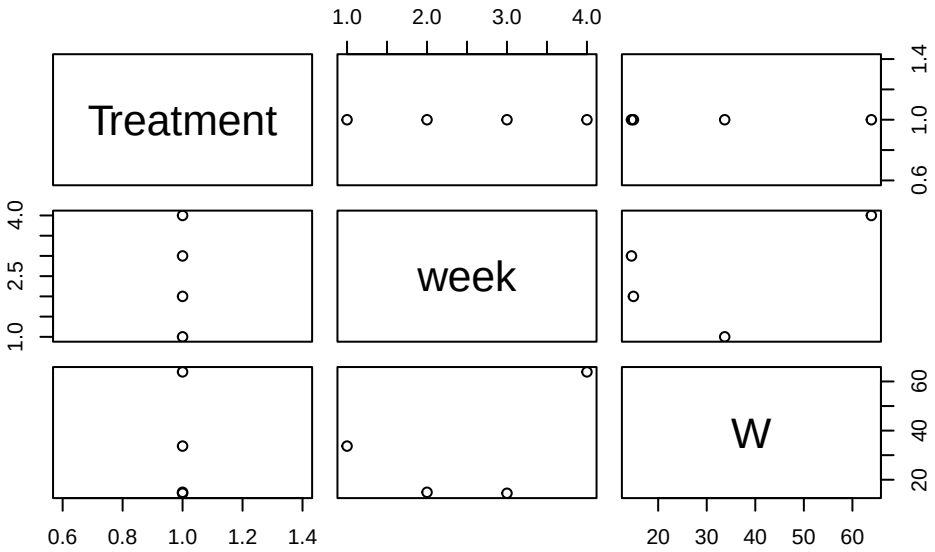


```
# Could also look (e.g.) by treatment, if it existed!
```

Some thoughts

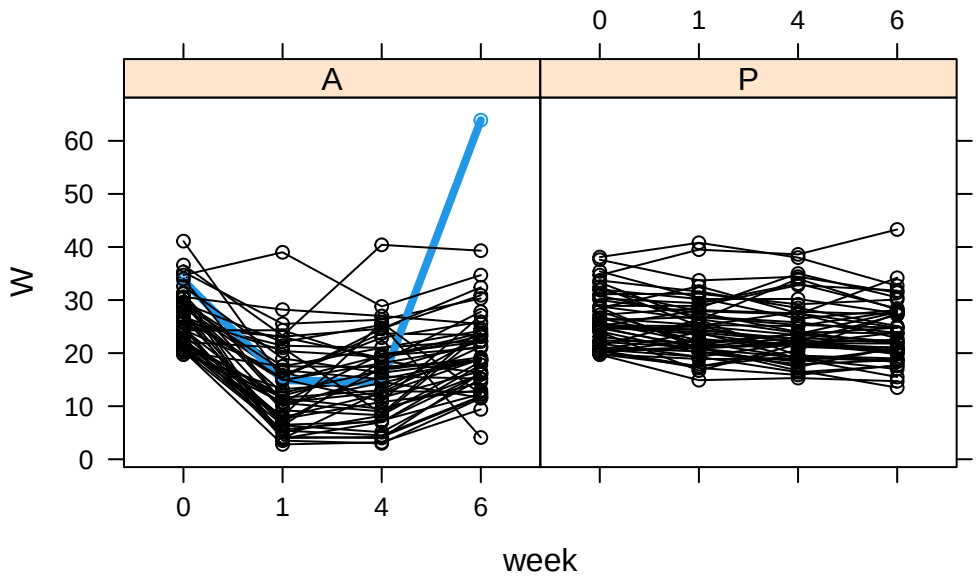
- Maybe we should investigate this residual... If this patient is outlying for indiosyncratic reasons we may want to remove them and redo the analysis
- The fitted xy plots look like the empirical ones - good sign
- Everything else looks pretty good

```
id=TLC_long$ID[which(residuals(model, type="pearson")>5)]

plot(TLC_long[TLC_long$ID==id,-1])
```

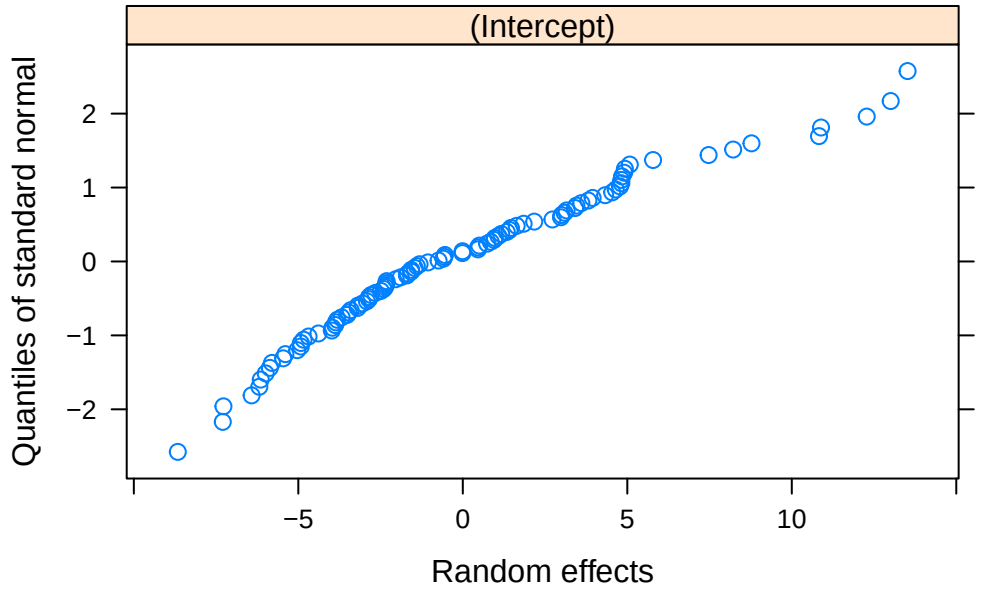



```
col=rep(1,400)
col[TLC_long$ID==id]=4

lwd=rep(1,400)
lwd[TLC_long$ID==id]=4

lattice::xyplot(W ~ week | Treatment,
                data = TLC_long,
```

```
        groups = ID,
        col = col,
        type = c('l', 'p'),lwd=lwd)
```




```
qqnorm(model, ~ranef(.))
```




```
TLC_long_3=TLC_long[TLC_long$ID!=id,]

model2 <- nlme::lme(fixed= W ~ 1 +Treatment+week+Treatment*week,random= ~1|ID, data = TLC_

summary(model)
```

```
Linear mixed-effects model fit by REML
  Data: TLC_long
       AIC      BIC    logLik
  2480.621 2520.334 -1230.311

Random effects:
 Formula: ~1 | ID
        (Intercept) Residual
StdDev:    5.112717 4.214287

Fixed effects:  W ~ 1 + Treatment + week + Treatment * week
                    Value Std.Error  DF    t-value p-value
(Intercept)        26.540 0.9370175 294  28.323912  0.0000
TreatmentP         -0.268 1.3251428  98  -0.202242  0.8401
week1             -13.018 0.8428574 294 -15.445080  0.0000
week4             -11.026 0.8428574 294 -13.081691  0.0000
```

```
week6             -5.778 0.8428574 294  -6.855252  0.0000
TreatmentP:week1  11.406 1.1919804 294   9.568950  0.0000
TreatmentP:week4   8.824 1.1919804 294   7.402807  0.0000
TreatmentP:week6   3.152 1.1919804 294   2.644339  0.0086
 Correlation:
                (Intr) TrtmnP week1  week4  week6  TrtP:1 TrtP:4
TreatmentP      -0.707
week1           -0.450  0.318
week4           -0.450  0.318  0.500
week6           -0.450  0.318  0.500  0.500
TreatmentP:week1  0.318 -0.450 -0.707 -0.354 -0.354
TreatmentP:week4  0.318 -0.450 -0.354 -0.707 -0.354  0.500
TreatmentP:week6  0.318 -0.450 -0.354 -0.354 -0.707  0.500  0.500

Standardized Within-Group Residuals:
        Min          Q1         Med          Q3         Max
-4.18502705 -0.46501691 -0.04732964  0.36499878  7.66712566

Number of Observations: 400
Number of Groups: 100
```

```
summary(model2)
```

```
Linear mixed-effects model fit by REML
  Data: TLC_long_3
      AIC     BIC    logLik
  2371.01 2410.62 -1175.505

Random effects:
 Formula: ~1 | ID
        (Intercept) Residual
StdDev:    5.083126 3.671238

Fixed effects:  W ~ 1 + Treatment + week + Treatment * week
                     Value Std.Error  DF    t-value p-value
(Intercept)       26.393878 0.8957514 291  29.465627  0.0000
TreatmentP        -0.121878 1.2604340  97  -0.096695  0.9232
week1            -12.900000 0.7417021 291 -17.392428  0.0000
week4            -10.859184 0.7417021 291 -14.640897  0.0000
week6             -6.512245 0.7417021 291  -8.780135  0.0000
TreatmentP:week1  11.288000 1.0436674 291  10.815707  0.0000
```

```
TreatmentP:week4   8.657184 1.0436674 291   8.294964  0.0000
TreatmentP:week6   3.886245 1.0436674 291   3.723643  0.0002
 Correlation:
                (Intr) TrtmnP week1  week4  week6  TrtP:1 TrtP:4
TreatmentP      -0.711
week1           -0.414  0.294
week4           -0.414  0.294  0.500
week6           -0.414  0.294  0.500  0.500
TreatmentP:week1  0.294 -0.414 -0.711 -0.355 -0.355
TreatmentP:week4  0.294 -0.414 -0.355 -0.711 -0.355  0.500
TreatmentP:week6  0.294 -0.414 -0.355 -0.355 -0.711  0.500  0.500

Standardized Within-Group Residuals:
        Min          Q1         Med          Q3         Max
-4.63582605 -0.49846350 -0.05679961  0.40026479  3.28119867

Number of Observations: 396
Number of Groups: 99
```

```
nlme::intervals(model)
```

```
Approximate 95% confidence intervals

 Fixed effects:
                     lower    est.      upper
(Intercept)       24.6958881  26.540  28.384112
TreatmentP        -2.8977028  -0.268   2.361703
week1            -14.6767987 -13.018 -11.359201
week4            -12.6847987 -11.026  -9.367201
week6             -7.4367987  -5.778  -4.119201
TreatmentP:week1   9.0601043  11.406  13.751896
TreatmentP:week4   6.4781043   8.824  11.169896
TreatmentP:week6   0.8061043   3.152   5.497896

 Random Effects:
  Level: ID
                  lower     est.    upper
sd((Intercept)) 4.33785 5.112717 6.025997

 Within-group standard error:
   lower     est.    upper
3.887059 4.214287 4.569062
```

```
nlme::intervals(model2)
```

```
Approximate 95% confidence intervals

 Fixed effects:
                     lower        est.      upper
(Intercept)       24.630905  26.3938776  28.156850
TreatmentP        -2.623490  -0.1218776   2.379735
week1            -14.359781 -12.9000000 -11.440219
week4            -12.318964 -10.8591837  -9.399403
week6             -7.972026  -6.5122449  -5.052464
TreatmentP:week1   9.233907  11.2880000  13.342093
TreatmentP:week4   6.603090   8.6571837  10.711277
TreatmentP:week6   1.832151   3.8862449   5.940338

 Random Effects:
  Level: ID
                   lower     est.    upper
sd((Intercept)) 4.334069 5.083126 5.961643

 Within-group standard error:
   lower     est.    upper
3.384769 3.671238 3.981952
```

```
nlme::fixef(model)-nlme::fixef(model2)
```

```
     (Intercept)       TreatmentP            week1            week4
       0.1461224       -0.1461224       -0.1180000       -0.1668163
           week6 TreatmentP:week1 TreatmentP:week4 TreatmentP:week6
       0.7342449        0.1180000        0.1668163       -0.7342449
```

```
anova(model2)
```

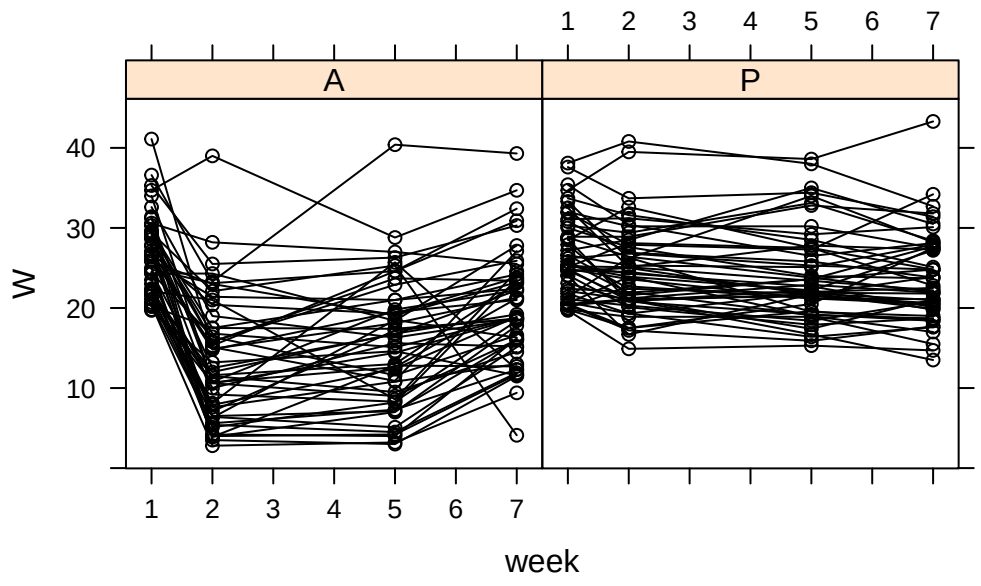
```
               numDF denDF   F-value p-value
(Intercept)        1   291 1606.9197  <.0001
Treatment          1    97   28.8577  <.0001
week               3   291   77.0542  <.0001
Treatment:week     3   291   46.2000  <.0001
```

1.6.3 Sensitivity analysis - We could have fit a quadratic or piece-wise linear model to the data.

```
TLC_long_pl=TLC_long_3
TLC_long_pl$week=as.numeric(as.character(TLC_long_3$week))+1
TLC_long_pl$time1=(TLC_long_pl$week<2)*TLC_long_pl$week
# TLC_long_pl$time2=(TLC_long_pl$week>=3)*(TLC_long_pl$week-TLC_long_pl$week)
head(TLC_long_pl)
```

```
    ID Treatment week    W time1
1.0  1         P    1 30.8     1
2.0  2         A    1 26.5     1
3.0  3         A    1 25.8     1
4.0  4         P    1 24.7     1
5.0  5         A    1 20.4     1
6.0  6         A    1 20.4     1
```

```
lattice::xyplot(W ~ week|Treatment,data=TLC_long_pl  , groups = ID,
                col = 'black',
                type = c('l', 'p'))
```



```r
model_pl <- nlme::lme(fixed= W ~ week+time1+Treatment+Treatment*week+Treatment*time1,rando
summary(model_pl)
```

```
Linear mixed-effects model fit by REML
  Data: TLC_long_pl
       AIC      BIC    logLik
  2382.985 2414.714 -1183.492

Random effects:
 Formula: ~1 | ID
        (Intercept) Residual
StdDev:    5.076697 3.706653

Fixed effects:  W ~ week + time1 + Treatment + Treatment * week + Treatment *      time1
                      Value Std.Error  DF    t-value p-value
(Intercept)       10.561547 1.0495337 293  10.063085       0
week               1.230397 0.1487828 293   8.269756       0
time1             14.601933 0.8194318 293  17.819584       0
TreatmentP        14.507927 1.4768248  97   9.823729       0
week:TreatmentP   -1.432713 0.2093560 293  -6.843432       0
time1:TreatmentP -13.197091 1.1530427 293 -11.445449       0
 Correlation:
                 (Intr) week   time1  TrtmnP wk:TrP
week             -0.662
time1            -0.549  0.666
TreatmentP       -0.711  0.470  0.390
week:TreatmentP   0.470 -0.711 -0.473 -0.662
time1:TreatmentP  0.390 -0.473 -0.711 -0.549  0.666

Standardized Within-Group Residuals:
        Min          Q1         Med          Q3         Max
-4.39986915 -0.47078045 -0.04895001  0.39625544  3.32467517

Number of Observations: 396
Number of Groups: 99
```

