GLS & LME

Lecture 6

LSM3257

AY22/23; Sem 2 | Ian Z.W. Chan

Summary (Learning Objectives)

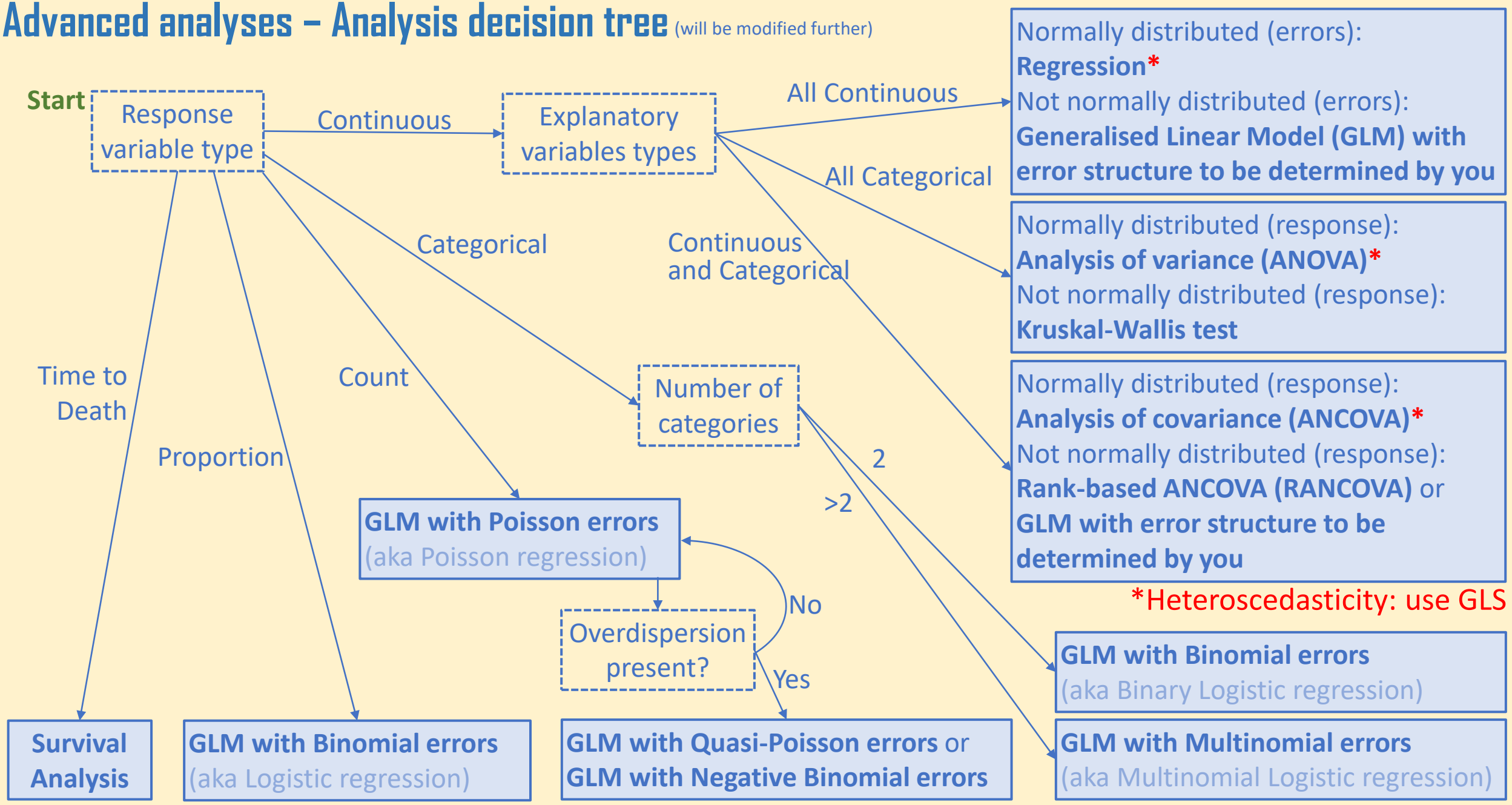
Generalised Least Squares (GLS)

- Purpose
- Types of variance structures: varIdent, varFixed, varPower, varExp, varConstPower, varComb
- Fitting, checking and comparing

Linear Mixed Effect (LME) models

- Random effects: what are they, why and when to use them?
- lmer() vs. lme()
- Fitting, checking and comparing
- Simplifying: random effects using REML, then fixed effects using ML

Advanced analyses – Analysis decision tree (will be modified further)



GLS

Generalised Least Squares

What is GLS?

the respnse variable having constant variance for all values of itself

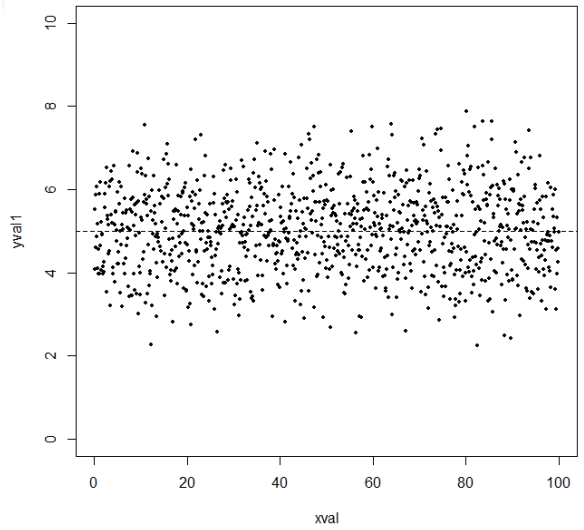
Used if:

one or more explanatory variable

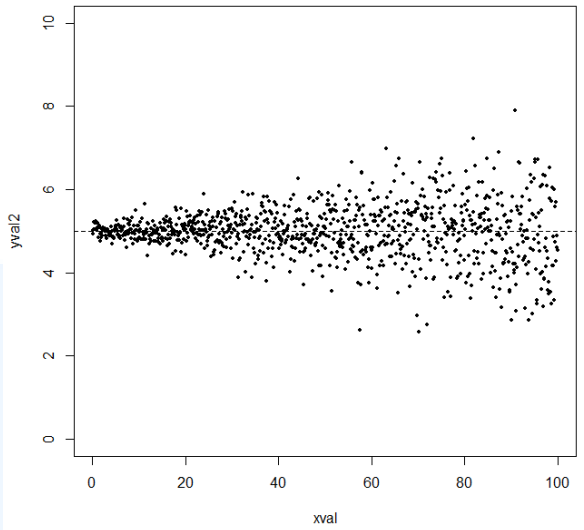
i) You're doing a linear model (i.e. a Regression, ANOVA or ANCOVA; you have one continuous response variable and one or more continuous or categorical response variables); and

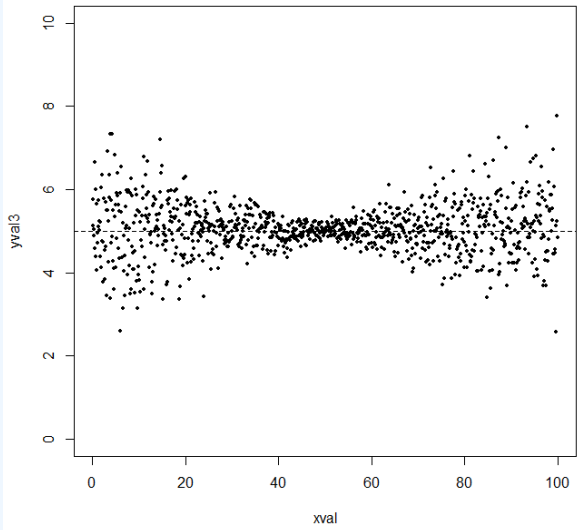
ii) Your model has a problem with heteroscedasticity: because variance changes with the values of a variable, R does not know how to pick the best model.

