9 Mixed models

9.1 Linear Mixed Models (LMM)

Assumption of the linear model:

- $\epsilon_i \sim N(0, \sigma^2)$
- ϵ_i are **independent** and identically distributed.

Why is independence so important?

Recall the formula for the SE of the mean

$$\frac{\sigma}{\sqrt{n}}$$

Violation of the independence assumption lead to an inflation of Type 1 errors (i.e., rejecting H_0 if it is actually true).

We are often in situations, were we have non-indepedent data points, for example:

- measurements from the same individual
- measurements from the similar study areas

Possible solutions:

- Include grouping variables in the model as factor.
- Fit individual models to each group
- Use mixed models (fixed and random effects)

9.1.1 Include grouping variables

- If we include a categorical variable, each level (= group) will have it's own intercept.
- If we want to have group-specific slopes for an other covariate, we can include interactions.

9.1.2 Fit individual models

- We can fit a model to each group/individual
- Do statistic with coefficients (-> list columns are handy here)

The advantage of this approach is:

- Fewer assumption (no distributional assumption)
- Easier to understand

9.1.3 Use mixed models (fixed and random effects)

We allow individual effects for intercept (and slope) to vary around a common mean.

Recall,

$$y_i = \beta_0 + \beta_1 x_i + \epsilon_i$$

9.1.4 Random Intercept Model

$$y_i = \underbrace{(\alpha_j + \beta_0)}_{\text{intercept for group } j} + \beta_1 x_i + \epsilon_i$$

We further assume that

$$\alpha_j \sim N(0, \sigma_\alpha^2)$$

with

- $i = 1, \dots, n$ individuals, and
- $j = 1, \dots, k$ groups.

9.1.5 Random intercept and slope

$$y_i = \underbrace{(\alpha_j + \beta_0)}_{\text{intercept for group } j} + \underbrace{(\gamma_j + \beta_1)x_i}_{\text{slope for group } j} + \epsilon_i$$

We further assume that

$$\begin{bmatrix}\alpha_j \\ \gamma_j\end{bmatrix} \sim N_2(\mathbf{0}, \Sigma)$$

with

- $i = 1, \dots, n$ individuals, and
- $j = 1, \dots, k$ groups, and
- And the variance covariance matrix Σ

$$\Sigma = \begin{bmatrix}\sigma^2_\alpha & \sigma_{\alpha\gamma} \\ \sigma_{\alpha\gamma} & \sigma^2_\gamma\end{bmatrix}$$

9.1.6 In R

The package `lme4` allows fitting such models, with the `lmer()` function.

- Random intercept model: `y ~ x + (1 | g)`
- Random intercept and slope model: `y ~ x + (x | g)`
- Random intercept and slope, and no correlation between slope and intercept (i.e., $\sigma_{\alpha\gamma} = 0$): `y ~ x + (1 | g) + (0 + x | g)`

9.2 Example

We will use again the data set on mountain pines.

The `broom` package:

```r
library(broom)
```

- The package `broom` provides a set of functions to work with models. The three most used functions are `augment()`, `glance()` and `tidy()`.

```r
m1 <- lm(Sepal.Width ~ Sepal.Length, data = iris)
```

The function `augment()` adds several new columns to the data set used to fit the model. These include the fitted ($\hat{y}$) values of the response (`.fitted`) and the residuals (`resid`).

```r
head(augment(m1))
```

```
# A tibble: 6 x 8
  Sepal.Width Sepal.Length .fitted  .resid    .hat .sigma   .cooksd .std.resid
        <dbl>        <dbl>   <dbl>   <dbl>   <dbl>  <dbl>     <dbl>      <dbl>
1         3.5          5.1    3.10  0.397  0.0121   0.435 0.00516        0.919
2         3            4.9    3.12 -0.116  0.0154   0.436 0.000563      -0.269
3         3.2          4.7    3.13  0.0719 0.0195   0.436 0.000277       0.167
4         3.1          4.6    3.13 -0.0343 0.0218   0.436 0.0000709    -0.0798
5         3.6          5      3.11  0.490  0.0136   0.434 0.00893        1.14
6         3.9          5.4    3.08  0.815  0.00859  0.431 0.0154         1.89
```

The function `glance()` provides the main overall information of the model (such as R^2 and adjusted R^2), σ and the values of the F-test as a tibble.

```r
glance(m1)
```

```
# A tibble: 1 x 12
  r.squared adj.r.squared sigma statistic p.value    df logLik   AIC   BIC
      <dbl>         <dbl> <dbl>     <dbl>   <dbl> <dbl>  <dbl> <dbl> <dbl>
1    0.0138       0.00716 0.434      2.07   0.152     1  -86.7  179.  188.
# i 3 more variables: deviance <dbl>, df.residual <int>, nobs <int>
```

`tidy()` summarizes a model in terms of the estimated coefficients and returns a *tidy* data.frame.

```r
tidy(m1)
```

```
# A tibble: 2 x 5
  term         estimate std.error statistic  p.value
  <chr>           <dbl>     <dbl>     <dbl>    <dbl>
1 (Intercept)    3.42      0.254      13.5  1.55e-27
2 Sepal.Length  -0.0619    0.0430     -1.44 1.52e- 1
```

```r
dat <- read.table(here::here("data/pines/Data2.txt"), header = TRUE)

head(dat, 2)
```

```
  code.tree         site aspect.tree         NS         EW z.coord.tree slope
1      D003 SNP.South.18         214 -0.8290376 -0.5591929       1919.6    34
2      D004 SNP.South.18         221 -0.7547096 -0.6560590       1915.9    30
  tree.height longest.core.code longest.core.age longest.core.dbhbb
1        10.3           D003.L1              144             14.946
2         8.3           D004.R1              140             17.258
  lc.growth.rate50 lc.growth.rate50.cat
1           0.8400          0.75-1.0 mm
2           0.9578          0.75-1.0 mm
```

```r
dat <- data.frame(
  dbh = dat$longest.core.dbhbb,
  gc = dat$lc.growth.rate50.cat,
  age = dat$longest.core.age,
  site = dat$site
)
```

