Mixed effects models

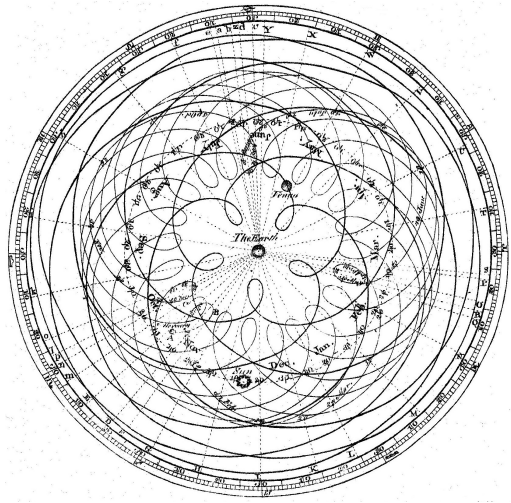
"Wheels within wheels"

Samuel Robinson, Ph.D.

Oct 6, 2023

Outline

- Linear mixed effects models (LMMs)
  - A bit of math
  - Fixed vs. random effects
  - Random intercepts and slopes
- Generalized linear mixed effects models (GLMMs)
  - Residuals checks
  - Some sage advice
- Hypothesis testing and inference
  - Slopes and intercepts
  - Entire terms
  - AIC and R^2



Part 1: Mixed effects models

Problem: group-level variation

- Sometimes we have to sample within groups: different field sites, individual organisms, etcetera
- However, often we're not really interested in each group *per se*, but in the **average group**
- e.g. "What is the effect of **x** if you remove group-to-group variation?"
- If you have a small number of groups, you can just include it in your model:
  - `lm(y ~ x + group)`
  - However, if you have few samples for each group, this can create problems
- Another solution is to use **mixed effects models**

What are mixed effects models?

Many different names:

1. Mixed effects models
2. Random effects models
3. Hierarchical models
4. Empirical/Bayesian hierarchical models
5. Latent variable models
6. Split-plot models
7. Variance partitioning

I usually use the term *heirarchical models*, as this is the closest to what I will teach you

Scary math

Unfortunately, we need a review of matrix algebra in order to explain this:

- This is a matrix:

$$A = \begin{bmatrix} 1 & 4 & 7 \\ 2 & 5 & 8 \\ 3 & 6 & 9 \end{bmatrix}$$

- This is a vector:

$$b = \begin{bmatrix} 1 & 2 & 3 \end{bmatrix}$$

- Multiplying them looks like this:

$$A \times b = Ab = 1 \times \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} + 2 \times \begin{bmatrix} 4 \\ 5 \\ 6 \end{bmatrix} + 3 \times \begin{bmatrix} 7 \\ 8 \\ 9 \end{bmatrix} = \begin{bmatrix} 30 \\ 36 \\ 42 \end{bmatrix}$$

Why do we call them "linear models"?

- *Linear* mapping of **coefficients** onto a **model matrix** (from your data)
- Coefficients:

$$\beta = \begin{bmatrix} 0.1 & 1.8 & -0.03 \end{bmatrix}$$

- Model matrix:

$$X = \begin{bmatrix} 1 & 1 & 10 \\ 1 & 1 & 12 \\ 1 & 0 & 9 \\ \vdots & \vdots & \vdots \end{bmatrix}$$

- Multiplying them looks like:

$$\hat{y} = X\beta = \begin{bmatrix} 1.60 \\ 1.54 \\ -0.17 \\ \vdots \end{bmatrix}$$

This is exactly what R does to fit models:

```r
head(dat)
```

```
##           y         x site
## 1  1.5101095 -4.248450    g
## 2  3.7190900  5.766103    j
## 3 -4.3737644 -1.820462    f
## 4 30.1459331  7.660348    n
## 5  0.2777422  8.809346    o
## 6 -3.6978175 -9.088870    p
```

```r
m1 <- lm(y~x,data=dat) #Uses x to predict y
summary(m1)
```

```
##
## Call:
## lm(formula = y ~ x, data = dat)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -18.2574  -4.1262   0.0296   3.1854  25.2780
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   2.7306     0.5772   4.731 4.92e-06 ***
## x             0.7213     0.1020   7.074 4.60e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.3 on 158 degrees of freedom
## Multiple R-squared:  0.2405, Adjusted R-squared:  0.2357
## F-statistic: 50.04 on 1 and 158 DF,  p-value: 4.6e-11
```

This is exactly what R does to fit models (cont.):

```r
head(model.matrix(m1))
```

```
##   (Intercept)         x
## 1           1 -4.248450
## 2           1  5.766103
## 3           1 -1.820462
## 4           1  7.660348
## 5           1  8.809346
## 6           1 -9.088870
```

```r
coef(m1)
```

```
## (Intercept)           x
##   2.7305689   0.7212867
```

```r
pred2 <- model.matrix(m1) %*% coef(m1) #predicted = matrix * coefs
head(data.frame(pred1=predict(m1),pred2)) #same thing!
```

```
##        pred1      pred2
## 1 -0.3337812 -0.3337812
## 2  6.8895819  6.8895819
## 3  1.4174942  1.4174942
## 4  8.2558758  8.2558758
## 5  9.0846325  9.0846325
## 6 -3.8251119 -3.8251119
```

Groups are coded by “dummy variables” (0s and 1s)

```
m2 <- lm(y~site,data=dat) #Use site to predict y
head(model.matrix(m2)) #0s and 1s used to identify groups
```

```
##   (Intercept) siteb sitec sited sitee sitef siteg siteh sitei sitej sitek sitel
## 1           1     0     0     0     0     0     1     0     0     0     0     0
## 2           1     0     0     0     0     0     0     0     0     1     0     0
## 3           1     0     0     0     0     1     0     0     0     0     0     0
## 4           1     0     0     0     0     0     0     0     0     0     0     0
## 5           1     0     0     0     0     0     0     0     0     0     0     0
## 6           1     0     0     0     0     0     0     0     0     0     0     0
##   sitem siten siteo sitep
## 1     0     0     0     0
## 2     0     0     0     0
## 3     0     0     0     0
## 4     0     1     0     0
## 5     0     0     1     0
## 6     0     0     0     1
```

```
coef(m2) #This uses the 1st site as the "control" group
```

```
## (Intercept)       siteb       sitec       sited       sitee       sitef
##    7.192416  -11.998464  -14.632803    1.983649   -7.765354   -4.523079
##       siteg       siteh       sitei       sitej       sitek       sitel
##   -3.439621   -8.280601   -4.306456   -4.085855   -5.663021   -5.155112
##       sitem       siten       siteo       sitep
##   -6.226642    8.403599   -8.626661  -10.934182
```

Structure of LMs... now with matrices!

