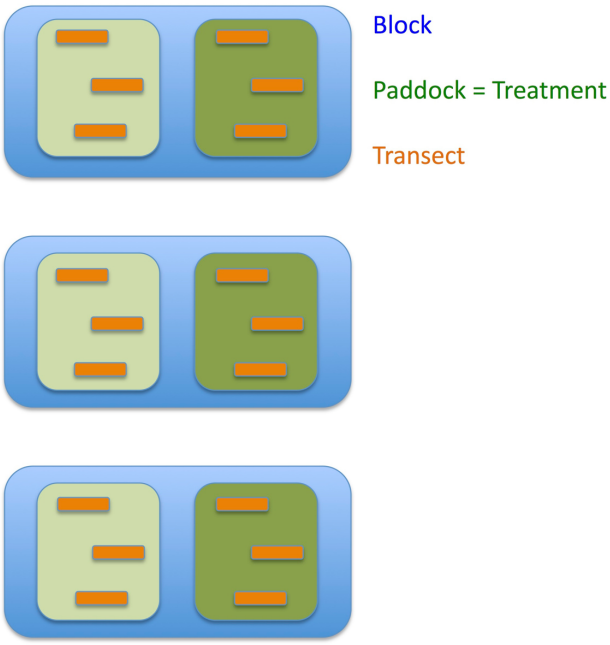
Lecture 21. Mixed Models III.

Over the last two lectures we spent a lot of time on the basic principles of random effects, how to model and interpret them, how to include fixed effects predictors (to get a mixed effects model), and how to do to inference on those predictors. Now I'm going to go through some additional examples that will help you get a sense for the different ways mixed effects models can be used, and some additional concepts that did not come up in the prior example.

Grassland grazing experiment: repeated measures, nested and crossed random effects, fixed vs. random effects.

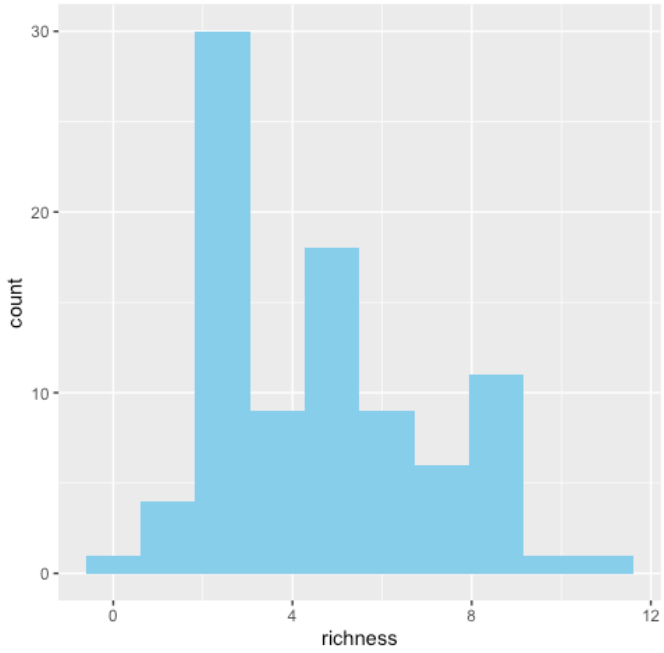
Here is a long-term experiment reproted by Jerrentrup et al. (2014; Journal of Applied Ecology, "Grazing intensity affects insect diversity…"). The authors designed an experiment to quantify how insect diversity/abundance is affected by cattle grazing in grasslands in Germany. The motivation is that these fields are used for livestock, but ideally one could manage grazing in such a way that would mitigate effects on biodiversity in the ecosystem. The authors manipulated the stocking of cattle in experimental paddocks to create 'moderate grazing' (MG) and 'lenient grazing' (LG) treatments. The experimental design looked like this:



There are three blocks. Within each block there are two paddocks, one MG and one LG. Within each paddock there are three permanent transects where butterflies and grasshoppers were counted. The treatments are maintained over many years, and insect counts were taken in five years, 2002-2004 and 2010-2011. The authors looked at a number of analyses, but here I will just focus on how butterfly species richness is affected by the grazing treatment.

The structure of this experiment is fairly complex: the data has a nested 3-level spatial structure, and the same transects are measured 5 times (repeated measures). Of course this kind of spatio-temporal complexity is actually quite common in biology, both for experiments and observational data, so it's important to use an approach that appropriately models this complexity.

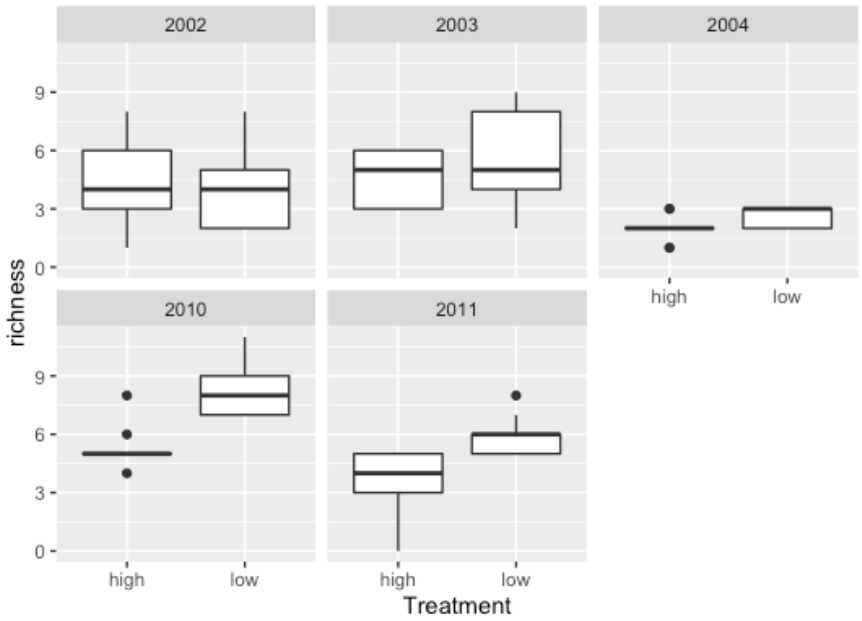
Let's start off with some exploratory plots. There are a total of 18 transects measured 5 times each, giving 90 observations. The response variable I'm looking at is butterfly species richness in a transect, and the raw data look like this:



Species richness is a kind of count, and so a Poisson distribution would probably be most appropriate for modeling this data. But I'm going to do GLMMs next lecture, and in this case the data looks like a normal distribution would probably do fine as well (recall that as the mean of the Poisson gets large enough, the distribution becomes approximately normal). And of course we will look at the residuals to see if this is true.

