Mixed-effect/Hierarchical Linear Models

September 26, 2014

Goals of this Lecture

The basic concept:

1. Why bother?
2. Distinction between random intercepts, random slopes, grouping factors, variance components.

Some considerations in application of LMMs:

1. Diagnostics for linear mixed models, additional to those of linear models
2. Issues with model selection and hypothesis testing in LMMs
3. The parametric bootstrap

Motivation

Common to collect data at different spatial, temporal scales.

- Within field sites/river systems/experimental units.
- On the same individual throughout time.
- Across multiple, nested scales: ie. individuals within sites within regions

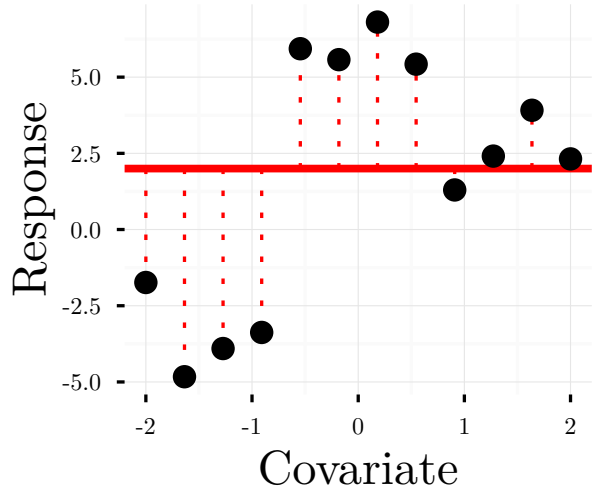
Mixed models (aka multilevel or heirarchical models): explicitly model response variable at different scales, and also heterogeneity in response at different scales.

Motivation: Statistical Dependence

With normal models, we always have *observation specific* random variation (aka residuals or error). For observation i:

$$y_i = \alpha + \beta x_i + \epsilon_i$$

Example with $\beta = 0$ (for clarity):



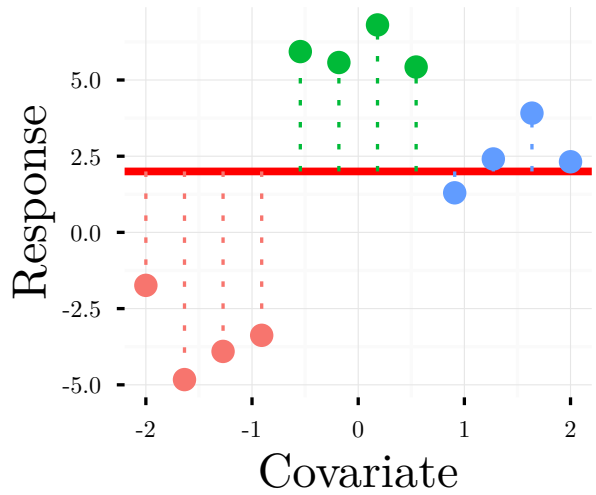
Residuals follow a normal distribution with mean 0 and scale σ.

$$\epsilon_i \sim \mathcal{N}(0, \sigma)$$

Motivation: Statistical Dependence

Could also consider random variation at **more coarse scale** than single observations.

For example **random variation among sites**, within which observations are collected.

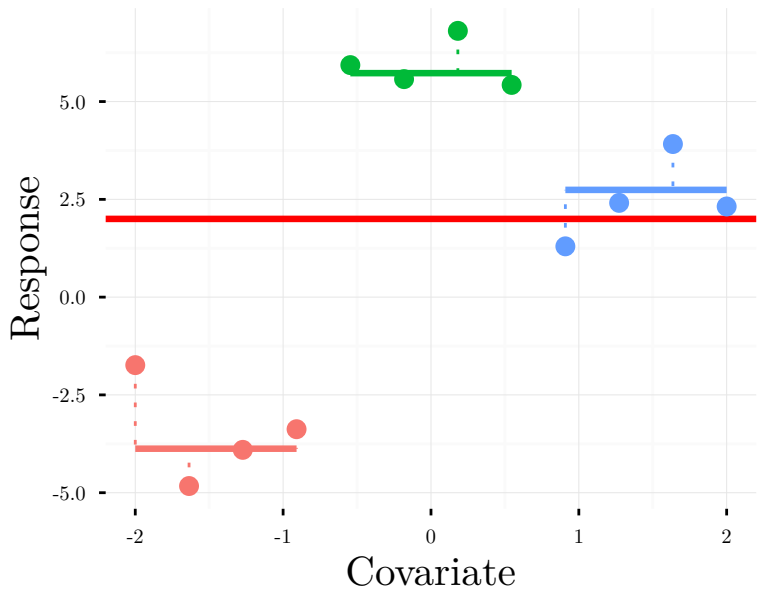


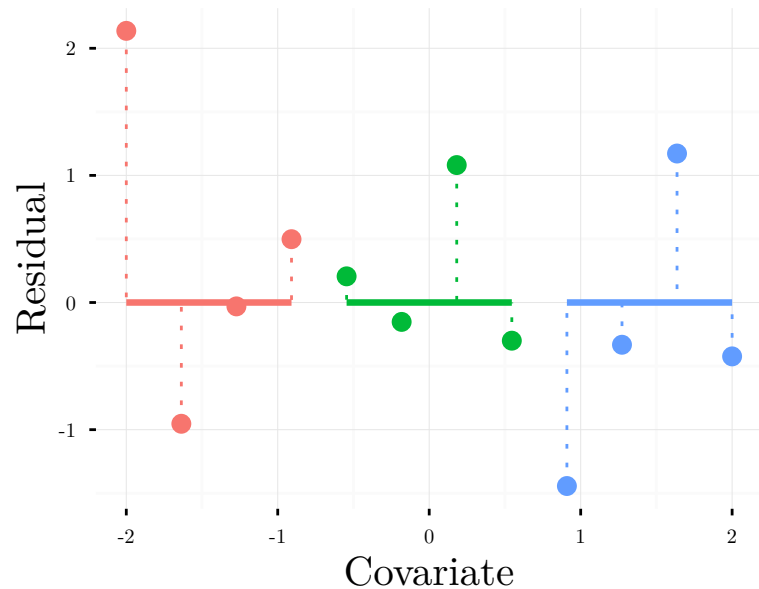
Colors indicate which observations belong to which site.

Motivation: Statistical Dependence

For this example, consider the site-level means.

Then define the observation-level random variation (residuals) as deviations from these site-specific means:

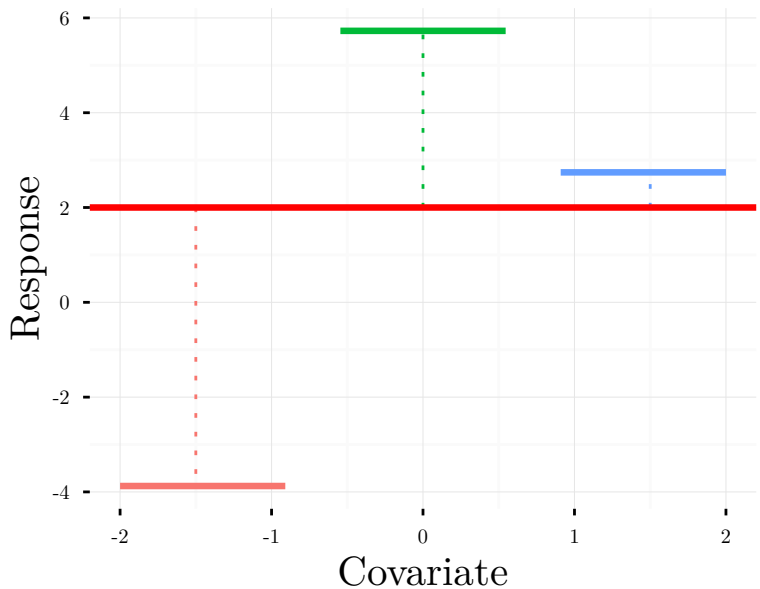


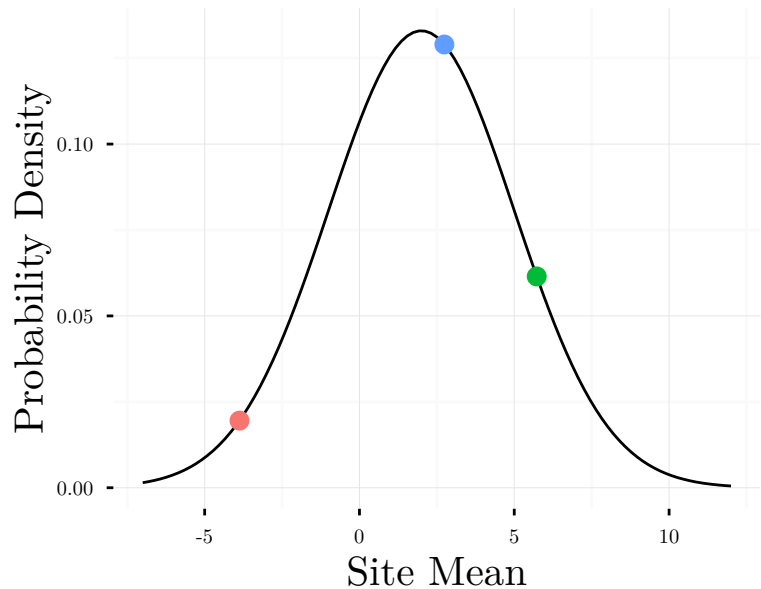


Motivation: Statistical Dependence

The site-level means are also a type of random error (just at a different spatial scale):

They follow a normal distribution, with mean α and scale τ:

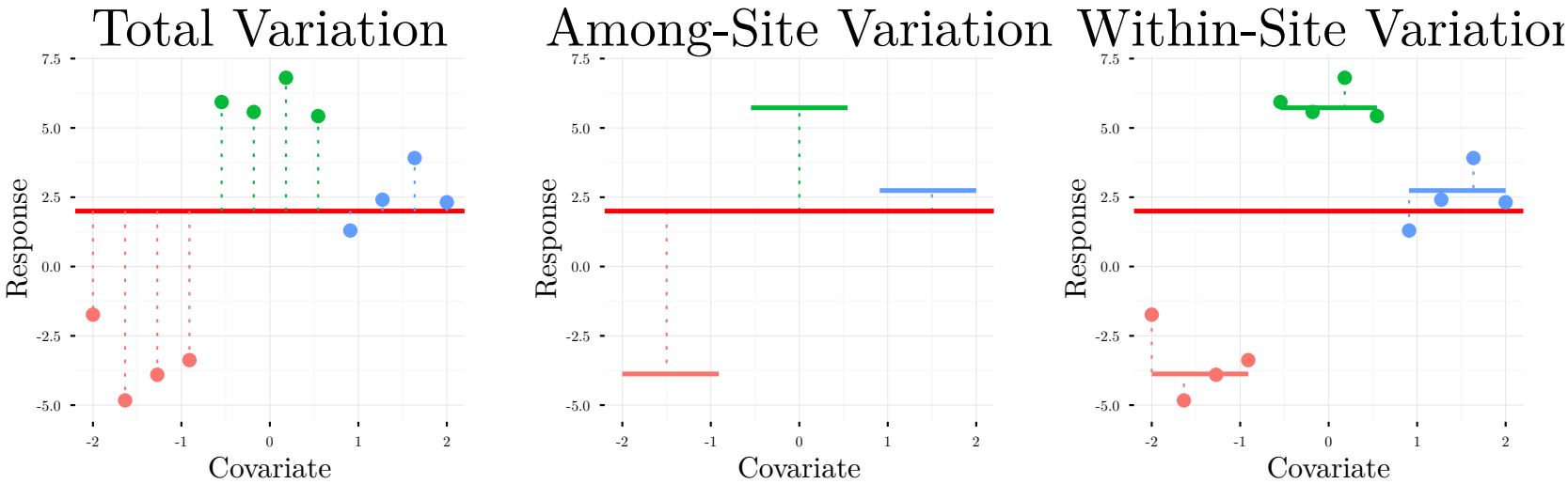




Motivation: Statistical Dependence

The difference between the observation and the overall mean is now the sum of two components.

Among-site variation and **Within-site variation**

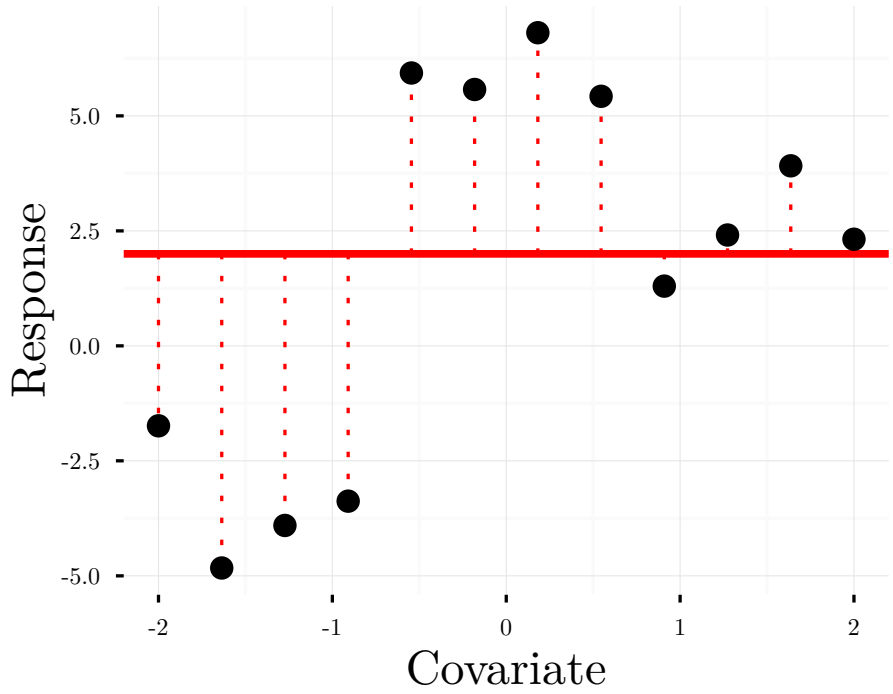


The among-site variation (the site level means), can be modelled as a **random effect**.

Motivation: Statistical Dependence

The residuals in a model ignoring site means are **not independent.**

Residuals from the same site are correlated, as they are deviations from the same site-level mean.



Two observations from the same site are closer in value than are observations between sites.

Motivation: Pseudoreplication

Pseudoreplication is replication where some replicates are not independent.

Examples:

- Observations from same individual.
- Observations from the same spatial location.
- Species that diverged (relatively) recently.

Treating pseudoreplicated data points as independent results in overconfident inference:
we think we have more info then we actually have.

Mixed effects models appropriately account for a known structure in observations while modelling an overall trend.

Motivation: Why use random effects?

Benefits of random effects:

- Account for pseudoreplication/non-independence of data
- Increase power by removing noise at various scales
- Estimate the variance of random effects
  ie. additive genetic variance in trait among individuals
- Predict response for some level higher than the observation
  ie. kriging over geographic space, longitudinal prediction of health
- Shrinkage: levels of same grouping factor *share information*

Structure of Mixed Effects Models

Let i denote observation within site, and j denote site:
so that response[ij] = response$[1, 3]$ is the first observation of the third group.

$$\begin{aligned}\text{response}[ij] = {} & \text{overall mean} + \text{overall slope} \cdot \text{covariate}[ij] \\ & + \text{random intercept}[j] + \text{random slope}[j] \cdot \text{covariate}[ij] \\ & + \text{residual}[ij]\end{aligned}$$

In this equation:

- **Fixed effects** (first line):
  intercept, regression coefficients are the 'overall' mean and response; not specific to any level of the grouping factor.
- **Random effects** (second line):
  deviations from overall intercept and regression coefficients that are specific to levels of the grouping factor; drawn from a distribution
- **Residuals** (third line):
  Observation-level error; drawn from a distribution

Structure of Mixed Effects Models

Random effects have four main components:

- A **grouping factor**
  ie. site, individual
- **Random intercept**
  deviations from overall mean, for each site/individual/etc.
- **Random slope**
  deviations in response to covariate from overall response, for each site/individual/etc.
- **Variance components**
  the variation of intercepts, slopes, controlling how much they deviate from the overall mean.