Homoscedasticity



Heteroscedasticity





equality of variance -> referes to categorical variable

Recall from previous lectures:

R fits the most likely model based on assumptions about: (i) the distribution of the errors and (ii) the variance in the dataset.

With GLS, you specify a variance structure (how the variance in the explanatory variable changes) and R will take this into account to fit the best model.

What is GLS?

Types of variance structures you can fit...

maybe Jan, Feb, Mar, alll diff

explanatory

Levels of variance (**varIdent**): variance changes with the levels of a categorical variable.

Fixed variance (**varFixed**): variance changes with the value of a continuous covariate (x-variable. Do not use if the covariate has negative values.

continuous explanatory variable increase -> variance increase/decrease
linear relationship

Power (**varPower**): variance changes with the power (e.g. x^{α}) of the covariate. Do not use if the covariate has 0 values.

exponential relationship

Constant+Power (**varConstPower**): variance changes with the power of the covariate plus a certain constant. Better than varPower if the covariate has values close to 0.

Exponential (**varExp**): variance changes with the exponent (e.g. α^{x}) of the covariate. Use this if the covariate has 0 values.

Combination (**varComb**): used to combine any of the relationships above.

if categorical -> varIdent, if continuous, fit all the variance structures and compare

R uses a few parameters (e.g. α) in the above. When you run the GLS, the computer will choose the best value based on maximum likelihood.

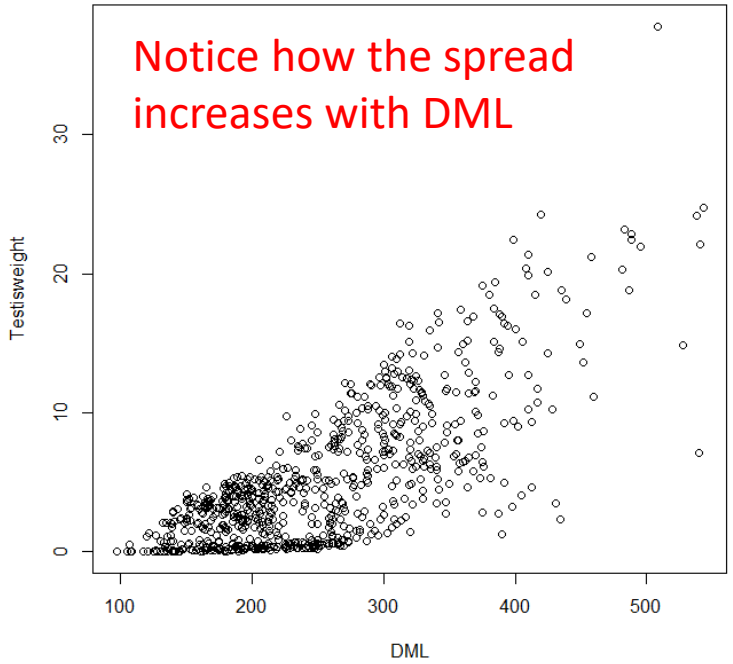
Finding variance structure problems in the dataset

We're using the Squid dataset from Zuur et al. (2009) where they use <MONTH> (categorical) and <DML> (Dorsal Mantle Length; continuous) to explain <Testisweight> (continuous): therefore an ANCOVA.

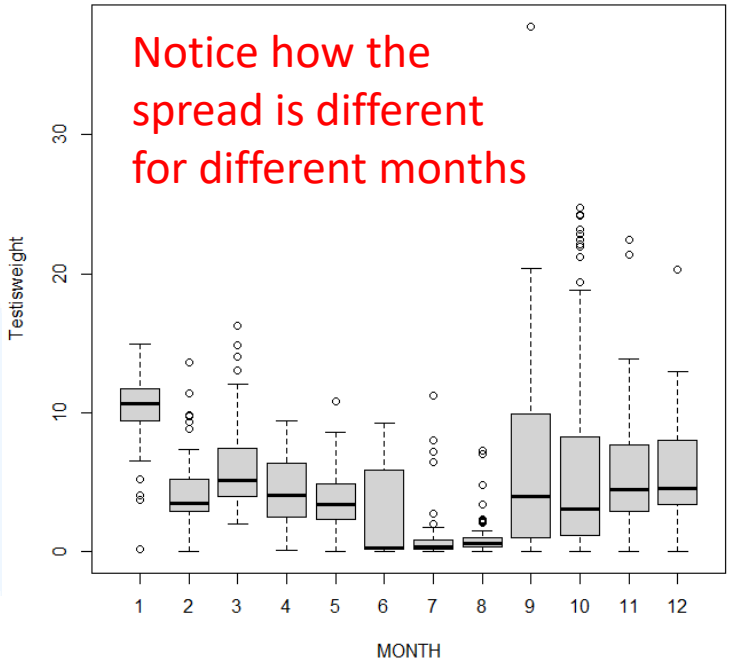
#Read in and visualise the dataset

```
d1=read.table("Squid.txt",header=T)
```

looks like problem with heteroschadicity and equality of variance

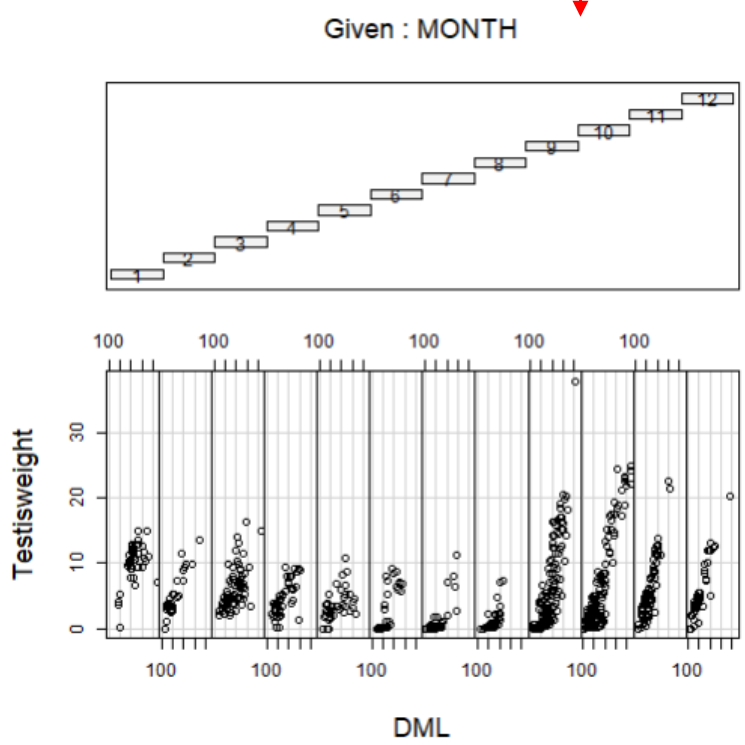



```
plot(Testisweight~DML,data=d1)
```




```
boxplot(Testisweight~MONTH,data=d1)
```

This coplot() is another useful plot to look for interactions between variables




```
coplot(Testisweight~DML|MONTH,data=d1,row=1)
```

Finding variance structure problems in the dataset

#Install the nlme package to do a GLS

```
require(nlme)
```

#Fit the ANCOVA and check the model

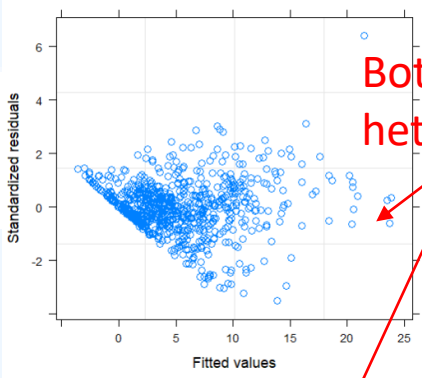
```
mod.lm=gls(Testisweight~DML*MONTH,data=d1)
plot(mod.lm)
```

only plot one heteroscedasticity

Without any "weights" specified, this is equivalent to a linear model, i.e. an ANCOVA (here) or ANOVA or regression