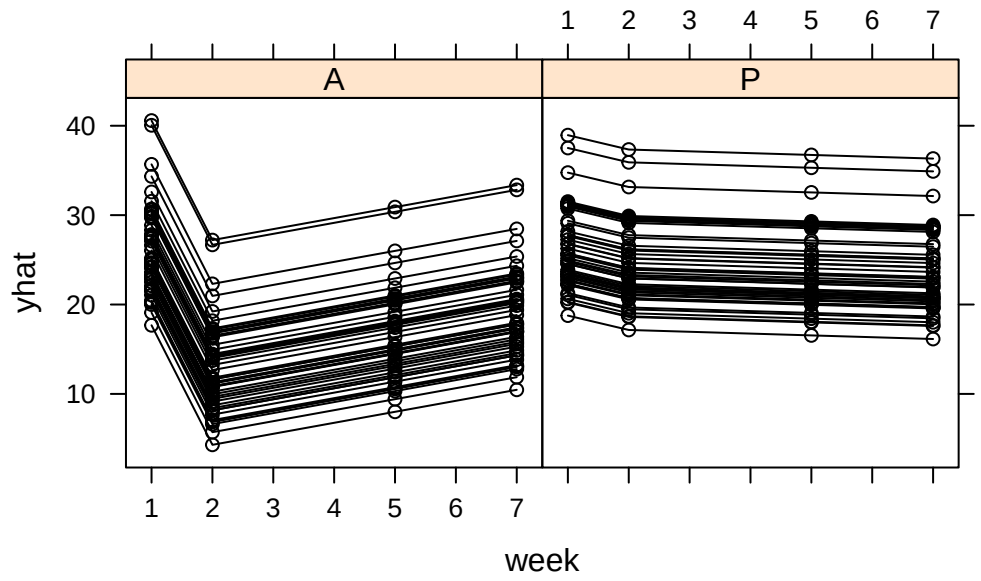
```r
summary(model2)
```

```
Linear mixed-effects model fit by REML
  Data: TLC_long_3
```

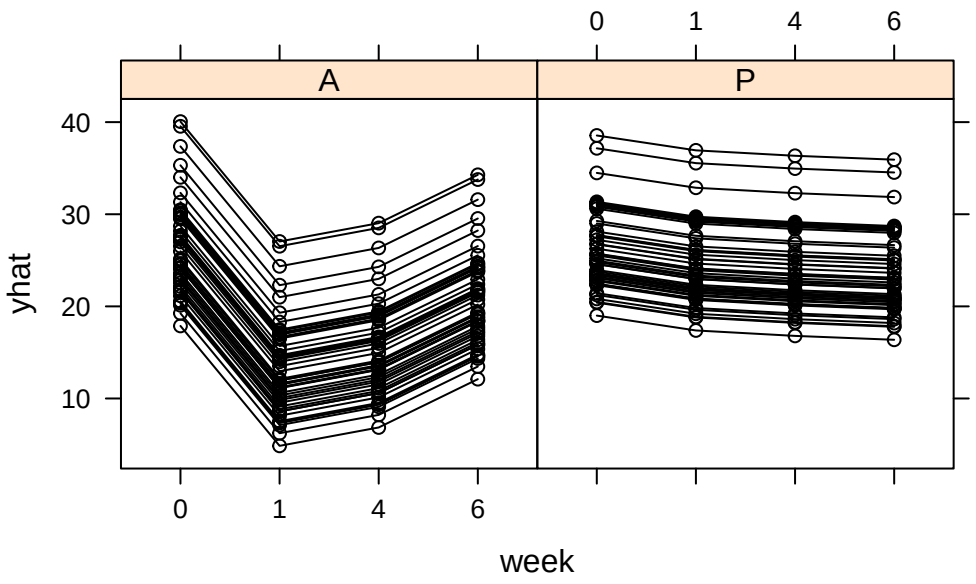
```
      AIC      BIC    logLik
  2371.01 2410.62 -1175.505

Random effects:
 Formula: ~1 | ID
        (Intercept) Residual
StdDev:    5.083126 3.671238

Fixed effects:  W ~ 1 + Treatment + week + Treatment * week
                     Value Std.Error  DF    t-value p-value
(Intercept)       26.393878 0.8957514 291  29.465627  0.0000
TreatmentP        -0.121878 1.2604340  97  -0.096695  0.9232
week1            -12.900000 0.7417021 291 -17.392428  0.0000
week4            -10.859184 0.7417021 291 -14.640897  0.0000
week6             -6.512245 0.7417021 291  -8.780135  0.0000
TreatmentP:week1  11.288000 1.0436674 291  10.815707  0.0000
TreatmentP:week4   8.657184 1.0436674 291   8.294964  0.0000
TreatmentP:week6   3.886245 1.0436674 291   3.723643  0.0002
 Correlation:
                  (Intr) TrtmnP week1  week4  week6  TrtP:1 TrtP:4
TreatmentP        -0.711
week1             -0.414  0.294
week4             -0.414  0.294  0.500
week6             -0.414  0.294  0.500  0.500
TreatmentP:week1   0.294 -0.414 -0.711 -0.355 -0.355
TreatmentP:week4   0.294 -0.414 -0.355 -0.711 -0.355  0.500
TreatmentP:week6   0.294 -0.414 -0.355 -0.355 -0.711  0.500  0.500

Standardized Within-Group Residuals:
        Min          Q1         Med          Q3         Max
-4.63582605 -0.49846350 -0.05679961  0.40026479  3.28119867

Number of Observations: 396
Number of Groups: 99
```

```r
#we can plot the xy plot of the fitted values


yhat=predict(model_pl,newdata = TLC_long_pl[,-4],level=0:1)
TLC_long_4=TLC_long_pl
TLC_long_4$yhat=yhat[,3]
```

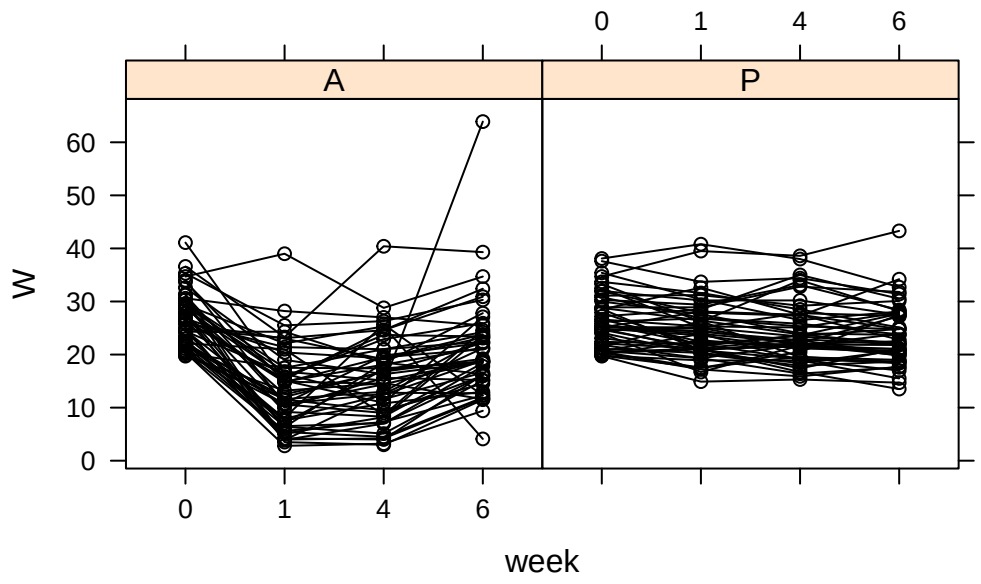
```
lattice::xyplot(yhat ~ week|Treatment,data=TLC_long_4  , groups = ID,
                col = 'black',
                type = c('l', 'p'))
```



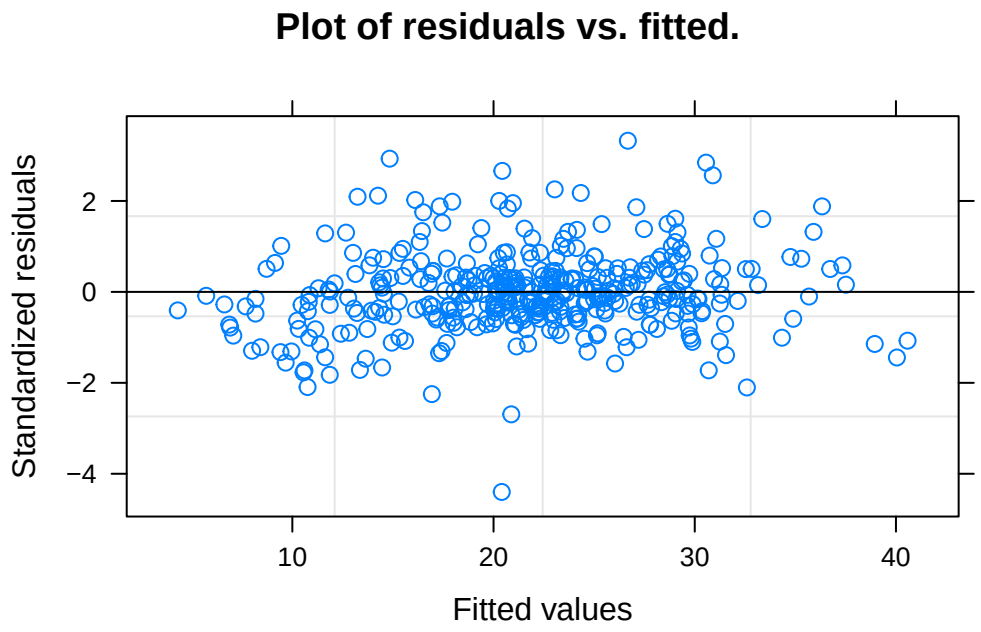

```
lattice::xyplot(yhat ~ week|Treatment,data=TLC_long_2  , groups = ID,
                col = 'black',
                type = c('l', 'p'))
```



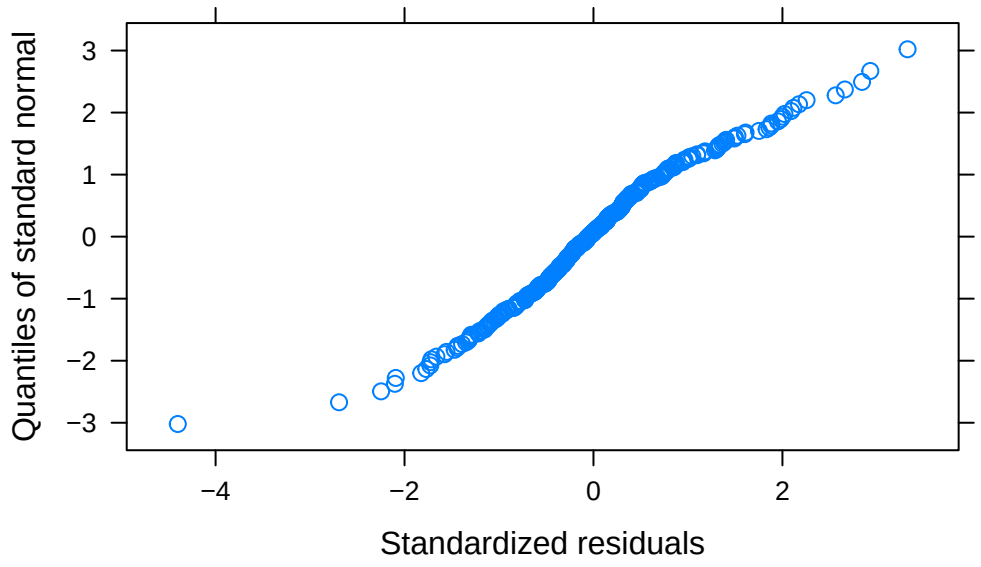

```
lattice::xyplot(W ~ week|Treatment,data=TLC_long_2  , groups = ID,
                col = 'black',
                type = c('l', 'p'))
```



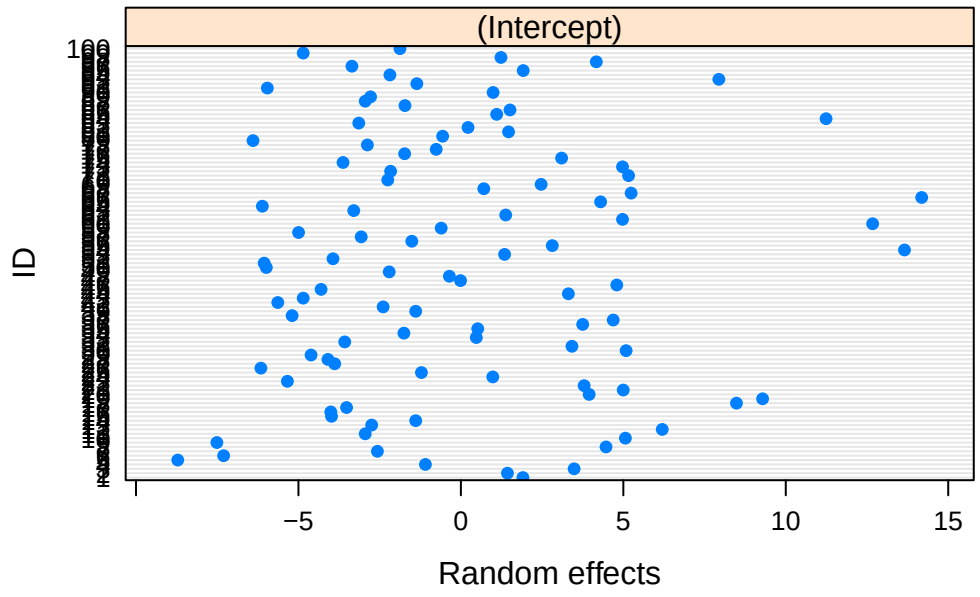

```
# residuals over time?

# Residuals vs. Fitted (no patterns)
plot(model_pl, main = "Plot of residuals vs. fitted.")
```



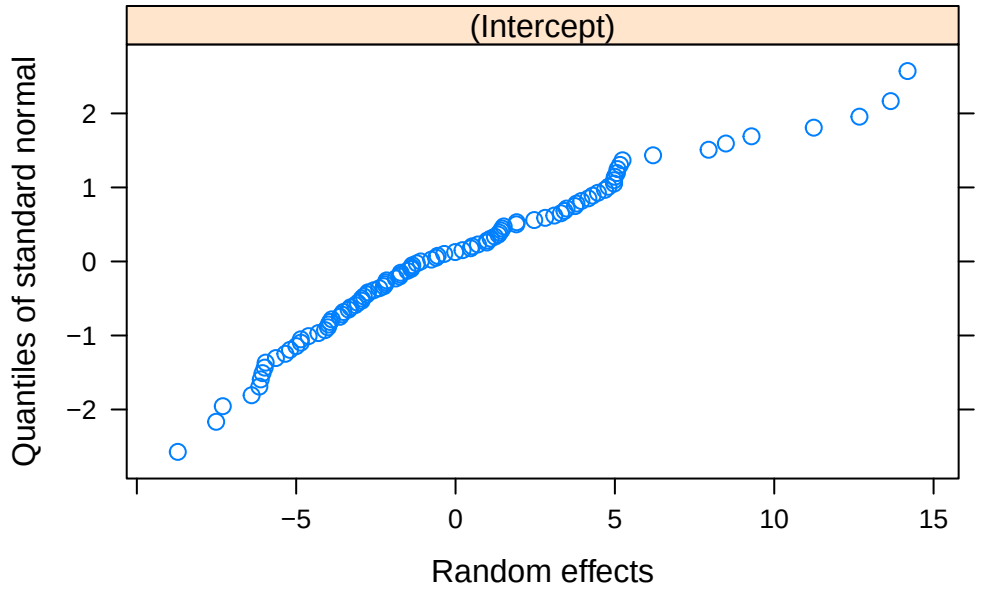

```
# QQPlot for normality of errors
qqnorm(model_pl, ~ residuals(., type="pearson")) # Some issues... probably
```