R syntax (packages nlme / lme4):
<rest of formula> +
(1 + <covariate> | <grouping factor>)

Structure of Mixed Effects Models

The **Grouping factor** is a label which defines structure among data points. Observed, not estimated.

- "Site1", "Site2", etc.

Do not include the grouping factor as both a random effect and fixed effect.

Better to have more levels of the grouping factor, than more replication within levels.

- Allows more accurate estimation of the between-group variance.

Random intercepts and no random slope

Random intercepts are the simplest sort of random effect.

Think of these as *residuals* at a higher scale than the observation:

- For example, site-level deviations from the *overall mean*.

If θ_j is site-level deviation from overall mean α:

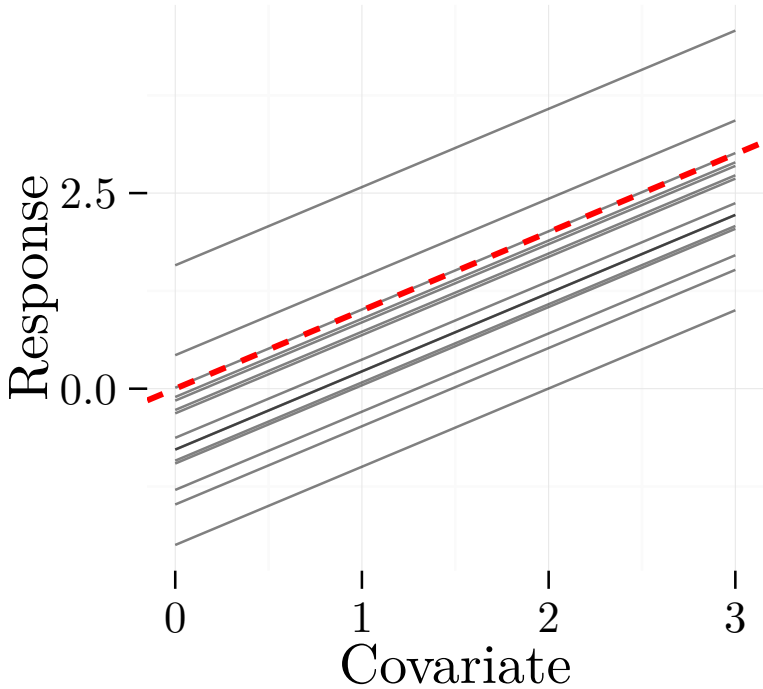
$$\begin{aligned} y_{ij} &= \alpha + \beta x_{ij} + \theta_j + \epsilon_{ij} \\ &= (\alpha + \theta_j) + \beta x_{ij} + \epsilon_{ij} \end{aligned}$$

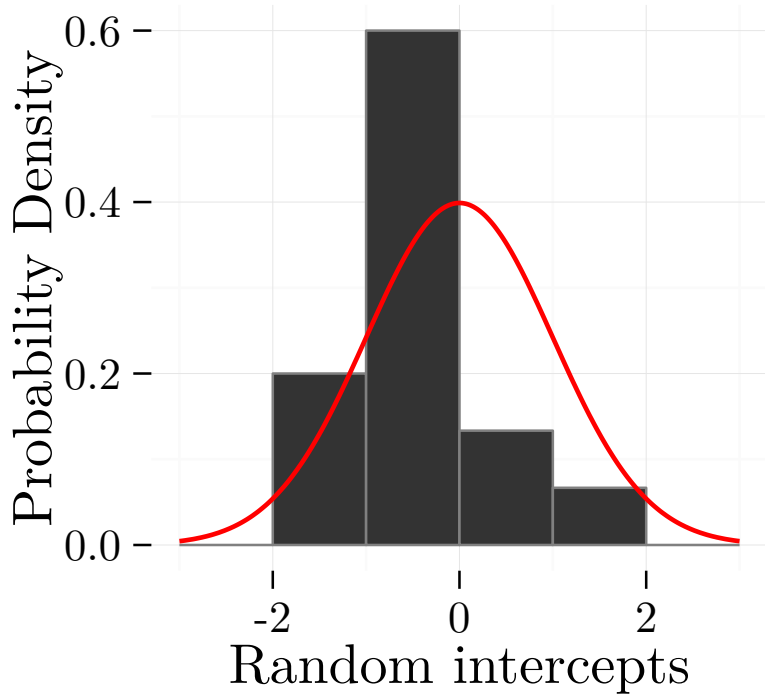
y is observed value, β is overall slope for covariate x, ϵ are residuals (observation-specific errors)

R formula syntax:
<covariate for overall slope> + (1|<grouping factor>)

Random intercepts and no random slope

A visual example: groups with different intercepts but same overall slope ($\beta = 1$).





Called **random intercepts** because they modify the overall intercept (red line) for each level of the grouping factor (black lines), and follow a probability distribution.

Random slopes and no random intercepts

Random slopes incorporate heterogeniety in a response to a covariate, among levels of the grouping factor (ie. different sites).

If our fixed effect slope is the *overall* slope, then the random slopes are site-level deviations from this *overall* slope.

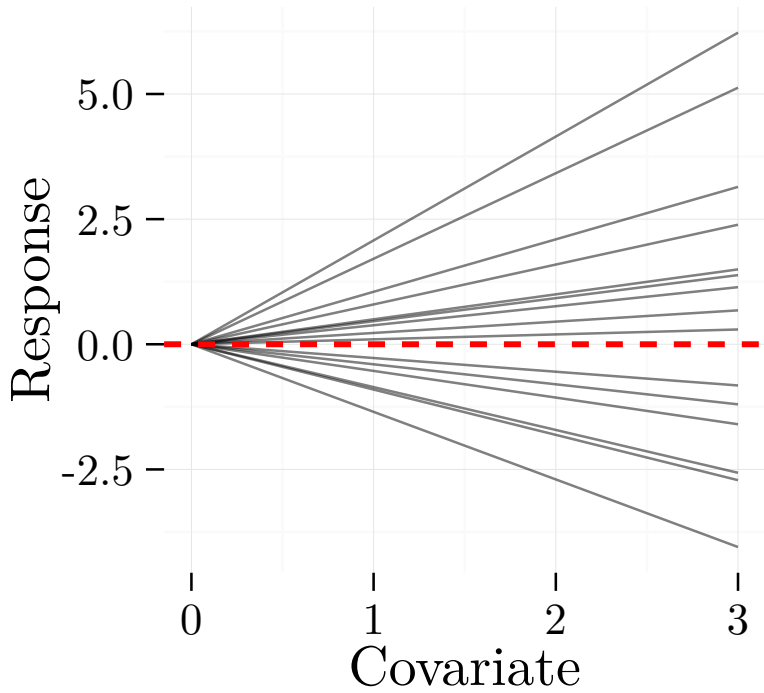
η_j is site-level deviation from overall slope β:

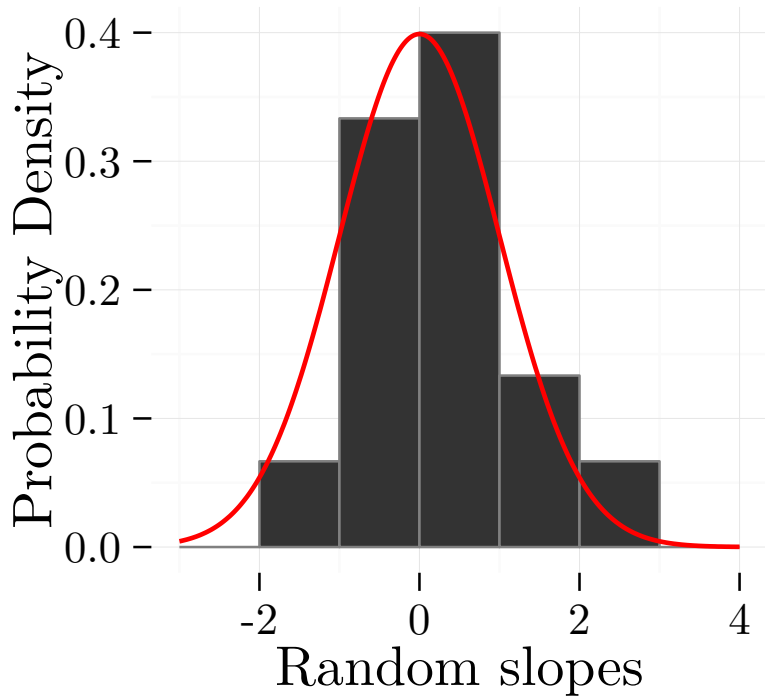
$$\begin{aligned} y_{ij} &= \alpha + \beta x_{ij} + \eta_j x_{ij} + \epsilon_{ij} \\ &= \alpha + (\beta + \eta_j) x_{ij} + \epsilon_{ij} \end{aligned}$$

R formula syntax:
<covariate for overall slope> +
(0 + <covariate>|<grouping factor>)

Random slopes and no random intercepts

Visual example, where the overall slope is set to 0 ($\beta = 0$):

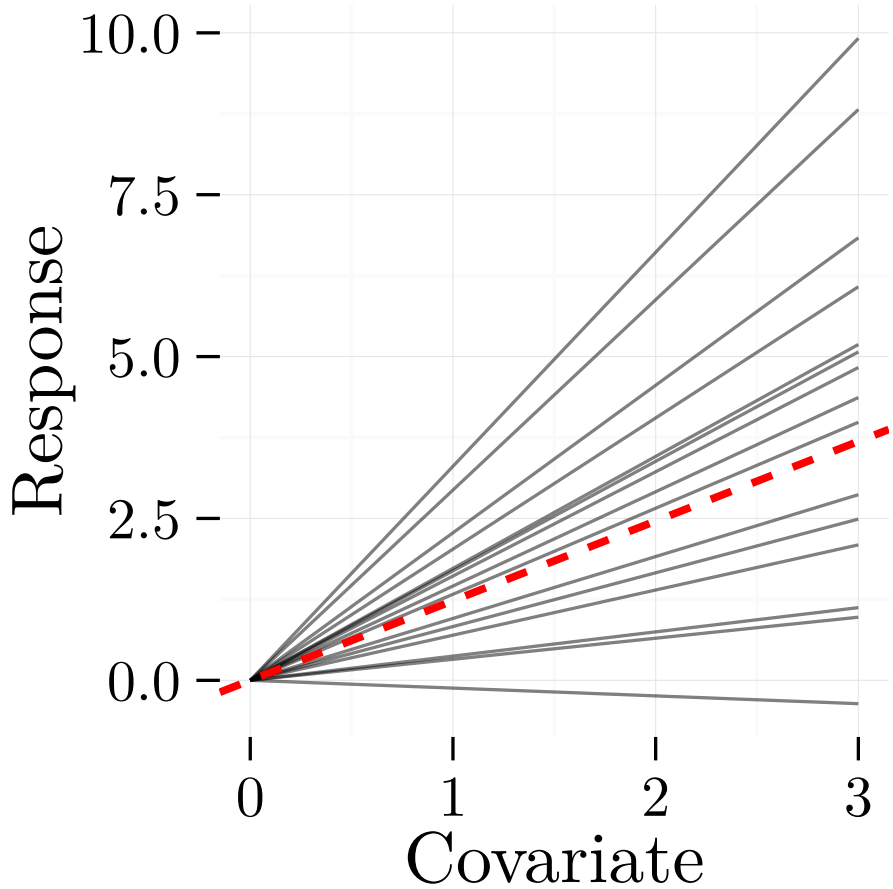




Called **random slopes** because they modify the overall slope (red line) for each level of the grouping factor (black lines).

Random slopes and no random intercepts

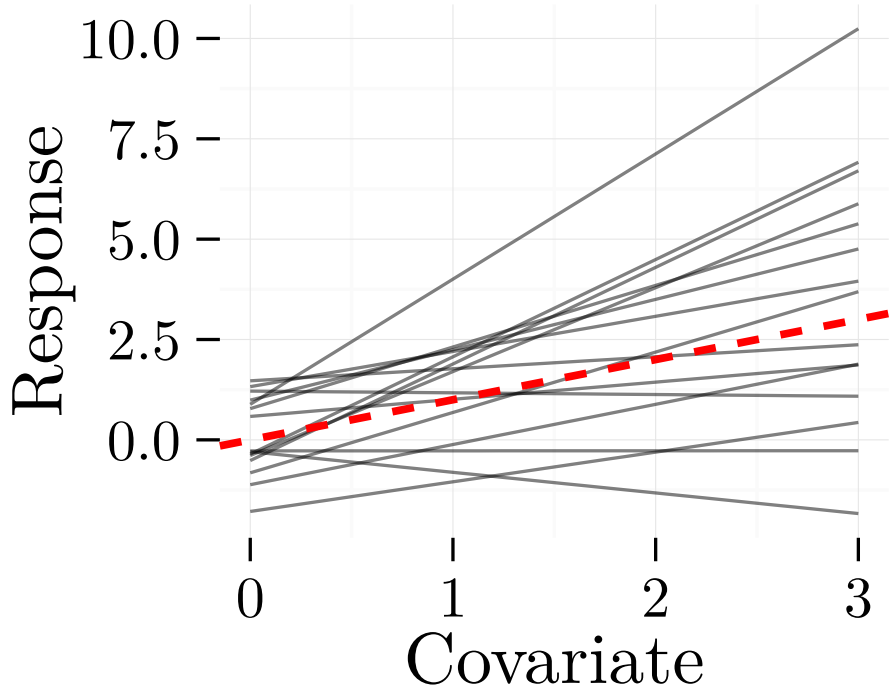
An example where the overall slope is non-zero ($\beta = 1.23$):



It rarely (if ever) makes sense to include a random slope for a covariate, without also including a fixed effect for that covariate.

Random slopes and random intercepts

Random slopes and intercepts can be combined:



In this case we have deviations for the mean for each level of the grouping factor, and also deviations for the slope.

R syntax: **<covariate, fixed effect> + (1 + <covariate, random effect>|<grouping factor>)**

The Distinction Between Fixed and Random

Fixed effects describe the **overall response**.

- For example, the *overall mean* and *overall regression slope*.
- **A common but inaccurate definition is that we care about the point estimates of individual fixed effects and not those of random effects.**
  For example, we might care about an estimated difference among sexes, but not an estimated difference among sites.
- Sometimes we do care about the specific random effects: for example in phylogenetic comparisons, tips and nodes are random effects.