The authors' question is whether the grazing treatment affects the insect community, and whether a reduced grazing regime can permit greater biodiversity. So let's do an exploratory plot of richness vs. treatment. Because this is a long-term experiment, the effect of the treatment might change over time, so I want to plot the two treatments by year. This is easily done with facet_wrap():

```
ggplot(butterfly, aes(Treatment, richness)) + geom_boxplot() + facet_wr
ap(~Year)
```



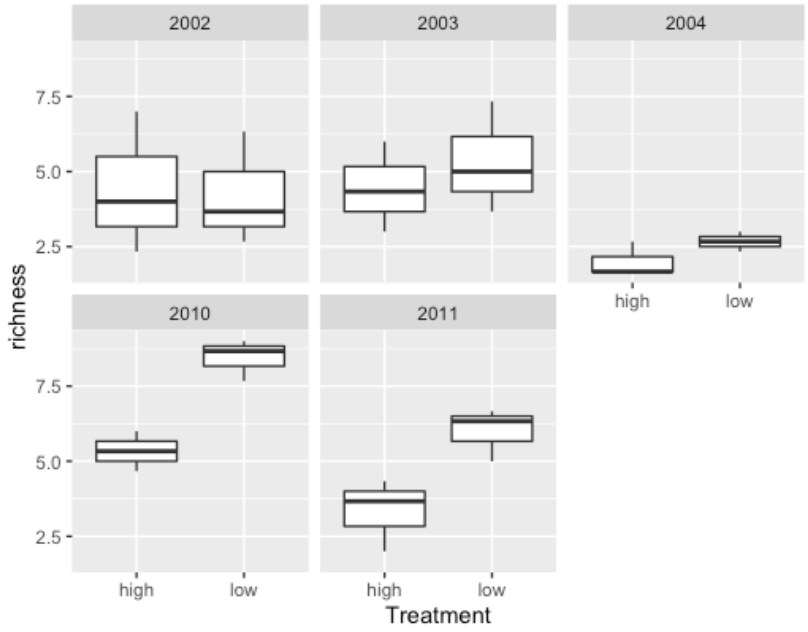
It looks like the moderate grazing treatment (which I've coded as 'high' in the dataset) tends to have lower richness than the low grazing treatment, but it also looks like the effect is larger in 2010 and 2011. So this is something we will consider in the model. It's worth noting that when we plot these data we are treating each individual transect as a separate observation. But there are 3 transects per experimental replicate, which are the paddocks that are assigned to low or high grazing. So for the purposes of plotting the data we may want to average richness by paddock (in each year). This is easily done with ddply in the plyr package:

```
butterfly.means = ddply(butterfly, .(Year, Block, Treatment, Paddock),
summarize, richness = mean(richness))
```

This function does 'split-apply-combine', as Hadley Wickham calls it. It takes a dataframe, splits it by some factor(s), applies some function to each piece, and then combines the results into a new dataframe. The second argument, .(Year, Block, Treatment, Paddock), tells the function what factors to use to split up the data. Note that TransectID is absent, because we want to averaged over transects within a paddock. The third argument indicates that we're going to take each piece of the dataframe and summarize it with some function. The fourth argument is the function to apply to each piece: we're going to make a variable 'richness', which is just the mean of richness in each piece. So this function will find the mean richness by Year and Paddock (and also code with Block and Treatment the Paddock is from).

The point of this is just to plot the paddock-mean richness by treatment:

```
ggplot(butterfly.means, aes(Treatment, richness)) + geom_boxplot() + fa
cet_wrap(~Year)
```



This looks pretty similar to the earlier plot, though now the plot will better reflect what a model will actually be analyzing for the treatment effect.

Nested groups

The experiment has three transects per paddock, and two paddocks per block. We want to test for a treatment effect, but to properly account for potential non-independence in the data, we need to include the additional grouping structure in our model. It's also important to include this structure if we want to use an approximate F-test to do inference, because the degrees of freedom calculation takes into account the number of groups at different levels.

The experiment clearly has a nested structure, transects within paddocks within blocks. The good news is that we don't have to explicitly tell lmer() about this nested structure, as long as each group is has a unique identifier. Let's consider what this means. In this dataset, there is a variable 'Transect', and a variable 'TransectID':

```
butterfly$Transect

##  [1] 1 2 3 1 2 3 1 2 3 1 2 3…

butterfly$TransectID

##  [1] A1_1 A1_2 A1_3 A2_1 A2_2 A2_3 B1_1 B1_2 B1_3 B2_1 B2_2 B2_3… ## 18 Lev
els: A1_1 A1_2 A1_3 A2_1 A2_2 A2_3 B1_1 B1_2 B1_3 B2_1 B2_2 ... C2_3
```

I have abbreviated the output to show the first twelve elements of the vectors. The vector TransectID uniquely codes each transect: there is transect 1 in Paddock A1,

transect 2 in Paddock A1, transect 3 in Paddock A1, transect 1 in Paddock A2, transect 2 in Paddock A2, etc. The vector Transect does not uniquely code each transect; rather, each transect is just called 1 or 2 or 3, because each Paddock has 3 transects. The important difference between these two labeling schemes is that using Transect requires us to tell the function what the nested structure is; *otherwise it will think the every transect labeled 1 is the same transect*. This could be specified in lmer like this:

```
(1|Block/Paddock/Transect)
```

This says 'let the intercept vary randomly by block, paddock, and transect, with transect nested within paddock and paddock nested within block'. This is perfectly legitimate, but in general its easier to just give each random effect its own term in the model, like this:

```
(1|Block) + (1|Paddock) + (1|TransectID)
```

Because each paddock and TransectID is uniquely labeled, we don't need to tell lmer() that the data has a nested structure. In essence this is 'automatically' accounted for by the model; in reality it doesn't matter, because we're going to assume that the amount of random variation between transects doesn't depend on which paddock or block they're in. The punchline is that you should make sure each level of a grouping variable is uniquely coded, and then you don't need to use a syntax for nesting.