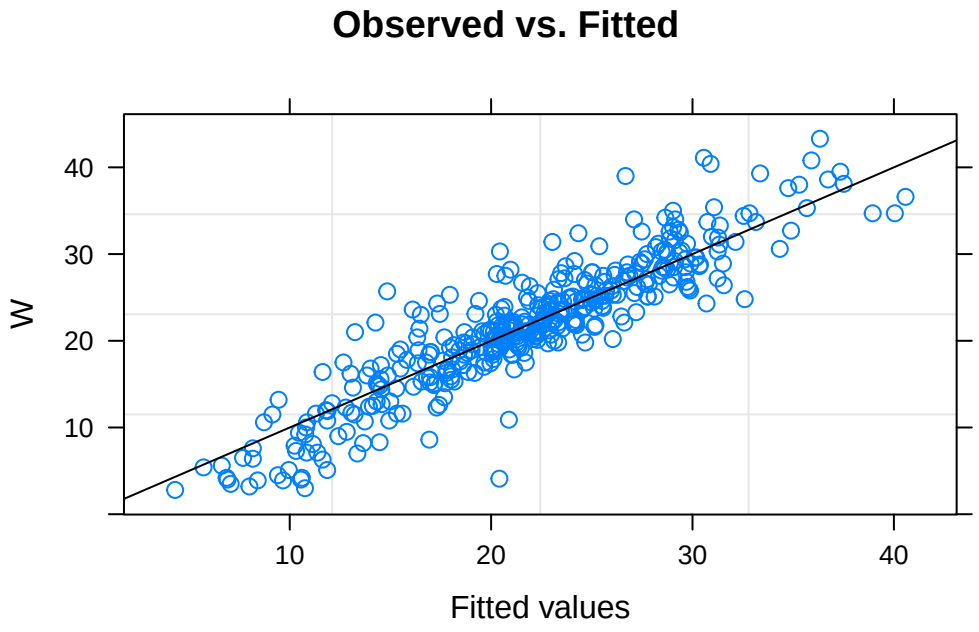
```
# Plots for the Predicted (BLUPs)
plot(nlme::ranef(model_pl))
```



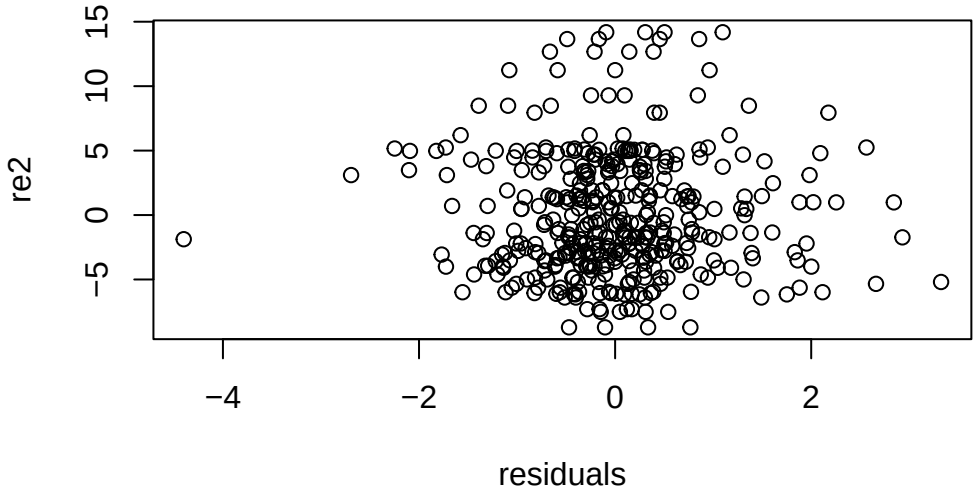

```
qqnorm(model_pl, ~ranef(.)) # These look okay!
```




```
# model$residuals


# Observed vs. Fitted
plot(model_pl, W ~ fitted(.), abline = c(0,1), main = "Observed vs. Fitted")
```




```
# Could also look (e.g.) by treatment, if it existed!

residuals=residuals(model_pl, type="pearson")
re=nlme::ranef(model_pl)
re2=rep(re$`(Intercept)` ,each=4)

plot(residuals,re2)
```




```
length(residuals)
```

```
[1] 396
```

```
length(re)
```

```
[1] 1
```

```
TLC_long_pl$weeksq= TLC_long_pl$week^2
model_q <- nlme::lme(fixed= W ~ week+weeksq+Treatment+week*Treatment+weeksq*Treatment,rand
summary(model_q)
```

```
Linear mixed-effects model fit by REML
  Data: TLC_long_pl
       AIC      BIC   logLik
  2485.78 2517.509 -1234.89

Random effects:
 Formula: ~1 | ID
```

```
        (Intercept) Residual
StdDev:    4.948122 4.346842

Fixed effects:  W ~ week + weeksq + Treatment + week * Treatment + weeksq * Treatment
                     Value Std.Error  DF    t-value p-value
(Intercept)       32.08886 1.2914204 293  24.847722   0.000
week              -9.43993 0.7337490 293 -12.865333   0.000
weeksq             1.12085 0.0908875 293  12.332330   0.000
TreatmentP        -5.11030 1.8171896  97  -2.812198   0.006
week:TreatmentP    8.33921 1.0324764 293   8.076897   0.000
weeksq:TreatmentP -1.02915 0.1278901 293  -8.047157   0.000
 Correlation:
                  (Intr) week   weeksq TrtmnP wk:TrP
week              -0.763
weeksq             0.707 -0.984
TreatmentP        -0.711  0.542 -0.502
week:TreatmentP    0.542 -0.711  0.699 -0.763
weeksq:TreatmentP -0.502  0.699 -0.711  0.707 -0.984

Standardized Within-Group Residuals:
        Min          Q1         Med          Q3         Max
-4.14185550 -0.45450738 -0.04543433  0.47296981  3.35222137

Number of Observations: 396
Number of Groups: 99
```

```r
# model_q <- nlme::lme(fixed= W ~ week+weeksq+Treatment+weeksq*Treatment,random= ~1|ID, da
summary(model_q)
```

```
Linear mixed-effects model fit by REML
  Data: TLC_long_pl
      AIC      BIC   logLik
  2485.78 2517.509 -1234.89

Random effects:
 Formula: ~1 | ID
        (Intercept) Residual
StdDev:    4.948122 4.346842

Fixed effects:  W ~ week + weeksq + Treatment + week * Treatment + weeksq * Treatment
                     Value Std.Error  DF    t-value p-value
```

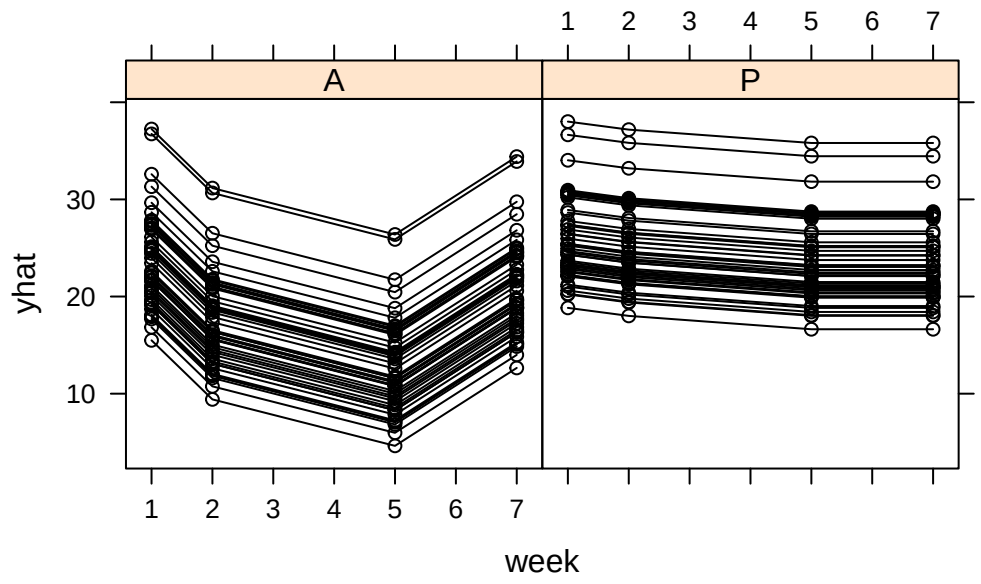
```
(Intercept)       32.08886 1.2914204 293  24.847722   0.000
week              -9.43993 0.7337490 293 -12.865333   0.000
weeksq             1.12085 0.0908875 293  12.332330   0.000
TreatmentP        -5.11030 1.8171896  97  -2.812198   0.006
week:TreatmentP    8.33921 1.0324764 293   8.076897   0.000
weeksq:TreatmentP -1.02915 0.1278901 293  -8.047157   0.000
 Correlation:
                  (Intr) week   weeksq TrtmnP wk:TrP
week              -0.763
weeksq             0.707 -0.984
TreatmentP        -0.711  0.542 -0.502
week:TreatmentP    0.542 -0.711  0.699 -0.763
weeksq:TreatmentP -0.502  0.699 -0.711  0.707 -0.984

Standardized Within-Group Residuals:
        Min          Q1         Med          Q3         Max
-4.14185550 -0.45450738 -0.04543433  0.47296981  3.35222137

Number of Observations: 396
Number of Groups: 99
```

```
summary(model2)
```

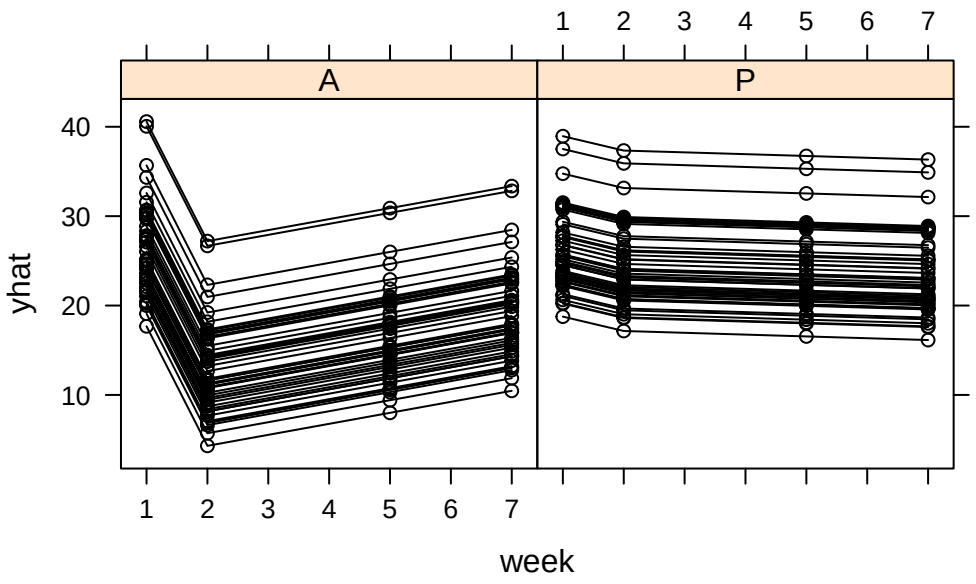
```
Linear mixed-effects model fit by REML
  Data: TLC_long_3
      AIC     BIC    logLik
  2371.01 2410.62 -1175.505

Random effects:
 Formula: ~1 | ID
        (Intercept) Residual
StdDev:    5.083126 3.671238

Fixed effects:  W ~ 1 + Treatment + week + Treatment * week
                     Value Std.Error  DF    t-value p-value
(Intercept)       26.393878 0.8957514 291  29.465627  0.0000
TreatmentP        -0.121878 1.2604340  97  -0.096695  0.9232
week1            -12.900000 0.7417021 291 -17.392428  0.0000
week4            -10.859184 0.7417021 291 -14.640897  0.0000
week6             -6.512245 0.7417021 291  -8.780135  0.0000
TreatmentP:week1  11.288000 1.0436674 291  10.815707  0.0000
```

```
TreatmentP:week4   8.657184 1.0436674 291   8.294964  0.0000
TreatmentP:week6   3.886245 1.0436674 291   3.723643  0.0002
 Correlation:
                (Intr) TrtmnP week1  week4  week6  TrtP:1 TrtP:4
TreatmentP      -0.711
week1           -0.414  0.294
week4           -0.414  0.294  0.500
week6           -0.414  0.294  0.500  0.500
TreatmentP:week1  0.294 -0.414 -0.711 -0.355 -0.355
TreatmentP:week4  0.294 -0.414 -0.355 -0.711 -0.355  0.500
TreatmentP:week6  0.294 -0.414 -0.355 -0.355 -0.711  0.500  0.500

Standardized Within-Group Residuals:
        Min          Q1         Med          Q3         Max
-4.63582605 -0.49846350 -0.05679961  0.40026479  3.28119867

Number of Observations: 396
Number of Groups: 99
```

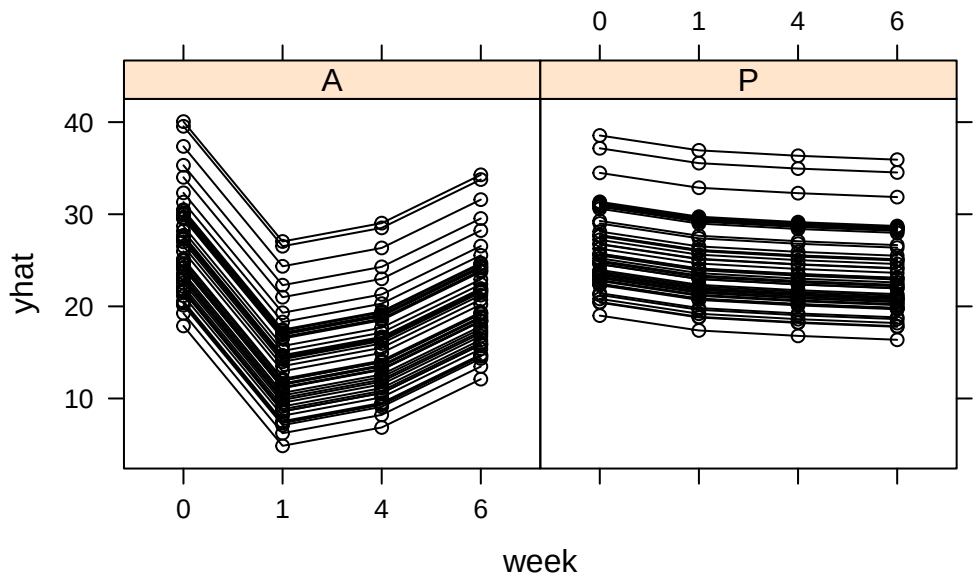
```
#we can plot the xy plot of the fitted values


yhat=predict(model_q,newdata = TLC_long_pl[,-4],level=0:1)
TLC_long_5=TLC_long_pl
TLC_long_5$yhat=yhat[,3]

par(mfrow=c(2,2))

lattice::xyplot(yhat ~ week|Treatment,data=TLC_long_5  , groups = ID,
                col = 'black',
                type = c('l', 'p'))
```



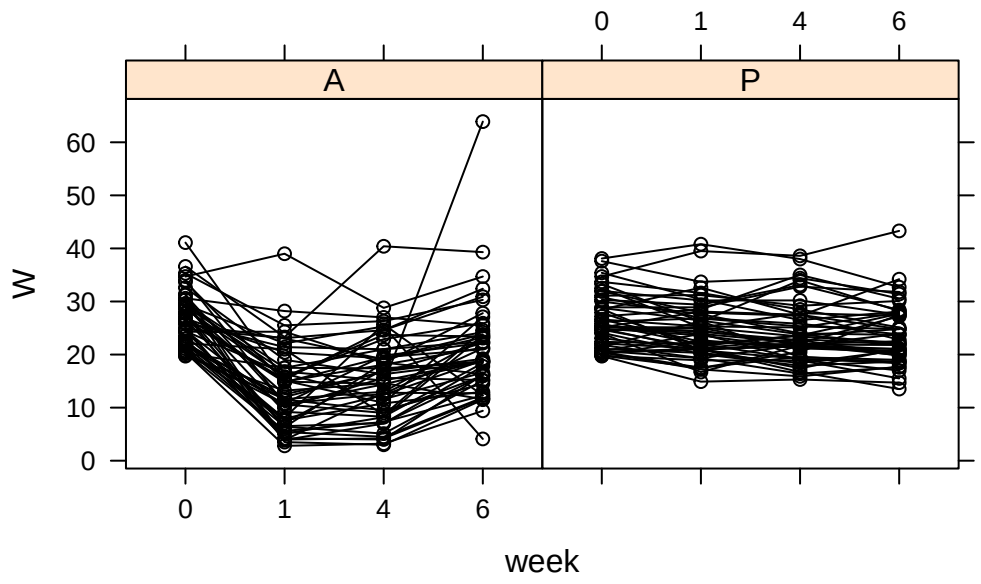

```
lattice::xyplot(yhat ~ week|Treatment,data=TLC_long_4  , groups = ID,
                col = 'black',
                type = c('l', 'p'))
```




```
lattice::xyplot(yhat ~ week|Treatment,data=TLC_long_2  , groups = ID,
                col = 'black',
                type = c('l', 'p'))
```




```
lattice::xyplot(W ~ week|Treatment,data=TLC_long_2  , groups = ID,
                col = 'black',
                type = c('l', 'p'))
```



Conclusions:

- All models indicate a significant effect of treatment, with the largest drop being a time point

    1.