- All linear models take the form:

$$\hat{y} = X\beta = b_0 1 + b_1 x_1 ... + b_i x_i$$
$$y \sim Normal(\hat{y}, \sigma)$$

- y is a vector of data you want to predict
- $\hat{y}$ is a vector of *predicted values* for y
- $X = \{1, x_1 ...\}$ is a matrix of *predictors* for y
- $\beta = \{b_0, b_1, ...\}$ is a vector of *coefficients*
- $y \sim Normal(\hat{y}, \sigma)$ means:
  - "y follows a Normal distribution with mean $\hat{y}$ and SD σ"

Fixed effects vs. Random effects

Say that X is a model matrix coding for a bunch of sites[1], and y is something we're interested in predicting

$$\hat{y} = b_0 + X\beta$$
$$y \sim Normal(\hat{y}, \sigma)$$

- Site coefficients (β) are unrelated to each other
- σ is the SD of *residuals*
- Site is a **fixed effect**

$$\hat{y} = b_0 + X\zeta$$
$$y \sim Normal(\hat{y}, \sigma)$$
$$\zeta \sim Normal(0, \sigma_{site})$$

- Site coefficients (ζ) are related to each other via a *Normal* distribution
- σ is the SD of *residuals*, σ_{site} is the SD of *sites*
- Site is a **random effect**

[1] Intercept is a separate variable

Mixed effects = fixed + random effects

A mixed effects model has both **fixed** and **random** effects

$$\hat{y} = X\beta + U\zeta$$
$$y \sim Normal(\hat{y}, \sigma)$$
$$\zeta \sim Normal(0, \sigma_{site})$$

- X = fixed effects matrix (e.g. intercept, temperature)
- β = fixed effects coefficients
- U = random effects matrix (e.g. sites)
- ζ = random effects coefficients
- σ, σ_{site} = variance terms

Mixed effect model example

Let's go back to our earlier example:

- We're interested in predicting y using x (fixed effects)
- Data was collected at a number of *sites*, which may affect y "somehow"
- Effect of each site is normally distributed

```
head(dat)
```

```
##           y         x site
## 1  1.5101095 -4.248450    g
## 2  3.7190900  5.766103    j
## 3 -4.3737644 -1.820462    f
## 4 30.1459331  7.660348    n
## 5  0.2777422  8.809346    o
## 6 -3.6978175 -9.088870    p
```

Mixed effect model example

```r
library(lme4) #Mixed effects library
#site is fit as a random intercept
mm1 <- lmer(y ~ x + (1|site),data=dat)
summary(mm1)
```

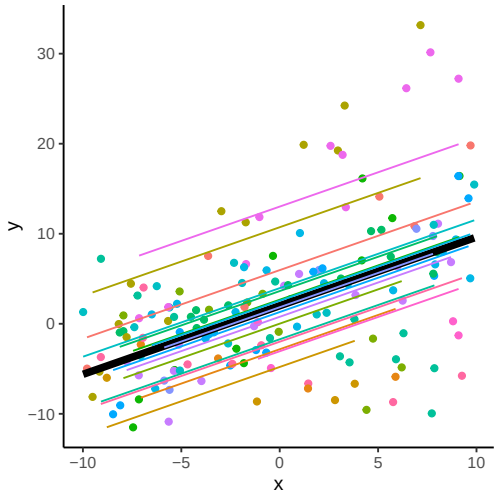
```
## Linear mixed model fit by REML ['lmerMod']
## Formula: y ~ x + (1 | site)
##    Data: dat
##
## REML criterion at convergence: 1040.4
##
## Scaled residuals:
##      Min       1Q   Median       3Q      Max
## -2.50816 -0.71380 -0.02682  0.69401  3.01951
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  site     (Intercept) 26.78    5.175
##  Residual             31.68    5.628
## Number of obs: 160, groups:  site, 16
##
## Fixed effects:
##             Estimate Std. Error t value
## (Intercept)  2.00331    1.38297   1.449
## x            0.76192    0.08083   9.426
##
## Correlation of Fixed Effects:
##   (Intr)
## x -0.002
```

Results from `lmer` model:

- Random effects:
  - *residual* and *site* variance (σ, σ_{site})
- Fixed effects:
  - Intercept and slope estimates (β)
  - No d.f. and p-value*
  - If you need p-values for parameters, you can use the *lmerTest* package (or just calculate them yourself using means/SEs)

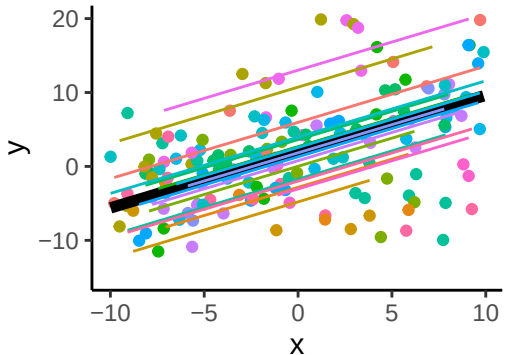
Mixed effect model results

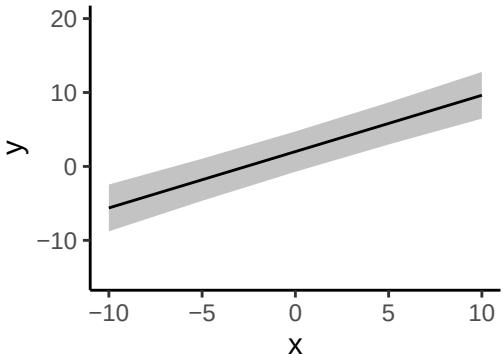
- In a *random intercepts* model, the regression line of x on y is allowed to move up or down around the main regression line for each site
- These changes in intercepts are *normally distributed*



Mixed effect model results (cont.)

- For plotting, we want a partial effects plot that marginalizes across sites (i.e. "What does the trend look like at the average site?")
- `ggpredict` works well for this. If you want partial residuals, you can add them in using `predict` and `residual`

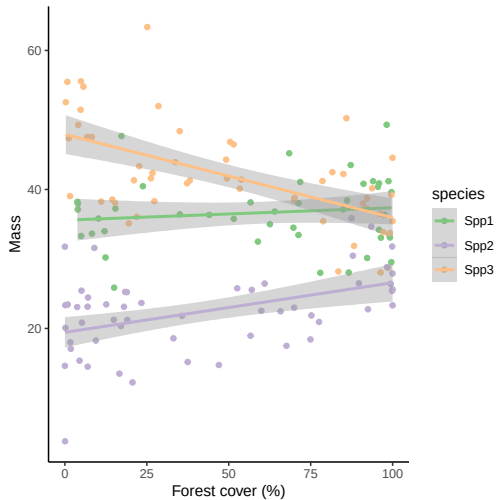




First challenge

How does forest cover influence fish size? Maybe some of the species do better in forested streams?

- You've weighed fish in streams with different forest covers (`fishMass.csv`). However, perhaps some of the variation is caused by "other things" about the site?
- Fit a mixed effects model with the fixed effects you're interested in (**forest cover**, **species**), and include **site** as a random *intercept*
- How does this compare to a simple linear model where you *ignore* site?