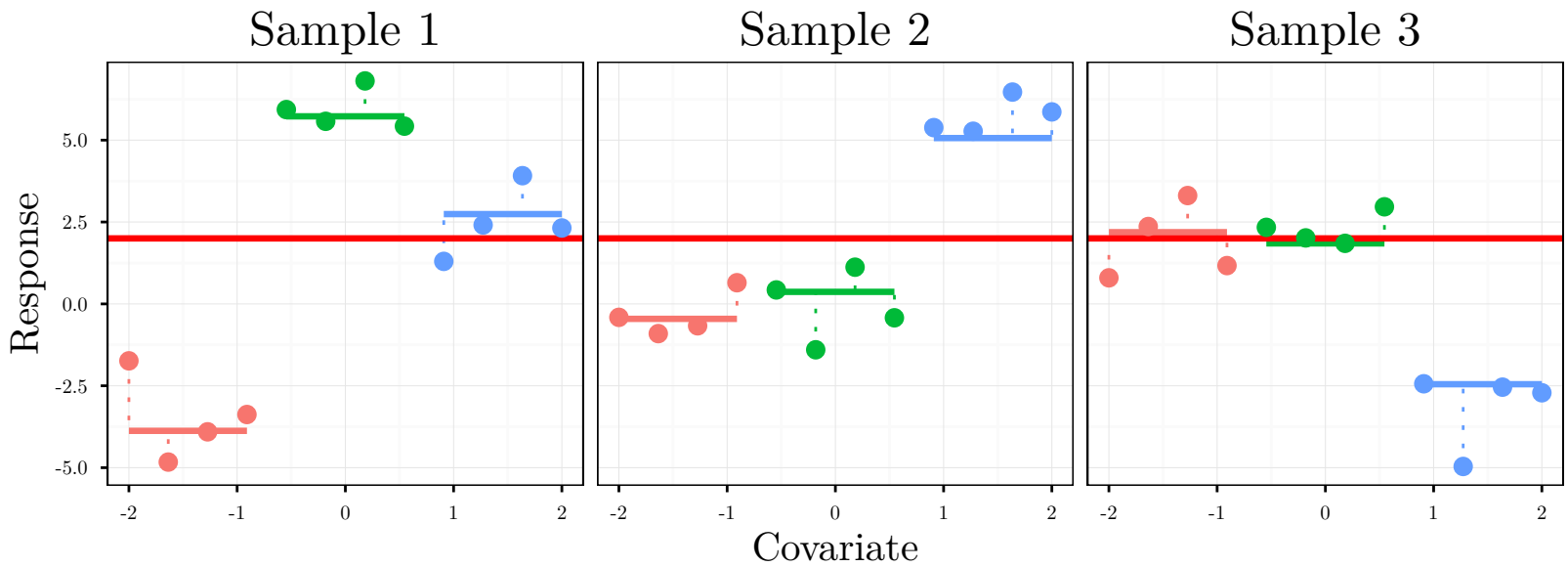
A more accurate definition:

- Random effects are drawn from some larger population, so the values of individual random effects change with each sample (ie. we sample different sites).
- Fixed effects are constant across the entire population, and so do not change in value every time we sample.

The Distinction Between Fixed and Random

Example: repeated sampling of the same system.

In each sample, we randomly select three spatial locations (sites) and collect 4 observations in each site.



Note that random effects (site means) change with each random sample, but the fixed mean (overall mean) does not.

Easier to understand with individuals (not sites): can't usually repeat experiment with same individuals.

Multiple random effects: crossed

We can have multiple grouping factors.

- For example, we might have site and species, or individual and activity.

If these factors are **not** subsets of each other, then they are said to be crossed.

- For example, Species A might occur in multiple sites.

Note that the levels of the grouping factors do not need to be 'completely' crossed (like in a two-way ANOVA)

- For example, Species A might be complete absent from some sites.

R syntax:
<fixed effects> +
(<random intercept/slope>|<grouping factor 1>) +
(<random intercept/slope>|<grouping factor 2>)

Multiple Random Effects: Nested

If multiple grouping factors **are** subsets, they are said to be **nested**.

For example:

- Site within region.
- Site 1 belongs to region B, Site 3 to region C.
- Thus a given site only occurs in **a single region**.

R syntax:

- If site labels are not unique, e.g. each region has sites called "Site1" "Site2" "Site3", etc:
  (1|Region/Site)
- If site labels are unique, e.g. "Region1.Site1", "Region3.Site1", etc.:
  (1|Region) + (1|Site)

Two Approaches to Estimation

Maximum likelihood (ML):

- Given data, finds value of parameters that maximize joint probability of data.
- Results in biased estimates of variance components.
- Applicable to generalized linear mixed models.

Restricted Estimated Maximum Likelihood (REML):

- Likelihood with the *fixed effects integrated out*.
- Finds value of variance components that maximize the joint probability of data over all possible values of fixed effects.
- Results in unbiased estimates of variance components.
- Not applicable to generalized linear mixed models.

The value of REML does not directly depend on values of fixed effects.

Two Approaches to Estimation

Why use REML:

- Gives unbiased (more accurate) estimates of variance components.

Why use ML in place of REML:

- REML cannot be used to select among models with differing fixed effects.
- This restriction includes AIC, likelihood-ratio-test model selection.
- ML becomes unbiased at large sample sizes.

If you want to select among fixed effects, use ML.

However, REML **can** be used to select among different random effects structures, **given that the fixed effects are held constant.**

Bats

Frick et al. (2012) Insectivorous bat pollinates columnar cactus more effectively per visit than specialized nectar bat. *Am. Nat.*

- Response: log pollen deposition
- Covariates: bat species, minutes since sunset
- Random effects (grouping factors): site, individual cactus

Question: do bat species deposit different amounts of pollen?

Bats

```
bats <- read.csv("bats.csv")
str(bats)

## 'data.frame': 89 obs. of  5 variables:
##  $ SiteID   : Factor w/ 5 levels "EB03","LT01",..: 2 1 5 5 5
##  $ CactusID : Factor w/ 44 levels "PPS02","PPS13A",..: 1 44 7
##  $ Species  : Factor w/ 2 levels "ANPA","LEPTO": 2 1 1 1 1 1
##  $ MinSunset: int  249 75 58 62 67 93 196 200 54 63 ...
##  $ Pollen   : int  4000 1287 687 1204 981 1550 147 224 83 840

bats$Pollen <- log(bats$Pollen)
```

Fitting Mixed Models

Two popular packages (**nlme, lme4**) for fitting mixed-effects models.

Why use nlme?

- correlated, heteroskedastic errors
- nonlinear mixed-effects models with normal errors
- more built-in options for plotting stuff by groups

Why use lme4?

- much faster, cutting-edge, under active development
- allows random effects variance to go to zero (drop out of model)
- easily use multiple random effects, crossed & nested
- nicer syntax, in general less cumbersome to extract things from models
- generalized linear mixed models

Fitting Mixed Models

Using **lmer(<formula>, <data>)**:

```
library(lme4)
# model with nested random intercepts
ranint_model <- lmer(Pollen ~ Species + MinSunset +
    (1 | SiteID/CactusID), data = bats)
# model with random slopes for Site
bats$UniqueCactusID <- as.factor(paste0(bats$SiteID,
    bats$CactusID))  #make sure each Cactus has unique ID
ranslope_model <- lmer(Pollen ~ Species +
    MinSunset + (MinSunset | SiteID) + (1 |
    UniqueCactusID), data = bats)

## Warning:  Model failed to converge with max|grad|
= 0.550441 (tol = 0.002)
```

Model summary

```
summary(ranslope_model)


## Linear mixed model fit by REML ['lmerMod']
## Formula:
## Pollen ~ Species + MinSunset + (MinSunset | SiteID) + (1 | UniqueCactusID)
##    Data: bats
##
## REML criterion at convergence: 388.7
##
## Scaled residuals:
##    Min     1Q Median     3Q    Max
## -2.289 -0.600  0.127  0.563  2.031
##
## Random effects:
##  Groups         Name        Variance Std.Dev. Corr
##  UniqueCactusID (Intercept) 0.870527 0.9330
##  SiteID         (Intercept) 3.962715 1.9907
##                 MinSunset   0.000347 0.0186   -1.00
##  Residual                   3.270622 1.8085
## Number of obs: 89, groups: UniqueCactusID, 46; SiteID, 5
##
## Fixed effects:
##               Estimate Std. Error t value
## (Intercept)   6.417968   1.162070    5.52
## SpeciesLEPTO -1.355236   0.550246   -2.46
## MinSunset    -0.000945   0.010323   -0.09
##
## Correlation of Fixed Effects:
##             (Intr) SLEPTO
## SpecisLEPTO  0.011
## MinSunset   -0.964 -0.142
```

Model summary

resid(), **fitted()**, **AIC()**, **model.matrix()**, etc. all work as for GLMs/LMs.