Number of trees per site:

```r
table(dat$site)
```

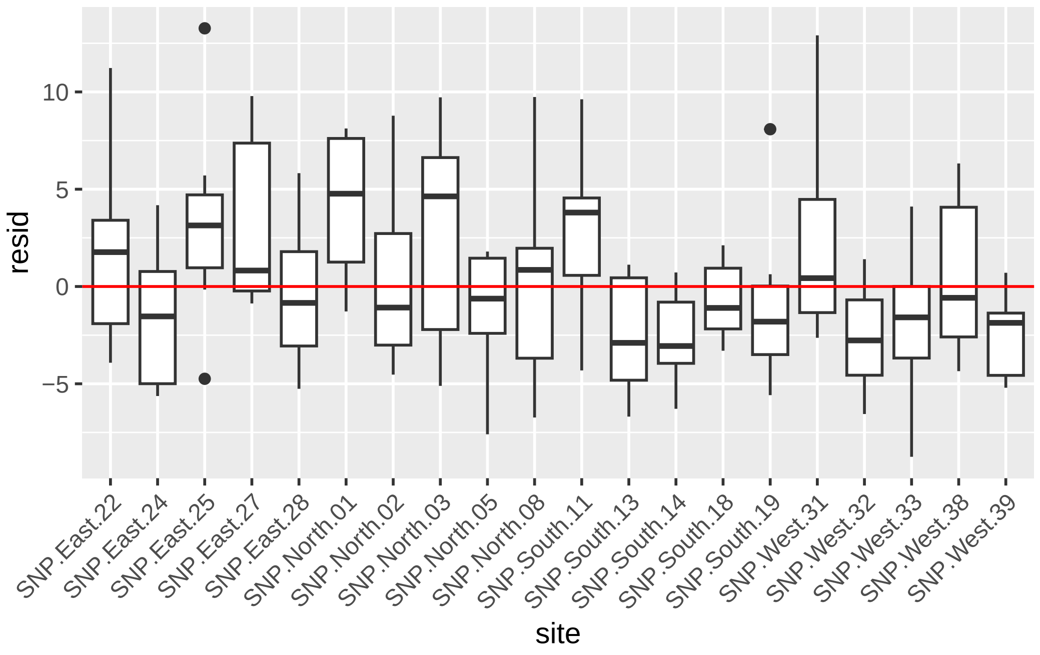
```
 SNP.East.22  SNP.East.24  SNP.East.25  SNP.East.27  SNP.East.28 SNP.North.01 
           9            9            7            7            8            4 
SNP.North.02 SNP.North.03 SNP.North.05 SNP.North.08 SNP.South.11 SNP.South.13 
          10            9            7            9            6            9 
SNP.South.14 SNP.South.18 SNP.South.19  SNP.West.31  SNP.West.32  SNP.West.33 
           7            7           10            7            8           10 
 SNP.West.38  SNP.West.39 
           9            8 
```

We start by fitting a simple linear model.

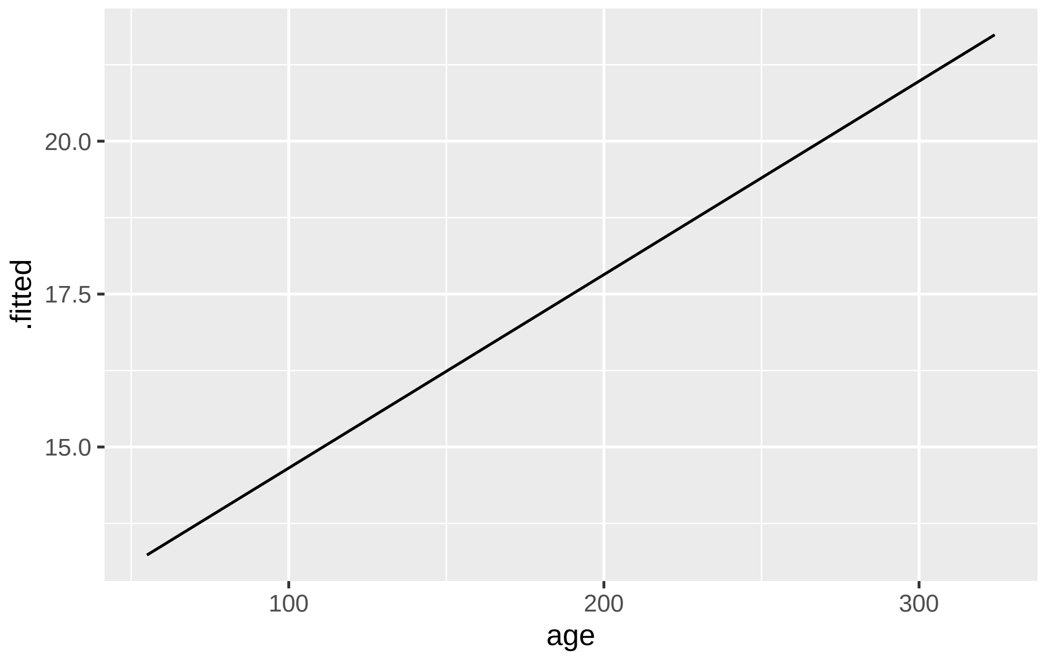
```r
m1 <- lm(dbh ~ age, data = dat)
```

- The residuals for each site (the function `add_residuals()` from the `modelr` package is used here to add the residuals to the data.
- If the model is correct, we expect that the site does not have an effect on the residuals.

```r
dat %>% add_residuals(m1) %>%
  ggplot(aes(site, resid)) + geom_boxplot() +
  geom_hline(yintercept = 0, col = "red") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