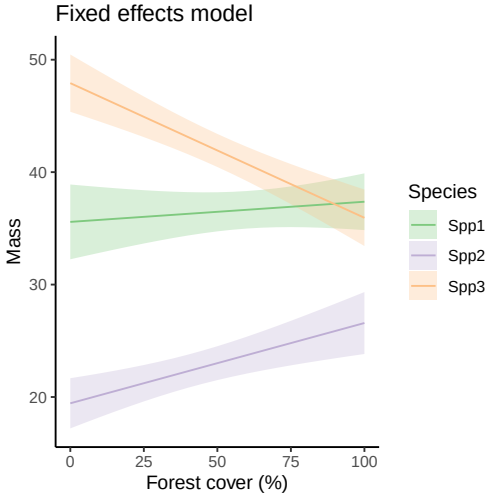
First challenge results

```
##
## Call:
## lm(formula = mass ~ species * forest, data = fishDat)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -15.6767  -3.1422   0.0415   3.3364  18.4631
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)          35.57610    1.68256  21.144  < 2e-16 ***
## speciesSpp2         -16.13571    2.02625  -7.963 4.60e-13 ***
## speciesSpp3          12.34080    2.11876   5.825 3.59e-08 ***
## forest                0.01792    0.02413   0.743   0.4590
## speciesSpp2:forest    0.05348    0.03152   1.697   0.0919 .
## speciesSpp3:forest   -0.13769    0.03187  -4.321 2.88e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.468 on 144 degrees of freedom
## Multiple R-squared:  0.7391, Adjusted R-squared:   0.73
## F-statistic: 81.57 on 5 and 144 DF,  p-value: < 2.2e-16
```

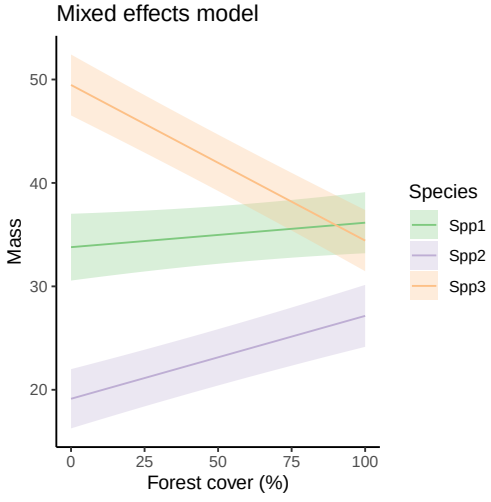
```
## Linear mixed model fit by REML ['lmerMod']
## Formula: mass ~ species * forest + (1 | sites)
##    Data: fishDat
##
## REML criterion at convergence: 807.4
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -3.3538 -0.5868  0.0548  0.6296  2.1122
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  sites    (Intercept) 25.931   5.092
##  Residual              8.381   2.895
## Number of obs: 150, groups:  sites, 15
##
## Fixed effects:
##                      Estimate Std. Error t value
## (Intercept)          33.78928    1.63228  20.701
## speciesSpp2         -14.65711    1.13614 -12.901
## speciesSpp3          15.68085    1.19467  13.126
## forest                0.02365    0.01353   1.747
## speciesSpp2:forest    0.05650    0.01720   3.285
## speciesSpp3:forest   -0.17411    0.01785  -9.754
##
## Correlation of Fixed Effects:
##             (Intr) spcsS2 spcsS3 forest spcS2:
## speciesSpp2 -0.501
## speciesSpp3 -0.480  0.681
## forest      -0.513  0.734  0.702
## spcsSpp2:fr  0.390 -0.815 -0.532 -0.771
## spcsSpp3:fr  0.387 -0.546 -0.835 -0.764  0.584
```

First challenge results (cont.)

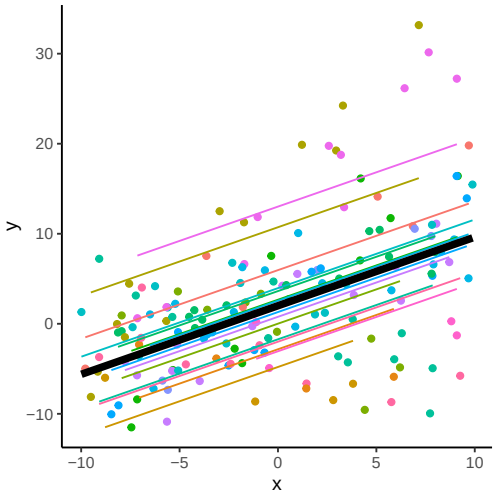
`lm(mass~forest*species)`

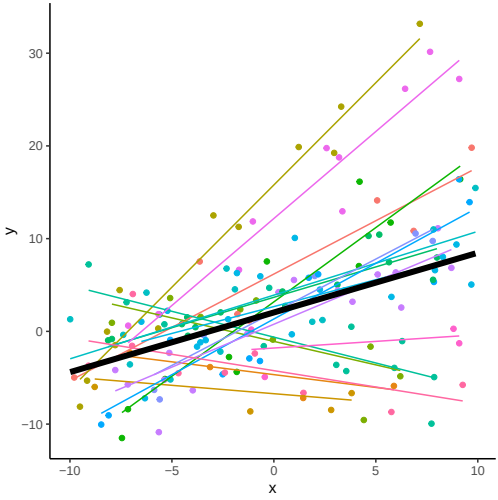


`lmer(mass~forest*species+(1|sites))`



More random effects: slopes!





Random slopes + intercepts

Suppose that y wasn't just higher or lower at each site, but that the effect of x on y was higher or lower at each site

$$\hat{y} = X\beta + U\zeta_{int} + U_x\zeta_{slope}$$
$$y \sim Normal(\hat{y}, \sigma)$$
$$\zeta_{int} \sim Normal(0, \sigma_{int})$$
$$\zeta_{slope} \sim Normal(0, \sigma_{slope})$$

- X = fixed effects matrix (e.g. intercept, temperature)
- β = fixed effects coefficients
- U = random intercept matrix (e.g. sites)
- U_x = random slopes matrix (e.g. temperature)
- $\zeta_{int}, \zeta_{slope}$ = random intercept and slope coefficients
- $\sigma, \sigma_{int}, \sigma_{slope}$ = variance terms

Random slope and intercept example:

```
#Intercept varies with site, and slope of x can
#   also vary with site (both hierarchical)
mm2 <- lmer(y ~ x + (x|site),data=dat)
summary(mm2)
```

```
## Linear mixed model fit by REML ['lmerMod']
## Formula: y ~ x + (x | site)
##    Data: dat
##
## REML criterion at convergence: 900.6
##
## Scaled residuals:
##      Min       1Q   Median       3Q      Max
## -2.10500 -0.64857  0.02414  0.61137  2.22996
##
## Random effects:
##  Groups   Name        Variance Std.Dev. Corr
##  site     (Intercept) 35.2210  5.9347
##           x            0.7889  0.8882   0.82
##  Residual              9.3162  3.0522
## Number of obs: 160, groups:  site, 16
##
## Fixed effects:
##             Estimate Std. Error t value
## (Intercept)   2.0383     1.5091   1.351
## x             0.6438     0.2275   2.830
##
## Correlation of Fixed Effects:
##   (Intr)
## x 0.790
```

Results from `lmer` model:

- Random effects:
  - *residual*, *slope*, and *site* variance (σ, σ_{int}, σ_{slope})
  - Correlation b/w intercept and slope = 0.82
    - Sites with higher intercept *also* have a higher slope
- Fixed effects:
  - Intercept and slope estimates
- Correlation of fixed effects:
  - Refers to correlation in *estimates*, not in *data*
  - Might be good to center the `x` variable in this case
  - Also occurs for interactions, but doesn't really matter

Model matrices

X: Fixed effects model matrix

```
##   (Intercept)         x
## 1           1 -4.248450
## 2           1  5.766103
## 3           1 -1.820462
## 4           1  7.660348
## 5           1  8.809346
## 6           1 -9.088870
```

