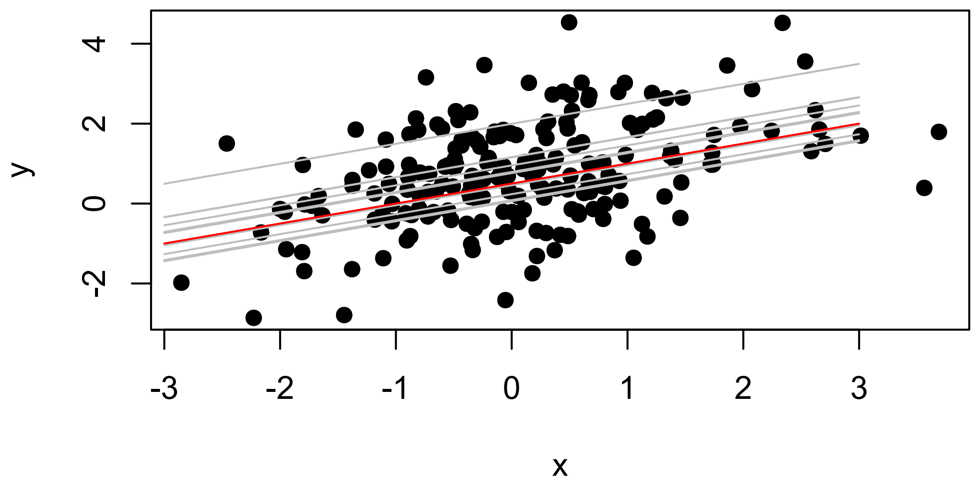
Into the multilevel model: Varying slopes and group level predictors

Varying Intercepts

Let’s start by simulating a varying intercepts model similar to our discussion last week:

$$
\begin{aligned}
y_i &\sim Normal(\mu_i, \sigma_y) \\
\mu_i &= \alpha_{j[i]} + \beta x_i \\
\alpha_j &\sim Normal(\overline{\alpha}, \sigma_\alpha)
\end{aligned}
$$

```r
set.seed(10)
n <- 200 # number of observations
j <- 10 # number of groups
a_bar <- .5 # mean of intercepts
b <- .5 # slope
sigma_a <- .75 # error standard deviation of the intercept
sigma_y <- 1.1 # individual level error standard deviation
group <- sample(1:j, n, replace = T) # assign individuals to groups
a <- rnorm(j, mean = a_bar, sd = sigma_a) # sample j intercepts
x <- rnorm(n) # sample n predictor values

mu <- a[group] + b * x # calculate the conditional means

y <- rnorm(n, mu, sigma_y) # add noise to the prediction.

x_pred <- seq(-3,3,l = 100)
plot(y ~ x, pch = 19)
for(i in 1:j) lines((a[i] + b*x_pred) ~ x_pred, col = "grey")
lines(a_bar + x_pred*b ~ x_pred, col = "red")
```



Let’s fit a varying intercepts model to this data.

```r
library(lme4)
library(arm)
grp <- as.numeric(group)
mod1 <- lmer(y ~ 1 + x + (1|grp))
display(mod1)
```

```
lmer(formula = y ~ 1 + x + (1 | grp))
            coef.est coef.se
(Intercept) 0.75     0.17   
x           0.45     0.07   

Error terms:
 Groups   Name        Std.Dev.
 grp      (Intercept) 0.48    
 Residual             1.08    
---
number of obs: 200, groups: grp, 10
AIC = 626.2, DIC = 607.7
deviance = 613.0
```

So the model has point estimates of $\overline{\alpha} = .75 \pm .34$, $\beta = .45 \pm .14$, $\sigma_y = 1.08$, $\sigma_\alpha = .48$. This all is pretty close to our simulated values.