- The lead levels seem to be returning to baseline over time
- The treatment certainly reduces lead levels for a few weeks
- Investigate subject data with ID 40.

We can get more specific too

- At time point 1, individual lead levels seem to drop by 11 points over the placebo.
- After that, the gap starts closing over time - valued at 11, 8, and then 3

2 Generalized linear mixed models

Packages you may need for this lesson:

```
library(ggplot2)
```

```
Warning: package 'ggplot2' was built under R version 4.2.3
```

```
library(GGally)
```

```
Warning: package 'GGally' was built under R version 4.2.3
```

```
Registered S3 method overwritten by 'GGally':
  method from
  +.gg   ggplot2
```

```
library(reshape2)
library(lme4)
```

```
Loading required package: Matrix
```

```
Warning: package 'Matrix' was built under R version 4.2.3
```

```
library(compiler)
library(parallel)
library(boot)
```

```
Warning: package 'boot' was built under R version 4.2.2
```

```
library(tidyverse)
```

```
Warning: package 'tidyverse' was built under R version 4.2.1
```

```
-- Attaching packages --------------------------------------- tidyverse 1.3.2 --
```

```
v tibble  3.2.1     v dplyr   1.1.4
v tidyr   1.3.1     v stringr 1.5.1
v readr   2.1.2     v forcats 0.5.2
v purrr   1.0.2
```

```
Warning: package 'tibble' was built under R version 4.2.3
```

```
Warning: package 'tidyr' was built under R version 4.2.3
```

```
Warning: package 'readr' was built under R version 4.2.1
```

```
Warning: package 'purrr' was built under R version 4.2.3
```

```
Warning: package 'dplyr' was built under R version 4.2.3
```

```
Warning: package 'stringr' was built under R version 4.2.3
```

```
Warning: package 'forcats' was built under R version 4.2.1
```

```
-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x tidyr::expand() masks Matrix::expand()
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
x tidyr::pack()   masks Matrix::pack()
x tidyr::unpack() masks Matrix::unpack()
```

```
library(lattice)
```

```
Attaching package: 'lattice'
```

```
The following object is masked from 'package:boot':

    melanoma
```

2.1 Methodology overview

2.1.1 Model structure

Here, we briefly cover generalized linear mixed models (GLMMs). GLMMs should be used when you have clustered data, and a response variable that is not continuous. They can mainly be fit with the glmm package in R. package in R. You may need other packages for multinomial (mclogit) and cumulative logit (clmm) regression. In general, the GLMM is the child of LMMs and GLMs. Link functions and random effects join forces!

We may consider the linear predictor:

$$\eta = X\alpha + Z\beta.$$

The link function allows us to LINK the response Y to η. We can then define:

- $\eta = X\alpha + Z\beta$
- g – the link function
- g^{-1} – the inverse link function

We can the concisely write our model as: $g(E(Y)) = \eta$, $E(Y) = g^{-1}(\eta)$ and $Y = h(\eta) + \epsilon$.

For instance, popular link functions include: - logit: $g(x) = logit(x) = \log(x/(1-x))$ – for binary response - log: $g(x) = \log(x)$ – for count response - logit: $g(x) = x$ – continuous

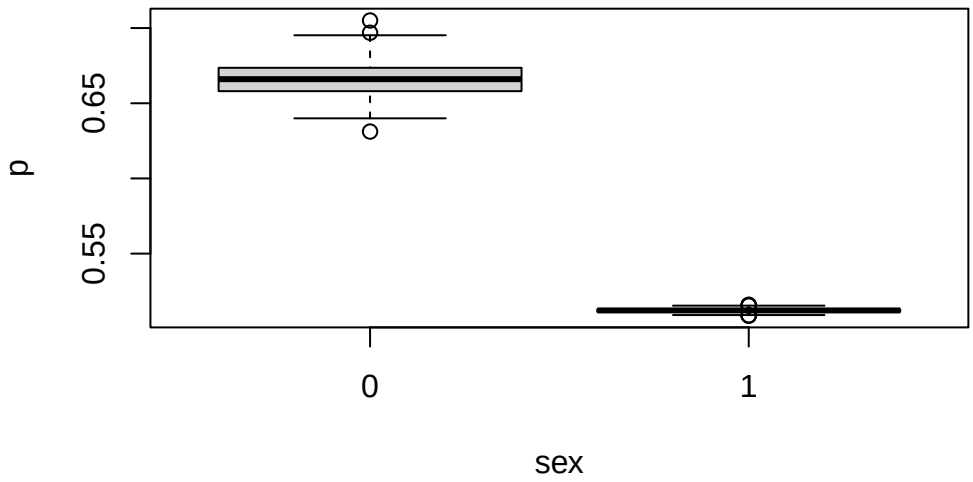
2.1.2 Interpretations

- The interpretations at the population level follow that of usual GLMs/mixed models. For instance, in logistic mixed regression, you may talk about the probability of a success for different levels of predictors. Or you might say an increase in a predictor causes a certain change in the log odds of a success.
- For example, you may wish to recall that $logit(P(A)) = \log(odds(A)) = \log(P(A)/(1 - P(A)))$ is the log odds of A happening.
- $odds(A)/odds(B)$ is the odds ratio of A to B
- $P(A)/P(B)$ is the relative risk of A to B

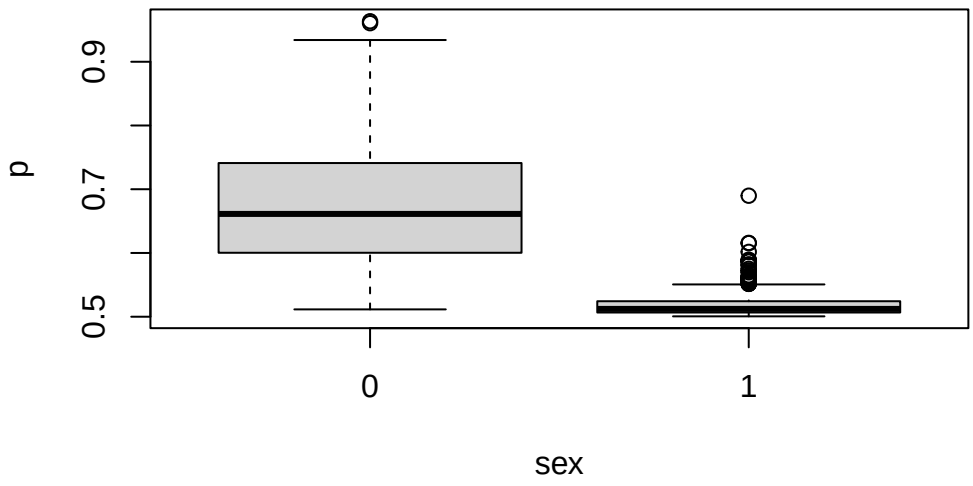
For instance, we may consider $OR = odds(male\ gets\ disease)/odds(female\ gets\ disease)$. In the logistic model, we have that $\log(odds(male\ gets\ disease)/odds(female\ gets\ disease)) = \alpha_{gender}$. The usual interpretation is holding all other variables fixed, in the case of a GLMM, this includes the random effect. Then, α_{gender} would be the increase in log odds for someone in the same cluster. If there is large variability between clusters, the fixed effects may be comparatively small. It is important to see the effect on the probability.

Let's see the impact of different levels of variability on the probability of, say recovery, based off of sex:

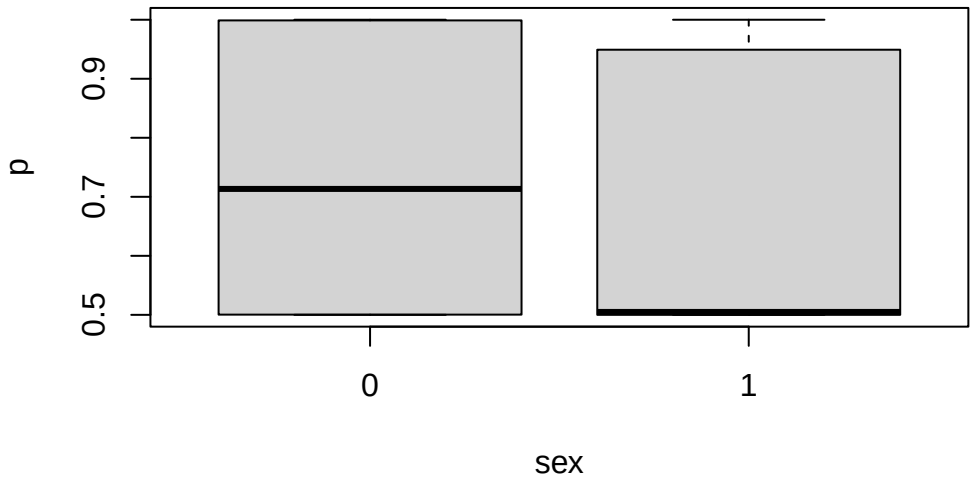
```
expit=function(x){(1+exp(-x))}
simulate_p=function(re_var=1,n=1000){


  sex=rbinom(n,1,1/2)
  re=rnorm(n,0,re_var)
  alpha=3
  odds=expit(sex*alpha+re)
  p=odds/(1+odds)
  boxplot(p~sex)
}
simulate_p(0.1)
```



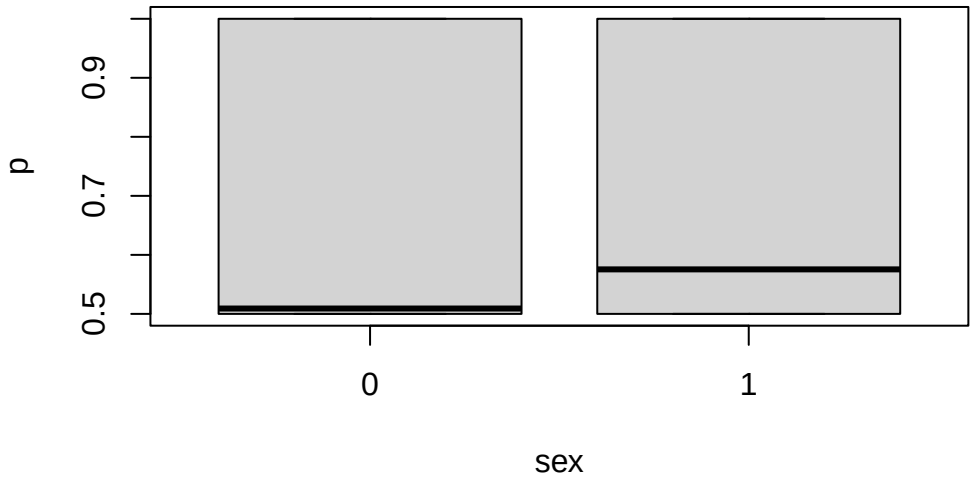

```
simulate_p(1)
```




```
simulate_p(10)
```

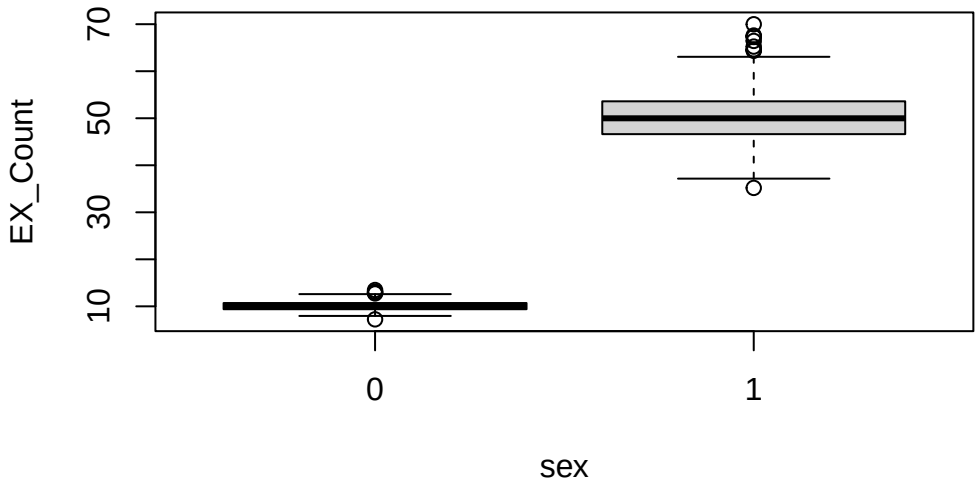


```
simulate_p(25)
```

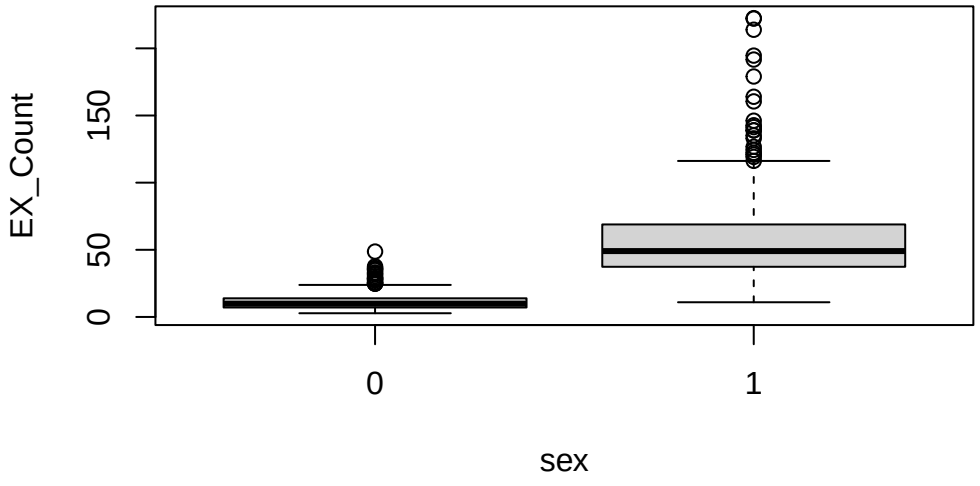


We can do something similar for count data. If we consider a Poisson model, with the log link, we have that

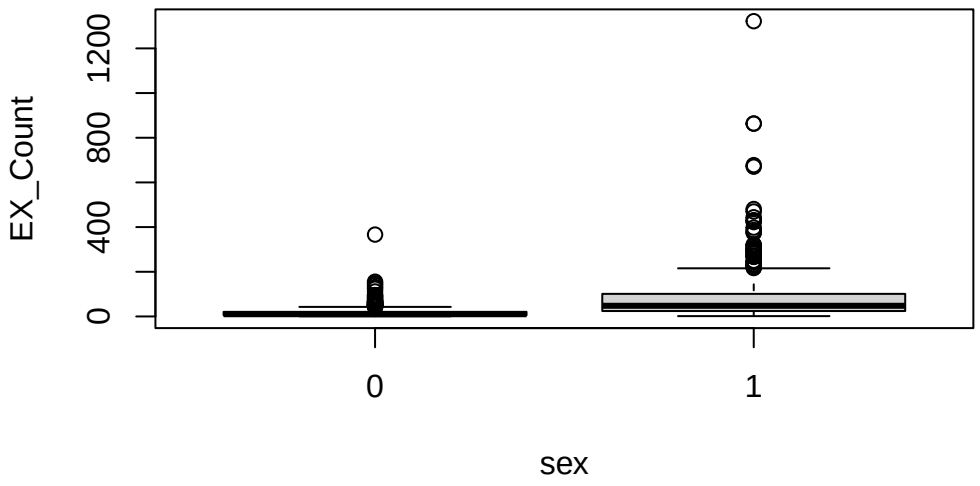
```
simulate_count=function(re_var=1,n=1000){


  sex=rbinom(n,1,1/2)
  re=rnorm(n,0,re_var)
  alpha=log(5)
  EX_Count=exp(log(10)+re+sex*alpha)
  boxplot(EX_Count~sex)
}
simulate_count(0.1)
```