U: Random intercept model matrix

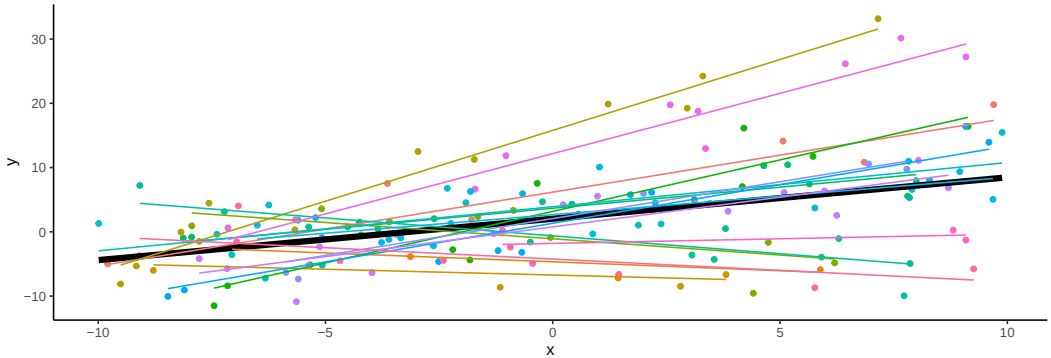
```
##   sitea siteb sitec sited sitee sitef siteg siteh sitei
## 1     0     0     0     0     0     0     1     0     0
## 2     0     0     0     0     0     0     0     0     0
## 3     0     0     0     0     0     1     0     0     0
## 4     0     0     0     0     0     0     0     0     0
## 5     0     0     0     0     0     0     0     0     0
## 6     0     0     0     0     0     0     0     0     0
##   sitej sitek sitel sitem siten siteo sitep
## 1     0     0     0     0     0     0     0
## 2     1     0     0     0     0     0     0
## 3     0     0     0     0     0     0     0
## 4     0     0     0     0     1     0     0
## 5     0     0     0     0     0     1     0
## 6     0     0     0     0     0     0     1
```

U_x: Random slope model matrix

```
##   sitea siteb sitec sited sitee sitef siteg siteh sitei
## 1     0     0     0     0     0  0.00 -4.25     0     0
## 2     0     0     0     0     0  0.00  0.00     0     0
## 3     0     0     0     0     0 -1.82  0.00     0     0
## 4     0     0     0     0     0  0.00  0.00     0     0
## 5     0     0     0     0     0  0.00  0.00     0     0
## 6     0     0     0     0     0  0.00  0.00     0     0
##   sitej sitek sitel sitem siten siteo sitep
## 1  0.00     0     0     0  0.00  0.00  0.00
## 2  5.77     0     0     0  0.00  0.00  0.00
## 3  0.00     0     0     0  0.00  0.00  0.00
## 4  0.00     0     0     0  7.66  0.00  0.00
## 5  0.00     0     0     0  0.00  8.81  0.00
## 6  0.00     0     0     0  0.00  0.00 -9.09
```

Mixed effect model results

- Regression line of x on y is allowed to move up or down for each site (random intercepts)
- Slope of regression line can be more or less steep for each site (random slopes)
- Changes in intercepts and slopes are *normally distributed*, and in this example are *correlated* with each other

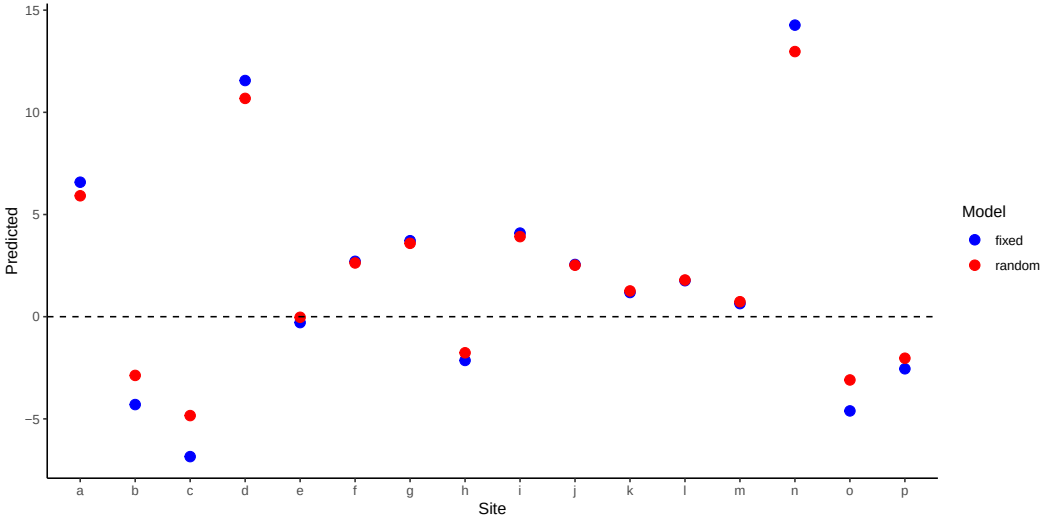


Why do we need to do any of this?

"My supervisor told me to just use site as a fixed effect. Why can't I do that?"

- You can do it this way, but you may encounter the following problems:
  - You lose the *partial pooling* that occurs in mixed effects models = Worse estimates of site effects!
  - You lose 1 d.f. for each site = Type II error ↑ = You may not find the fixed effect of interest, even if it's there!
  - Sites with low sample sizes may cause your models to break
  - People who have read statistics books published after 1980 may ask questions
- However, if you have a low number of sites (1-10), fixed effects may work better
  - Hard to estimate σ_{site} if number of sites is low
  - If stakes are high, it may be better to be more conservative about site intercepts
  - Easier to interpret (p-values, ANOVA, etc.)

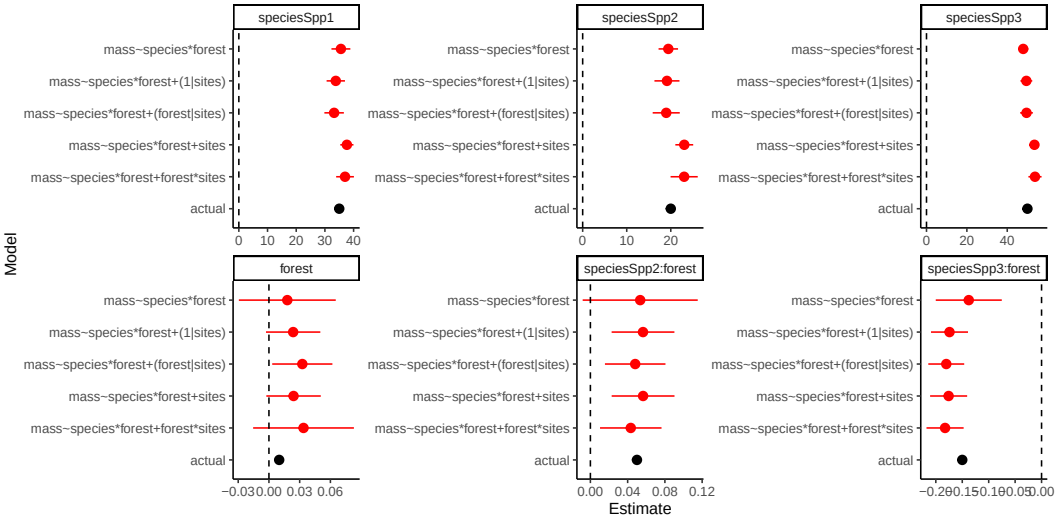
Example of partial pooling effect (shrinkage)