Intro to varying intercepts and varying slopes

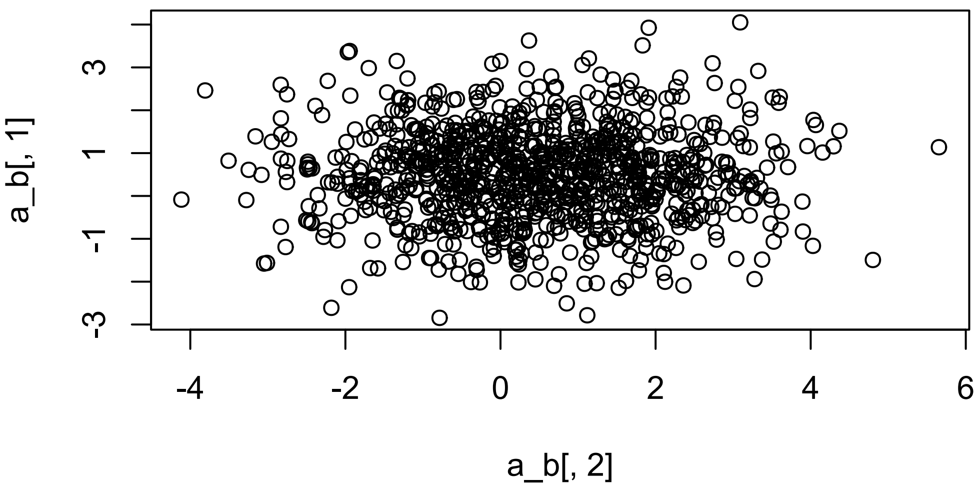
It would be nice if we could just add an independent model: $\beta_j \sim Normal\left(\overline{\beta}, \sigma_\beta\right)$. Unfortunately, we are not so lucky. We could do this if we expect α_j and β_j to be uncorrelated, but they will often be correlated, so we need to account for this. In this case, we will use the multi-variate normal distribution to model the coefficients. Deep breaths, here we go:

$$ \begin{align*} y_i &\sim Normal(\mu_i, \sigma_y)\\ \mu_i &= \alpha_{j[i]} + \beta_{j[i]} x_i\\ \begin{bmatrix} \alpha_j \\ \beta_j \end{bmatrix} &\sim MVN(\begin{bmatrix} \overline{\alpha}\\\overline{\beta}\end{bmatrix}, \Sigma)\\ \Sigma &= \begin{bmatrix} \sigma_{\alpha}^2 \space \space \rho\sigma_{\alpha}\sigma_{\beta} \\ \rho\sigma_{\alpha}\sigma_{\beta} \space \space \sigma_{\beta}^2\end{bmatrix} \end{align*} $$

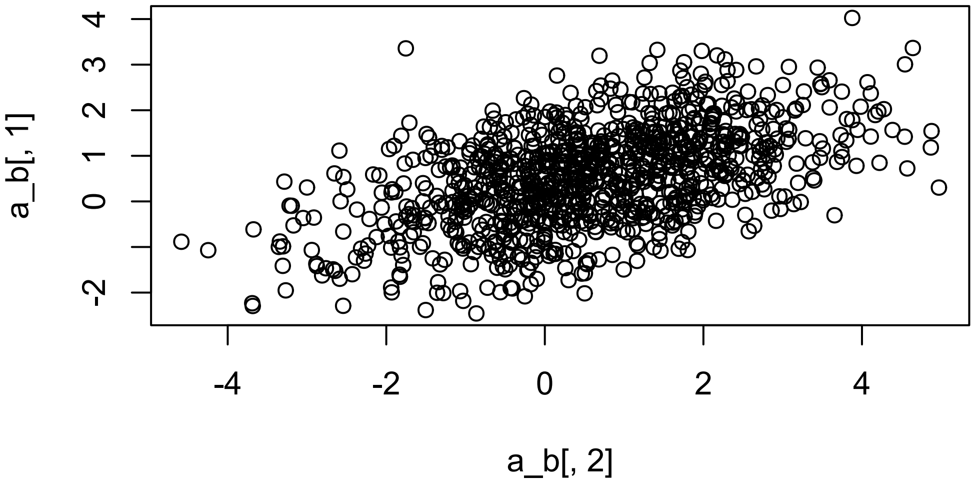
Always be simulatin'!. The big new thing here is that now the coefficients are allowed to co-vary. The strength of this covariance is controlled by ρ, or the correlation. Let's simulate a bunch of α's and β's with different values of ρ. Because correlation is easier to understand than covariance and standard deviations are easier to understand than variances, we will break down the covariance matrix above into: $\Sigma = SRS$, where S is a diagonal matrix with the standard deviations of the coefficients on the diagonal and 0's everywhere else, and R is a correlation matrix with 1's on the diagonal and correlations elsewhere:

$$ \Sigma = \begin{bmatrix} \sigma_{\alpha} \space 0 \\ 0 \space \sigma_{\beta} \end{bmatrix}\begin{bmatrix} 1 \space \rho\\ \rho \space 1 \end{bmatrix}\begin{bmatrix} \sigma_{\alpha} \space 0 \\ 0 \space \sigma_{\beta} \end{bmatrix} $$

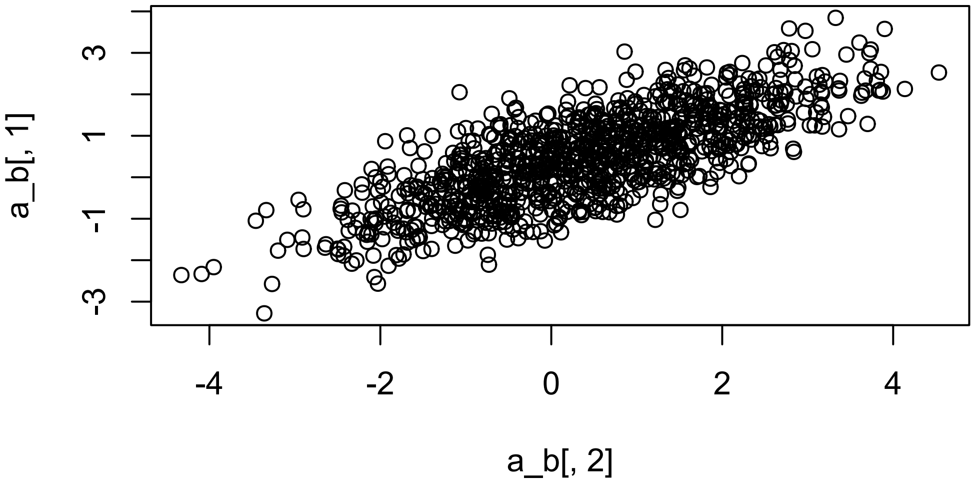
```r
coef_sims <- function(rho){
  n <- 1000
  a_bar <- .5
  b_bar <- .5
  sig_a <- 1.1
  sig_b <- 1.5
  rho <- rho
  S <- diag(c(sig_a, sig_b))
  R <- matrix(c(1, rho, rho, 1), ncol = 2)

  a_b <- MASS::mvrnorm(n, c(a_bar, b_bar), S %*% R %*% S)
  plot(a_b[,1] ~ a_b[,2])
}
coef_sims(0)
```