Repeated measures

This experiment also has 'repeated measures', i.e. each transect is measured in 5 years (and the same is true for each paddock and block). One of the great values of mixed models is that they can account for repeated measures, with some limitation. If we include a random effect for transect, then this will account for the fact that observations within a transect are likely to be correlated, in the sense that observations within a transect will be more similar to each other than they are to observations from other transects.

In most cases a random effect will successfully account for non-independence caused by repeated measures. However, a random effect will not account for *temporal autocorrelation*, which is what happens when *observations close together in time are more similar than observations further apart in time*. Temporal autocorrelation can be modeled, but requires additional methods beyond random effects. We will cover this later in the course, but in reality it can be difficult to combined mixed models with temporal autocorrelation, so it is easier to just use a mixed model when possible. With only 5 observations over time, this dataset is not really much of a time series, and so temporal autocorrelation is unlikely to have

much effect on the results. It is more of an issue when trying to do regression with a long time series.

Crossed random effects

In addition to the spatial structure of the data, there is structure in time: observations were taken in 5 years. The exploratory plot suggests that butterfly species richness varies across years, regardless of the experimental treatment. 2004 looks low and 2010 looks high. So this is yet another source of variation that we should account for, in order to better capture any effect of treatment (by accounting for other sources of variation), and to account for non-independence. You could make an argument for putting Year in the model as a fixed effect factor (more on this later), but we can also add it in as another random effect. Because Year is 'crossed' with the spatial variables (block, paddock, transect), in the sense that each paddock is measured in multiple years, this will mean the model has 'crossed random effects'. Some software that fits mixed models cannot easily handle crossed random effects, but lme4 can.

It's worth noting what an effect of Year will mean in the model: this will be the difference between Years, *averaged over the spatial variation* (due to transect, paddock, block, treatment). Likewise, a random effect for Paddock will quantify the different between paddocks, *averaged over years*.

The model

Now that we've discussed nested and crossed random effects, and repeated measures, let's specify a model:

```
mod.rand = lmer(richness ~ Treatment + (1|Year) + (1|TransectID) + (1|Block) +
(1|Paddock), data = butterfly)
```

So after all the thinking, the specification is quite easy. We have a fixed effect for treatment, and random effects for Year, TransectID, Block, and Paddock.

```
summary(mod.rand)
```

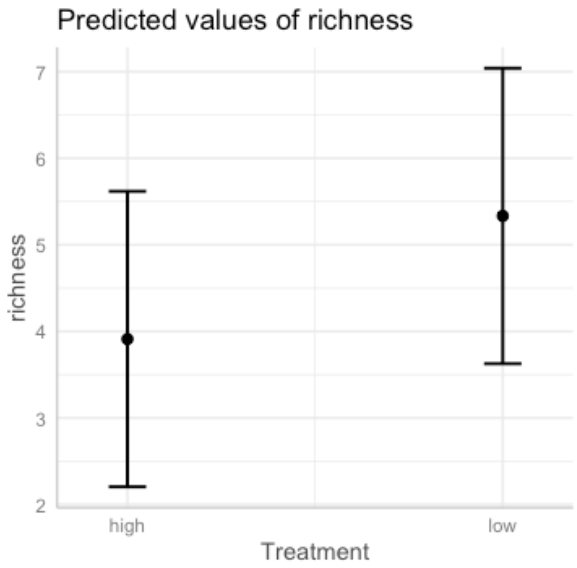
```
## Linear mixed model fit by REML ['merModLmerTest']
## Formula:
## richness ~ Treatment + (1 | Year) + (1 | TransectID) + (1 | Block) +
##     (1 | Paddock)
##    Data: butterfly
##
## REML criterion at convergence: 356
##
## Scaled residuals:
##    Min     1Q Median     3Q    Max
## -2.133 -0.590 -0.098  0.581  2.220
##
```

```
## Random effects:
##  Groups     Name        Variance Std.Dev.
##  TransectID (Intercept) 0.000    0.000
##  Paddock    (Intercept) 0.549    0.741
##  Year       (Intercept) 2.494    1.579
##  Block      (Intercept) 0.000    0.000
##  Residual               2.513    1.585
## Number of obs: 90, groups:  TransectID, 18; Paddock, 6; Year, 5; Block, 3
##
## Fixed effects:
##              Estimate Std. Error    df t value Pr(>|t|)
## (Intercept)     3.911      0.859 6.500    4.55   0.0032 **
## Treatmentlow    1.422      0.691 4.000    2.06   0.1087
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr)
## Treatmentlw -0.402
```