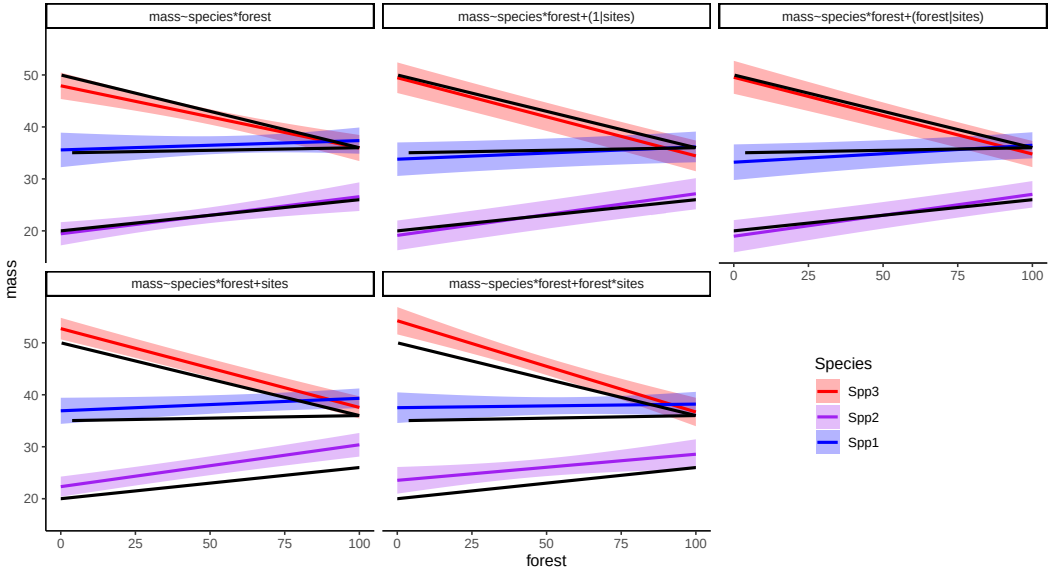
Second challenge

- Let's go back to the fish weight model...
- Fit a mixed effects model with the fixed effects you're interested in (**forest cover**, **species**), and include **site** as a random effect (*intercept* **or** *slope*)
- Your supervisor doesn't like hierarchical models, and tells you to just use site as another fixed term in an `lm` model. Do you get different results if you use their approach?

Second challenge results



Second challenge results (cont.):



Part 2: GLMMs

What if my response variable is non-normal?

- Linear model (LM)

$$\hat{y} = X\beta$$
$$y \sim Normal(\hat{y}, \sigma)$$

- Linear mixed effects model (LMM)

$$\hat{y} = X\beta + U\zeta$$
$$y \sim Normal(\hat{y}, \sigma)$$
$$\zeta \sim Normal(0, \sigma_{site})$$

- Generalized linear model (GLM)

$$logit(\hat{\phi}) = X\beta$$
$$y \sim Binomial(\hat{\phi})$$

- Generalized linear mixed effects model (GLMM)

$$logit(\hat{\phi}) = X\beta + U\zeta$$
$$y \sim Binomial(\hat{\phi})$$
$$\zeta \sim Normal(0, \sigma_{site})$$

How do I fit GLMMs?

- `glmer()` and `glmer.nb()` from `lme4` work for Binomial, Poisson, and Negative Binomial data

```
library(lme4)
glmm1 <- glmer.nb(y~x+(x|site),data=dat2) #Negative binomial GLMM
summary(glmm1) #glmer.nb takes a long time to run
```

- `glmmTMB()` from `glmmTMB` works for those above, *plus* a bunch of others (Zero-inflation, Beta-binomial), and it's generally faster

```
library(glmmTMB)
glmm2 <- glmmTMB(y~x+(x|site),data=dat2,family=nbinom2())
summary(glmm2) #Similar results, but quicker
```

Fitting GLMMs

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: Negative Binomial(5.1294)  ( log )
## Formula: y ~ x + (x | site)
##    Data: dat2
##
##      AIC      BIC   logLik deviance df.resid
##    627.8    646.3   -307.9    615.8      154
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -1.3745 -0.7098 -0.3946  0.5108  2.5367
##
## Random effects:
##  Groups Name        Variance Std.Dev. Corr
##  site   (Intercept) 1.43500  1.1979
##         x           0.02878  0.1697   0.92
## Number of obs: 160, groups:  site, 16
##
## Fixed effects:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.32746    0.32166   1.018   0.3087
## x            0.10830    0.04681   2.314   0.0207 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##   (Intr)
## x 0.799
```

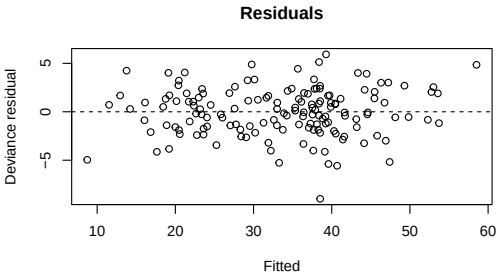
```
##  Family: nbinom2  ( log )
## Formula:          y ~ x + (x | site)
## Data: dat2
##
##      AIC      BIC   logLik deviance df.resid
##    627.8    646.2   -307.9    615.8      154
##
## Random effects:
##
## Conditional model:
##  Groups Name        Variance Std.Dev. Corr
##  site   (Intercept) 1.43543  1.1981
##         x           0.02892  0.1701   0.92
## Number of obs: 160, groups:  site, 16
##
## Dispersion parameter for nbinom2 family (): 5.12
##
## Conditional model:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.34132    0.32172   1.061   0.2887
## x            0.11026    0.04697   2.348   0.0189 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

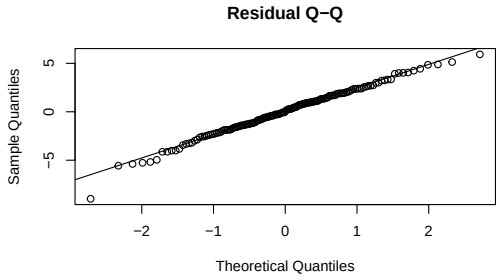
Residual checks for LMMs/GLMMs

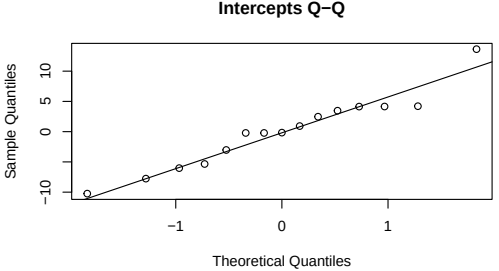
- Unfortunately, residual plotting functions aren't set up for mixed effects models. However, you can extract deviance residuals from `lmer`, `glmer`, or `glmmTMB` models and make your own plots:
- Added complication: **we also need to check whether the random effects are normally distributed**

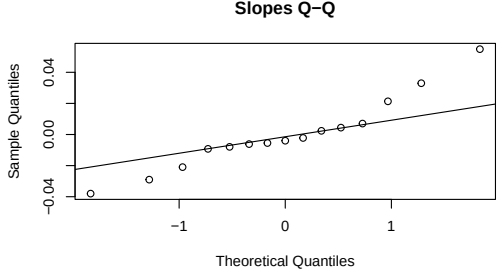
```
#Example with LMMs (fish model from before)
devRes <- residuals(m2,type='deviance') #Get deviance residuals
m2RE <- ranef(m2)$sites #Get matrix of random effects (intercept + slope)
par(mfrow=c(2,2))
#Plots deviance residuals + 0 line
plot(fitted(m2),devRes,xlab='Fitted',ylab='Deviance residual',main='Residuals')
abline(h=0,lty=2)
qqnorm(devRes,main='Residual Q-Q'); qqline(devRes) #Deviance residual Q-Q
qqnorm(m2RE[,1],main='Intercepts Q-Q'); qqline(m2RE[,1]) #Intercepts
qqnorm(m2RE[,2],main='Slopes Q-Q'); qqline(m2RE[,2]) #Slopes Q-Q
par(mfrow=c(1,1))
```

Residual checks for LMMs/GLMMs (cont.)