```r
augment(m1) %>% ggplot(aes(age, .fitted)) + geom_line()
```

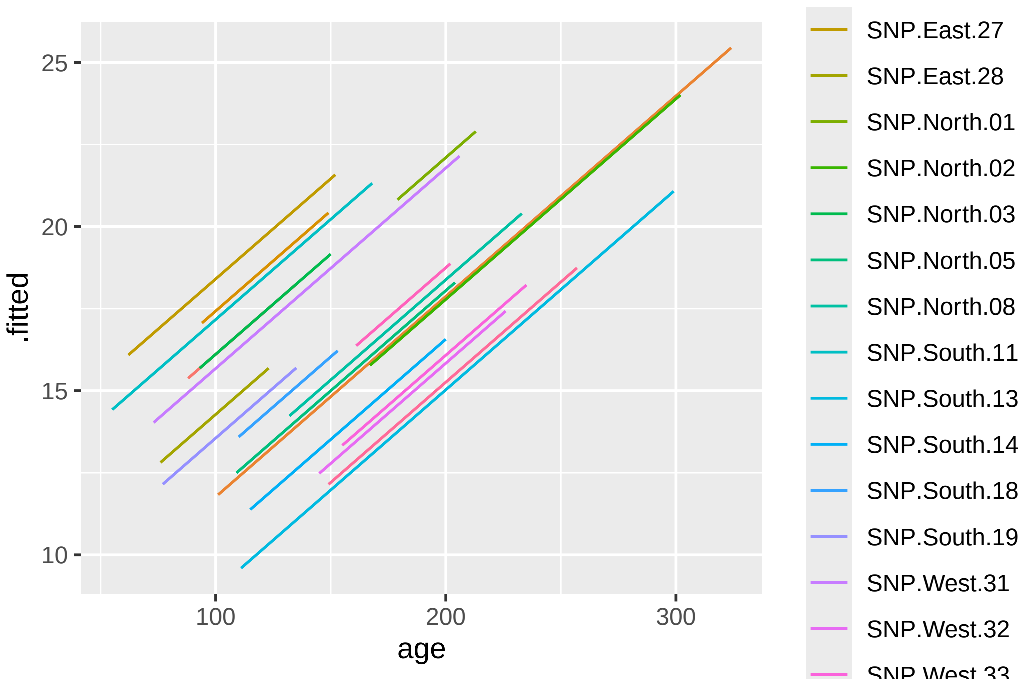


We could add site as a covariate in the model.

```r
m2 <- lm(dbh ~ age + site, data = dat)
```

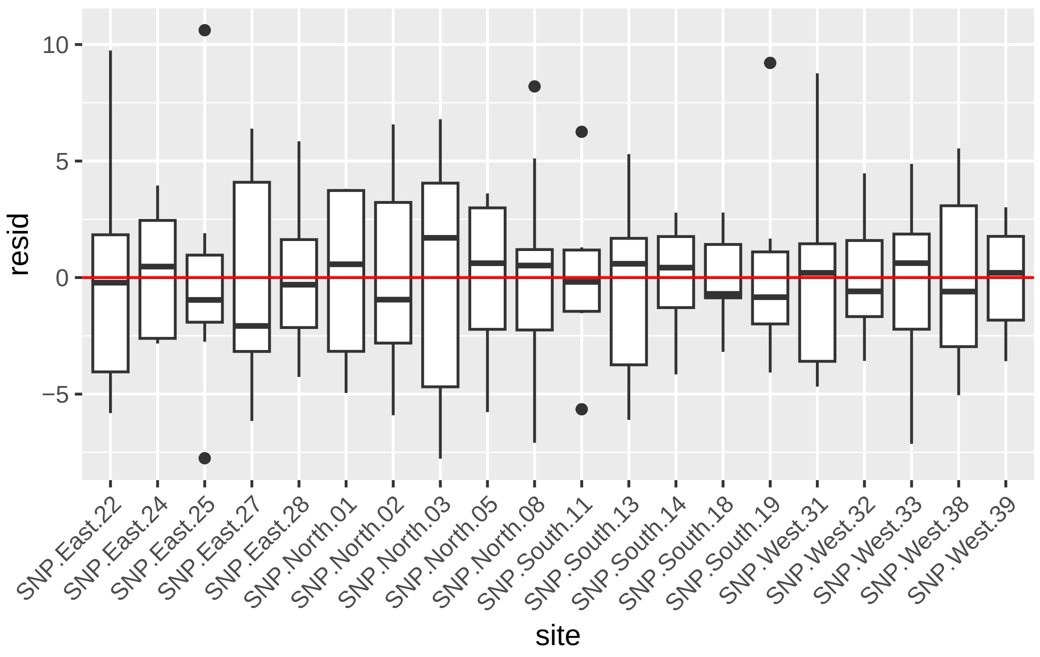
This allows each site to have its own intercept.

```r
augment(m2) %>% ggplot(aes(age, .fitted, col = site)) + geom_line()
```



This improves the residuals

```
dat %>% add_residuals(m2) %>%
  ggplot(aes(site, resid)) + geom_boxplot() +
  geom_hline(yintercept = 0, col = "red") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



But take a look at the number of coefficients that we are fitting now

```r
length(coef(m2))
```

```
[1] 21
```

```r
nrow(dat)
```

```
[1] 160
```

- Assuming we need 10 - 20 data points for each parameter that we want to estimate, this can be problematic.
- Interpretation becomes tedious at best.

```r
coef(m2)
```