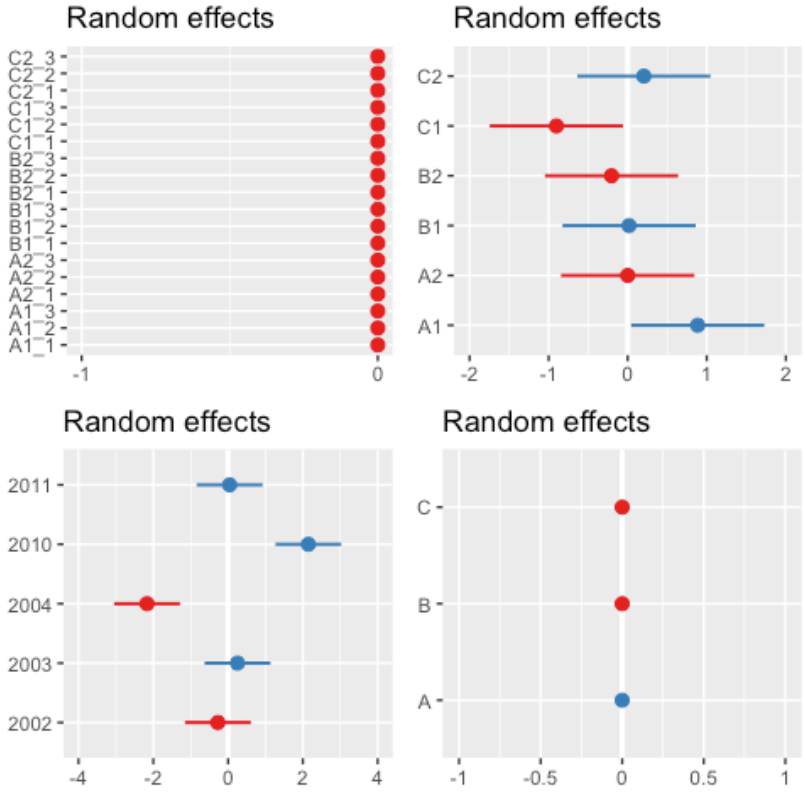
The fixed effect for treatment suggests that the low grazing treatment tends to have higher species richness, but this different is probably not significant (later we'll see if the effect varies over time). How about the random effects? We have four of them, and some of them appear to be important while others are not. Species richness varies across years with a standard deviation of 1.5, consistent with the exploratory plots. Species richness also varies across paddocks, which are the unit of experimental manipulation, with a standard deviation of 0.74. So that represent spatial variation in butterfly richness, which is moderate but not huge. The residual variation is 1.59. Let's consider what the residuals are in this model: we have terms for transect as well as year, so the residuals will be the variation within transects over time. This is of a comparable magnitude to the spatially-averaged variation over time, which is quantified by Year.

Let's look at the estimated effects for these terms:

```
plot(ggeffect(mod.rand))
```

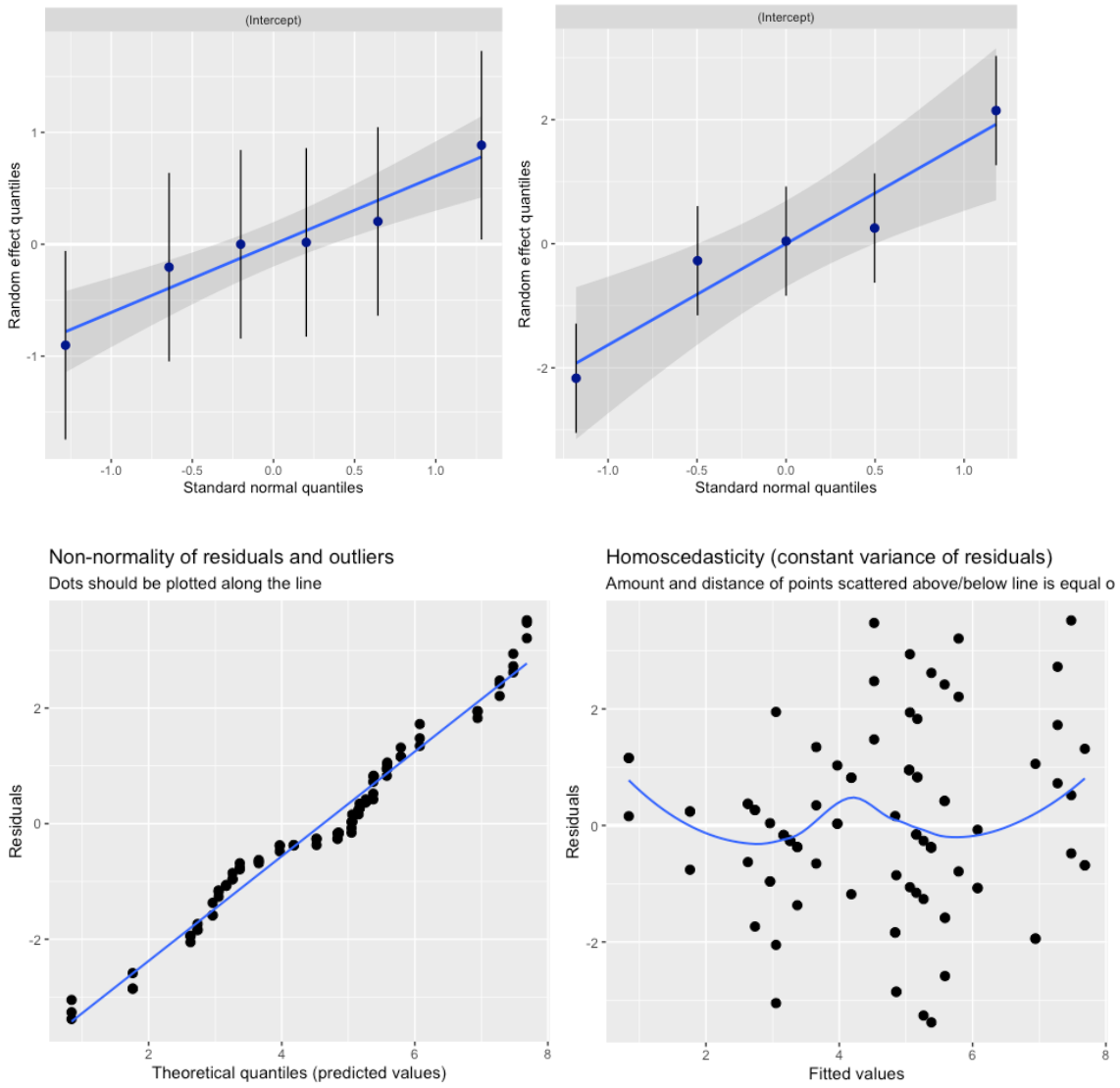



```
grid.arrange(grobs = plot_model(mod.rand, type = 're'))
```



These plots give similar info to what we already discussed. We can also see that 2004 and 2010 were the most extreme years (perhaps due to some large-scale environmental variation). We can also look at some diagnostics for the random effects and residuals:

```
plot_model(mod.rand, type = 'diag')
```



Clearly for Year and Paddock, which do not have many levels, it is hard to assess the normality assumption. But there is no strong indication to the contrary. The residuals look pretty good, i.e. the normal approximation for the discrete richness data seems fine.