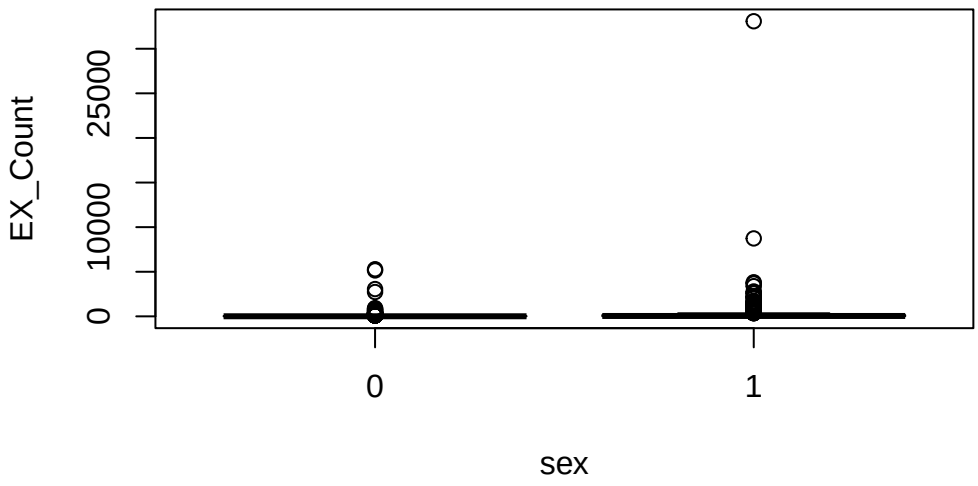

```
simulate_count(0.5)
```



```
simulate_count(1)
```




```
simulate_count(2)
```



2.1.3 Final notes

- The estimates are usually computed via QMLE, or MCMC methods
- Can take long to fit them
- Need enough samples at each level

2.2 Case study 2.1

In general, you will have to learn about specific GLMMs individually/case-by-case, so we will proceed with a case study. The following is adapted from: here .

Case information: What patient and physician factors explain lung cancer remission after treatment?

Setup:

```
hdp = read.csv("https://stats.idre.ucla.edu/stat/data/hdp.csv")
hdp = within(hdp, {
  Married = factor(Married, levels = 0:1, labels = c("no", "yes"))
  DID = factor(DID)
  HID = factor(HID)
```

```
  CancerStage = factor(CancerStage)
})
```

Let's explore the data, focusing on :

- Il6, CRP: Biological measurements
- LengthofStay
- CancerStage (I, II, III, or IV),
- Experience (doctor)
- doctor ID - cluster variable

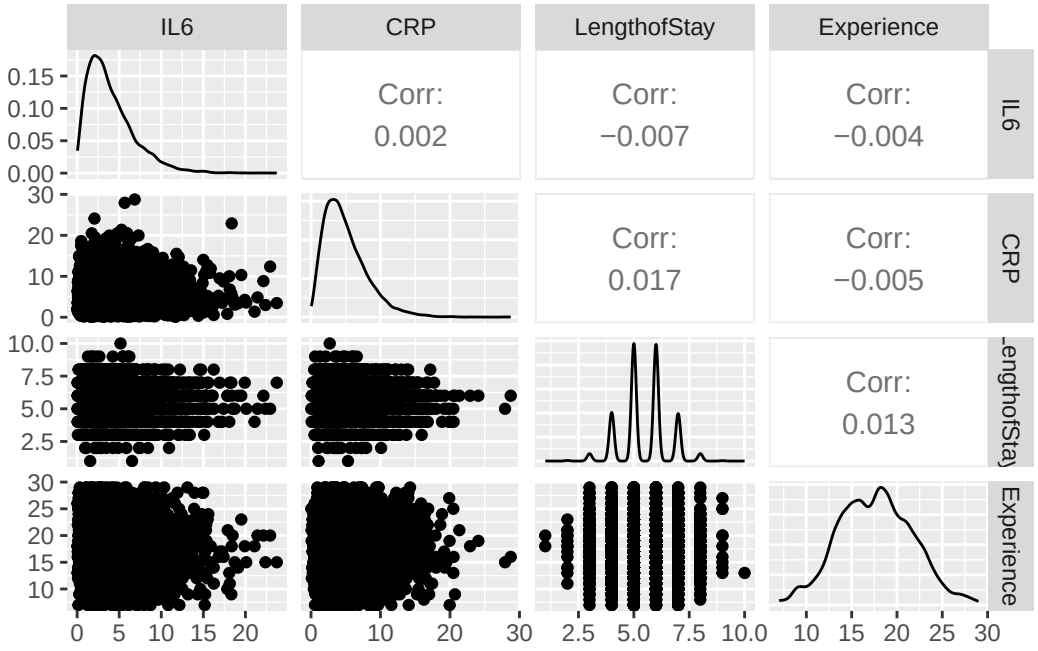
```
head(hdp)
```

```
  tumorsize      co2 pain wound mobility ntumors nmorphine remission
1  67.98120 1.534333    4     4        2       0         0         0
2  64.70246 1.676132    2     3        2       0         0         0
3  51.56700 1.533445    6     3        2       0         0         0
4  86.43799 1.453300    3     3        2       0         0         0
5  53.40018 1.566348    3     4        2       0         0         0
6  51.65727 1.417868    4     5        2       0         0         0
  lungcapacity      Age Married FamilyHx SmokingHx    Sex CancerStage
1    0.8010882 64.96824      no       no    former   male          II
2    0.3264440 53.91714      no       no    former female          II
3    0.5650309 53.34730     yes       no     never female          II
4    0.8484109 41.36804      no       no    former   male           I
5    0.8864910 46.80042      no       no     never   male          II
6    0.7010307 51.92936     yes       no     never   male           I
  LengthofStay      WBC      RBC      BMI       IL6       CRP DID Experience
1            6 6087.649 4.868416 24.14424  3.698981 8.0864168   1         25
2            6 6700.310 4.679052 29.40516  2.627481 0.8034876   1         25
3            5 6042.809 5.005862 29.48259 13.896153 4.0341565   1         25
4            5 7162.697 5.265058 21.55726  3.008033 2.1258629   1         25
5            6 6443.440 4.984259 29.81519  3.890698 1.3493239   1         25
6            5 6800.549 5.199714 27.10252  1.418219 2.1946941   1         25
   School Lawsuits HID  Medicaid
1 average        3   1 0.6058667
2 average        3   1 0.6058667
3 average        3   1 0.6058667
4 average        3   1 0.6058667
5 average        3   1 0.6058667
6 average        3   1 0.6058667
```

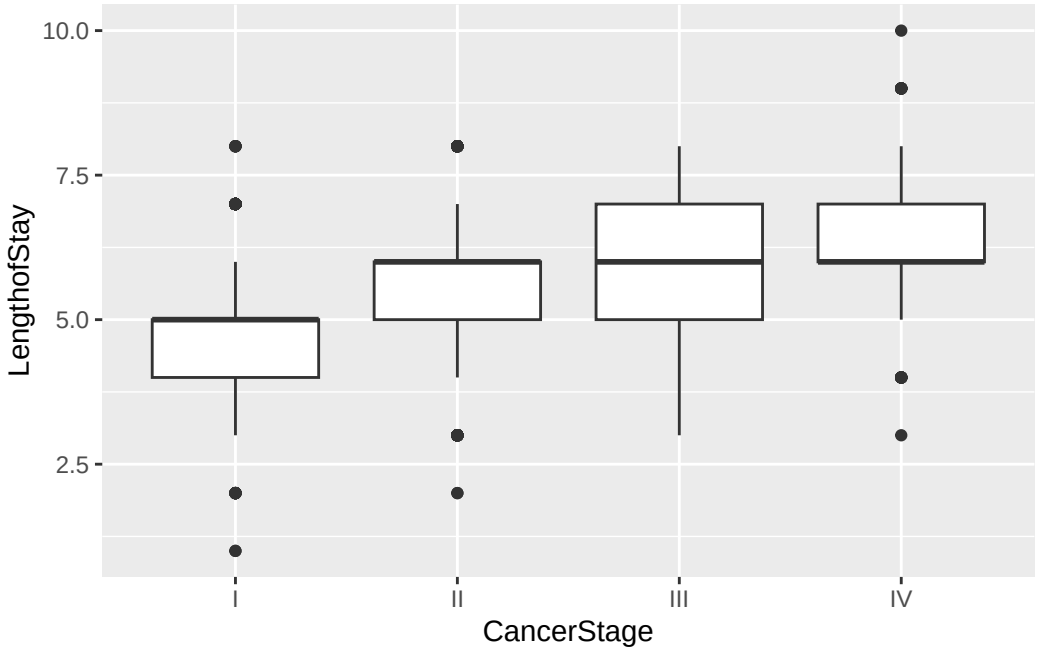
```
names(hdp)
```

```
 [1] "tumorsize"    "co2"          "pain"         "wound"        "mobility"
 [6] "ntumors"      "nmorphine"    "remission"    "lungcapacity" "Age"
[11] "Married"      "FamilyHx"     "SmokingHx"    "Sex"          "CancerStage"
[16] "LengthofStay" "WBC"          "RBC"          "BMI"          "IL6"
[21] "CRP"          "DID"          "Experience"   "School"       "Lawsuits"
[26] "HID"          "Medicaid"
```

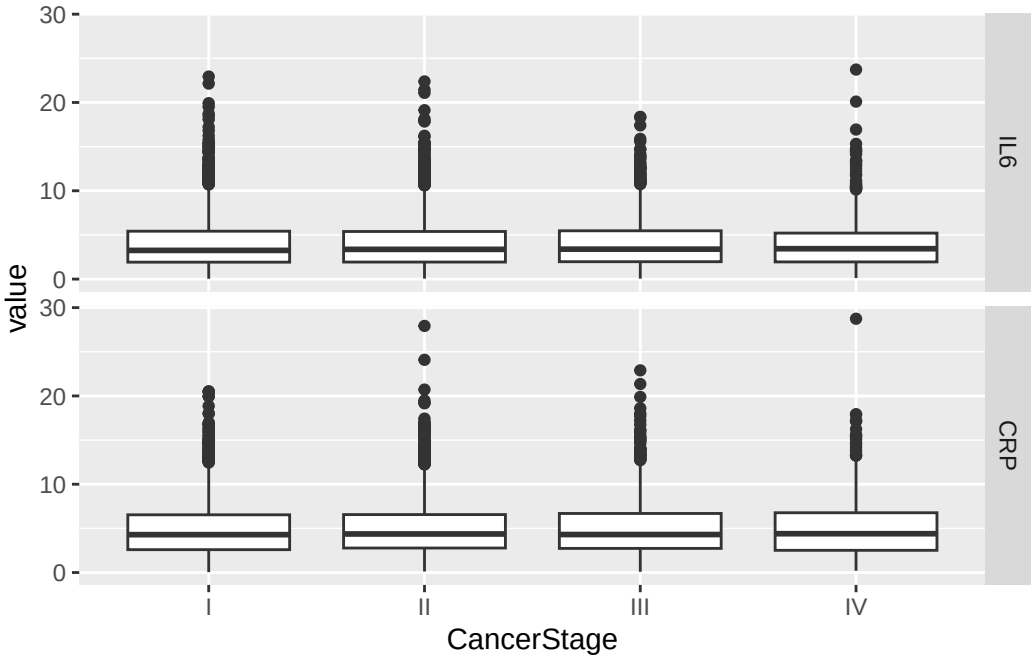
```
# We can use ggplot2 for this one...
# This si
ggpairs(hdp[, c("IL6", "CRP", "LengthofStay", "Experience")])
```



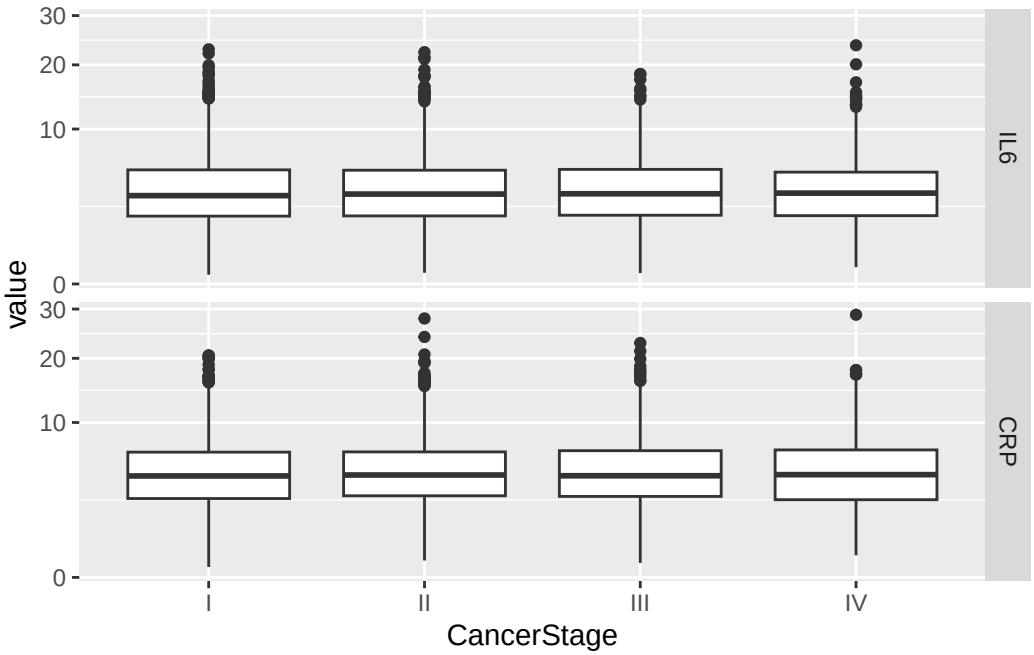

```
ggplot(hdp, aes(x = CancerStage, y = LengthofStay)) +
  geom_boxplot()
```



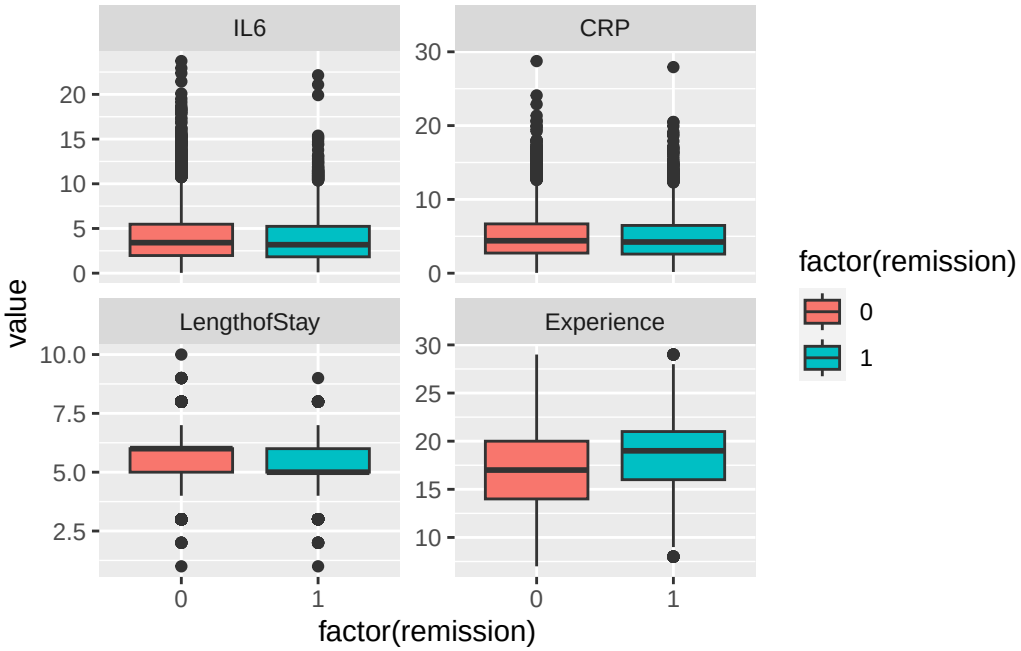

```
tmp = melt(hdp[, c("CancerStage", "IL6", "CRP")], id.vars="CancerStage")
ggplot(tmp, aes(x = CancerStage, y = value)) +
  geom_boxplot() +
  facet_grid(variable ~ .)
```




```
ggplot(tmp, aes(x = CancerStage, y = value)) +
  geom_boxplot() +
  facet_grid(variable ~ .) +
  scale_y_sqrt()
```




```
tmp = melt(hdp[, c("remission", "IL6", "CRP", "LengthofStay", "Experience")],
  id.vars="remission")
ggplot(tmp, aes(factor(remission), y = value, fill=factor(remission))) +
  geom_boxplot() +
  facet_wrap(~variable, scales="free_y")
```