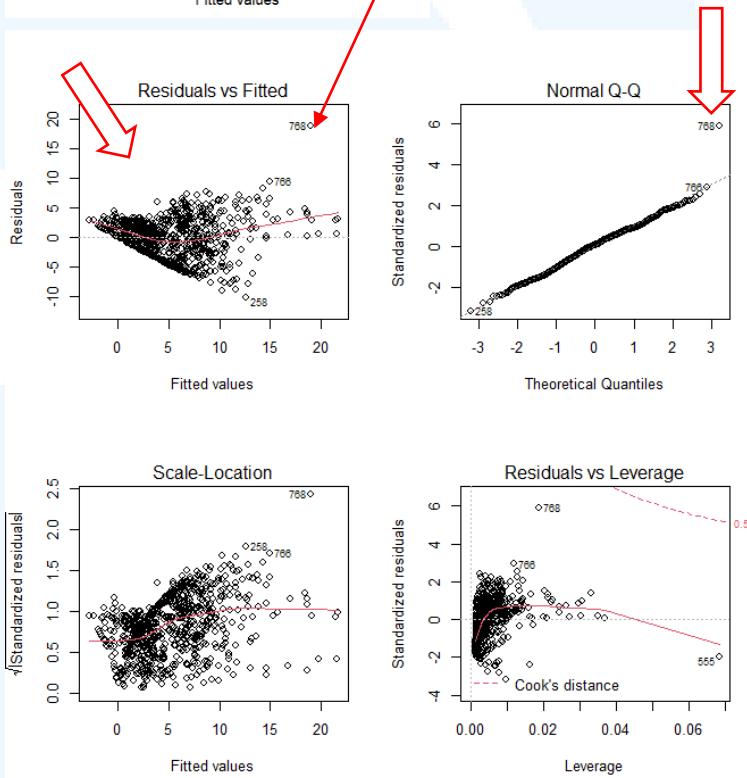
```
mod.lm2=lm(Testisweight~DML*MONTH,data=d1)
par(mfrow=c(2,2))
plot(mod.lm2)
```

plot(mod.lm)



Both show the same heteroscedasticity

plot(mod.lm2)



#Looks like problems with heteroscedasticity that changes with DML and MONTH

We could try transforming the variable, but we suspect (from the graphs) that variance is changing with the explanatory variables, so we decide to try fitting a GLS to solve the issue.

Fitting a GLS

#Specify variance structure for different levels of <MONTH> (categorical variable)

```
vs1=varIdent(form=~1|MONTH)
```

This is the format for specifying a categorical variable

#Fit the GLS

This "weights" command makes this a GLS and tells R how to account for the heteroscedasticity.

```
mod.ident=gls(Testisweight~DML*MONTH,data=d1,weights=vs1)
```

#Compare the two models

```
AIC(mod.lm,mod.ident)
```

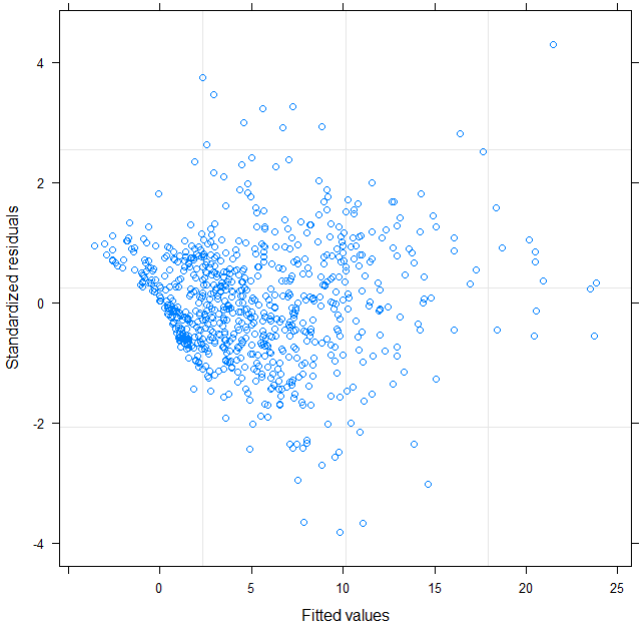
```
> AIC(mod.lm,mod.ident)
          df      AIC
mod.lm    25 3641.877
mod.ident 36 3614.436
Warning message:
In AIC.default(mod.lm, mod.ident) :
  models are not all fitted to the same number of observations
```

AIC for mod.Ident is lower so it is better!

#Check results

```
summary(mod.ident)
```

```
plot(mod.ident)
```




```
Variance function:
 Structure: Different standard deviations per stratum
 Formula: ~1 | MONTH
 Parameter estimates:
        2         9        12        11         8        10         5
1.0000000 2.9913588 1.2736165 1.5090471 0.9821378 2.2162169 1.6396315
        7         6         4         1         3
1.3783514 1.6473098 1.4232366 1.9584902 1.9788666
```

The plot suggests there is still more variance structure to account for.

These numbers compare the variance of each month to month 1 (in the boxplots, month 9 is the biggest).

for diffferent months, they are multiplying this number

Fitting a GLS

#Specify variance structure for <DML> (continuous variable)

```
vs2=varFixed(~DML)
```

If we're using varFixed, we don't need the "form=" and the vertical bar (note: varFixed cannot take categorical variables).

#Fit the GLS

```
mod.fixed=gls(Testisweight~DML*MONTH,data=d1,weights=vs2)
```

#Compare the models

```
AIC(mod.lm,mod.ident,mod.fixed)
```

#mod.ident is still the best so far

```
> AIC(mod.lm,mod.ident,mod.fixed)
           df      AIC
mod.lm     25 3641.877
mod.ident  36 3614.436
mod.fixed  25 3620.898
Warning message:
In AIC.default(mod.lm, mod.ident, mod.fixed) :
  models are not all fitted to the same number of observations
```

Fitting a GLS

#Specify power variance structure for <DML>

```
vs3a=varPower(form=~DML)
```

random slope | random intercept

#Specify power variance structure and also allow it to vary with <MONTH>

```
vs3b=varPower(form=~DML|MONTH)
```

The left side of the bar usually specifies the slope of a linear equation, and is for continuous variables.

The right side of the bar usually specifies the intercept of a linear equation and is for categorical variables.

Together, this code allows the effect of DML to be different in different months: we're accounting for both a categorical variable and a continuous variable at the same time

#Do the same for constant+power and exponential variance structures for <DML>

```
vs4a=varConstPower(form=~DML)
vs4b=varConstPower(form=~DML|MONTH)
vs5a=varExp(form=~DML)
vs5b=varExp(form=~DML|MONTH)
```

Fitting a GLS

#Fit all the models and compare

```
mod.powA=gls(Testisweight~DML*MONTH,data=d1,weights=vs3a)
mod.powB=gls(Testisweight~DML*MONTH,data=d1,weights=vs3b)
mod.conPowA=gls(Testisweight~DML*MONTH,data=d1,weights=vs4a)
mod.conPowB=gls(Testisweight~DML*MONTH,data=d1,weights=vs4b)
mod.expA=gls(Testisweight~DML*MONTH,data=d1,weights=vs5a)
mod.expB=gls(Testisweight~DML*MONTH,data=d1,weights=vs5b)
AIC(mod.ident,mod.powA,mod.powB,mod.conPowA,mod.conPowB,mod.expA,mod.expB)
```

	df	AIC
mod.ident	36	3614.436
mod.powA	26	3473.019
mod.powB	37	3407.511
mod.conPowA	27	3475.019
mod.conPowB	49	3431.511
mod.expA	26	3478.152
mod.expB	37	3419.719

mod.powB looks the best

#Check what mod.powB is doing

```
summary(mod.powB)
```

```
Variance function:
 Structure: Power of variance covariate, different strata
 Formula: ~DML | MONTH
 Parameter estimates:
       2        9       12       11        8       10        5        7
1.728526 1.789499 1.733557 1.749259 1.617693 1.789182 1.746517 1.673240
       6        4        1        3
1.754484 1.711367 1.698942 1.722739
```

These are the powers applied to <DML> for each month (e.g. in month 2, the variance is a factor of $DML^{1.72}$).