What about Block and TransectID? Both of these have variance estimates of zero. What does this mean? It may seem odd at first that a variable like TransectID has a variance estimate of zero; surely there is variation in species richness across transects. But the random effect for TransectID is quantifying *whether transect explains variation in richness, while simultaneously accounting for variation within transects (residual) as well as variation among paddocks and blocks*. If the variance estimate is zero, that means the maximum likelihood estimate for the variation

explained by TransectID is zero, and any raw variation among transects is caused by within-transect variation, and/or variation at higher levels.

Random effects: how many levels are necessary? When to use fixed instead?

The random effect for block also has a variance estimate of zero. Block only has 3 levels (there are 3 blocks), and so we might be concerned that the model has a hard time estimating this term. After all, the model is trying to figure out how variation there is across block, with only 3 examples, which is a paltry amount of information. At the same time, the random effects for Year and Paddock only have 5 and 6 levels, respectively, and they have fairly substantial variance estimates.

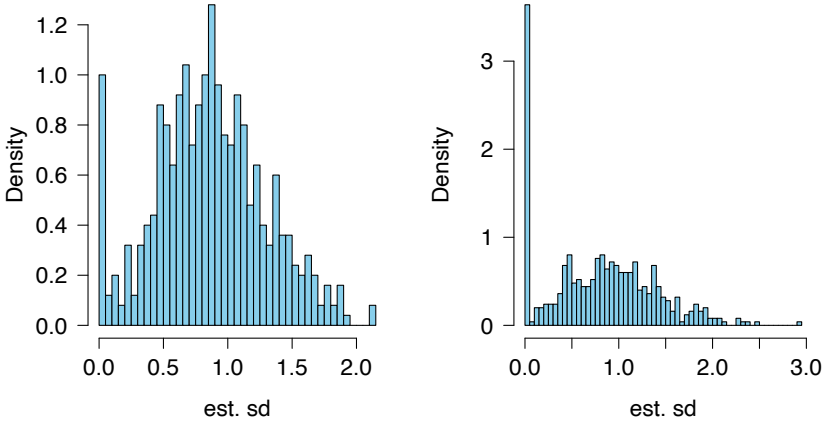
These concerns raise the question of when one should use random vs. fixed effects, and what is the rationale for doing so. This is one of those questions that does not have a consensus answer, and you will see various opinions. I think we can classify the approaches into two categories.

The philosophical approach. The idea is that using fixed vs. random effects for a grouping factor depends on exactly what you're interested in. If you're really interested in comparing particular groups, such as different experimental treatments, or the eight major Hawaiian islands, or 6 species that you're comparing, then you should use fixed effects. This makes sense because a fixed effect will fit a separate parameter for each group, and contrasting these parameters is what the model is designed to do.

Random effects are more appropriate when you're saying 'I want to quantify variation among these groups, but I'm not particularly interested in individual groups and the differences between them.' Random effects are natural for this situation, because you explicitly fit the variance among groups, and get the advantage of not fitting a separate parameter for each group. Examples include a number of different sites or years where observations were taken, different individuals from a population that were sampled multiple times, etc.

I have been emphasizing the importance of random effects for accounting for non-independence in the data. In reality fixed and random effects can both do this, but often random effects make more sense due to the considerations I just outlined.

The pragmatic approach. I think the pragmatic approach mostly agrees with the rational I just sketched, but also pays attention to how many groups were actually sampled. The issue is that when few groups are sample, the variance among those groups will be poorly estimated, and simulation shows that you are more likely to get a variance estimate of zero even when the true variance is greater than zero. Here's a simulation from Ben Bolker (http://rpubs.com/bbolker/4187) than shows this in action:



The true value of the standard deviation is 1, and the distribution of estimates has a peak near 1, but the estimates have an extra peak at 0. When n=5, 5% of the estimates are 0, and when n=3, 18% of the estimates are zero. This suggests that fitting a random effect to a factor with 3 levels is a somewhat risky proposition. In this case the recommendation is that you might want to use a fixed effect instead (it only costs 1 more parameter anyways), to ensure a more robust estimate of differences between groups.

Let's see if this matters for our model, where I used a random effect for block. We can change it to a fixed effect:

```
mod.fix = lmer(richness ~ Treatment + (1|Year) + (1|TransectID) + Block + (1|P
addock), data = butterfly)

summary(mod.fix)

## Random effects:
##  Groups     Name        Variance Std.Dev.
##  TransectID (Intercept) 0.000    0.000
##  Paddock    (Intercept) 0.709    0.842
##  Year       (Intercept) 2.494    1.579
##  Residual               2.513    1.585
## Number of obs: 90, groups:  TransectID, 18; Paddock, 6; Year, 5
##
## Fixed effects:
##              Estimate Std. Error     df t value Pr(>|t|)
## (Intercept)     4.489      1.041  4.890    4.31    0.008 **
## Treatmentlow    1.422      0.764  2.000    1.86    0.204
## BlockB         -0.700      0.936  2.000   -0.75    0.533
## BlockC         -1.033      0.936  2.000   -1.10    0.385
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##

Anova(mod.fix)
```

```
## Analysis of Deviance Table (Type II Wald chisquare tests)
##
## Response: richness
##           Chisq Df Pr(>Chisq)
## Treatment  3.46  1      0.063 .
## Block      1.27  2      0.530
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The parameter estimates for Block are not tiny, but they have large standard errors, suggesting they are not well distinguished from zero. And a likelihood ratio tests shows no evidence for a significant block effect. In this case I would say that the random and fixed effects give a similar result: variation among blocks is not important. In general, deciding which way to go in this situation is a judgement call and may depend on how many other terms are in the model.

Testing the treatment effect

We're taking a long journey through the intricacies of mixed models while doing this analysis. The ultimate goal here is to test the effect of the experimental treatment. So let's do that. We've learned that an approximate F-test and parametric bootstrap are both OK options, and we can compare that to the chi-square likelihood ratio test as well:

```
anova(mod, ddf = "Kenward-Roger")
```

```
## Analysis of Variance Table of type 3  with  Kenward-Roger
## approximation for degrees of freedom
##           Sum Sq Mean Sq NumDF DenDF F.value Pr(>F)
## Treatment   13.8    13.8     1     2    4.24   0.18
```

Looks like treatment is not significant. Look at the denominator degrees of freedom: only 2! It seems like the degrees of freedom should be 4, because there are 6 paddocks and 2 treatment levels, and the anova formula denominator df is N-k. Let's see what the Satterthwaite approximation for the degrees of freedom produces:

```
anova(mod, ddf = "Satterthwaite")
```

```
## Analysis of Variance Table of type 3  with  Satterthwaite
## approximation for degrees of freedom
##           Sum Sq Mean Sq NumDF DenDF F.value Pr(>F)
## Treatment   13.8    13.8     1  5.33    5.49  0.063 .
```

This version has df = 5.33, and a somewhat larger F-value as well, and a p-value 3 times lower. Clearly these F-test approximations get pretty sketchy when you have a complex mixed model with a small amount of effective degrees of freedom. So we shouldn't inhale too deeply on the numbers.

I tend to prefer the parametric bootstrap, so let's try that:

```
mod.null = lmer(richness ~ 1 + (1|Year) + (1|TransectID) + (1|Block) +
(1|Paddock), data = butterfly, REML = FALSE)
```

```
PBmodcomp(mod, mod.null)
```

```
## Parametric bootstrap test; time: 50.02 sec; samples: 1000 extremes: 140;
## large : richness ~ Treatment + (1 | Year) + (1 | TransectID) + (1 | Block)
+
##     (1 | Paddock)
## small : richness ~ 1 + (1 | Year) + (1 | TransectID) + (1 | Block) +
##     (1 | Paddock)
##         stat df p.value
## LRT    3.79  1   0.052 .
## PBtest 3.79      0.141
```

It looks like the chi-squared LRT approximation thinks the treatment effect is marginally signficant, but the parametric boostrap more than doubles that p-value. This is the kind of data where the chi-squared approximation is problematic: there's really only 3 replicates of the experimental manipulation, i.e. 6 total paddocks, and so the degrees of freedom are quite small and the asymptotic assumption of the chi-squared LRT is too optimistic. The parametric bootstrap is somewhere between the F-test results, so at least the results are somewhat consistent.

We can also compare this to AICc out of curiosity:

```
library(MuMIn)
AICc(mod)

## [1] 373.6

AICc(mod.null)

## [1] 375
```

This is actually pretty similar to the parametric bootstrap. Removing treatment from the model increases AICc, but only by about 1.5, which is weak evidence that you need treatment to make a good model. So based on these results, it would seem like the authors' experiment did not have a convincing effect on butterfly species richness. But our exploratory plot (and our thinking) suggested that the effect of the treatment may have increased over time, so we should consider an interaction.

Note you don't generally want to report multiple kinds of inference in a publication, it smacks of 'p-hacking', i.e. trying everything until you find a 'significant' result. I'm just doing this for pedagogical purposes.

Random treatment*year interaction: multivariate random effects

We want to see if the effect of treatment varies by year. Treatment is a fixed effect and Year is a random effect. One option would be to change Year to a fixed effect (it only has 5 levels after all), and fit a standard fixed effects interaction, which would require 2*5 parameters. I think that would be defensible, though not ideal due to the large number of parameters relative to the experimental replication. But right now let's learn about interactions between fixed and random effects.

An interaction between a fixed effect and a random effect has to be specified as a new random effect. There are two ways to do this with lmer, and I will compare both. Right now the model looks like this:

```
mod.rand = lmer(richness ~ Treatment + (1|Year) + (1|TransectID) + (1|Block) +
(1|Paddock), data = butterfly)
```

With a fixed effect for Treatment and a random effect for Year. To say "let the effect of Treatment vary randomly by year", we can do this:

```
mod.inter = lmer(richness ~ Treatment + (Treatment|Year) + (1|TransectID) +
(1|Block) + (1|Paddock), data = butterfly)
```

Let's look at what gets estimated for the random effects in this model:

```
VarCorr(mod.inter)