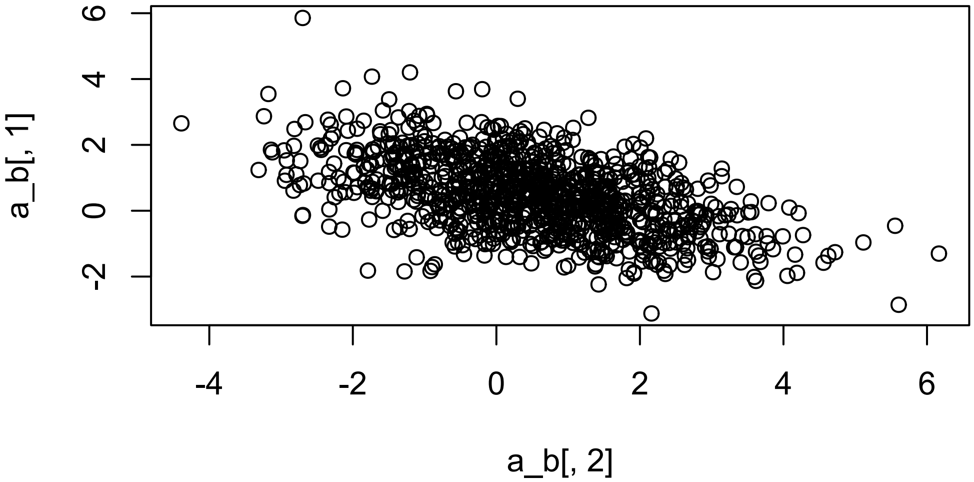
```
coef_sims(.5)
```



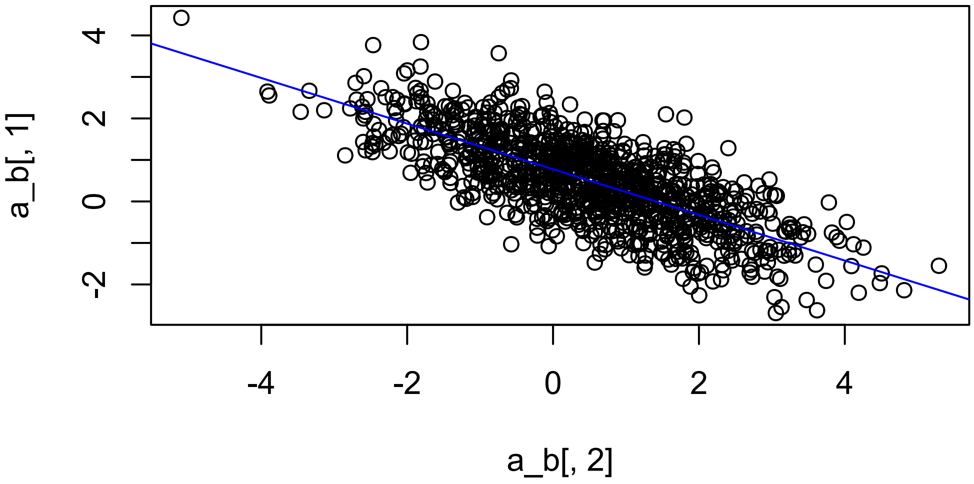
```
coef_sims(.75)
```



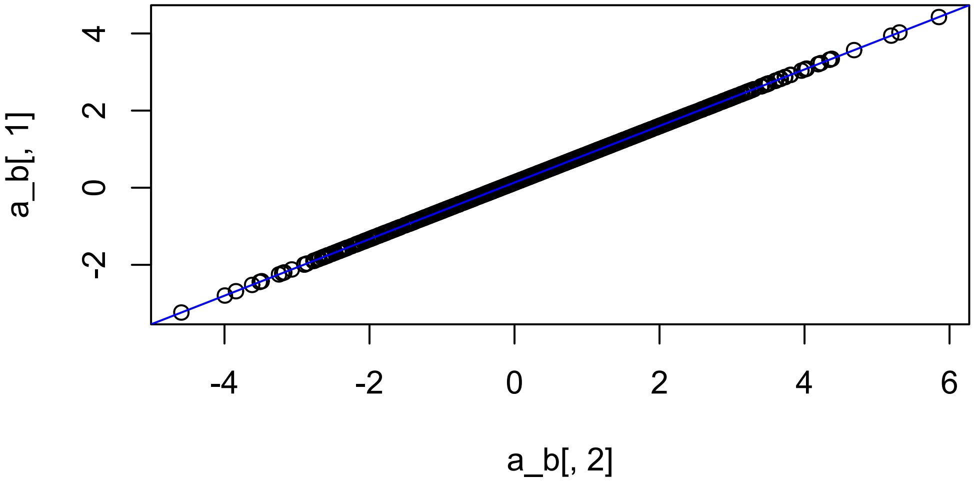
```r
coef_sims(-.5)
```



```r
coef_sims(-.75)
abline(a = .5 - .5*-.75*1.1/1.5, b = -.75*1.1*1.5/(1.5^2), col = "blue")
```