Fitting a GLS

#Combining the various functions

```
vs6=varComb(varFixed(~DML),varExp(form=~DML|MONTH))
```

First structure

Second structure

```
mod.combi=gls(Testisweight~DML*MONTH,data=d1,weights=vs6)
AIC(mod.powB,mod.combi)
```

```
> AIC(mod.powB,mod.combi)
           df      AIC
mod.powB   37 3407.511
mod.combi  37 3415.297
```

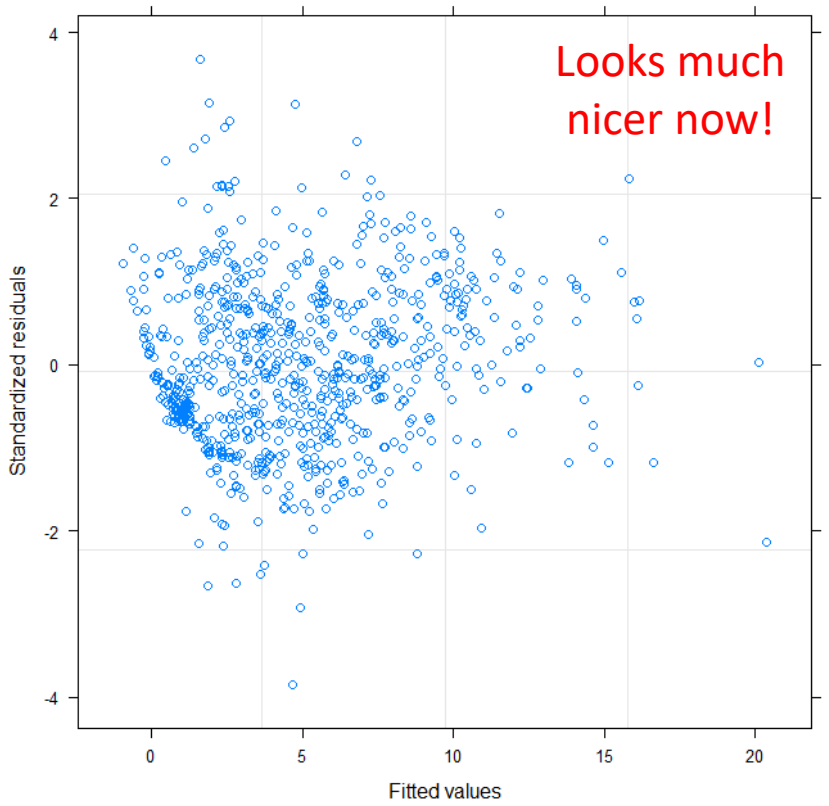
#Check mod.powB

```
plot(mod.powB)
```

#Now proceed with model simplification and comparison using AIC()

```
summary(mod.powB)
```

#A little harder to read but the information is there, including coefficients and p-values



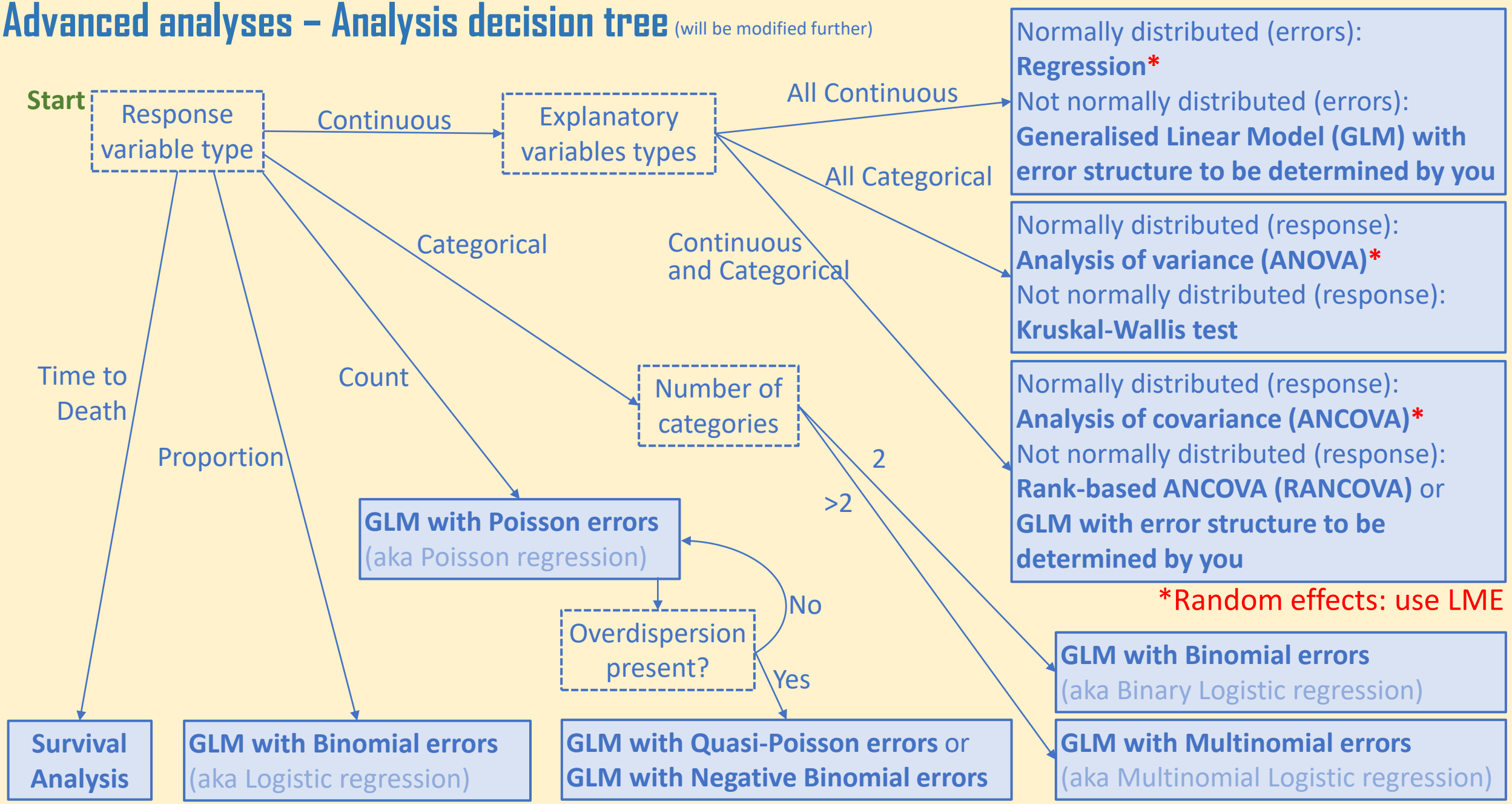
What if GLS doesn't solve my heteroscedasticity problems

Your original normal error distribution assumption may be fatally flawed: therefore do a **Generalised** LM (GLM; in later lectures)

LME

Linear Mixed Effect models

Advanced analyses – Analysis decision tree (will be modified further)



What are Random Effects?

"Random effects" are NOT RANDOM.