Some new extractor functions:

- **fixef(model)**: returns estimated fixed effects
- **ranef(model)**: returns estimated random effects point estimates
- **VarCorr(model)**: returns random effects variances
- **getME(model, <whatever you want>)**: returns all sorts of nifty things, see **?getME**

Model summary

ranef(<lmer object>) returns point estimates of random effects (BLUPs) as list:

```
str(ranef(ranslope_model))
```

```
## List of 2
##  $ UniqueCactusID:'data.frame': 46 obs. of  1 variable:
##   ..$ (Intercept): num [1:46] 0.23555 0.13387 0.0323 0.29986 0.00282 ...
##  $ SiteID        :'data.frame': 5 obs. of  2 variables:
##   ..$ (Intercept): num [1:5] 0.591 -1.938 -0.815 2.32 -0.158
##   ..$ MinSunset  : num [1:5] -0.00554 0.01814 0.00763 -0.02172 0.00148
##  - attr(*, "class")= chr "ranef.mer"
```

```
head(ranef(ranslope_model)$SiteID)
```

```
##       (Intercept) MinSunset
## EB03       0.5914 -0.005536
## LT01      -1.9376  0.018137
## LT01B     -0.8154  0.007633
## LT03       2.3199 -0.021716
## PB01      -0.1583  0.001482
```

Convergence

The model optimisation **converges** when it reaches maximum likelihood (or REML) estimates for the parameters.

Why might a model fail to converge?

- **Collinearity among covariates.**
  Remove covariate, use PCA to reduce dimension.
- **Very different scales among covariates.**
  Scale variables with **scale(<covariate>)**
- **Correlation among random effects.**
  Center variables $\Rightarrow$ reduce correlation between intercept and mean, simplify random effects structure.
- **Not enough iterations of optimisation algorithm.**
  See ?lmerControl

Convergence

Current lme4 (1.1-6 or so) has new convergence checks, a few of which are false positives.

Distinguish between errors and warnings:

- An error will cause the model fitting to fail, returning nothing.
- A warning tells you something might be up, and returns the (potentially flawed) fit.

Warnings to take seriously:

- *False (singular) convergence*
- *Model nearly unidentifiable*

Warnings that are often false positives:

- Model failed to converge with max grad ...

See this SO post [1] for more info.

[1] http://stackoverflow.com/questions/21344555/convergence-error-for-development-version-of-lme4

Convergence

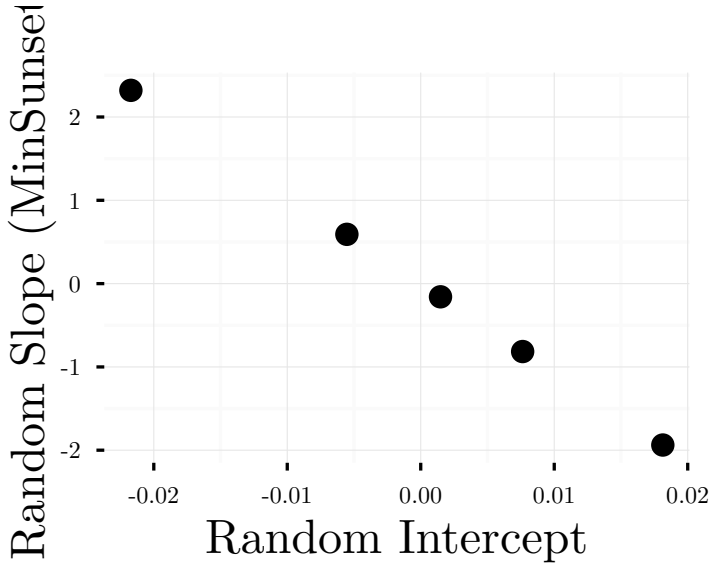
Examples of a convergence error and warning:

```
fails_with_error <- lmer(Pollen ~ MinSunset +
    (MinSunset | SiteID/CactusID), data = bats)

## Error:  number of observations (=89) <= number of
random effects (=92) for term (MinSunset |
CactusID:SiteID); the random-effects parameters and the
residual variance (or scale parameter) are probably
unidentifiable

# create screwy variable
bats$fckd_covariate <- (bats$MinSunset +
    rnorm(bats$MinSunset, 0, 3)) * 1e+05
works_with_warnings <- lmer(Pollen ~ MinSunset +
    fckd_covariate + (1 | SiteID), data = bats)

## Warning:  Some predictor variables are on very
different scales:  consider rescaling
```

Random Effects Correlations

```
print(ranslope_model)

## Linear mixed model fit by REML ['lmerMod']
## Formula:
## Pollen ~ Species + MinSunset + (MinSunset | SiteID) + (1 | Un
##    Data: bats
## REML criterion at convergence: 388.7
## Random effects:
##  Groups          Name        Std.Dev. Corr
##  UniqueCactusID (Intercept) 0.9330
##  SiteID         (Intercept) 1.9907
##                 MinSunset   0.0186   -1.00
##  Residual                   1.8085
## Number of obs: 89, groups: UniqueCactusID, 46; SiteID, 5
## Fixed Effects:
##  (Intercept)  SpeciesLEPTO     MinSunset
##     6.417968     -1.355236     -0.000945
```

Random Effects Correlations



What does the high (> 0.9) correlation between **SiteID:(Intercept)** and **SiteID:MinSunset** indicate?

- Groups with high means have large slopes
- Indicates there is too little information to independently estimate mean and slope
- A bad sign–can result in overfitting, bias of fixed effects estimates.

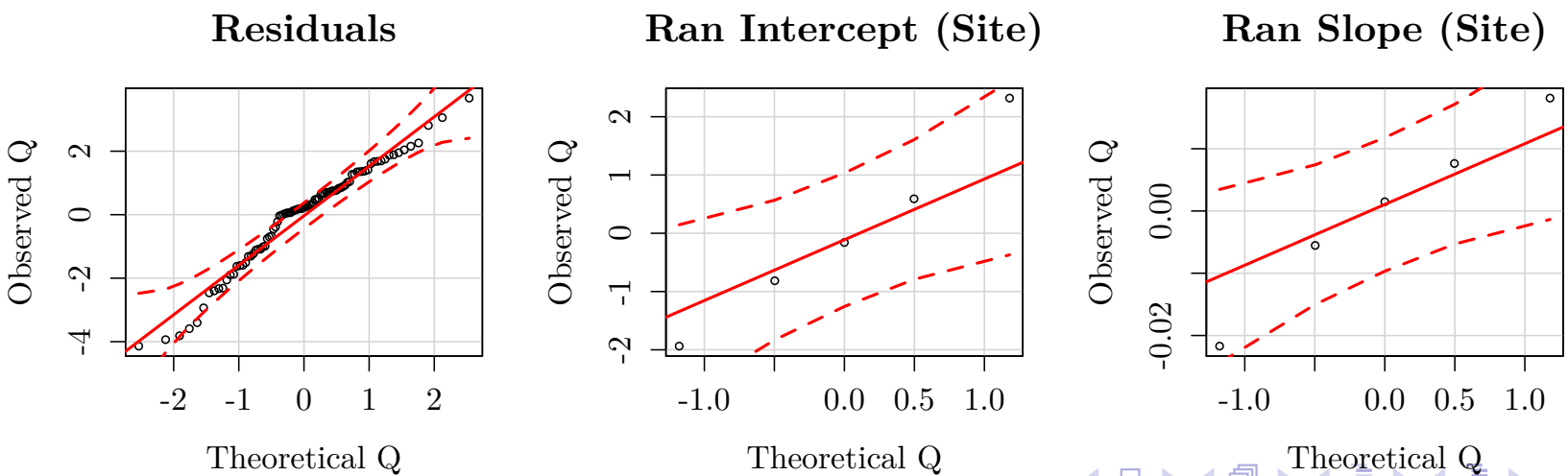
Centering continuous variables may reduce this correlation, but probably best to simplify model.

Normality of Random Effects

All assumptions from linear models apply. (residual normality, homoskedasticity, linearity, independence).

A new assumption for linear mixed models: the random effects estimates (intercepts/slopes) are normally distributed.

```
library(car)
par(mfrow = c(1, 3))
qqPlot(resid(ranslope_model), main = "Residuals",
    xlab = "Theoretical Q", ylab = "Observed Q")
qqPlot(ranef(ranslope_model)[["SiteID"]][,
    1], main = "Ran Intercept (Site)", xlab = "Theoretical Q",
    ylab = "Observed Q")
qqPlot(ranef(ranslope_model)[["SiteID"]][,
    2], main = "Ran Slope (Site)", xlab = "Theoretical Q",
    ylab = "Observed Q")
```



Prediction

Often we want to predict response for certain levels of random effect.