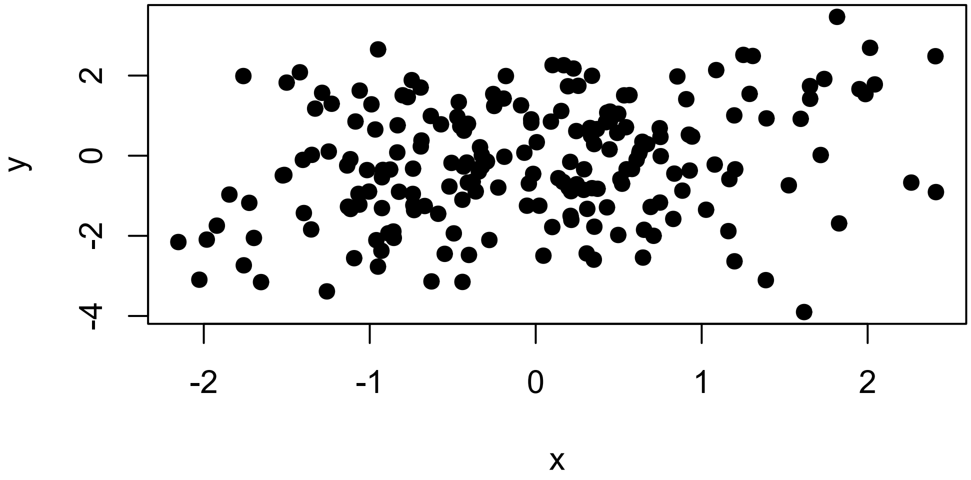


```
coef_sims(1)
abline(a = .5 - .5*1.1/1.5, b = 1*1.1*1.5/(1.5^2), col = "blue")
```



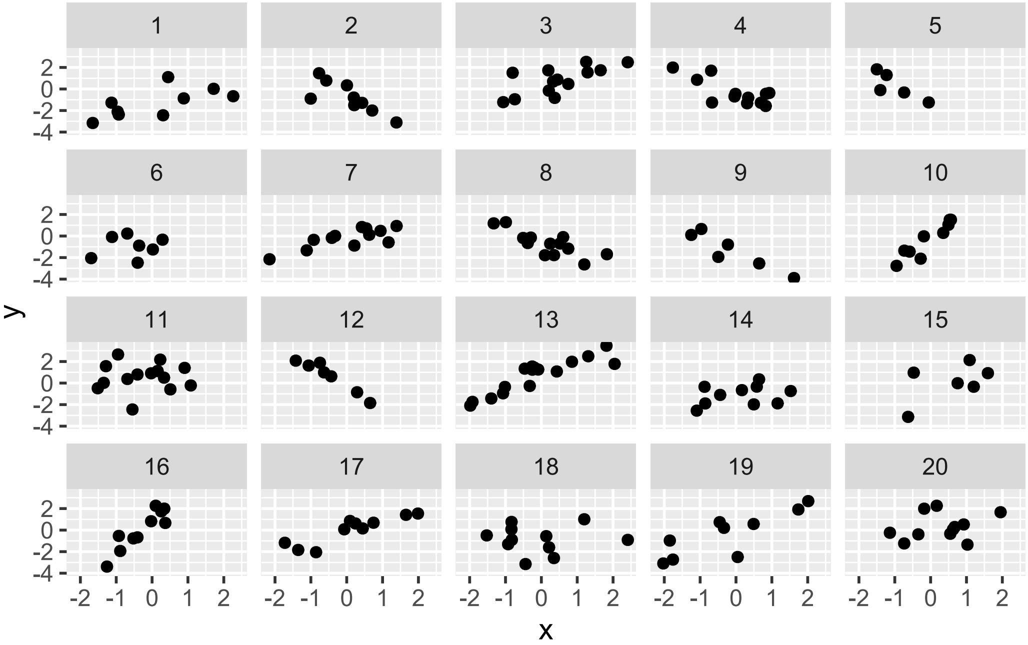
So let's go ahead and simulate a varying slopes and varying intercepts data generating process and fit it.

```r
n <- 200
a_bar <- 0
b_bar <- .5
sig_a <- .75
sig_b <- 1.1
sig_y <- 1
rho <- .5
S <- diag(c(sig_a, sig_b))
R <- matrix(c(1, rho, rho, 1), ncol = 2)

a_b <- MASS::mvrnorm(20, c(a_bar, b_bar), S %*% R %*% S)

group <- sample(1:20, n, replace = T)

x <- rnorm(n)

mu <- a_b[group, 1] + a_b[group, 2] * x

y <- rnorm(n, mu, sig_y)

plot(y ~ x, pch = 19)
```