So far when we fit a(n) regression/ANOVA/ANCOVA, we partition the variation to:

1) the variables we are interested in (the main explanatory variables aka **Fixed effects**).

2) the residuals (Errors due to random chance).

Additionally, an LME allows us to partition the variation to:

3) variables which we expect to have an effect but which we are not interested in for this particular experiment/research question (the **Random effects**).

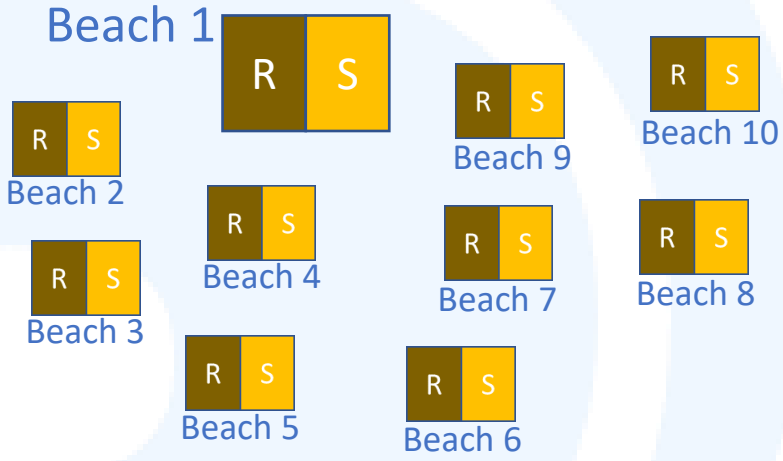
- A particular variable can be a Fixed effect in one experiment and a Random effect in another—it all depends on your research question.

Random effects can be categorical ("random intercept") or continuous ("random slope").

Examples of Random Effects

Experiment 1: I measure the effect of shore substrate (**R**ocky vs. **S**andy) on the number of crabs at 10 different beaches.

- Response variable: number of crabs.
- Explanatory variable (Fixed effect): substrate.
- Random effect: 10 beaches.



Experiment 2: I measure the effect of the location of 10 beaches on the number of crabs at the beach, and control for the effects of shore substrate.

- Response variable: number of crabs.
- Explanatory variable: 10 different beaches.
- Random effect: substrate (but not ideal because < 5 levels).

Examples of Random Effects

Experiment 3: I use the **number of people living near a protected area** to explain the **success of the protected area** in **10 different countries** with varying **GDP**.

Response variable: success of protected area.

Explanatory variable: Answer

Random effects: Answer

Experiment 4: I test the effects of **butterfly colour** (brown and black) and **light levels** (dim and bright) on **mantid attack rates** using **25 different mantids**.

Response variable: attack rate.

Explanatory variables: Answer

Random effects: Answer

Why include Random Effects?

Accounting for the variation caused by these effects will produce a **better model** of our fixed effects: **more realistic** (a better approximation of what is happening in reality) and **more powerful** (better able to find a true effect).

Why include them as Random effects?

Random effects are estimated differently from fixed effects (Linear unbiased prediction instead of Maximum Likelihood): fewer parameters are estimated, multiple comparisons are avoided and precious Degrees of Freedom are saved.

2 Guidelines for using random effects

- Random effects are important variables that are NOT your research question.

- In general, you want categorical random effects to have at least 5 levels (if the variable has < 5 levels, it is probably better to include it as a fixed effect).

Fitting Linear Mixed Effect models – linear models with random effects

Two functions from two different packages:

1) lme4 package, **lmer()** function.

Pros: Can have multiple independent random effects. Faster.

Cons: Cannot specify variance structure. Does not give p-values by default, need to install the lmerTest package as well.

```
#Example code: mod.lme=lmer(y~x1+x2*x3/x4+(1|A/B/C)+(E|D))
```

Adding a second random effect with a random slope "E"

Fixed effects. Can include interactions and nestedness as we have previous learnt

Random effects for variables A, B and C specified like this. C is nested in B is nested in A (e.g. C is leaves, B is trees, A is forest)

2) nlme package, **lme()** function.

Pros: You can also specify variance structure using "weights=" like a GLS.

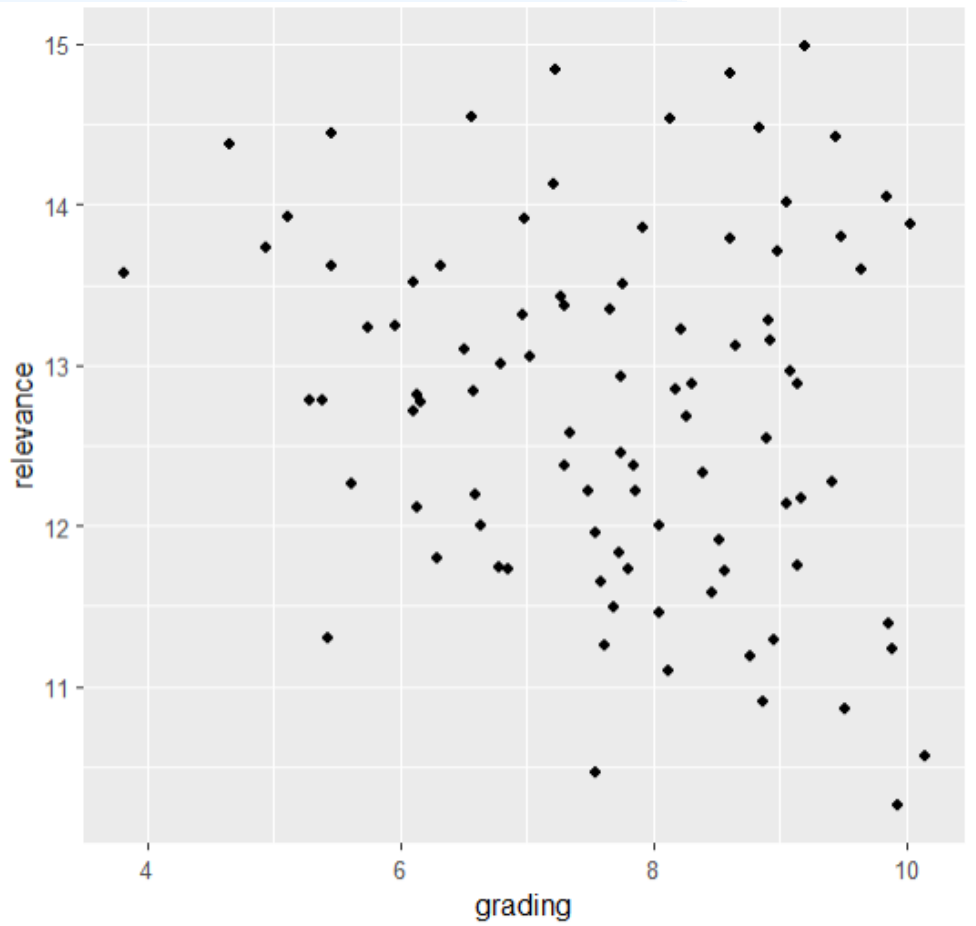
Cons: Cannot have multiple random intercepts (except if they are nested).

```
#Example code: mod.lme=lme(y~x1+x2*x3/x4,random=~1|A/B/C)
```

Fitting LMEs

#Prepare the dataset

```
d2=read_excel("relevance.xlsx")
d2$origin=as.factor(d2$origin)
d2$process=as.factor(d2$process)
library(ggplot2)
ggplot(data=d2,aes(x=grading,y=relevance))+
geom_point()
```



#Explain <relevance> using <grading>:

#Normal regression

```
mod1=lm(relevance~grading,data=d2)
summary(mod1)
```

```
> summary(mod1)

Call:
lm(formula = relevance ~ grading, data = d2)

Residuals:
     Min       1Q   Median       3Q      Max
-2.30065 -0.86516 -0.03685  0.63938  2.48045

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 13.92287    0.62701  22.205   <2e-16 ***
grading     -0.15431    0.08027  -1.922   0.0577 .
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Without random effects, the effect of <grading> is marginal

Fitting LMEs – 1 random effect (categorical)

lme4 package

```
require(lme4)
require(lmerTest) #for p-values
```

nlme package

```
require(nlme)
```

Maybe <origin> will help to explain:

```
#1 categorical random effect
mod2a=lmer(relevance~grading+
(1|origin),data=d2)
```

```
#1 categorical random effect
mod2b=lme(relevance~grading,
random=~1|origin,data=d2)
```

<origin> is now included as a random effect in the model. Note where it is inserted because it is a categorical variable.

Fitting LMEs – interpreting results

lme4 package

`summary(mod2a)`

```
> summary(mod2a)
Linear mixed model fit by REML. t-tests use
  Satterthwaite's method [lmerModLmerTest]
Formula: relevance ~ grading + (1 | origin)
   Data: d2

REML criterion at convergence: 277.7

Scaled residuals:
     Min       1Q   Median       3Q      Max
-1.97887 -0.90830 -0.00838  0.63659  2.19413

Random effects:
 Groups   Name        Variance Std.Dev.
 origin   (Intercept) 0.1026   0.3203
 Residual             1.1298   1.0629
Number of obs: 92, groups:  origin, 2

Fixed effects:
            Estimate Std. Error       df t value
(Intercept) 14.19631    0.66866 31.22828  21.231
grading     -0.18990    0.08061 89.97977  -2.356
            Pr(>|t|)
(Intercept)   <2e-16 ***
grading       0.0207 *
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Correlation of Fixed Effects:
        (Intr)
grading -0.926
```

`require(MuMIn)`

`r.squaredGLMM(mod2a)`

```
> r.squaredGLMM(mod2a)
              R2m       R2c
[1,] 0.05556494 0.1342041
```

Variation explained by the random effect

Residual variation

Now significant!

R2m: marginal R^2, the variation explained by the **fixed effects**

R2c: conditional R^2, the variation explained by the **fixed + random effects**

nlme package

`summary(mod2b)`

```
> summary(mod2b)
Linear mixed-effects model fit by REML
  Data: d2
       AIC     BIC    logLik
  285.7207 295.72 -138.8604

Random effects:
 Formula: ~1 | origin
        (Intercept) Residual
StdDev:   0.3203465 1.062942

Fixed effects:  relevance ~ grading
                Value Std.Error DF   t-value p-value
(Intercept) 14.196307 0.6686619 89 21.230920  0.0000
grading     -0.189904 0.0806112 89 -2.355798  0.0207
 Correlation:
        (Intr)
grading -0.926

Standardized Within-Group Residuals:
         Min           Q1          Med           Q3
-1.978871578 -0.908302014 -0.008382594  0.636584922
         Max
 2.194128304

Number of Observations: 92
Number of Groups: 2
```

`require(MuMIn)`

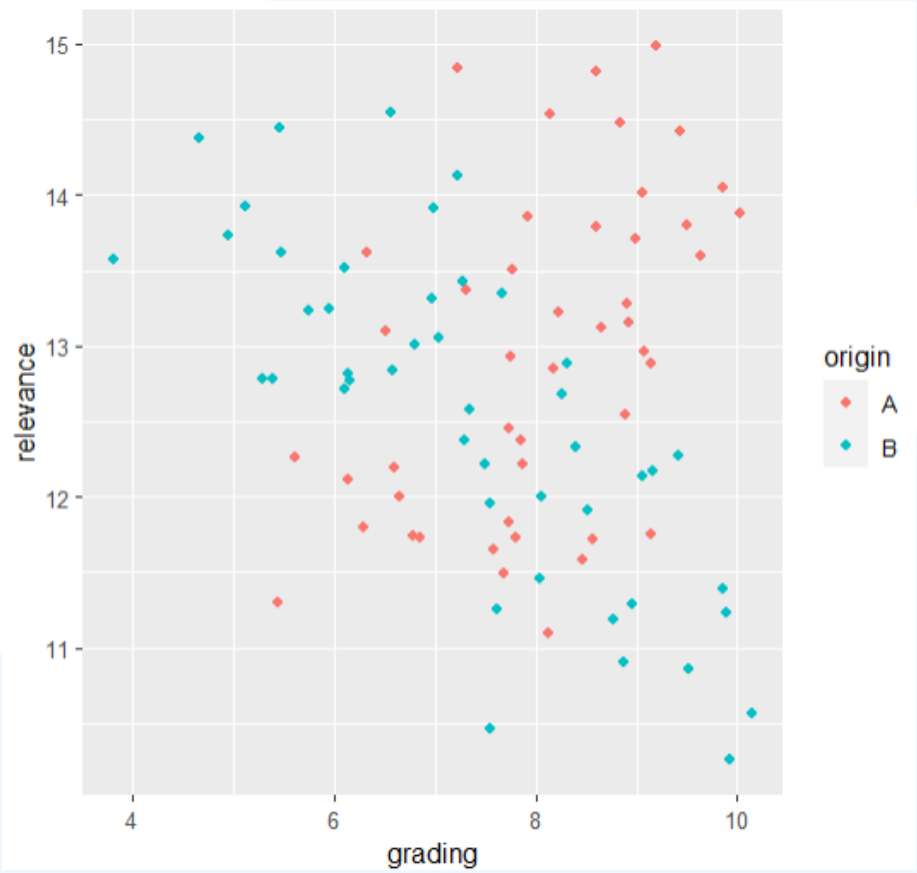
`r.squaredGLMM(mod2b)`

```
> r.squaredGLMM(mod2b)
              R2m       R2c
[1,] 0.05556494 0.1342037
```

```
> summary(mod1)

Call:
lm(formula = relevance ~ grading, data = d2)

Residuals:
     Min       1Q   Median       3Q      Max
-2.30065 -0.86516 -0.03685  0.63938  2.48045

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 13.92287    0.62701  22.205   <2e-16 ***
grading     -0.15431    0.08027  -1.922   0.0577 .
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Fitting LMEs – interpreting results

Intuitively, adding <origin> as a random effect accounts for the fact that there are two different levels of <origin>: A and B

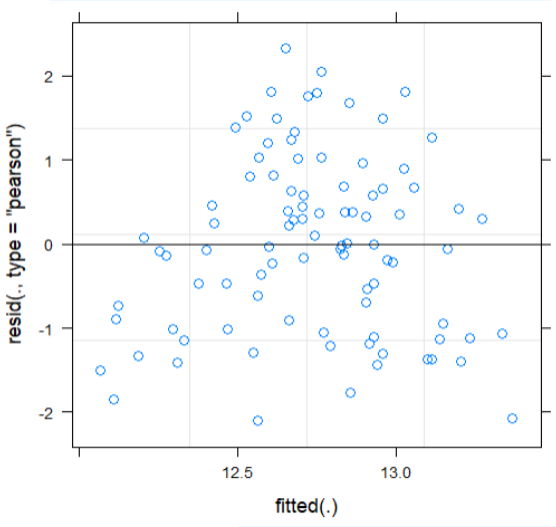


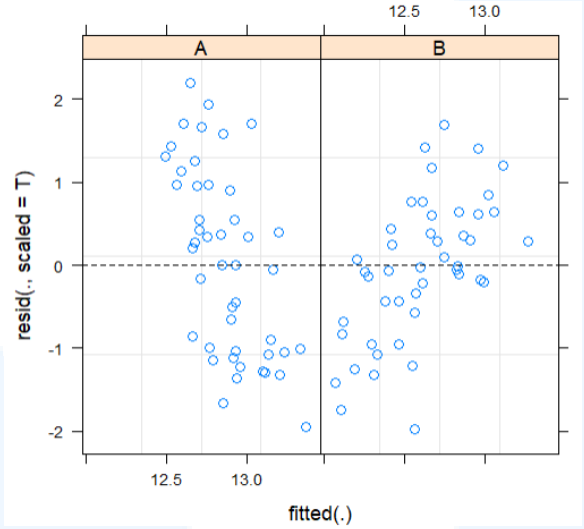
Test your understanding!: if you had fit <origin> as a fixed effect, do you think <grading> and <origin> would have a significant interaction?