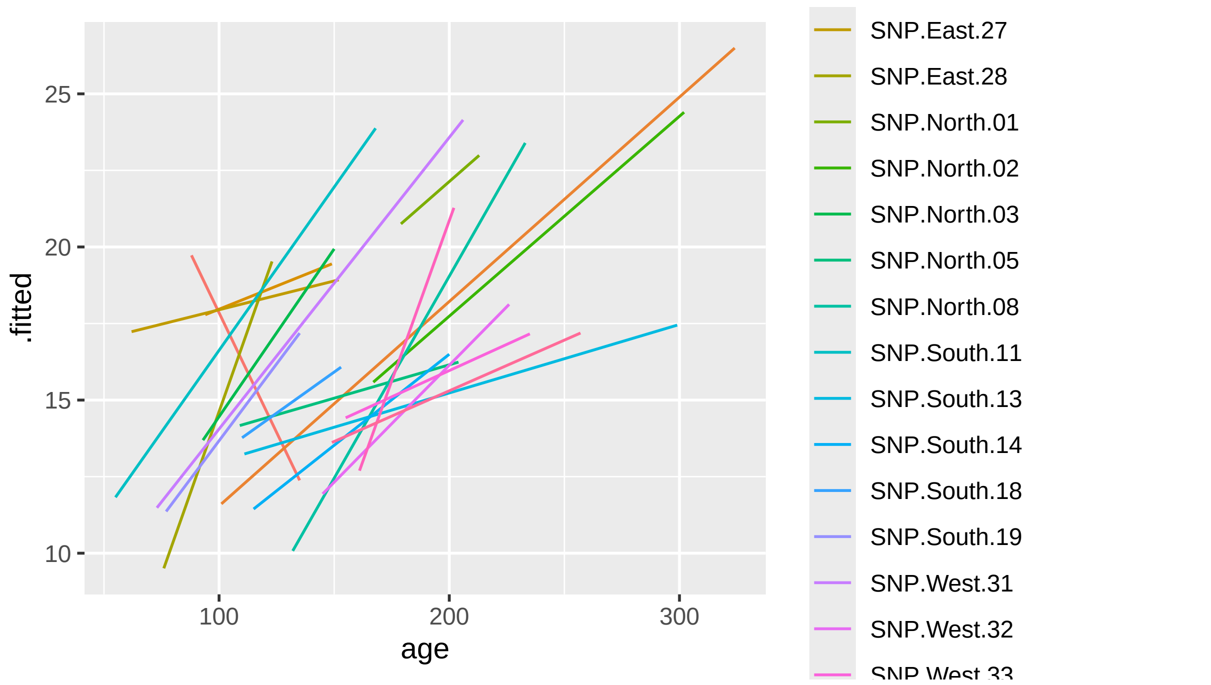
```
     (Intercept)              age  siteSNP.East.24  siteSNP.East.25 
    10.005204952      0.061058802     -4.342966637      1.318461455 
  siteSNP.East.27  siteSNP.East.28 siteSNP.North.01 siteSNP.North.02 
     2.295838994     -1.830702836     -0.113083117     -4.431706270 
siteSNP.North.03 siteSNP.North.05 siteSNP.North.08 siteSNP.South.11 
     0.001242395     -4.163689066     -3.833972067      1.059542570 
```

```
siteSNP.South.13 siteSNP.South.14 siteSNP.South.18 siteSNP.South.19
    -7.187161295     -5.646999929     -3.128496175     -2.553214615
 siteSNP.West.31  siteSNP.West.32  siteSNP.West.33  siteSNP.West.38
    -0.431453859     -6.378068692     -6.133800721     -3.467017925
 siteSNP.West.39
    -6.956053532
```

- The previous model allowed different sites to have different intercepts, but all sites have the same slope.
- We could add an interaction between `age` and `site` to allow for different slopes.

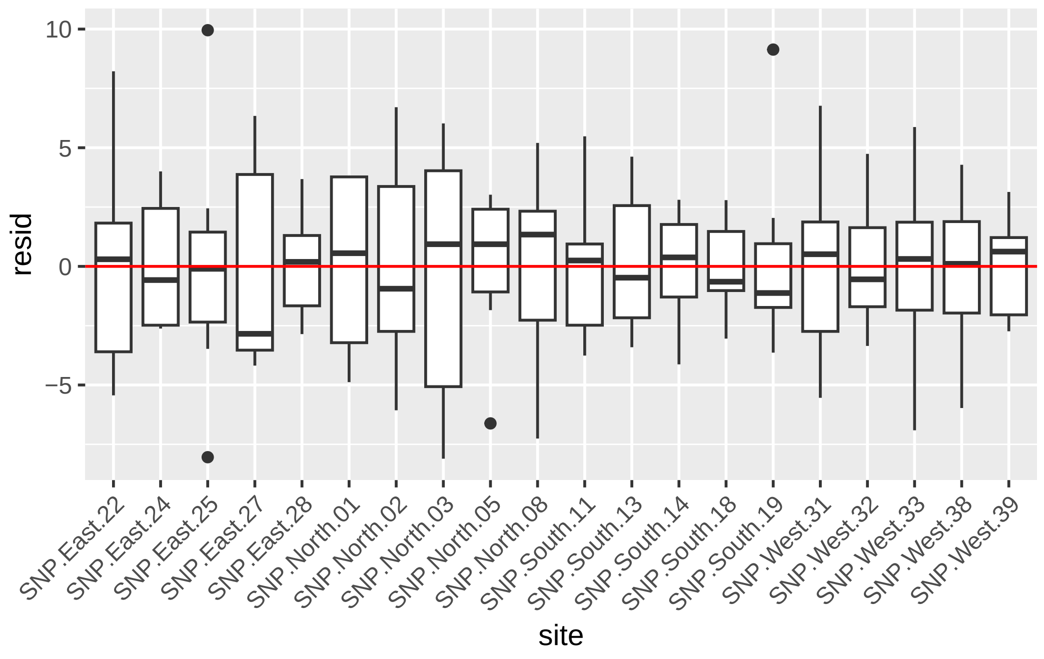
```
m3 <- lm(dbh ~ age * site, data = dat)

augment(m3) %>% ggplot(aes(age, .fitted, col = site)) + geom_line()
```



And again check the residuals, which look even better now.

```
dat %>% add_residuals(m3) %>%
  ggplot(aes(site, resid)) + geom_boxplot() +
  geom_hline(yintercept = 0, col = "red") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



The number of coefficients becomes huge.

```r
length(coef(m3))
```

```
[1] 40
```

```r
nrow(dat)
```

```
[1] 160
```

An alternative approach would be, to fit for each site the simple model `dbh ~ age`.