2.2.1 Fitting the model

```
# mixed effects logistic regression model
# estimate the model
# model = glmer(remission ~ IL6 + CRP + CancerStage + LengthofStay + Experience +
    # (1 | DID), data = hdp, family = binomial)

model = glmer(remission ~ IL6 + CRP + CancerStage + LengthofStay + Experience +
    (1 | DID), data = hdp, family = binomial, control = glmerControl(optimizer = "bobyqa")
```

```
# print the mod results without correlations among fixed effects
print(model, corr = FALSE)
```

```
Generalized linear mixed model fit by maximum likelihood (Adaptive
  Gauss-Hermite Quadrature, nAGQ = 10) [glmerMod]
 Family: binomial  ( logit )
Formula: remission ~ IL6 + CRP + CancerStage + LengthofStay + Experience +
    (1 | DID)
   Data: hdp
      AIC       BIC    logLik  deviance  df.resid
```

```
 7397.276  7460.733 -3689.638  7379.276      8516
Random effects:
 Groups Name        Std.Dev.
 DID    (Intercept) 2.015
Number of obs: 8525, groups:  DID, 407
Fixed Effects:
   (Intercept)             IL6             CRP   CancerStageII  CancerStageIII
      -2.05269         -0.05677         -0.02148         -0.41393         -1.00346
 CancerStageIV    LengthofStay      Experience
      -2.33704         -0.12118          0.12009
```

```
# broom::tidy(model)
```

```
se = sqrt(diag(vcov(model)))
# table of estimates with 95% CI
tab = cbind(Est = fixef(model), LL = fixef(model) - 1.96 * se, UL = fixef(model) + 1.96 *s
```

```
                     Est          LL           UL
(Intercept)    -2.05269401 -3.09430566 -1.011082369
IL6            -0.05677185 -0.07934786 -0.034195848
CRP            -0.02148294 -0.04151100 -0.001454893
CancerStageII  -0.41393409 -0.56243123 -0.265436940
CancerStageIII -1.00346486 -1.19609923 -0.810830493
CancerStageIV  -2.33703717 -2.64682992 -2.027244418
LengthofStay   -0.12118214 -0.18710336 -0.055260913
Experience      0.12008835  0.06628404  0.173892667
```

```
#Odds ratios
exp(tab)
```

```
                    Est         LL        UL
(Intercept)    0.12838856 0.04530646 0.3638250
IL6            0.94480960 0.92371854 0.9663822
CRP            0.97874617 0.95933879 0.9985462
CancerStageII  0.66104452 0.56982201 0.7668708
CancerStageIII 0.36660700 0.30237140 0.4444888
CancerStageIV  0.09661346 0.07087554 0.1316979
LengthofStay   0.88587259 0.82935801 0.9462382
Experience     1.12759647 1.06853018 1.1899278
```

Inference - use normal approximation, or bootstrapping... For boostrapping, we need to sample one level at a time.

```
#resamples single level clustered data
sampler = function(dat, clustervar, replace = TRUE, reps = 1) {
    # Unique clusters
    cid = unique(dat[, clustervar[1]])
    # num clusters
    ncid = length(cid)
    # sampled clusters for each rep
    recid = sample(cid, size = ncid * reps, replace = TRUE)
    if (replace) {
      # This line is grabbing all the rows corresponding to each cluster, sampling them, a
        rid = lapply(seq_along(recid), function(i) {
            cbind(NewID = i, RowID = sample(which(dat[, clustervar] == recid[i]),
                size = length(which(dat[, clustervar] == recid[i])), replace = TRUE))
        })
    }

    else {
      # This line is grabbing all the rows corresponding to each cluster and assigning a n
        rid = lapply(seq_along(recid), function(i) {
            cbind(NewID = i, RowID = which(dat[, clustervar] == recid[i]))
        })
    }
    #put the above info in a dataframe
    dat = as.data.frame(do.call(rbind, rid))

    # put the above info in a dataframe, cut divides the above long samples into the repli
    dat$Replicate = cut(dat$NewID, breaks = c(1, ncid * 1:reps), include.lowest = TRUE,lab
    # change to factor
    dat$NewID = factor(dat$NewID)
    return(dat)
}
```

```
head(sampler(hdp, "DID", reps = 1))
```

```
  NewID RowID Replicate
1     1  1487         1
2     1  1512         1
3     1  1492         1
4     1  1513         1
```

```
5     1  1504         1
6     1  1476         1
```

```r
set.seed(1241)
#sample 100 samples, use more in reality

tmp = sampler(hdp, "DID", reps = 100)
```

```r
bigdata = cbind(tmp, hdp[tmp$RowID, ])
head(bigdata)
```

```
     NewID RowID Replicate tumorsize      co2 pain wound mobility ntumors
1137     1  1137         1  61.60908 1.385558    4     5        5       4
1131     1  1131         1  62.90957 1.489077    3     5        4       1
1158     1  1158         1  61.15147 1.694328    5     6        4       0
1151     1  1151         1  94.27969 1.752081    5     7        5       2
1152     1  1152         1  85.51482 1.532822    3     6        6       6
1148     1  1148         1  73.29905 1.770178    5     6        6       5
     nmorphine remission lungcapacity      Age Married FamilyHx SmokingHx
1137         7         0    0.9750009 55.83614      no       no     never
1131         4         0    0.9268863 51.04217     yes       no     never
1158         0         0    0.6443683 43.13214     yes       no     never
1151         7         0    0.8582457 62.36135     yes       no   current
1152         5         0    0.4208578 61.63105      no      yes     never
1148         4         0    0.9638132 49.00940      no       no   current
        Sex CancerStage LengthofStay      WBC      RBC      BMI       IL6
1137 female          II            6 4967.423 5.412072 29.58874 6.1558399
1131 female          II            6 5605.937 4.793538 23.66554 3.7553295
1158 female           I            5 5483.016 5.040751 28.64414 2.5412189
1151   male           I            5 6337.338 6.064870 30.33128 3.2348010
1152 female          IV            7 4039.674 5.266622 25.28456 0.7059072
1148   male           I            4 5365.597 4.970115 39.41294 0.7627430
           CRP DID Experience  School Lawsuits HID  Medicaid
1137  1.9068986  52         19 average        3   5 0.2188103
1131  0.6346611  52         19 average        3   5 0.2188103
1158  3.7357859  52         19 average        3   5 0.2188103
1151  8.2530591  52         19 average        3   5 0.2188103
1152  4.4492342  52         19 average        3   5 0.2188103
1148 10.7412680  52         19 average        3   5 0.2188103
```

```
f = fixef(model); f
```

```
  (Intercept)           IL6           CRP  CancerStageII CancerStageIII
   -2.05269401    -0.05677185    -0.02148294    -0.41393409    -1.00346486
CancerStageIV   LengthofStay     Experience
   -2.33703717    -0.12118214     0.12008835
```

```
r = getME(model, "theta"); r
```

```
DID.(Intercept)
       2.014552
```

```
# Make cluster
cl = makeCluster(10)
clusterExport(cl, c("bigdata", "f", "r"))
a=clusterEvalQ(cl, require(lme4))
```

```
myboot = function(i) {
  # fit the mdoel for every bootstrap sample
    object = try(glmer(remission ~ IL6 + CRP + CancerStage + LengthofStay +
        Experience + (1 | NewID), data = bigdata, subset = Replicate == i, family = binomi
        nAGQ = 1, start = list(fixef = f, theta = r)), silent = TRUE)
    if (class(object) == "try-error")
        return(object)
    c(fixef(object), getME(object, "theta"))
}
```

```
start = proc.time()
res = parLapplyLB(cl, X = levels(bigdata$Replicate), fun = myboot)
end = proc.time()
# shut down the cluster
stopCluster(cl)
```

```
#Check how many models converge
success = sapply(res, is.numeric)
mean(success)

# combine successful results
```

```r
bigres = do.call(cbind, res[success])

# calculate 2.5th and 97.5th percentiles for 95% CI
(ci = t(apply(bigres, 1, quantile, probs = c(0.025, 0.975))))

# All results
finaltable = cbind(Est = c(f, r), SE = c(se, NA), BootMean = rowMeans(bigres),ci)
# round and print
round(finaltable, 3)
```

2.2.2 Predicted probabilities and graphing

We may wish to plot some of the fitted functions/probabilities.

Now we can look at by length of stay:

```r
# temporary data
tmpdat = hdp[, c("IL6", "CRP", "CancerStage", "LengthofStay", "Experience","DID")]

summary(hdp$LengthofStay)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  1.000   5.000   5.000   5.492   6.000  10.000
```

```r
jvalues =  seq(from = min(hdp$LengthofStay), to = max(hdp$LengthofStay), l = 100)

# calculate predicted probabilities and store in a list
pp = lapply(jvalues, function(j) {
    tmpdat$LengthofStay = j
    predict(model, newdata = tmpdat, type = "response")
})

# average marginal predicted probability across a few different Lengths of
# Stay
jvalues[c(1, 20, 40, 60, 80, 100)]
```

```
[1]  1.000000  2.727273  4.545455  6.363636  8.181818 10.000000
```

```r
sapply(pp[c(1, 20, 40, 60, 80, 100)], mean)
```

```
[1] 0.3652319 0.3366360 0.3075494 0.2796359 0.2530109 0.2277694
```

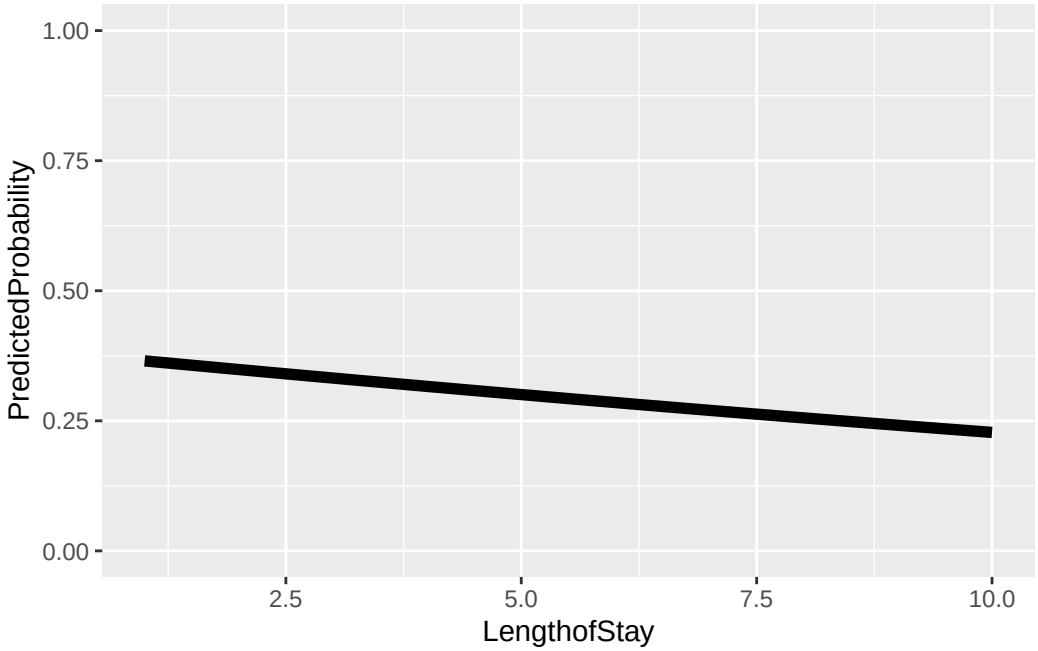
```
# get the means with lower and upper quartiles
plotdat = t(sapply(pp, function(x) {
    c(M = mean(x), quantile(x, c(0.25, 0.75)))
}))
head(plotdat)
```

```
            M        25%       75%
[1,] 0.3652319 0.08489874 0.6155638
[2,] 0.3637056 0.08404676 0.6129535
[3,] 0.3621815 0.08320255 0.6103367
[4,] 0.3606597 0.08236605 0.6077135
[5,] 0.3591402 0.08153722 0.6050841
[6,] 0.3576230 0.08071600 0.6024486
```

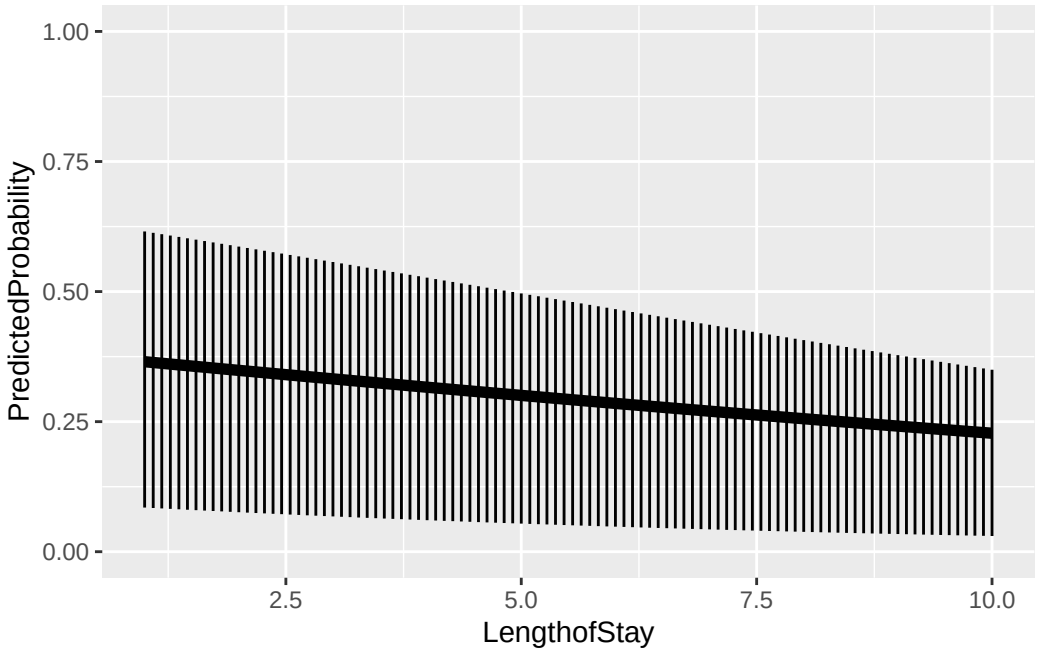
```
# add in LengthofStay values and convert to data frame
plotdat = as.data.frame(cbind(plotdat, jvalues))

# better names and show the first few rows
colnames(plotdat) = c("PredictedProbability", "Lower", "Upper", "LengthofStay")
head(plotdat)
```

```
  PredictedProbability      Lower     Upper LengthofStay
1            0.3652319 0.08489874 0.6155638     1.000000
2            0.3637056 0.08404676 0.6129535     1.090909
3            0.3621815 0.08320255 0.6103367     1.181818
4            0.3606597 0.08236605 0.6077135     1.272727
5            0.3591402 0.08153722 0.6050841     1.363636
6            0.3576230 0.08071600 0.6024486     1.454545
```

```
# plot average marginal predicted probabilities
ggplot(plotdat, aes(x = LengthofStay, y = PredictedProbability)) + geom_line(linewidth = 2
    ylim(c(0, 1))
```




```
ggplot(plotdat, aes(x = LengthofStay, y = PredictedProbability)) + geom_linerange(aes(ymin
    ymax = Upper)) + geom_line(linewidth = 2) + ylim(c(0, 1))
```



```
#################
```