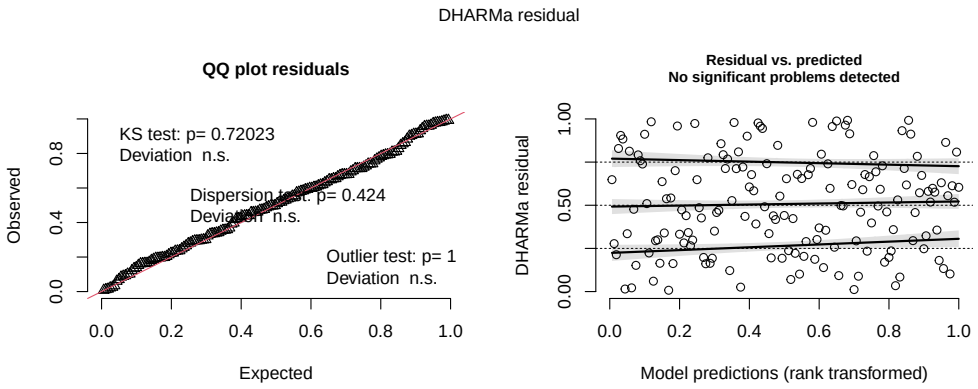






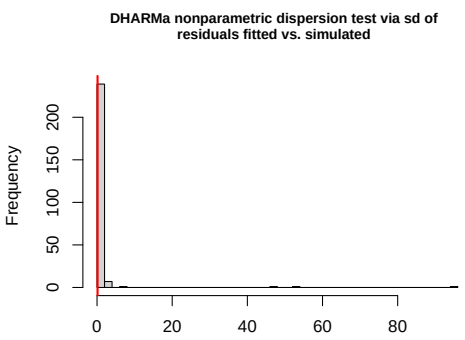
Alternative approach: simulation

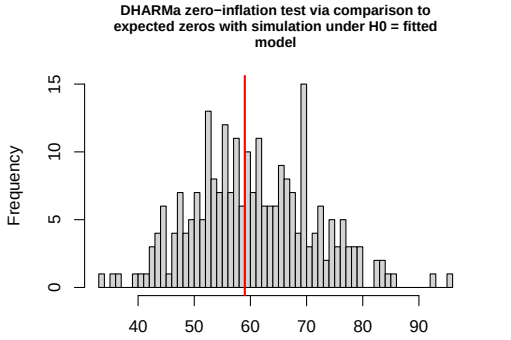
- Residual plots can be misleading, and can hide information from you. A better way is to compare a set of *simulated data* from your model to the *actual data*
- The `simulateResiduals` from `DHARMa` (see here) works well for `glmmTMB` models, as well as LMs, GLMs, and more, so it's worth learning



Simulation (cont.)

- `DHARMa` also has useful functions for checking overdispersion and zero-inflation. Both of these tests indicate that a) this model is not overdispersed, and b) there seems to be no zero-inflation (see here for more examples)



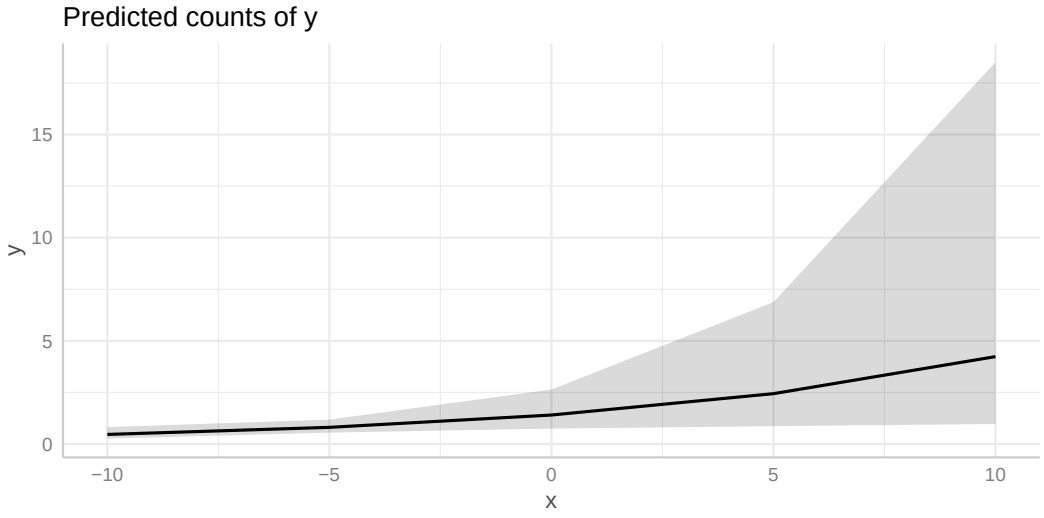



```
##
##  DHARMa nonparametric dispersion test via sd of residuals fitted vs.
##  simulated
##
```

```
##
##  DHARMa zero-inflation test via comparison to expected zeros with
##  simulation under H0 = fitted model
##
```

Partial residual plots for `glmmTMB` models

- `ggpredict()` from `library(ggeffects)` works with all `glmm` models

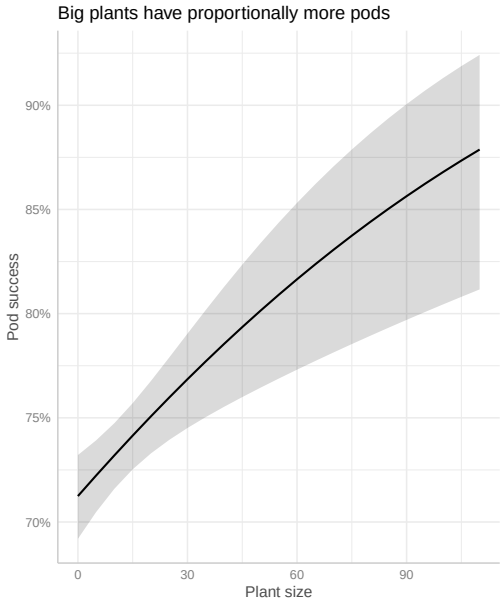


Third challenge

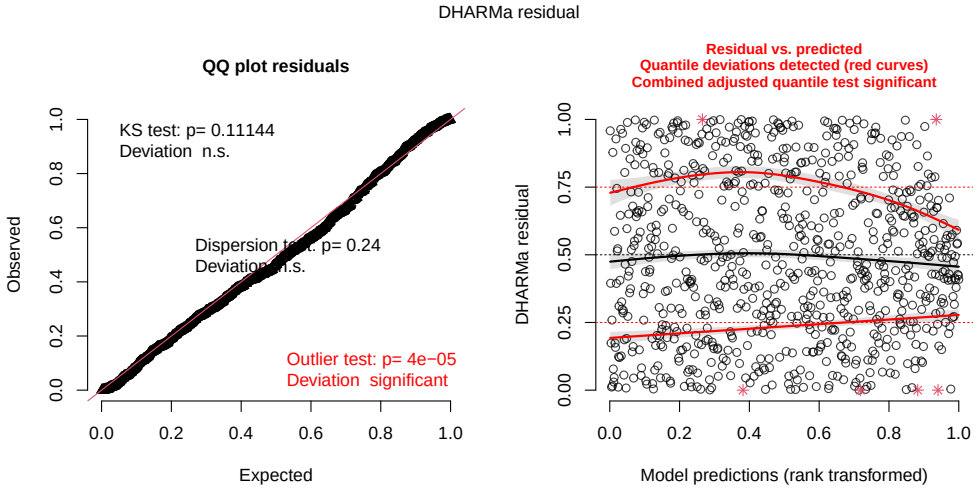
- Remember that `canolaPlants.csv` data I gave you last week, which you all dutifully fit GLMs of? (See here)
- That `Field` column indicates which farmer's field the plants came from. You'll also notice that there are groupings at each `Distance`, indicating distinct `Plots` that each plant came from.
- Fit a GLMM of pod success with distance, using Field and Plot as random effects.
- Check the assumptions of your model with `DHARMa`.