The **predict()** method for lmer objects:
predict(<model object>, <new data>, re.form = <formula>)

Omitting **<new data>** gives predictions for each data point.

Argument **re.form** specifies which random effects to predict for:

- if **re.form = 0**, then 'overall' predictions are generated.
- if **re.form = (MinSunset|SiteID)**, then predictions for each site are generated.

Prediction

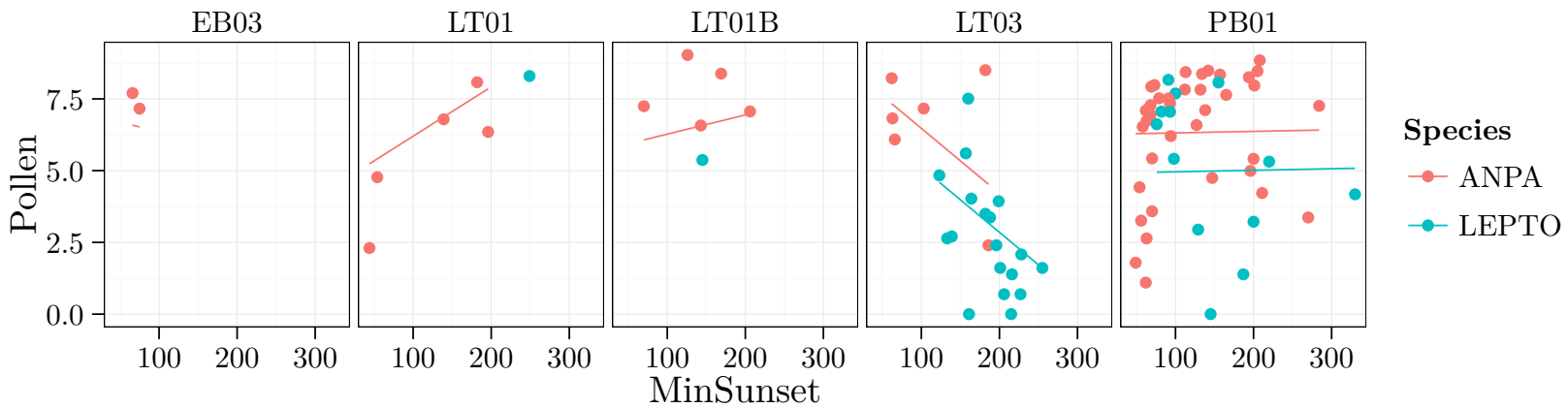
Examples:

```
# overall predictions
bats$preds_overall <- predict(ranslope_model,
    re.form = ~0)
head(bats$preds_overall)

## [1] 4.827 6.347 6.363 6.359 6.355 6.330

# site-level predictions
bats$preds_site <- predict(ranslope_model,
    re.form = ~(MinSunset | SiteID))
head(bats$preds_site)

## [1] 7.406 6.523 6.291 6.293 6.296 6.310
```

Prediction

To visualize: plot predicted, observed values in panels for each level of grouping effect.

```
ggplot(bats, aes(x = MinSunset, y = Pollen,
    color = Species)) + geom_point() + geom_line(aes(y = preds_site)) +
    facet_grid(~SiteID) + theme_minimal() +
    theme(panel.border = element_rect(fill = NA))
```



Residual Degrees Freedom and Hypothesis Tests

In a mixed model, the residual degrees freedom are unspecified.

Why?

- How many data points do we have in a pseudoreplicated dataset?
- With 5 sites and 89 observations:
  do we have 5 or 89 data points?
- The effective number of data points is somewhere inbetween these two extremes.

Consequences:

- Residual DF neccessary for T-tests, F-tests.
- Therefore, no null hypothesis tests $\Rightarrow$ no p-values in output.

Residual Degrees Freedom and Hypothesis Tests

See **?pvalues** for more information.

Three suggested solutions:

- Use approximation to Residual DF
  lmerTest package.
- Use likelihood ratio tests
  anova(<model 1>, <model 2>)
- Use parametric bootstrap
  bootMer() function.

Approximation to Residual Degrees Freedom

Two approximations to the residual DF are common in the literature: **Satterthwaite and Kenward-Roger.**

Both use the random effect variance (ie. between-site variance) to estimate how close the effective degrees freedom is to the # of grouping factors or the # observations.

The greater the between-site variance (ie. as sites are more distinct), the more correlation among data points from the same site.

Both are implemented in the **lmerTest** package.

lmerTest modifies the **lmer()** function directly to provide p-values for T-statistics, and implements p-values for F-tests in **anova(<lmer object>)**.

Approximation to Residual Degrees Freedom

```
library(lmerTest)
new_ranslope_model <- lmer(Pollen ~ Species +
    MinSunset + (MinSunset | SiteID) + (1 |
    UniqueCactusID), data = bats)

## Warning:  Model failed to converge with max|grad| =
0.550441 (tol = 0.002)

summary(new_ranslope_model)$coef

## Warning:  Model failed to converge with max|grad| =
0.550441 (tol = 0.002)
## Warning:  Model failed to converge with max|grad| =
0.550441 (tol = 0.002)

##                 Estimate Std. Error     df  t value Pr(>|t|)
## (Intercept)   6.4179675    1.16207  2.782  5.52287  0.01423
## SpeciesLEPTO -1.3552359    0.55025 70.177 -2.46297  0.01624
## MinSunset    -0.0009454    0.01032  3.521 -0.09158  0.93200
```

Approximation to Residual Degrees Freedom

Syntax for F-tests:
anova(<lmer object>, type = <1 or 3>, ddf=<approximation type>)

```
anova(new_ranslope_model, type = 1, ddf = "Satterthwaite")

## Warning:  Model failed to converge with max|grad| =
0.541634 (tol = 0.002)
## Warning:  Model failed to converge with max|grad| =
0.541634 (tol = 0.002)

## Analysis of Variance Table of type 1  with  Satterthwaite
## approximation for degrees of freedom
##           Sum Sq Mean Sq NumDF DenDF F.value Pr(>F)
## Species    20.47   20.47     1 11.40    4.01  0.069 .
## MinSunset   0.03    0.03     1  3.52    0.01  0.932
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' '

anova(new_ranslope_model, type = 3, ddf = "Kenward-Roger")
```

Likelihood Ratio Tests

A common tool for model selection/hypothesis testing is the **likelihood ratio test**.

Basic idea:

- **improvement in fit** = difference in deviance between models = ΔD.
- **increase in complexity** = difference in number of parameters between models = ΔP.

We ask: is improvement in fit greater than we would expect at random, given increase in complexity?

The null distribution for ΔD is χ^2 with degrees of freedom ΔP. (ie. this is the improvement in fit which could happen by chance, when adding an arbitrary covariate to the model.)

Likelihood Ratio Tests: Application

In R, calculate likelihood ratio test with
anova(<model 1>, <model 2>)

```
full_model <- update(ranslope_model, REML = F)


## Warning:  Model failed to converge with max|grad| = 0.860524 (tol = 0.002)

reduced_model <- update(full_model, . ~ . -
    Species)

## Warning:  Model failed to converge with max|grad| = 0.565527 (tol = 0.002)

anova(full_model, reduced_model)


## Data: bats
## Models:
## ..1: Pollen ~ MinSunset + (MinSunset | SiteID) + (1 | UniqueCactusID)
## object: Pollen ~ Species + MinSunset + (MinSunset | SiteID) + (1 | UniqueCactusID)
##        Df AIC BIC logLik deviance Chisq Chi Df Pr(>Chisq)
## ..1     7 400 418   -193      386
## object  8 397 417   -190      381  5.35      1      0.021 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Models should be nested (differing in single covariate).

Likelihood Ratio Tests: Application

Two approaches:

- **Sequential**: add parameters from null model
- **Marginal**: remove parameters from full model

Analogous to Type I and Type III sums of squares. Easy way to do all marginal tests is with **drop1()**:

```
drop1(full_model, test = "Chisq")  ## gives all marginal likelihood ratio tests


## Warning:  Model failed to converge with max|grad| = 0.565527 (tol = 0.002)
## Warning:  Model failed to converge with max|grad| = 0.890189 (tol = 0.002)

## Single term deletions
##
## Model:
## Pollen ~ Species + MinSunset + (MinSunset | SiteID) + (1 | UniqueCactusID)
##           Df AIC  LRT Pr(Chi)
## <none>       397
## Species    1 400 5.35   0.021 *
## MinSunset  1 395 0.01   0.911
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Hypothesis Testing: Considerations with REML

As mentioned previously, restricted estimated maximum likelihood (REML) does not directly depend on the fixed effects.