Kinda hard to see what's going on, let's plot all the groups.

```r
library(tidyverse)
data.frame(y, x, group) %>%
ggplot(aes(x = x, y = y)) +
  geom_point() +
  facet_wrap(group ~ .)
```



Now let's fit the model with `lmer`.

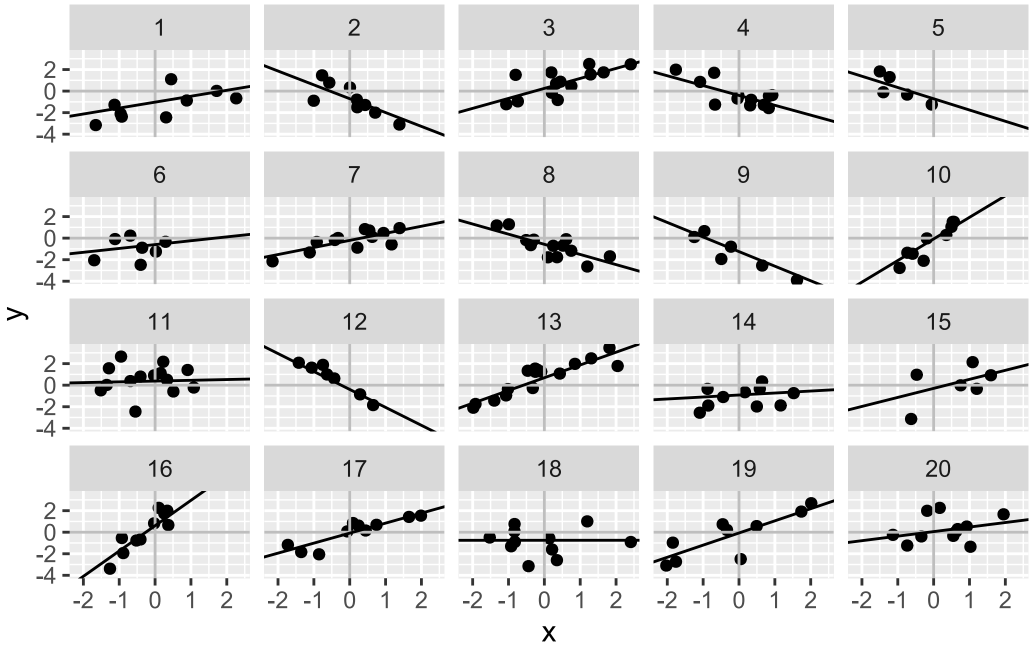
```
grp <- factor(group)
mod2 <- lmer(y ~ 1 + x + (1 + x|grp))
display(mod2)
```

```
lmer(formula = y ~ 1 + x + (1 + x | grp))
            coef.est coef.se
(Intercept) -0.30     0.16
x            0.22     0.28

Error terms:
 Groups   Name        Std.Dev. Corr
 grp      (Intercept) 0.61
          x           1.18     0.56
 Residual             0.97
---
number of obs: 200, groups: grp, 20
AIC = 645.2, DIC = 627.4
deviance = 630.3
```

```
fits <- data.frame(group = 1:20, a = coef(mod2)$grp[,1], b = coef(mod2)$grp[,2])
```