```r
m4 <- dat %>% nest(data = -site)
```

Note, the use of the `nest()` function here. - `nest(data = -site)` groups the data here by site.

```r
m4[1:2, ]
```

```
# A tibble: 2 x 2
  site         data
  <chr>        <list>
1 SNP.South.18 <tibble [7 x 3]>
2 SNP.North.03 <tibble [9 x 3]>
```

Finally, a new column `model` (also a list-column) is created with a model for each site.

```
m4 <- m4 %>%  mutate(model = map(data, ~ lm(dbh ~ age, data = .)))
```

Each model can now be augmented (entries in the list-column `model`)

```
m4$model[[1]] %>% augment()
```

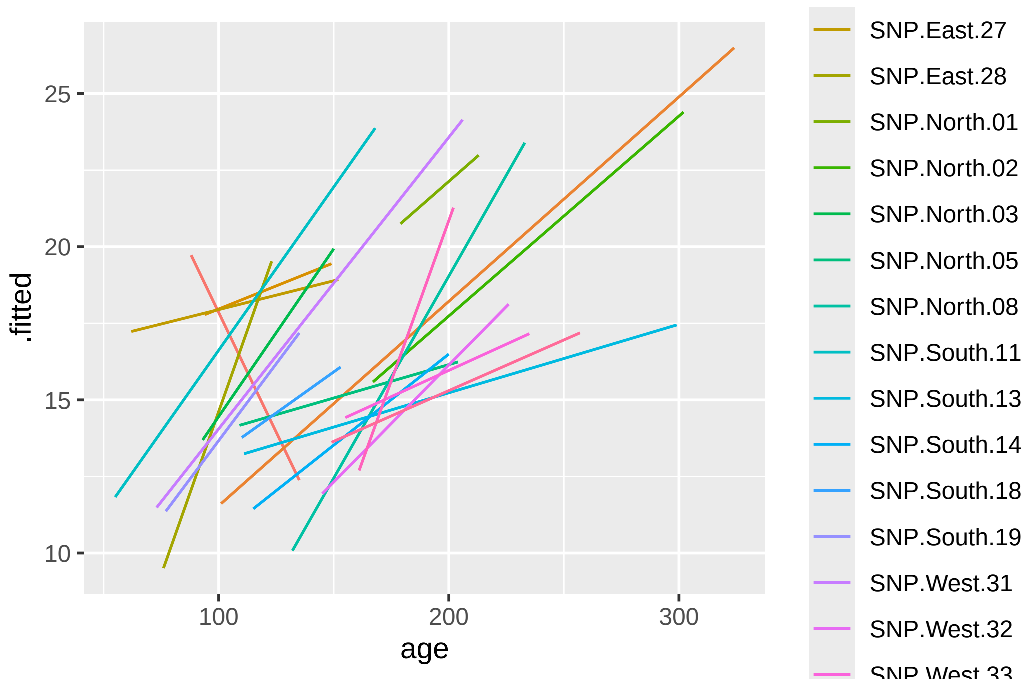
```
# A tibble: 7 x 8
    dbh   age .fitted .resid  .hat .sigma .cooksd .std.resid
  <dbl> <int>   <dbl>  <dbl> <dbl>  <dbl>   <dbl>      <dbl>
1  14.9   144    15.6 -0.647 0.212   2.44  0.0147     -0.331
2  17.3   140    15.4  1.88  0.169   2.24  0.0889      0.936
3  13.0   153    16.1 -3.05  0.387   1.51  0.985      -1.77
4  16.5   141    15.4  1.06  0.178   2.39  0.0305      0.531
5  17.8   134    15.1  2.79  0.143   1.95  0.156       1.37
6  13.1   114    14.0 -0.848 0.398   2.40  0.0815     -0.496
7  12.6   110    13.8 -1.19  0.512   2.31  0.316      -0.776
```

To this, we have to iterate over `model` list-column and finally `unnest()` the list-column `model`.

```
m4_plot_dat <- m4 %>% mutate(model = map(model, augment)) %>%
  unnest(cols = model)
m4_plot_dat[1:2, ]
```

```
# A tibble: 2 x 10
  site       data       dbh   age .fitted .resid  .hat .sigma .cooksd .std.resid
  <chr>      <list>   <dbl> <int>   <dbl>  <dbl> <dbl>  <dbl>   <dbl>      <dbl>
1 SNP.South~ <tibble>  14.9   144    15.6 -0.647 0.212   2.44  0.0147     -0.331
2 SNP.South~ <tibble>  17.3   140    15.4  1.88  0.169   2.24  0.0889      0.936
```

```
m4_plot_dat %>% ggplot(aes(age, .fitted, col = site)) + geom_line()
```



- Results look very similar to model `m3`.
- At some `sites` the relationship between age and DBH is negative.

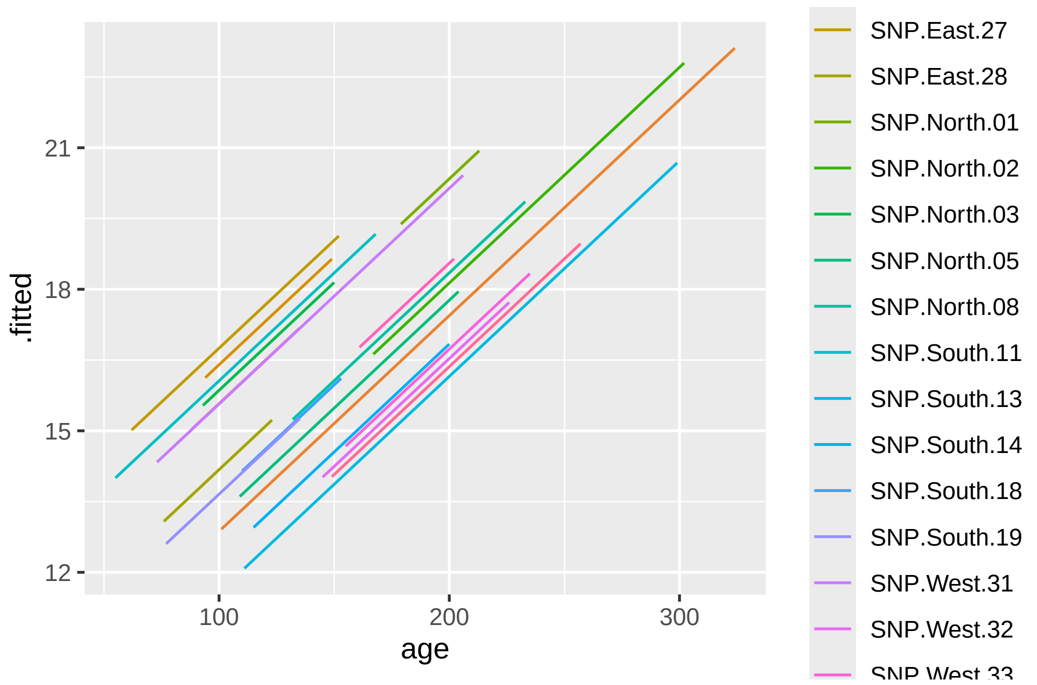
Now lets use a random intercept model.