Fitting LMEs – diagnostic plots

lme4 package

```
#Variance for whole dataset
plot(mod2a)
```

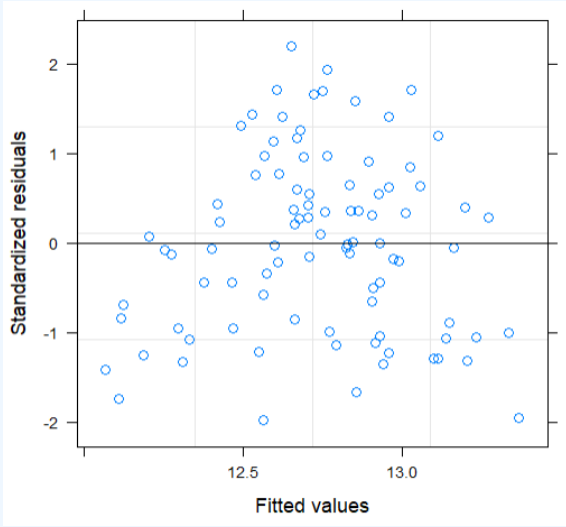


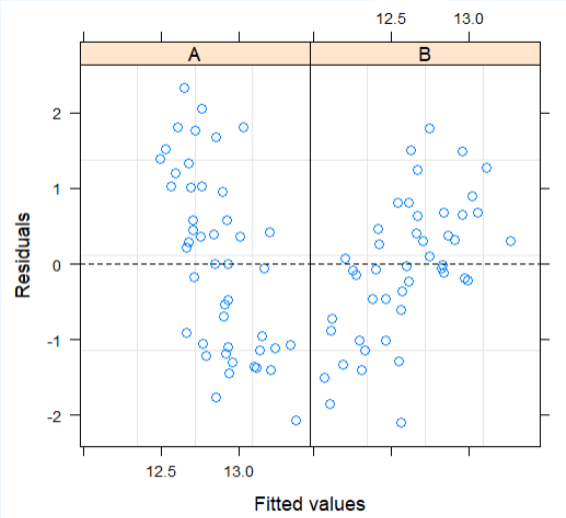



```
require(lattice)
#Variance for levels of random effect
plot(mod2a,resid(.,scaled=T)~fitted(.)
|origin,abline=0,lty=2)
```

nlme package

```
#Variance for whole dataset
plot(mod2b)
```



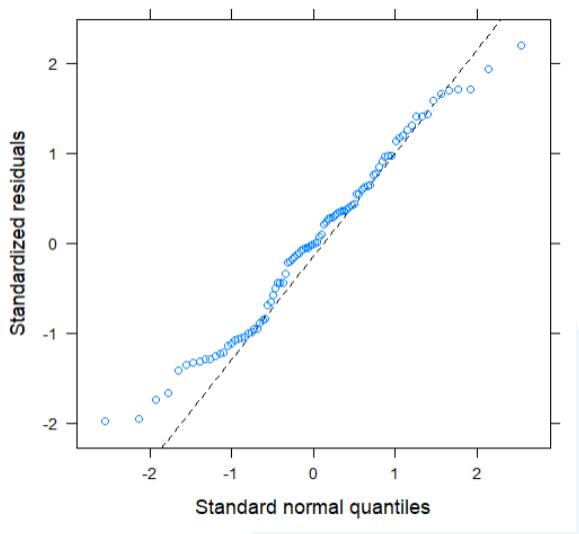


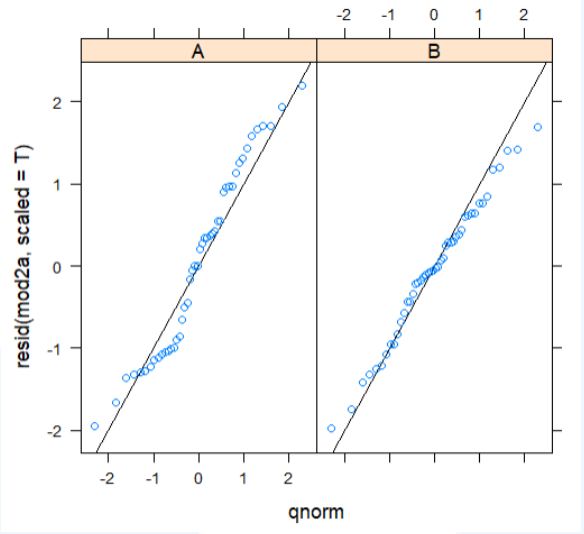

```
require(lattice)
#Variance for levels of random effect
plot(mod2b,resid(.,scaled=T)~fitted(.)
|origin,abline=0,lty=2)
```

Fitting LMEs – diagnostic plots

lme4 package

```
#Normality for whole dataset
qqmath(mod2a,lty=2)
```

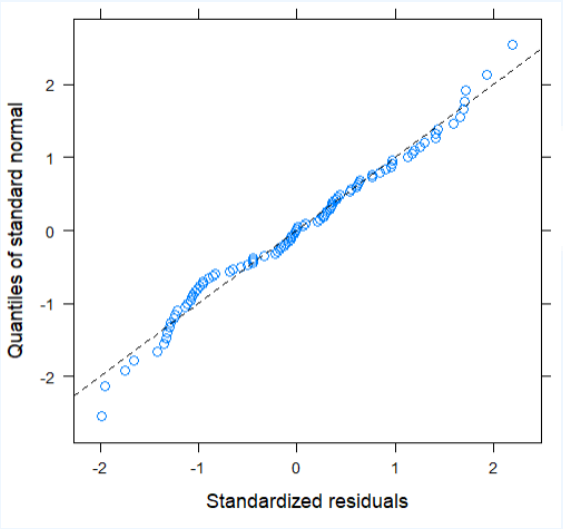


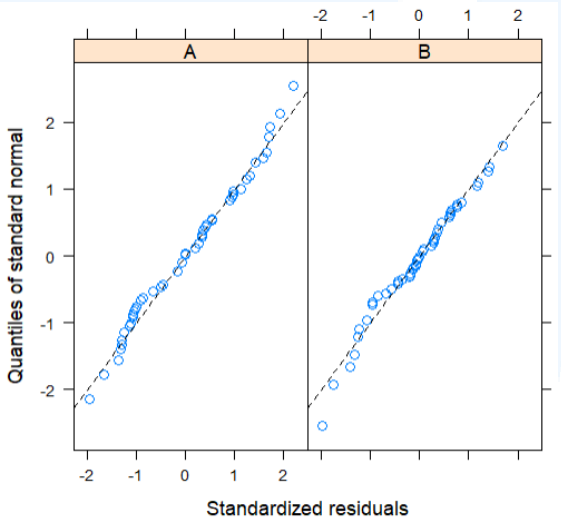



```
#Normality for levels of random effect
qqmath(~resid(mod2a,scaled=T)|origin,
data=d2,abline=c(0,1))
```

nlme package

```
#Normality for whole dataset
qqnorm(mod2b,abline=c(0,1),lty=2)
```






```
#Normality for levels of random effect
qqnorm(mod2b,~resid(.,type="p")|origin
,abline=c(0,1),lty=2)
```

What if heteroscedasticity is present?

Option 1: transform the y-variable.