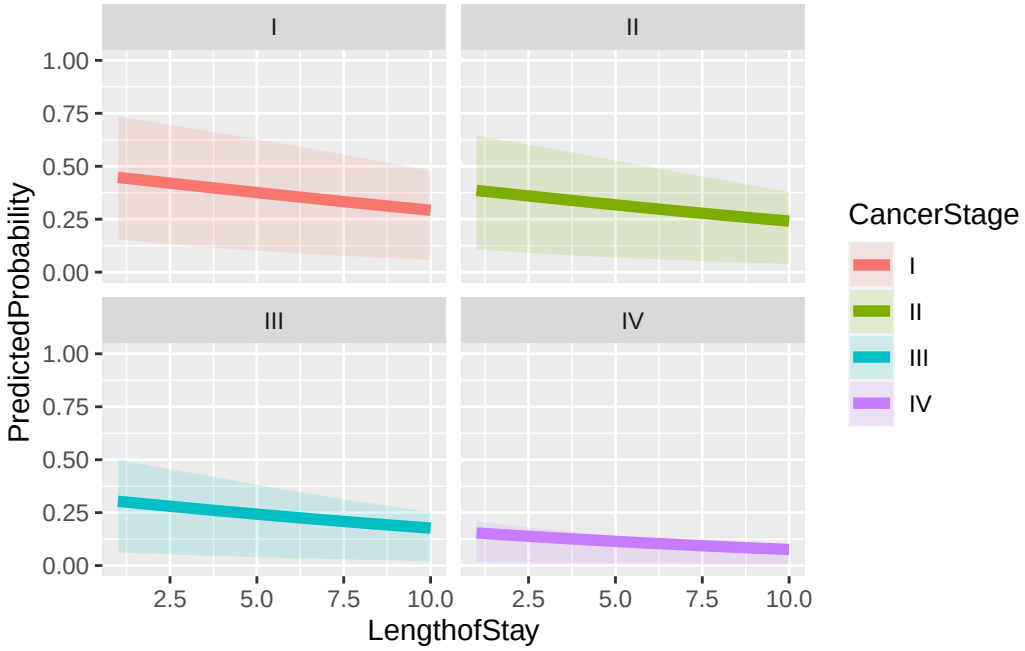
Now we can look at by stage:

```
# calculate predicted probabilities for each stage and store in a list
biprobs = lapply(levels(hdp$CancerStage), function(stage) {
  tmpdat$CancerStage = stage
  lapply(jvalues, function(j) {
    tmpdat$LengthofStay = j
    predict(model, newdata = tmpdat, type = "response")
  })
})

# get means and quartiles for all jvalues for each level of CancerStage
plotdat2 = lapply(biprobs, function(X) {
  temp = t(sapply(X, function(x) {
    c(M=mean(x), quantile(x, c(.25, .75)))
  }))
  temp = as.data.frame(cbind(temp, jvalues))
  colnames(temp) = c("PredictedProbability", "Lower", "Upper", "LengthofStay")
  return(temp)
})



# collapse to one data frame
plotdat2 = do.call(rbind, plotdat2)

# add cancer stage
plotdat2$CancerStage = factor(rep(levels(hdp$CancerStage), each = length(jvalues)))

# show first few rows
head(plotdat2)
```

```
  PredictedProbability     Lower     Upper LengthofStay CancerStage
1            0.4474662 0.1547407 0.7328360     1.000000           I
2            0.4458001 0.1533052 0.7306736     1.090909           I
3            0.4441352 0.1518807 0.7285001     1.181818           I
4            0.4424716 0.1504671 0.7263157     1.272727           I
5            0.4408092 0.1490643 0.7241204     1.363636           I
6            0.4391481 0.1476723 0.7219142     1.454545           I
```

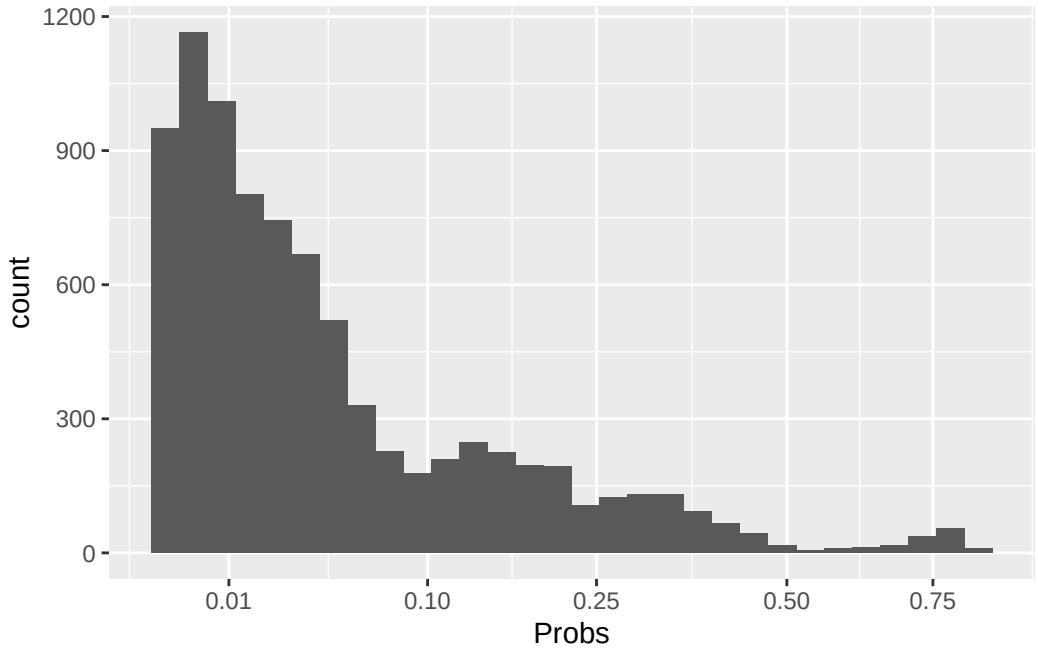
```
# graph it
ggplot(plotdat2, aes(x = LengthofStay, y = PredictedProbability)) +
  geom_ribbon(aes(ymin = Lower, ymax = Upper, fill = CancerStage), alpha = .15) +
  geom_line(aes(colour = CancerStage), size = 2) +
  ylim(c(0, 1)) + facet_wrap(~  CancerStage)
```

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




```
ggplot(data.frame(Probs = biprobs[[4]][[100]]), aes(Probs)) + geom_histogram() +
    scale_x_sqrt(breaks = c(0.01, 0.1, 0.25, 0.5, 0.75))
```

```
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



We may have considered the variation within hospital as well... Or toher variables etc.

```
# estimate the model and store results in m
m3a = glmer(remission ~ Age + LengthofStay + FamilyHx + IL6 + CRP +
  CancerStage + Experience + (1 | DID) + (1 | HID),
  data = hdp, family = binomial, nAGQ=1)
```

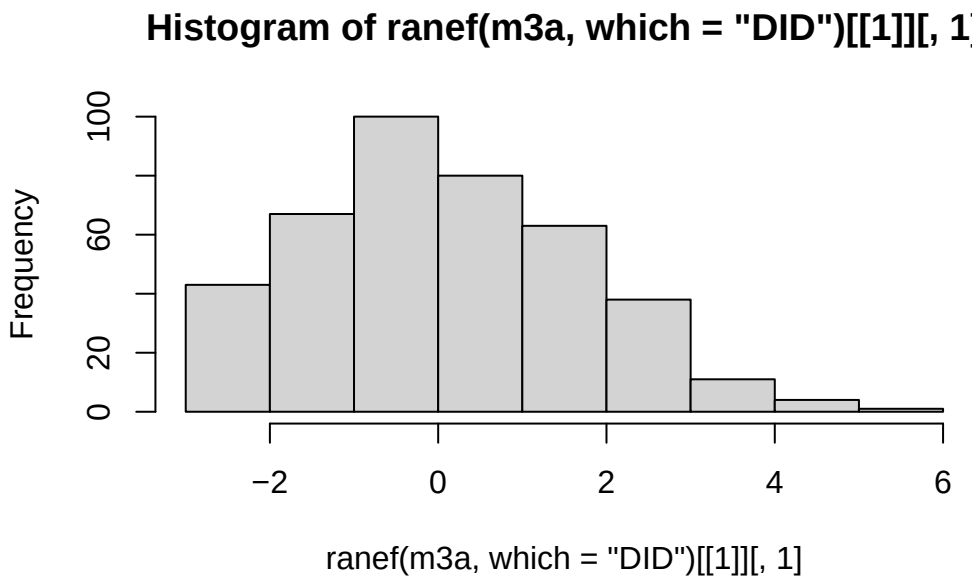
```
Warning in checkConv(attr(opt, "derivs"), opt$par, ctrl = control$checkConv, :
Model failed to converge with max|grad| = 0.393634 (tol = 0.002, component 1)
```

```
# print the mod results without correlations among fixed effects
print(m3a, corr=FALSE)
```

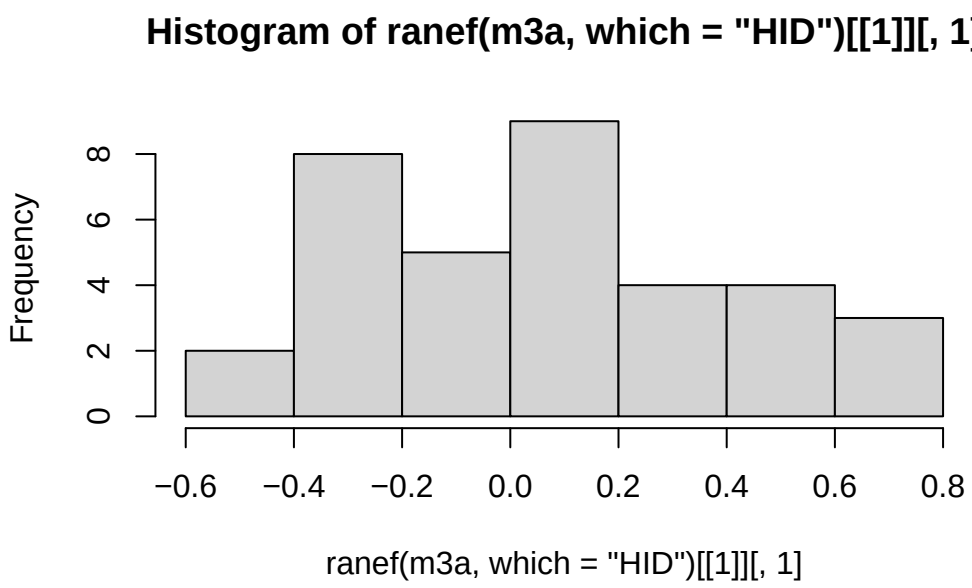
```
Generalized linear mixed model fit by maximum likelihood (Laplace
  Approximation) [glmerMod]
 Family: binomial  ( logit )
Formula: remission ~ Age + LengthofStay + FamilyHx + IL6 + CRP + CancerStage +
    Experience + (1 | DID) + (1 | HID)
   Data: hdp
      AIC       BIC    logLik  deviance  df.resid
```

```
 7199.081  7283.690 -3587.541  7175.081      8513
Random effects:
 Groups Name        Std.Dev.
 DID    (Intercept) 1.9513
 HID    (Intercept) 0.5432
Number of obs: 8525, groups:  DID, 407; HID, 35
Fixed Effects:
   (Intercept)             Age    LengthofStay     FamilyHxyes             IL6
      -1.68299        -0.01496        -0.04577        -1.30789        -0.05729
           CRP   CancerStageII  CancerStageIII   CancerStageIV      Experience
      -0.02209        -0.31739        -0.85462        -2.13138         0.12703
optimizer (Nelder_Mead) convergence code: 0 (OK) ; 0 optimizer warnings; 1 lme4 warnings
```

```r
hist(ranef(m3a, which = "DID")[[1]][,1])
```




```r
hist(ranef(m3a, which = "HID")[[1]][,1])
```




```
# lattice::dotplot(ranef(m3a, which = "DID", condVar = TRUE), scales = list(y = list(alter
# lattice::dotplot(ranef(m3a, which = "HID", condVar = TRUE))
```

Homework: Try this other model. What is the dotplot saying?

```
# estimate the model and store results in m
m3b = glmer(remission ~ Age + LengthofStay + FamilyHx + IL6 + CRP + CancerStage +
    Experience + (1 + LengthofStay | DID) + (1 | HID), data = hdp, family = binomial,
    nAGQ = 1)
```

```
Warning in checkConv(attr(opt, "derivs"), opt$par, ctrl = control$checkConv, :
Model failed to converge with max|grad| = 1.66877 (tol = 0.002, component 1)
```

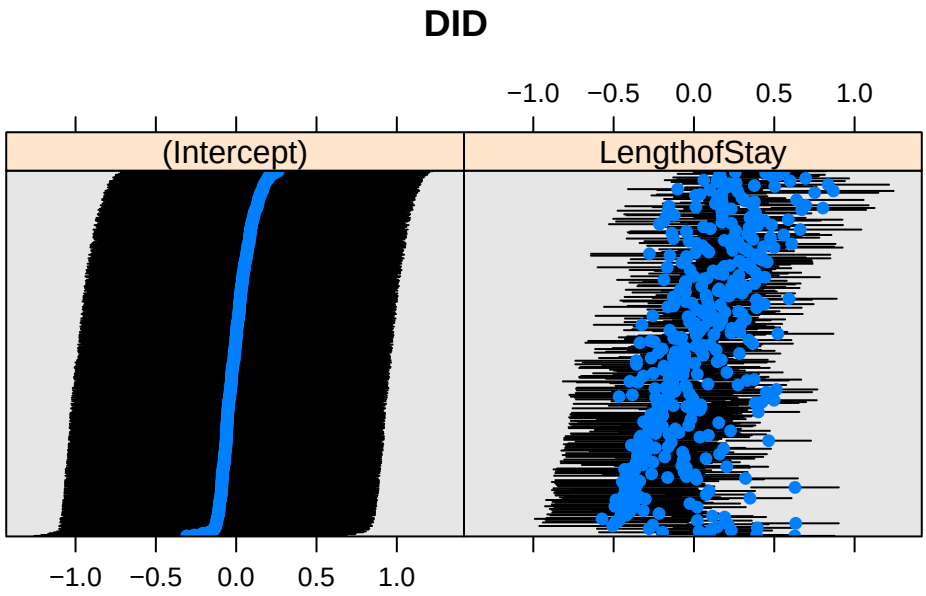
```
print(m3b, corr = FALSE)
```

```
Generalized linear mixed model fit by maximum likelihood (Laplace
  Approximation) [glmerMod]
 Family: binomial  ( logit )
Formula: remission ~ Age + LengthofStay + FamilyHx + IL6 + CRP + CancerStage +
    Experience + (1 + LengthofStay | DID) + (1 | HID)
```