##  Groups     Name         Std.Dev. Corr
##  TransectID (Intercept)  0.00e+00
##  Paddock    (Intercept)  7.56e-01
##  Year       (Intercept)  1.18e+00
##             Treatmentlow 1.23e+00 0.50
##  Block      (Intercept)  1.78e-07
##  Residual                1.47e+00
```

The other terms look the same as before, but now the results for Year have three components: a St Dev for (Intercept), a St Dev for Treatmentlow, and a 'Corr'. Understanding these components requires a bit of thought. The model has a fixed effects term for Treatment, and this will estimate two parameters: the Intercept, which quantifies the mean richness in the high grazing treatment, and Treatmentlow, which quantifies the difference in mean richness between the low grazing treatment and the high grazing treatment. When we put in the random effect (Treatment|Year), *this allows each of the two parameters to vary randomly by Year*. I.e., the mean richness in the high grazing treatment can vary by Year, and the difference between the low and high treatment can also vary by Year. The variance component for Year called "(Intercept)" is quantifying the former, and the variance component for Year called "Treatmentlow" is quantifying the latter. So in this formulation, the variance component for Year:Treatmentlow is telling us the magnitude of the interaction between Year and Treatment, because it quantifies how much the effect of Treatment varies by year. The estimate is 1.23, which is of similar magnitude to the variation in Intercept across years (1.18).

The 'Corr' parameter is something we haven't seen before. *This quantifies the correlation, across years, between the Year-specific Intercept and the Year-specific Treatment effect*. I.e., this is the correlation between the random effect for Intercept and Treatmentlow. The model estimates this correlation because we are now drawing multiple random effects by year (the intercept and the treatment effect), and the default approach is to assume that these two random effects are not necessarily independence of each other. For example, years that have high overall species richness may also tend to have a larger difference between treatments, and vice versa. Because we assume that the random effects are individually normally distributed, when we have multiple random effects by group, they are assumed to follow a *multivariate normal distribution*.

A multivariate distribution produces a *random vector*, as opposed to a single random number (a *scalar*, as opposed to a vector). A multivariate normal distribution with dimension 2 can be written like this:

$$\vec{\mu} = \begin{bmatrix} \mu_I \\ \mu_T \end{bmatrix}, \Sigma = \begin{bmatrix} \sigma_I^2 & \sigma_{IT} \\ \sigma_{IT} & \sigma_T^2 \end{bmatrix}$$

Where $\vec{\mu}$ is the vector of the means of the distribution; I've written the means as μ_I and μ_T, to correspond to the mean Intercept and the mean Treatment effect. The symbol Σ represent the *variance-covariance matrix*. This matrix specifies 1) the variance for each component of the distribution, and 2) the *covariance* between the two components. So σ_I^2 would be the random effects variance for Intercept, σ_T^2 would be the random effects variance for Treatmentlow, and σ_{IT} would be the covariance across years of the random effects for Intercept and Treatmentlow. When random numbers are drawn from a multivariate distribution, the whole vector is drawn at the same time, and the elements of the vector will tend to be correlated or not, depending on the covariance parameter(s) of the distribution.

Mathematically the covariance is related to the correlation like this: $\sigma_{IT} = \rho_{IT}\sigma_I\sigma_T$, where ρ is the correlation between *I* and *T*. That means we could equivalently write the multivariate normal distribution in terms of the correlation between the random effects: $\begin{bmatrix} \sigma_I^2 & \rho_{IT} \\ \rho_{IT} & \sigma_T^2 \end{bmatrix}$. This is info returned by summary() or VarCorr() for an lmer object; the estimated variance (or St. Dev) for each random effect term, as well as the estimated correlation when there are multiple random effects per grouping variable. In this case the estimate is 0.5, which means that Years with higher overall butterfly richness also tend to have a stronger Treatment effect (low grazing treatments have a larger benefit for butterfly richness). You should also note that although I wrote the multivariate distribution for the random effects in terms of $\vec{\mu}$ and Σ, the random effects returned by ranef() are centered around zero. The means $\vec{\mu}$ are quantified by the fixed effects for Intercept and Treatmentlow.

So we've seen that when a grouping variable (like Year) is associated with multiple random effects, the default syntax will estimate any correlation between those random effects. This is probably the best approach to use, because it makes no assumption about whether the effects are correlated or not. However, the covariance (=correlation) is an extra parameter to estimate, and sometimes we may want to remove this parameter, for example if the model doesn't seem to have enough information to estimate it (when this happens, the correlation parameter will get driven to exactly 1 or -1). To remove the correlation, we can do this:

```
mod.nocorr = lmer(richness ~ Treatment + (1|Year) + (1|Year:Treatment)
+ (1|TransectID) + (1|Block) + (1|Paddock), data = butterfly)
```

Now I'm specifying the Year effects as (1|Year) and (1|Year:Treatment). The first term will quantify overall variation across years; the second term will quantify additional variation among year:treatment combinations. This second term will quantify any year*treatment interaction, but it treats each year:treatment combination as an independent group. In other words, it does not know care that some of the random effects are coming from the same year and might be correlated; this is the same as assuming that the random effects drawn from the same year are independent. For this model the output looks like this:

```
## Random effects:
##  Groups          Name        Variance Std.Dev.
##  TransectID      (Intercept) 0.000    0.000
##  Year:Treatment (Intercept) 0.753    0.868
##  Paddock         (Intercept) 0.571    0.756
##  Year            (Intercept) 2.136    1.461
##  Block           (Intercept) 0.000    0.000
##  Residual                    2.174    1.475
## Number of obs: 90, groups:
## TransectID, 18; Year:Treatment, 10; Paddock, 6; Year, 5; Block, 3
```

There are 5 Years and 10 Year:Treatment combinations. The former has a St Dev of 1.46, and the latter 0.868. These are not so different from what we got from the previous model, though they are not identical. It looks like there is variation in Year:Treatment, which is consistent with an interaction between these terms, but we need to test this somehow.

Testing random effects

So far I've talked about testing fixed effects predictors. In some cases you might want to test the importance of a random effect term. Often this isn't necessary, or a good idea, because random effects terms are often used in models to account for non-independence in the data, as opposed to being an interesting predictor. This was the case for the lakes example, and this is the case for the random effects for Block and Paddock in the current example. However, there are cases when the random effect represents a hypothesis and you want to see what evidence there is

for that hypothesis. This is the case for the Treatment*Year interaction we've been talking about.

Testing random effects adds more complexity to the mix, because the rules of thumb for fixed effects don't apply. Likelihood ratio tests tend to be anti-conservative for fixed effects, so I recommended using a parametric bootstrap or F-test. In constrast, likelihood ratio tests tend to be *conservative* when testing random effects. This means that p-values tend to be a bit too high (usually about 2x too high), leading to reduced power. The cause has to do with the fact that the null hypothesis is that the random effects variance is 0, but it's not possible for a variance to go below zero. The upshot is that you can use a Chi-square LRT, and you will know that your answer is conservative. This is often the recommended approach. The alternative is to use a parametric bootstrap; the advantage is that fewer assumptions are necessary. Finally, you can use AIC, but like the LRT it tends to be conservative.

Let's test the random treatment*year interaction:

```
mod = lmer(richness ~ Treatment + (1|Year) + (1|TransectID) + (1|Block) +
(1|Paddock), data = butterfly)
mod.inter = lmer(richness ~ Treatment + (Treatment|Year) + (1|TransectID) +
(1|Block) + (1|Paddock), data = butterfly)
```

```
anova(mod, mod.inter, test = "Chisq")

## refitting model(s) with ML (instead of REML)

## Data: butterfly
## Models:
## mod: richness ~ Treatment + (1 | Year) + (1 | TransectID) + (1 | Block) +
## mod:     (1 | Paddock)
## mod.inter: richness ~ Treatment + (Treatment | Year) + (1 | TransectID) +
## mod.inter:     (1 | Block) + (1 | Paddock)
##           Df AIC BIC logLik deviance Chisq Chi Df Pr(>Chisq)
## mod        7 372 390   -179      358
## mod.inter  9 368 391   -175      350   7.8      2       0.02 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

I forgot to put REML=FALSE in order to compare models, but the anova() function cleverly noticed and refit the models for me. The Chi-square LRT says p = 0.02. So we can conclude that the effect of grazing treatment does seem to vary by year. Based on our plots of the raw data, this is likely because the treatment effect increased over time, which makes sense because the butterfly community may take some years to respond to the grazing regimes.