Third challenge results

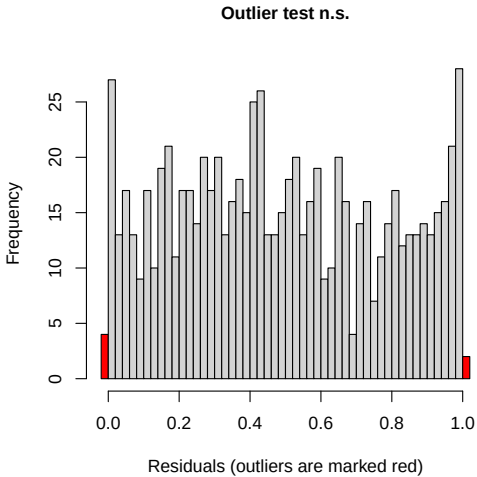
```
##  Family: binomial  ( logit )
## Formula:
## cbind(Pods, Missing) ~ Year + VegMass + Distance + (VegMass |
##     Field/Plot)
## Data: canolaDat
##
##      AIC      BIC   logLik deviance df.resid
##   7455.3   7502.1  -3717.7   7435.3      781
##
## Random effects:
##
## Conditional model:
##  Groups     Name        Variance  Std.Dev. Corr
##  Plot:Field (Intercept) 0.2525897 0.50258
##             VegMass     0.0005972 0.02444  -0.83
##  Field      (Intercept) 0.0510166 0.22587
##             VegMass     0.0001530 0.01237  -0.21
## Number of obs: 791, groups:  Plot:Field, 246; Field, 59
##
## Conditional model:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)  3.678e+01  1.564e+02   0.235 0.814090
## Year        -1.781e-02  7.764e-02  -0.229 0.818560
## VegMass      9.758e-03  2.650e-03   3.683 0.000231 ***
## Distance    -5.661e-05  1.026e-04  -0.552 0.580994
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

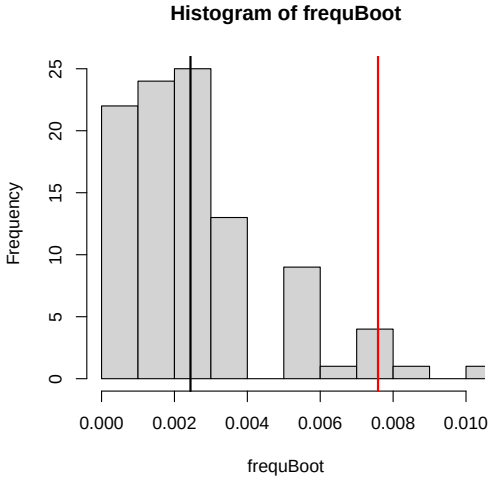


Third challenge results (cont.)



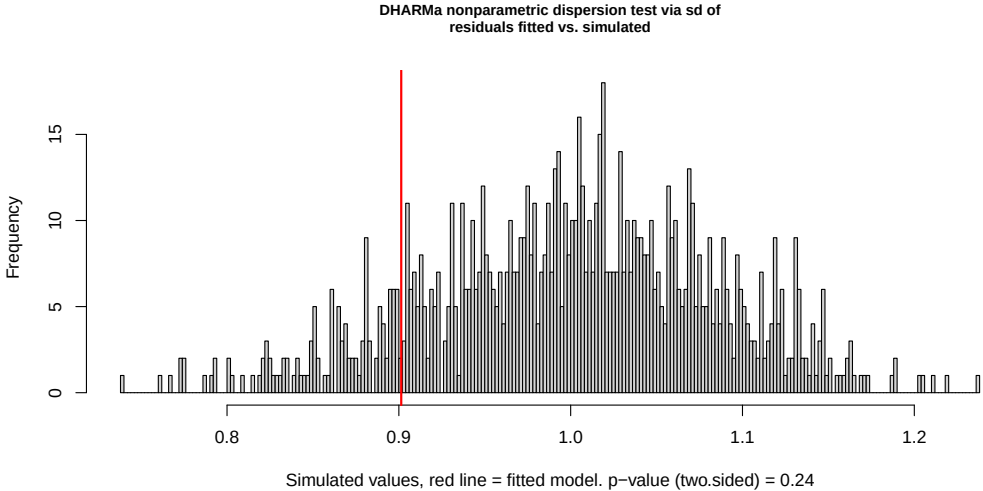
Third challenge results (cont.)






```
##
##  DHARMa bootstrapped outlier test
##
## data:  cenModRes
```

Third challenge results (cont.)




```
##
##  DHARMa nonparametric dispersion test via sd of
##  residuals fitted vs. simulated
##
```

Part 3: Hypothesis testing and inference

I fit a model... now what?

- Congratulations, your LM/GLM/LMM/GLMM ran and it met the assumptions of regression!
- Time to see if your predictions are supported by your data...
- For each of the terms in your model:
  - Was the term "important" for your model?
  - If so, what direction was the effect in?
- How well did your model fit your data (overall)?
- Some other sage advice

Step 1: was the term "important"?

In linear models, this is done using an ANOVA F-test (also shown at the bottom of a `summary()` statement):

```
lmMod1 <- lm(mpg~disp+gear,data=mtcars)
anova(lmMod1)
```

```
## Analysis of Variance Table
##
## Response: mpg
##           Df Sum Sq Mean Sq F value    Pr(>F)
## disp       1 808.89  808.89 73.9959 1.788e-09 ***
## gear       1   0.14    0.14  0.0132    0.9093
## Residuals 29 317.01   10.93
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

But wait, there's more!