```
library(lme4)
library(broom.mixed)
```

- The package `lme4` provides functions to fit mixed effects models.
- The package `broom.mixed` is very similar for the package `broom` (the functions `augment()`, `glance()` and `tidy()`), but for mixed models.

```
m5 <- lmer(dbh ~ age + (1 | site), data = dat)
```

```
m5 %>% augment() %>%
  ggplot(aes(age, .fitted, col = site)) + geom_line()
```

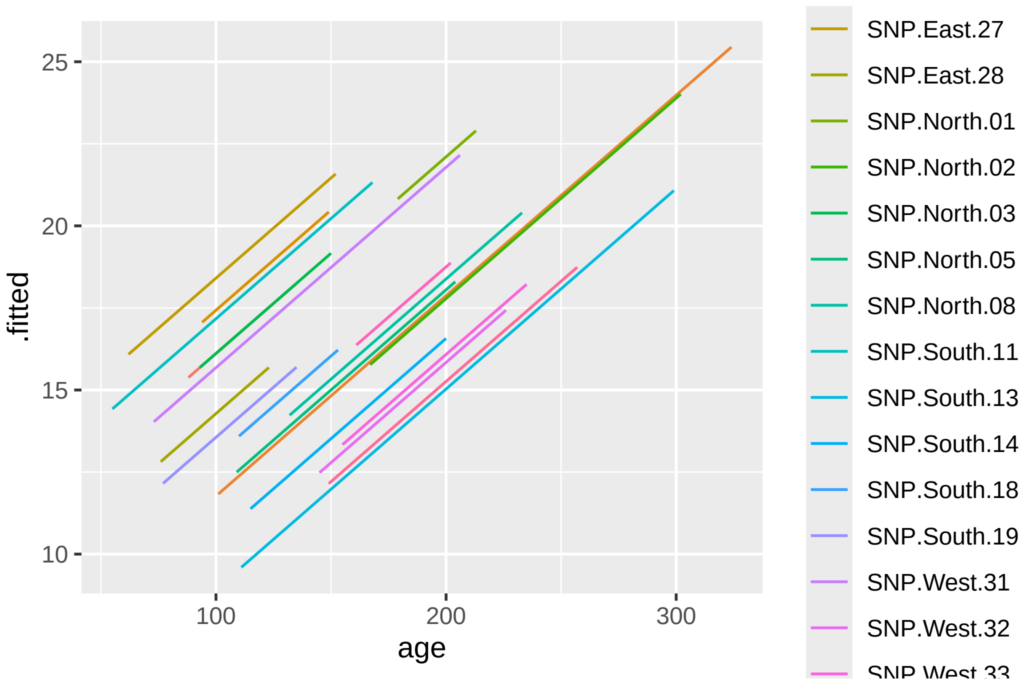


This is very similar to model m2.

```
m2$call
```

```
lm(formula = dbh ~ age + site, data = dat)
```

```
m2 %>% augment() %>%
  ggplot(aes(age, .fitted, col = site)) + geom_line()
```



But compare the number of coefficients between `m2`

```
coef(m2)
```

```
     (Intercept)              age  siteSNP.East.24  siteSNP.East.25
    10.005204952      0.061058802     -4.342966637      1.318461455
  siteSNP.East.27  siteSNP.East.28 siteSNP.North.01 siteSNP.North.02
     2.295838994     -1.830702836     -0.113083117     -4.431706270
siteSNP.North.03 siteSNP.North.05 siteSNP.North.08 siteSNP.South.11
     0.001242395     -4.163689066     -3.833972067      1.059542570
siteSNP.South.13 siteSNP.South.14 siteSNP.South.18 siteSNP.South.19
    -7.187161295     -5.646999929     -3.128496175     -2.553214615
  siteSNP.West.31  siteSNP.West.32  siteSNP.West.33  siteSNP.West.38
    -0.431453859     -6.378068692     -6.133800721     -3.467017925
  siteSNP.West.39
    -6.956053532
```

```
sigma(m2)
```

```
[1] 3.976816
```

and `m5`

```r
fixef(m5)
```

```
(Intercept)         age 
 9.45932393  0.04572627 
```

```r
VarCorr(m5)
```

```
 Groups   Name        Std.Dev.
 site     (Intercept) 2.0533  
 Residual             4.0183  
```

In summary

```r
# m2
length(coef(m2)) + 1
```

```
[1] 22
```

```r
# m5
length(fixef(m5)) + 2
```

```
[1] 4
```

Now let us add a random intercept to the existing model.