Option 2: fit variance structures using lme() (using “weights”, similar to GLS: just add a “weights=” argument to the model).

Option 3: use GLM.

What if residuals are non-normal?

Option 1: transform the y-variable.

Option 2: use GLM.

Fitting LMEs – 2 nested and 2 separate random effects (categorical)

lme4 package

#Nested categorical random effects

```
mod3a=lmer(relevance~grading+(1|origin/process),data=d2)

summary(mod3a)
```

This tell tells R that <process> is nested within <origin>

#Separate random effects

```
mod4a=lmer(relevance~grading+(1|origin)*(1|process),data=d2)
```

Just add more variables using "+" or "*" (for interacting random effects, though this is rare)

#Compare the two models

```
AIC(mod3a,mod4a)
```

nlme package

#Nested categorical random effects

```
mod3b=lme(relevance~grading,random=~1|origin/process,data=d2)

summary(mod3b)
```

#Separate random effects

lme() is not able to do this.

```
> mod3a=lmer(relevance~grading+(1|origin/process),data=d2)
boundary (singular) fit: see ?isSingular
> summary(mod3a)
Random effects:
 Groups         Name        Variance Std.Dev.
 process:origin (Intercept) 0.0000   0.0000
 origin         (Intercept) 0.1026   0.3203
 Residual                   1.1298   1.0629
optimizer (nloptwrap) convergence code: 0 (OK)
boundary (singular) fit: see ?isSingular
```

A "boundary fit" doesn't necessarily mean that there's something wrong: here it is because process:origin doesn't explain anything, so you could remove it from the model

Fitting LMEs – 2 random effects (1 continuous and 1 categorical)

lme4 package

```
mod5a=lmer(relevance~grading+(time|origin/process),data=d2)
```

Continuous variable goes in front of the bar: this adjusts the slope for <time> for the different levels of <origin> and <process>

nlme package

```
mod5b=lme(relevance~grading,random=~time|origin/process,data=d2)
```

#Convergence problems: no results/cannot trust results!

```
> mod5b=lme(relevance~grading,random=~time|origin/process,da
ta=d2) #convergence problems!
Error in lme.formula(relevance ~ grading, random = ~time | o
rigin/process,  :
  nlminb problem, convergence error code = 1
  message = iteration limit reached without convergence (10)
In addition: There were 50 or more warnings (use warnings()
 to see the first 50)
```

Convergence problems are frustratingly common in LMEs, GLMs and GLMMs. This usually happens when your **model is too complex** for the package to handle. You CANNOT trust the results! You must adjust your model until there are no more convergence problems.

There are some ways you can handle this issue (I will go through these on another week):

1) Fit a simpler model (fewer explanatory variables and/or random effects);
2) Adjust some of the settings in the function;
3) Try another package;
4) Choose a different response variable.

We will take a closer look at this in a later lecture.

Note: continuous random effects are less common than categorical. If you only want to fit one continuous random effect, fit it as a fixed effect.

Simplifying mixed effect models

We simplify the random effects first because we don't want them to explain any of the variation that our fixed effects can explain.

FIRST: simplify random effects—fit models using **REML (Restricted Maximum Likelihood which ignores fixed effects)** and compare models using AIC() .

NEXT: simplify fixed effects—fit models using **ML (Maximum Likelihood)** and compare models using either anova() (only if they're subsets) or AIC() (preferred).

lme4 package

#Fitting models using REML

```
mod3a=lmer(relevance~grading+(1|origin
/process),data=d2,REML=T)
```

REML=T is the default, so you actually don't need to specify this

#Fitting models using ML

```
mod3a=lmer(relevance~grading+(1|origin
/process),data=d2,REML=F)
```

nlme package

#Fitting models using REML

```
mod3b=lme(relevance~grading,random=~1|
origin/process,data=d2,method="REML")
```

method="REML" is the default, so you actually don't need to specify this

#Fitting models using ML

```
mod3b=lme(relevance~grading,random=~1|
origin/process,data=d2,method="ML")
```

Example – simplify random effects first

Compare models with 3 random effects structures: (i) no random effects (mod6a); (ii) <origin>; and (iii) <process> nested within <origin>.

lme4 package

#Fit models using REML

```
mod7a1=lmer(relevance~grading+time+
(1|origin),data=d2)

mod7a2=lmer(relevance~grading+time+
(1|origin/process),data=d2)
```

#Test models using AIC()

```
AIC(mod6a,mod7a1,mod7a2)
```

```
> AIC(mod6a,mod7a1,mod7a2)
       df      AIC
mod6a   4 281.3751
mod7a1  5 294.9092
mod7a2  6 296.9092
```

nlme package

#Fit models using REML

```
mod6a=lm(relevance~grading+time,data=d2)

mod7b1=lme(relevance~grading+time,
random=~1|origin,data=d2)

mod7b2=lme(relevance~grading+time,
random=~1|origin/process,data=d2)
```

#Test models using AIC()

```
AIC(mod6a,mod7b1,mod7b2)
```

```
> AIC(mod6a,mod7b1,mod7b2)
       df      AIC
mod6a   4 281.3751
mod7b1  5 294.9092
mod7b2  6 296.9092
```

In both, the model with no random effects is the best so we could choose the lm() with no random effect structure. But if we feel that including <origin> is more accurate biologically (i.e. in real life), we can choose to keep it: *"the art of data analysis."*

Example – simplify fixed effects after

Now we use the random effect structure we chose in the previous step, **fit a new model using "ML"**, and simplify using stepwise deletion.

lme4 package

#Re-fit using ML

```
mod8a=lmer(relevance~grading+time+(1|origin),data=d2,REML=F)
summary(mod8a) #<time> can be removed
mod9a=update(mod8a,~.-time)
summary(mod9a) #min. adequate model
```

#Compare using AIC() or anova()

```
AIC(mod8a,mod9a)
anova(mod8a,mod9a)
```

```
> AIC(mod8a,mod9a)
       df      AIC
mod8a   5 282.5272
mod9a   4 281.1875
```

```
> anova(mod8a,mod9a)
Data: d2
Models:
mod9a: relevance ~ grading + (1 | origin)
mod8a: relevance ~ grading + time + (1 | origin)
      npar    AIC    BIC  logLik deviance  Chisq Df Pr(>Chisq)
mod9a    4 281.19 291.27 -136.59   273.19
mod8a    5 282.53 295.14 -136.26   272.53 0.6603  1     0.4165
```

nlme package

#Re-fit using ML

```
mod8b=lme(relevance~grading+time,random=~1|origin,data=d2,method="ML")
summary(mod8b) #<time> can be removed
mod9b=update(mod8b,~.-time)
summary(mod9b) #minimum adequate model
```

#Compare using AIC() or anova()

```
AIC(mod8b,mod9b)
anova(mod8b,mod9b)
```

```
> AIC(mod8b,mod9b)
       df      AIC
mod8b   5 282.5272
mod9b   4 281.1875
```

```
> anova(mod8b,mod9b)
      Model df      AIC      BIC    logLik   Test  L.Ratio p-value
mod8b     1  5 282.5272 295.1361 -136.2636
mod9b     2  4 281.1875 291.2746 -136.5937 1 vs 2 0.6602715  0.4165
```

In both cases, the model without <time> is better (i.e. AIC is lower; and no difference in variation explained so we prefer the simpler model). This is our final model.

Summary (Learning Objectives)

Generalised Least Squares (GLS)

- Purpose
- Types of variance structures: varIdent, varFixed, varPower, varExp, varConstPower, varComb
- Fitting, checking and comparing

Linear Mixed Effect (LME) models

- Random effects: what are they, why and when to use them?
- lmer() vs. lme()
- Fitting, checking and comparing
- Simplifying: random effects using REML, then fixed effects using ML

Advanced analyses – Analysis decision tree (will be modified further)