```
   Data: hdp
      AIC       BIC    logLik  deviance  df.resid
 7147.749  7246.460 -3559.875  7119.749      8511
Random effects:
 Groups Name         Std.Dev. Corr
 DID    (Intercept)  0.5029
        LengthofStay 0.3729   -0.12
 HID    (Intercept)  0.7318
Number of obs: 8525, groups:  DID, 407; HID, 35
Fixed Effects:
   (Intercept)             Age    LengthofStay     FamilyHxyes             IL6
      -0.53725         -0.01523         -0.19062         -1.33822         -0.05865
           CRP   CancerStageII  CancerStageIII   CancerStageIV      Experience
      -0.02095         -0.29471         -0.86500         -2.30039          0.10412
optimizer (Nelder_Mead) convergence code: 0 (OK) ; 0 optimizer warnings; 1 lme4 warnings
```

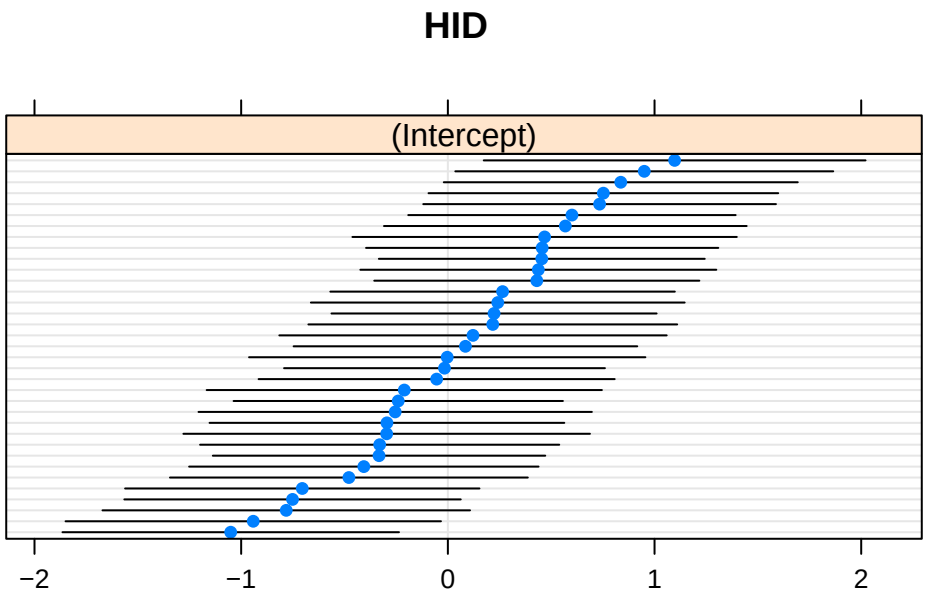
```
lattice::dotplot(ranef(m3b, which = "DID", condVar = TRUE), scales = list(y = list(alterna
```

```
$DID
```



```
lattice::dotplot(ranef(m3b, which = "HID", condVar = TRUE), scales = list(y = list(alterna
```

```
$HID
```



3 Permutation Tests

The following notes are adapted from these notes.

3.1 Introduction

Here, we briefly cover permutation tests. These are a class of nonparametric hypothesis tests for checking equality of distributions. Let's start with an example.

Suppose we sample n observations from F, denoted $(X_1, \dots, X_n)$ and m from G, denoted $(X_{n+1}, \dots, X_{n+m})$. Then, we wish to test

$$H_0 : F = G \qquad vs. \qquad H_1 : F \neq G.$$

Suppose we have some test statistic which is a measure of a difference between groups, for example, the Kolmogrov-Smirnoff statistic:

$$T^* = T(\ (X_1, \dots, X_n),\ (X_{n+1}, \dots, X_{n+m})\ ) = ||F_m - G_n||.$$

Here, T measures the distance between the empirical distributions from each sample. Recall then that if we want to do a hypothesis test, we need to compute the distribution of T under the null hypothesis.

Under the null hypothesis, we have that $(X_1, \dots, X_n)$ is equal in distribution to $(X_{n+1}, \dots, X_{n+m})$. This means that the labels 1,...,$n+m$ are arbitrary, in the sense that all observations come from the same population, so the division into the two groups holds no meaning. Furthermore, for any permutation $\pi_1, \dots, \pi_{n+m}$, it holds that $(X_{\pi_1}, \dots, X_{\pi_n})$ is equal in distribution to $(X_{\pi_{n+1}}, \dots, X_{\pi_{n+m}})$. This implies that $T(\ (X_1, \dots, X_n),\ (X_{n+1}, \dots, X_{n+m})\ )$ is equal in distribution to $T(\ (X_{\pi_1}, \dots, X_{\pi_n}),\ (X_{\pi_{n+1}}, \dots, X_{\pi_{n+m}})\ )$. This fact motivates the following procedure:

1. Draw a random permutation π from the set of $(n+m)!$ permutations
2. Compute $T(\ (X_{\pi_1}, \dots, X_{\pi_n}),\ (X_{\pi_{n+1}}, \dots, X_{\pi_{n+m}})\ )$
3. Repeat 1 and 2 K times to obtain K values $T_1, ..., T_K$
4. Let μ_K be the empirical distribution of $T^*, T_1, ..., T_K$
5. Compute the p-value: $\Pr_{\mu_K}(X \geq T^*)$

The above is a two-sample permutation test! Let's add some rigor.

3.2 The Permutation Lemma

Consider again, $X_1, \ldots, X_n \sim F$ and $X_{n+1}, \ldots, X_{n+m} \sim F$. Denote by $Y_1, \ldots, Y_{n+m}$ the order statistics of the combined sample $X_1, \ldots, X_{n+m}$. Now, let $V_i = 1(Y_i \in \{X_1, \ldots, X_m\})$.

Now, note that $Y_1, \ldots, Y_{n+m} = X_{\pi_1}, \ldots, X_{\pi_n}, X_{\pi_{n+1}}, \ldots, X_{\pi_{n+m}}$ for some permutation π. Furthermore, π has to be uniformly distributed across all possible permutations, on account of the X_i being iid. Therefore, the probability of a given permutation generated by the order statistics is $1/(n+m)!$.

Next, note each permutation generates a $(V_1, \ldots, V_{n+m})$. Now, the probability of observing any $(V_1, \ldots, V_{n+m})$ is the number of permutations such that the same m elements are in the first sample. This is given by $n!m!$. Given that the permutations are uniformly distributed, we have that the probability that $(V_1, \ldots, V_{n+m}) = (v_1, \ldots, v_{n+m}) = n!m!/(n+m)!$. Therefore, under the null hypothesis, every permutation is equally likely and so the distribution of the vector $(V_1, \ldots, V_n)$ is uniform over $\binom{n+m}{m}$. This has been referred to as the **permutation lemma**.

3.3 Adding in T

Now, we may ask where T comes in. Let $\mathbf{X}_1 = (X_1, \ldots, X_n)$ and $\mathbf{X}_2 = (X_{n+1}, \ldots, X_{n+m})$. Let $\mathbf{Y}$ be the combined sample order statistics and $\mathbf{V}$ be the label indicators. We can write:

$$\begin{aligned}
\Pr(T(\mathbf{X}_1, \mathbf{X}_2) \geq t | \mathbf{Y}) &= \Pr(T(\mathbf{V}, \mathbf{Y}) \geq t | \mathbf{Y}) \\
&= \sum_{i=1}^{\binom{n+m}{m}} \Pr(T(v, \mathbf{Y}) \geq t, \mathbf{V} = v | \mathbf{Y}) \\
&= \sum_{i=1}^{\binom{n+m}{m}} 1(T(v, \mathbf{Y}) \geq t) \Pr(\mathbf{V} = v | \mathbf{Y}) \\
&= \#\text{ labellings such that } T(v, \mathbf{Y}) \geq t \Big/ \binom{n+m}{m}.
\end{aligned}$$

This probability is: **the probability that we observe a value for our statistic at least as extreme as, assuming that the null hypothesis is true, given the set of values that we have observed.** Now then,

$$p = \#\text{ labellings such that } T(v, \mathbf{Y}) \geq t) \Big/ \binom{n+m}{m}$$

is the p-value, conditional on the combined sample order statistics we observed.

3.4 An example

```
set.seed(8235)
n=10
Y=rnorm(n)
X=rnorm(n,2)

TS=ks.test(X,Y)$statistic


permutation_test=function(X,Y,test_stat=function(x,y){ks.test(x,y)$statistic},K=1000){

  combined=c(X,Y)
  m=length(X)
  n=length(c(X,Y))
  permutations=replicate(K,sample(1:n,n))
  com_Tstar=function(permutation){test_stat(combined[permutation[1:m]],combined[permutatio
  Ts=apply(permutations,2,com_Tstar)
  Tss=test_stat(X,Y)
  p=mean(Tss<=c(Tss,Ts))
  return(p)
}

permutation_test(X,Y)
```

```
[1] 0.000999001
```

```
# How many samples do we need?



simulate_test=function(){
  Y=rnorm(n)
  X=rnorm(n,2)
  permutation_test(X,Y)}


mean(replicate(100,simulate_test())<= 0.05)
```

```
[1] 0.88
```

```
simulate_t_test=function(){
  Y=rnorm(n)
  X=rnorm(n,2)
  t.test(X,Y)$p.value}

mean(replicate(100,simulate_t_test())<= 0.05)
```

```
[1] 0.99
```

```
# A second exmaple

simulate_test=function(){
  Y=rexp(n)
  X=rexp(n,2)
  permutation_test(X,Y)}

simulate_test_2=function(){
  Y=rexp(n)
  X=rexp(n,2)
  permutation_test(X,Y,function(x,y){t.test(x,y)$statistic})
  }

simulate_t_test=function(){
  Y=rexp(n)
  X=rexp(n,2)
  t.test(X,Y)$p.value}

mean(replicate(50,simulate_test())<= 0.05)
```

```
[1] 0.08
```

```
mean(replicate(50,simulate_test_2())<= 0.05)
```

```
[1] 0
```

```
mean(replicate(50,simulate_t_test())<= 0.05)
```

```
[1] 0.24
```

A second example – heavy tails:

```
set.seed(8235)
n=35


simulate_test=function(){
  Y=rnorm(n)
  X=rt(n,3)
  permutation_test(X,Y)}

simulate_t_test=function(){
  Y=rnorm(n)
  X=rt(n,3)
  t.test(X,Y)$p.value}

mean(replicate(100,simulate_test())<= 0.05)
```

```
[1] 0.06
```

```
mean(replicate(100,simulate_t_test())<= 0.05)
```

```
[1] 0.07
```

Notes:

- We should take K as large as feasible. 1000 is a rule of thumb.
- The test statistic chosen has a large impact on the power. It is important to choose a test statistic that will perform well for the problem at hand. For instance, does it need to be robust, efficient computationally? What distributional differences are we most concerned about?
- Of course in simple problems, they will have lower power than optimal tests. However, they are suitable for situations where an optimal test is not easily derived, or the sample size is too low for asymptotic approximations. They are also easy to implement and relatively intuitive (you don't need to understand the CLT.)

- Permutation tests are mathematically valid because the data are exchangeable under the null hypothesis. We have to be careful that this is directly implied by our assumptions and null hypothesis. For instance, in the setup from the introduction, the null hypothesis $ E_F(X)=E_G(X)$ is not enough to give exchangeability of the data, since we have only assumed the data come from F and G, and it could be that $F \neq G$ but $E_F(X) = E_G(X)$. In that case, assuming H_0 alone is not enough to imply that $(X_1, \dots, X_n)$ is equal in distribution to $(X_{n+1}, \dots, X_{n+m})$. However, if in addition, we assume that F and G are in the family of normal distributions with variance 1, then we have that $(X_1, \dots, X_n)$ is equal in distribution to $(X_{n+1}, \dots, X_{n+m})$.

3.5 Permutation test for independence

Suppose that instead we observe $((X_1, Y_1), \dots, (X_n, Y_n)) \sim F_{XY}$, where $X_i \sim F_X$ and $Y_i \sim F_Y$. Suppose that we wish to test if X_i are independent of Y_i. One way to phrase this is

$$H_0 : F_{XY} = F_X F_Y \qquad vs. \qquad H_1 : F_{XY} \neq F_X F_Y.$$

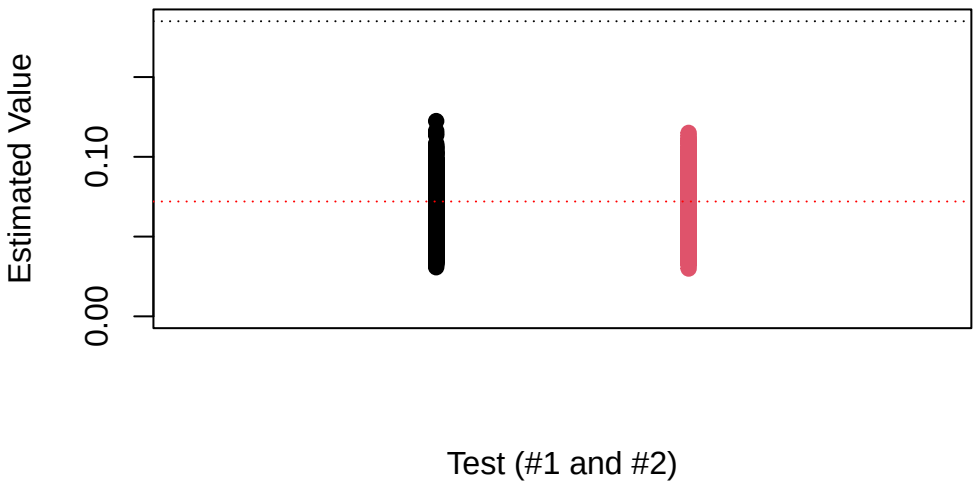
Now, under the null hypothesis, by definition, conditioning on X_i tells us nothing about the distribution of Y_i. Therefore, $((X_1, Y_1), \dots, (X_n, Y_n))$ is equal in distribution to $((X_1, Y_{\pi_1}), \dots, (X_n, Y_{\pi_n}))$ for any permuation π. Then, still under the null hypothesis, the pairings we observed were arbritrary. In fact, it is easy to see that the pairings are uniformly distributed accross the permutations of the Y_is. In this case, we draw many "permutation samples'' $((X_1, Y_{\pi_1}), \dots, (X_n, Y_{\pi_n}))$, and compute some statistic T which measures the dependence between X and Y. For instance, we may use

$$\sup_{(x_1, x_2) \in \mathbb{R}^2} |F_{XY}((x_1, x_2)) - F_X(x_1) F_Y(x_2)|.$$

This test is implemented in the `robusTest` package.

```
mv.ks.statistic = function(X, Y) {
  n = length(X)

  # The matrix under the assumption of independence is simply the
  # product of [i/n][j/n] for the (i,j)-th entry of the matrix
  indep_mat = as.matrix((1:n)/n) %*% t(1:n/n)

  # Return the maximum difference
  max(abs(robusTest::ecdf2D(X, Y)$ecdf - indep_mat))
}

set.seed(31415)
```

```r
# Generate X and Y dependent; X and W independent
K = 10000
n = 100
X = rnorm(n)
Y = 4*X + rnorm(n, 0, 3)
W = rexp(n)

# Compute the statistics
t1 = mv.ks.statistic(X, Y)
t2 = mv.ks.statistic(X, W)

results_mat = matrix(nrow = K, ncol = 2)

for(ii in 1:K) {
  Xs = X[sample(1:n, n)]
  results_mat[ii, ] = c(mv.ks.statistic(Xs, Y),
                        mv.ks.statistic(Xs, W))
}

# Plot the results
matplot(y = results_mat,
        x = matrix(c(rep(1, nrow(results_mat)),
                     rep(2, nrow(results_mat))),
                   nrow = nrow(results_mat)),
        pch = 19,
        ylab = "Estimated Value",
        xaxt = 'n',
        xlim = c(0, 3),
        ylim = c(0, max(c(results_mat, t1, t2))),
        xlab = 'Test (#1 and #2)')


abline(h = t1, lty = 3, col = 'black')
abline(h = t2, lty = 3, col = 'red')
```



We could just directly use the function: `robusTest::indeptest(X,Y)`.

For more information, see these notes.

References

Cabrera, Javier, and Andrew McDougall. 2002. *Statistical Consulting*. Springer New York. https://doi.org/10.1007/978-1-4757-3663-2.

Gelman, Andrew, and Jennifer Hill. 2006. *Data Analysis Using Regression and Multilevel/Hierarchical Models*. Cambridge University Press. https://doi.org/10.1017/cbo9780511790942.

Pinheiro, J. C., and D. Bates. 2009. *Mixed-Effects Models in s and s-PLUS*. Statistics and Computing. Springer. https://books.google.ca/books?id=y54QDUTmvDcC.

Wu, Lang. 2019. *Mixed Effects Models for Complex Data*. Chapman & Hall/CRC Monographs on Statistics & Applied Probability. Philadelphia, PA: Chapman & Hall/CRC.