```r
m6a <- lmer(dbh ~ age + (age | site), data = dat)
```

```
Warning in checkConv(attr(opt, "derivs"), opt$par, ctrl = control$checkConv, :
Model failed to converge with max|grad| = 1.16634 (tol = 0.002, component 1)
```

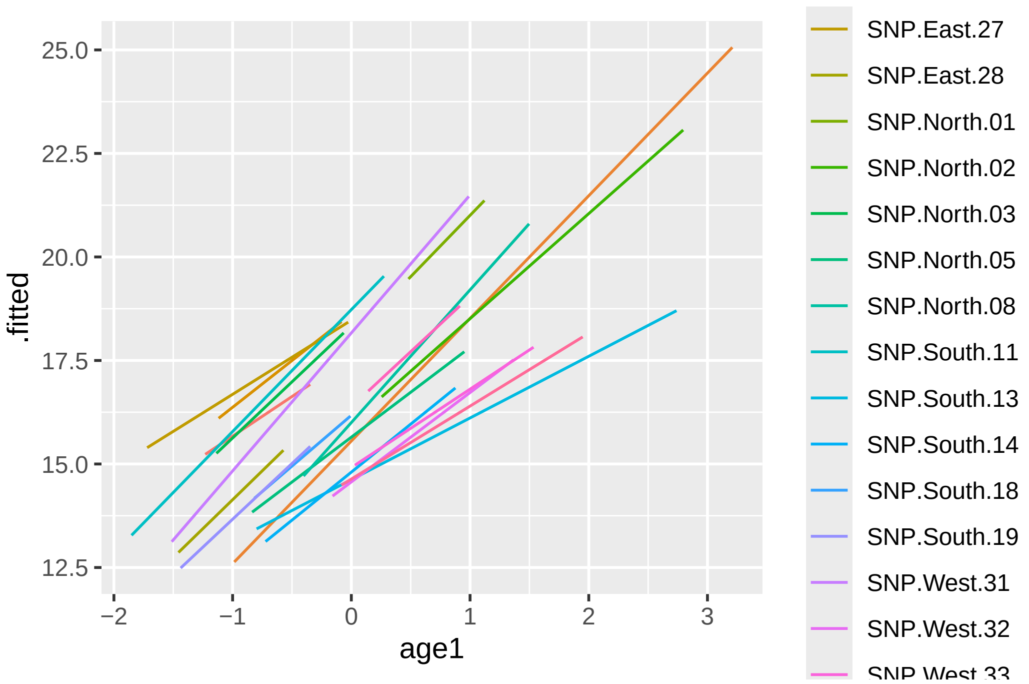
```
Warning in checkConv(attr(opt, "derivs"), opt$par, ctrl = control$checkConv, : Model is near
 - Rescale variables?
```

We get a warning that the optimizer had difficulties to fit the model. The warning message suggest to rescale the covariates (here `age`).

```r
m.age <- mean(dat$age)
sd.age <- sd(dat$age)
dat$age1 <- (dat$age - m.age) / sd.age
m6 <- lmer(dbh ~ age1 + (age1 | site), data = dat)
```

After rescaling the covariates, no more warnings are present.

```r
m6 %>% augment() %>%
  ggplot(aes(age1, .fitted, col = site)) + geom_line()
```



We observe two things:

- Slopes for different sites differ.
- Slopes are more similar then when fitting individual models. This is called **shrinkage to the mean**.

```r
m6
```

```
Linear mixed model fit by REML ['lmerMod']
Formula: dbh ~ age1 + (age1 | site)
   Data: dat
REML criterion at convergence: 917.3578
```

```
Random effects:
 Groups   Name        Std.Dev. Corr
 site     (Intercept) 1.934
          age1        1.059    0.25
 Residual             3.954
Number of obs: 160, groups:  site, 20
Fixed Effects:
(Intercept)         age1
      16.53         2.45
```

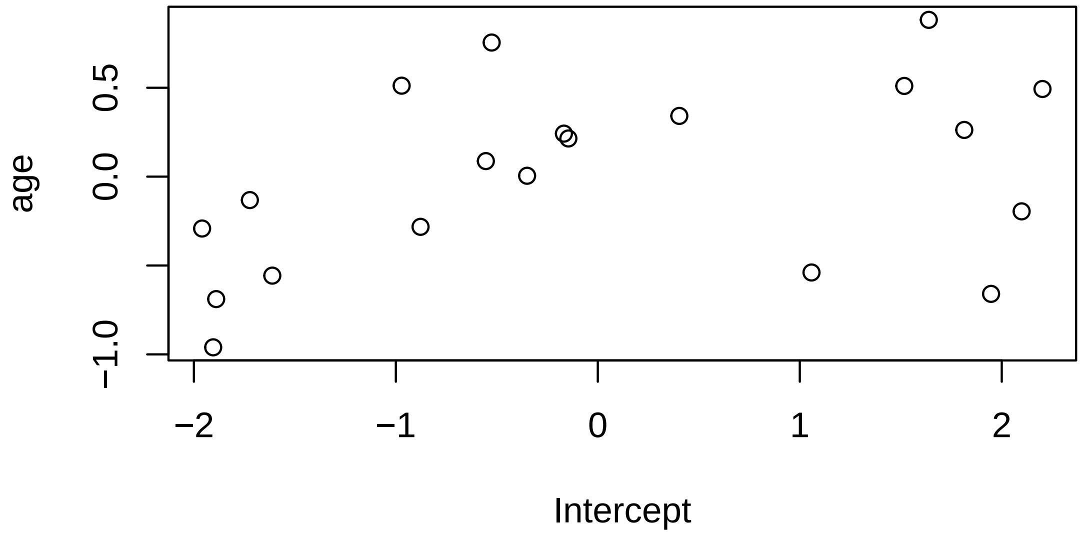
Correlation between intercept and slope:

```r
VarCorr(m6)
```

```
 Groups   Name        Std.Dev. Corr
 site     (Intercept) 1.9338
          age1        1.0594   0.253
 Residual             3.9536
```

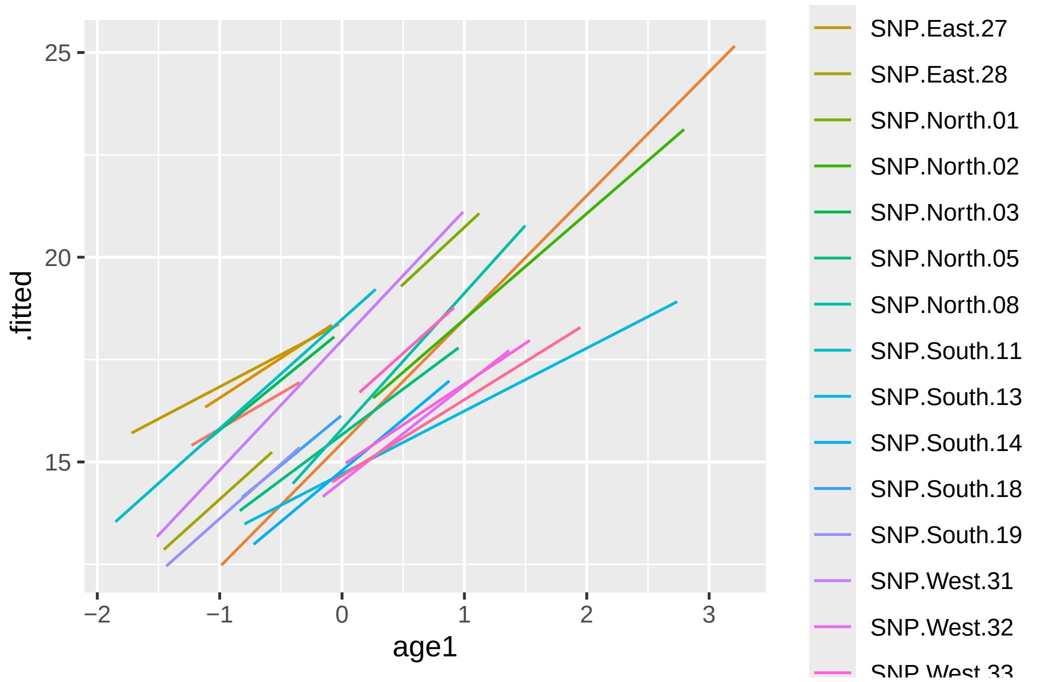
This means that sites with a higher intercept also grow faster.

```r
plot(ranef(m6)$site[, 1], ranef(m6)$site[, 2],
     xlab = "Intercept", ylab = "age")
```



If no correlation between the slope (`age`) and the intercept is needed, the model would look like this:

```r
# Now lets also add a random slope, but no interaction
m7 <- lmer(dbh ~ age1 + (1 | site) + (0 + age1 | site), data = dat)
m7 %>% augment() %>%
  ggplot(aes(age1, .fitted, col = site)) + geom_line()
```



Lets compare the width of confidence intervals:

```r
diff(confint(m1)[2, ])
```

```
    97.5 %
0.02595968
```

```r
diff(confint(m5)[4, ])
```

```
    97.5 %
0.03480329
```

```r
diff(confint(m6)[6, ])
```

```
  97.5 %
2.260624
```

Exercise: Linear Mixed Models

The file `pines/Data1.txt` contains estimates of DBH and age for each tree at different ages. Each tree has an individual id (`core.code`). Perform the following tasks:

1. Plot a growth curve for each tree (in one plot). Where you plot the age on the x-axis and the DBH on the y-axis.
2. Fit the following models:

    a. A model $DBH = \beta_0 + \beta_1 age$ for each individual tree.
    b. A global model $DBH = \beta_0 + \beta_1 age$.
    c. A global model $DBH = \beta_0 + \beta_1 age + \beta_2 id + \beta_3 age \cdot id$.
    d. The same model as in b), but with a random intercept.
    e. The same model as in b), but with a random intercept and random slope.

3. Create a plot with the model type (2b, 2d, 2e) on the x-axis and the estimate (with a confidence interval) on the y-axis. Distinguish between the two terms (intercept and slope) by using different panels. Hint, you may find the function(s) `broom::tidy()` with the argument `conf.int = TRUE`, and `bind_rows` useful.

9.3 Some model strategies

9.3.1 How many data levels are needed?

- Literature suggest at least 5 levels are needed.
- Problems can arrise if data are spread unequally among levels.
- But see also Oberpriller et al. 2021 [https://www.biorxiv.org/content/10.1101/2021.05.03.442487v1].

9.3.2 Choosing random slopes

- Often only random intercepts are used, this can lead to inflated Type 1 error rates (see also Schielzeth and Forstmeier 2009 for details[1]).
- It is recommended to fit the most complex model that the data allow.
- See Bates *et al* 2015[2] for strategies to find the optimal structure of mixed models.

[1]https://academic.oup.com/beheco/article/20/2/416/218997
[2]https://arxiv.org/pdf/1506.04967.pdf

9.3.3 Model complexity

- Try to avoid fitting the most complex model possible.
- Try to avoid interaction beyond order two.
- You should be able to draw interpretations and visualize your model for different predictors.
- Try to stick to $n/k = 10$, where n is the number of data points and k the number of predictors.

9.3.4 Assessing model fit

- For LMMs residuals analyses as with the linear models are possible.
- Similarely to GLMs we, can use randomized quantile residuals to asses model fit for mixed models with the package `DHARMa`.

9.4 GLMMs

- For generalized linear models with repeated measures and/or hierarchical data, GLMMs can be used. The idea is analogue to LMMs.
- Fitting GLMMs in R can be done with the function `glmer()` from the package `lme4`.
- Randomized quantile resiudals can be used to asses model fit.

9.5 Recommended reading

A good place to start is:

- Generalized linear mixed models: a practical guide for ecology and evolution (https://www.sciencedirect.com/science/article/abs/pii/S0169534709000196)

More practical guides:

- A brief introduction to mixed effects modelling and multi-model inference in ecology (https://peerj.com/articles/4794/)
- Perils and pitfalls of mixed-effects regression models in biology (https://peerj.com/articles/9522/)