Unfortunately, if we have more than 1 term in the model, the order of terms can change your answer:

```
lmMod1 <- lm(mpg~disp+gear,data=mtcars)
lmMod2 <- lm(mpg~gear+disp,data=mtcars)
```

```
anova(lmMod1)
```

```
## Analysis of Variance Table
##
## Response: mpg
##           Df Sum Sq Mean Sq F value    Pr(>F)
## disp       1 808.89  808.89 73.9959 1.788e-09 ***
## gear       1   0.14    0.14  0.0132    0.9093
## Residuals 29 317.01   10.93
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
anova(lmMod2)
```

```
## Analysis of Variance Table
##
## Response: mpg
##           Df Sum Sq Mean Sq F value    Pr(>F)
## gear       1 259.75  259.75  23.762 3.595e-05 ***
## disp       1 549.28  549.28  50.248 8.465e-08 ***
## Residuals 29 317.01   10.93
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Solution: use a Type II ANOVA

- We're usually interested in the importance of a term on its own, *not just after other terms are accounted for* (Type I ANOVA).
- For this, we use a Type II ANOVA using `drop1()` or `car::Anova()`

```
drop1(lmMod1,test='F')
```

```
## Single term deletions
##
## Model:
## mpg ~ disp + gear
##        Df Sum of Sq    RSS     AIC F value    Pr(>F)
## <none>              317.01  79.383
## disp    1    549.28 866.30 109.552 50.2476 8.465e-08 ***
## gear    1      0.14 317.16  77.397  0.0132    0.9093
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
drop1(lmMod2,test='F') #Same as above
```

```
## Single term deletions
##
## Model:
## mpg ~ gear + disp
##        Df Sum of Sq    RSS     AIC F value    Pr(>F)
## <none>              317.01  79.383
## gear    1      0.14 317.16  77.397  0.0132    0.9093
## disp    1    549.28 866.30 109.552 50.2476 8.465e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Interactions make ANOVA testing a bit strange

- If interactions are present, it doesn't really make sense to test the main terms **because they depend on the interactions**

```
## Single term deletions
##
## Model:
## mpg ~ gear * disp
##           Df Sum of Sq    RSS    AIC F value  Pr(>F)
## <none>                 278.60 77.250
## gear:disp  1    38.412 317.01 79.383  3.8604 0.05943 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

- `Anova` from the `car` package tests other terms *after* interactions, but the meaning isn't the same. I prefer to keep things simple and just use `drop1()`

```
## Anova Table (Type II tests)
##
## Response: mpg
##            Sum Sq Df F value    Pr(>F)
## gear         0.14  1  0.0145   0.90502
## disp       549.28  1 55.2038 4.312e-08 ***
## gear:disp  38.41  1  3.8604   0.05943 .
## Residuals 278.60 28
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

GLMs, LMMs, and GLMMs

- `drop1` also works with GLMs, LMMs, and GLMMs, but we use a χ^2 *likelihood ratio test* rather than an F-test
- Unfortunately, different numbers of data points change your likelihood, so this can wreck LRTs. This happens a lot if you have NAs in your predictor columns, so *clean up your data before using it in models*.

```
drop1(m1,test='Chisq')
```

```
## Single term deletions
##
## Model:
## mass ~ species * forest - 1 + (1 | sites)
##                npar    AIC    LRT   Pr(Chi)
## <none>              806.04
## species:forest    2 932.51 130.47 < 2.2e-16 ***
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

ML vs REML - a mathematical aside

- Maximum likelihood (ML) estimates of variance (e.g. SD) are always smaller than the actual variance (biased)
- Restricted maximum likelihood (REML) uses a mathematical trick to get around this, but...
- This means that models with *different numbers of fixed effects* don't have the same REML estimates
- Likelihood between these models technically can't be compared!
- Solution:
  1. Use ML if comparing between models with different fixed effects, then...
  2. Re-fit with REML once you've decided on a model

Step 2: what was the direction of the effect?

- How do I know this effect is different from x?
- Use Wald Z-test (2-sided p-value from Z-test)

```
#Get mean
meanEst <- fixef(mm1)[2]
#Get standard error
seEst <- sqrt(vcov(mm1)[2,2])
zEst <- meanEst/seEst #Z-score
#p-value from 2-sided Z-test
(1-pnorm(zEst,0,1))*2 #Large difference from zero
```

```
## x
## 0
```

- If you're testing multiple things at the same time, you need to account for *multiple comparisons*
- `glht` from `library(multcomp)` works with most linear models for comparing between coefficients
  - e.g. "Is treatment A different from B and C?"

```
library(multcomp)
mod <- lm(mass~species,data=fishDat)
sppComp <- glht(mod,linfct=mcp(species= "Tukey"))
cld(sppComp) #Letter display
```

```
## Spp1 Spp2 Spp3
##  "a"  "b"  "c"
```

Step 3: How well did your model fit your data?

In a simple linear model, a common measure of model fit is *adjusted* R^2, which can be found in the `summary()` statement

```
##
## Call:
## lm(formula = mass ~ species * forest - 1, data = fishDat)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -15.6767  -3.1422   0.0415   3.3364  18.4631
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## speciesSpp1         35.57610    1.68256  21.144  < 2e-16 ***
## speciesSpp2         19.44039    1.12903  17.219  < 2e-16 ***
## speciesSpp3         47.91690    1.28769  37.212  < 2e-16 ***
## forest               0.01792    0.02413   0.743   0.4590
## speciesSpp2:forest   0.05348    0.03152   1.697   0.0919 .
## speciesSpp3:forest -0.13769    0.03187  -4.321 2.88e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.468 on 144 degrees of freedom
## Multiple R-squared:  0.9766, Adjusted R-squared:  0.9756
## F-statistic:  1002 on 6 and 144 DF,  p-value: < 2.2e-16
```

How well did your model fit your data? (cont.)

- For GLMs, LMMs, and GLMMs, there isn't really one standard way to get "R^2". See here for a widely-read paper on the topic
- Likelihood Ratio Tests or AIC (*Akaike's Information Criterion*) can be used to compare between different models *of the same dataset*, but likelihood and AIC don't mean anything on their own
- Both `MuMIn` and `r2glmm` implement versions of the Nakagawa R^2 for mixed effects models. *Caveat emptor*:

```
library(MuMIn)
r.squaredGLMM(m1)
```

```
##            R2m       R2c
## [1,] 0.7094007 0.9290197
```

```
library(r2glmm)
r2beta(m1, method = 'nsj',partial=FALSE)
```

```
##   Effect  Rsq upper.CL lower.CL
## 1  Model 0.96    0.969    0.951
```

- Size of the random effects (variance components) can give you an idea of how large the between-group variance was compared to residual variance. Useful for planning future field work or experiments!

Some final advice

- LMMs and GLMMs are *hard* to both understand and fit. They are not as forgiving as LMs and GLMs, and will sometimes fail to fit, take a very long time, or give you weird answers without any explanation.
  - Check the model output and make sure the coefficients and SEs aren't weirdly large or small
  - Andrew Gelman's Modeling Rule-of-Thumb: "It's not me, it's you!"
- My advice: Once you have a model you'd like to fit, start off with simpler "incorrect" version of it, and add terms until you have the final "correct" model
  - In the canola model, I started off with a Field-level intercept, then added a Plot-level intercept, and finally a random slope term
  - One of my interim models had a random `Distance` slope term, which caused the model to fail (reason: only 1 distance per plot)
- **Avoid selecting models based on R^2 or AIC alone. Think about how the system works!**

To do this week: update your models!

- Some of you have mixed effects... time to update those models!
- For those who don't need to run LMMs or GLMMS, try this:
- I have a dataset of honey bee visitation in canola crops (found here) that I collected during my PhD. I was interested in whether distance from the edge of the field (`Distance`) and number of honey bee hives affected the visitation rate of honey bees.
- Fit a GLMM that answers my question, check the assumptions using `DHARMa` and make some plots of your results. Bonus if you calculate an R^2-like number!
  - I used different lengths of observation time, so you'll have to use `log(time)` as an offset. You also may need to use a zero-inflated distribution (see `glmmTMB` help file for details)
  - How does the size of the variance components look? Should I have a) taken more samples at each field, or b) used more fields?