We can compare to a parametric bootstrap:

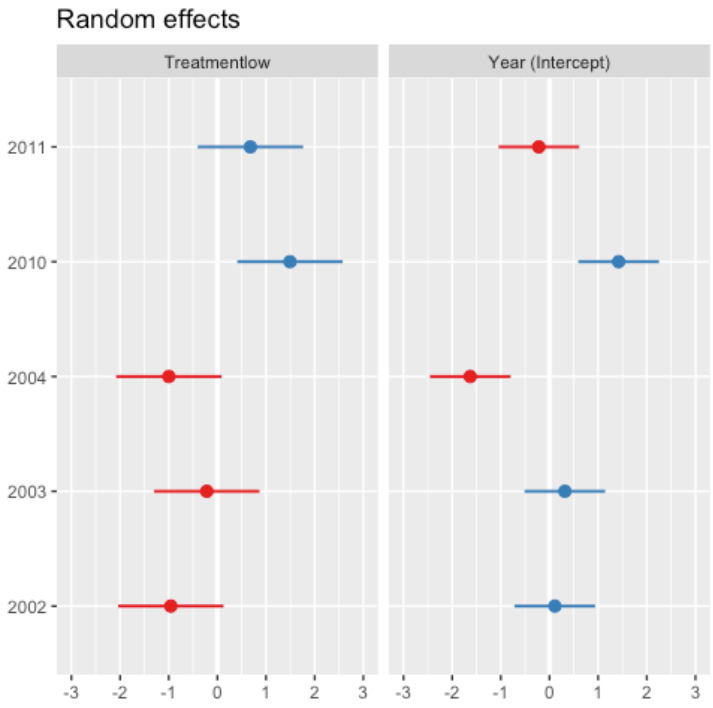
```
PBmodcomp(mod.inter, mod)
```

```
## Parametric bootstrap test; time: 64.27 sec; samples: 1000 extremes:
4;
## Requested samples: 1000 Used samples: 997 Extremes: 4
## large : richness ~ Treatment + (Treatment | Year) + (1 | TransectID)
 +
##     (1 | Block) + (1 | Paddock)
## small : richness ~ Treatment + (1 | Year) + (1 | TransectID) + (1 |
Block) +
##     (1 | Paddock)
##        stat df p.value
## LRT     7.8  2   0.020 *
## PBtest  7.8      0.005 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Looks like the term is slightly more significant using simulation, which is consistent with the fact that LRT tends to be conservative for random effect terms.

If we want to visualized the interaction, we can use a caterpillar plot:

```
plot_model(mod.inter, type = 're')
```

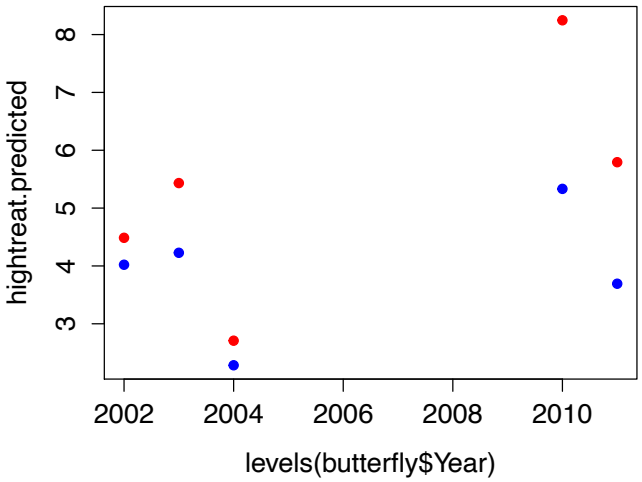


Indeed, 2010 and 2011 have the largest treatment effects. To combine fixed and random effects to see the full prediction, we have to do some work:

```
hightreat.predicted = fixef(mod.inter)["(Intercept)"] + ranef(mod.inter, which
el = "Year")$Year[,"(Intercept)"]

lowtreat.predicted = fixef(mod.inter)["(Intercept)"] + ranef(mod.inter, whiche
l = "Year")$Year[,"(Intercept)"] + fixef(mod.inter)["Treatmentlow"] + ranef(mo
d.inter, whichel = "Year")$Year[,"Treatmentlow"]

plot(hightreat.predicted ~ levels(butterfly$Year), pch = 19, col = 'blue', yli
m = c(min(c(hightreat.predicted, lowtreat.predicted)), max(c(hightreat.predict
ed, lowtreat.predicted))))
points(lowtreat.predicted ~ levels(butterfly$Year), pch = 19, col = 'red')
```



The low grazing treatment is in red, the high grazing in blue. We can see that the years with bigger treatment effects also had higher overall richness, which is what the random effects correlation parameter was quantifying.

When to drop random effects? One last thing to consider. In our model, the random effect variances for Block and TransectID are estimated at zero. Should we just drop them from the model? This is one of those tricky questions without a clear answer, but in general I like to leave the terms in the model, unless it's a situation where it seems like the data is limited and there's too many parameters in the model for it to give good parameter estimates. Even if the variance terms are estimated as zero, they will affect the degrees of freedom calculation if you use an approximate F-test, which makes sense because they are encoding information about how much replication is in the data at different levels.