This makes REML **inappropriate** for likelihood ratio tests (and AIC) when:

- select models with differing fixed effects.

In **lmer()** the argument **REML = F** fits models with maximum likelihood.

With models fit by ML, it's possible to compare fixed effects as long as random effects structures are identical.

Many functions will automatically refit the model with ML if needed, **but you should always check.**

Hypothesis Testing: Considerations with REML

REML works for F-tests, which rely on variance components to calculate the test statistic.

REML can be used with likelihood ratio tests and AIC, when selecting among random effects structures. In this case, the **same fixed effect structure must be used**.

```
model1 <- lmer(Pollen ~ Species + MinSunset +
    (MinSunset | SiteID) + (1 | UniqueCactusID),
    data = bats)


## Warning:  Model failed to converge with max|grad| = 0.550441 (tol = 0.002)

model2 <- lmer(Pollen ~ Species + MinSunset +
    (MinSunset | SiteID), data = bats)

## Warning:  Model failed to converge with max|grad| = 1.02204 (tol = 0.002)

anova(model1, model2)


## Data: bats
## Models:
## ..1: Pollen ~ Species + MinSunset + (MinSunset | SiteID)
## object: Pollen ~ Species + MinSunset + (MinSunset | SiteID) + (1 | UniqueCactusID)
##        Df AIC BIC logLik deviance Chisq Chi Df Pr(>Chisq)
## ..1     7 396 414   -191      382
## object  8 397 417   -190      381  1.44      1       0.23
```

Parametric Bootstrap

The **parametric bootstrap** is the most accurate, and most flexible option for hypothesis testing.

- Does not rely on approximations or additional assumptions.
- Can generate confidence intervals for any statistic derived from a model.
- However, may be infeasible for huge datasets (millions of observations).

The core idea is to use brute-force simulation to generate the sampling distribution for a statistic of interest, given a statistical model.

Parametric Bootstrap: Confidence Intervals

Basic recipe for confidence intervals of a model statistic:

1. **Fit model** of interest to data.
2. **Simulate data** from model, many times.
3. **Refit the model** to each simulated dataset.
4. **Calculate the statistic of interest** from each simulated model fit.
5. Construct distribution from simulated statistic, calculate confidence intervals, etc.

The PB works because the simulated datasets mimic data where the model is absolute truth.

With enough datasets, can estimate the sampling distribution of any statistic we wish (even those that don't have a closed form).

Parametric Bootstrap: Confidence Intervals

The **bootMer()** function automates steps 2-5.

First, we need to define a function to extract the statistic of interest from the model as a vector.

As an example, consider a function that extracts the point estimates for the site-level random intercepts.

```
getRE <- function(model) {
    return(as.vector(ranef(model)$SiteID[, 1]))
}
```

Parametric Bootstrap: Confidence Intervals

Then run:

bootMer(<lmer object>, <function>, nsim = <number sims>, type = "parametric", use.u = T)

The argument **use.u = T** is necessary if interested in specific random effects (otherwise, it'll simulate new random effects).

Generating CI's for individual random effects is useful application of PB, because the usual approximated confidence intervals are often very inaccurate.

Parametric Bootstrap: Confidence Intervals

Use **boot.ci(<bootMer object>, <conf>, <type>, <index>)** to generate various sorts of bootstrap confidence intervals.

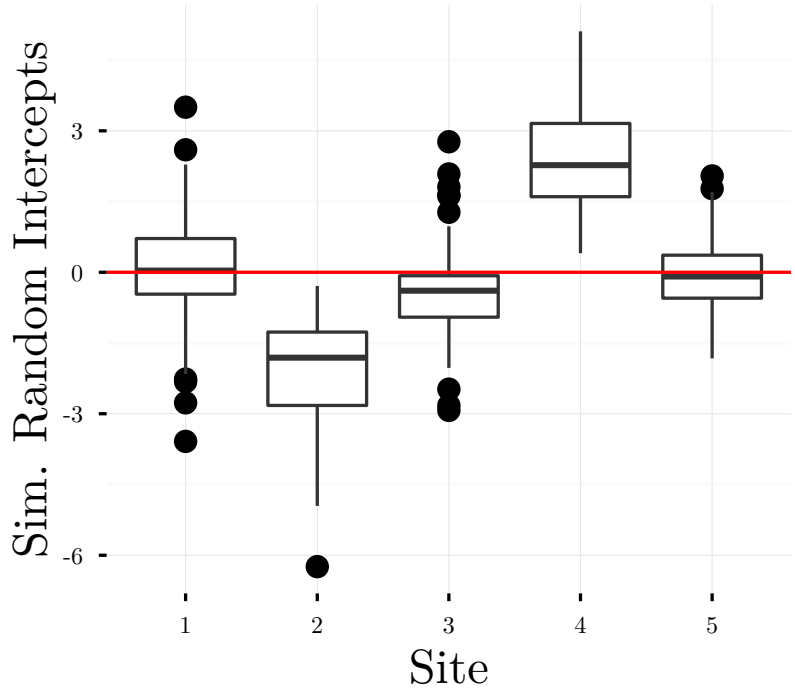
Here <conf> is the confidence level (ie. 0.95), <index> is the index of the statistic to calculate the interval for (we have 5 sites, so could be from 1 to 5).

```
library(boot)
re_boots <- bootMer(ranslope_model, getRE,
    nsim = 99, type = "parametric", use.u = T)
boot.ci(re_boots, conf = 0.9, "norm", 1)
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 99 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = re_boots, conf = 0.9, type = "norm", index = 1)
##
## Intervals :
## Level      Normal
## 90%   (-0.9621,  3.1000 )
## Calculations and Intervals on Original Scale
```

Parametric Bootstrap: Confidence Intervals

```
ggplot(melt(re_boots$t), aes(x = factor(X2),
    y = value)) + geom_boxplot() + geom_hline(yintercept = 0,
    color = "red") + ylab("Sim. Random Intercepts") +
    xlab("Site") + theme_minimal() + theme(axis.title = element_text(size = 8),
    axis.text = element_text(size = 4), axis.ticks.length = unit(0.05,
        "cm"))
```



Note that if the model fits your data poorly, the results of the PB will be rubbish (the same goes for any test from a badly fitting model).

Parametric Bootstrap: Confidence Intervals

You can use the parametric bootstrap for many, many tasks:

- Calculate confidence intervals of fixed effects.
- Calculate confidence intervals of random effects.
- Calculate confidence intervals of variance components.
- Calculate prediction intervals/std. errors of predictions.
- Calculate mean and confidence intervals of any derived function of model parameters.
- A bootstrapped version of the likelihood ratio test, where the χ^2 distribution is replaced with a simulated, empirical distribution.
- Bootstrapped AIC model selection, and model averaged parameters.

Very useful for generalized linear mixed models, where approximations and asymptotic assumptions often fail miserably.

Advanced topics:

- categorical random slopes (called **random coefficients**)
- generalized linear mixed models
  More of the same.
- spatial/phylogenetic random effects
  Assume a distance-based correlation structure among random effects.
- Bayesian approaches to GLMMs
  Fit models of arbitrary complexity, with missing data, with various distributions.

If time permits...

MuMIn is a useful package for information-theoretic model selection/averaging.

AIC is most commonly used. Note that AICc **cannot** be used with mixed models, as AICc formula incorporates residual degrees freedom.

Basic Recipe:

- Fit models, saving as objects in a list
- Use **model.sel()** on list for model selection table
- Use **model.avg()** on list for model averaged coefficients.