Let’s plot the mean fits on the data

```
data.frame(y, x, group) %>%
  ggplot(aes(x = x, y = y)) +
  geom_point() +
  facet_wrap(group ~ .) +
  geom_vline(xintercept = 0, color = "grey") +
  geom_hline(yintercept = 0, color = "grey") +
  geom_abline(data = fits, aes(slope = b, intercept = a))
```



Adding Group Level Predictors

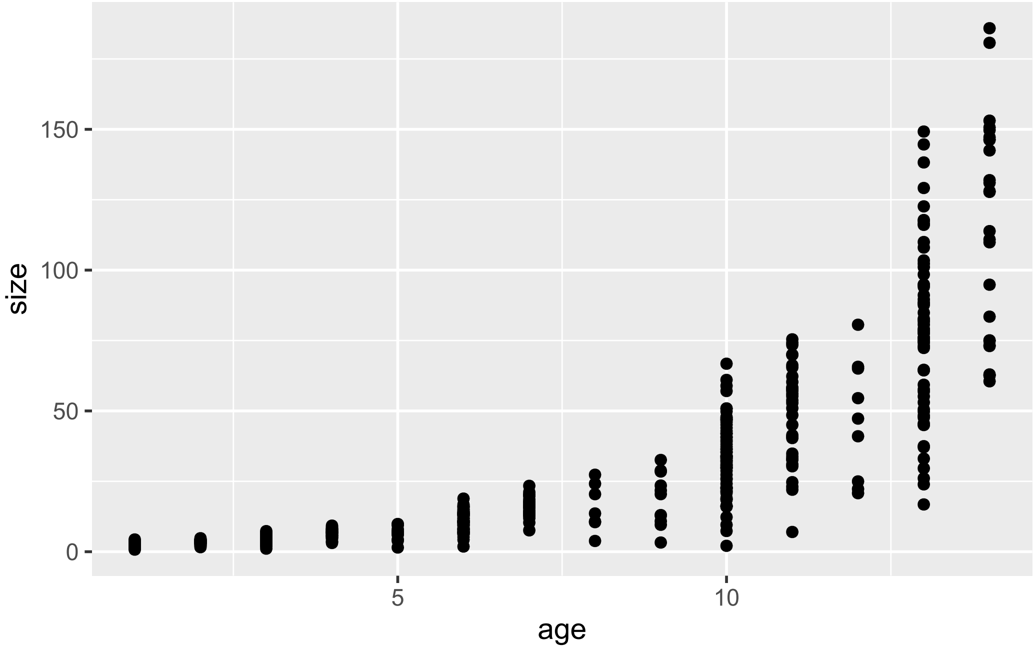
Let’s load in the plant growth data to play around with some real data. Let’s plot it to see what it looks like. There are many things that we might be interested about in this data set. Let’s start off by trying to estimate growth rates. In this case, growth rate will be the regression coefficient on age.

```
library(here)
growth_dat <- read.csv(here("data/week_10/plant_growth.csv"))
head(growth_dat)
```

```
   id   seed_size   size age
1 1_1  0.04398394  1.915   1
```

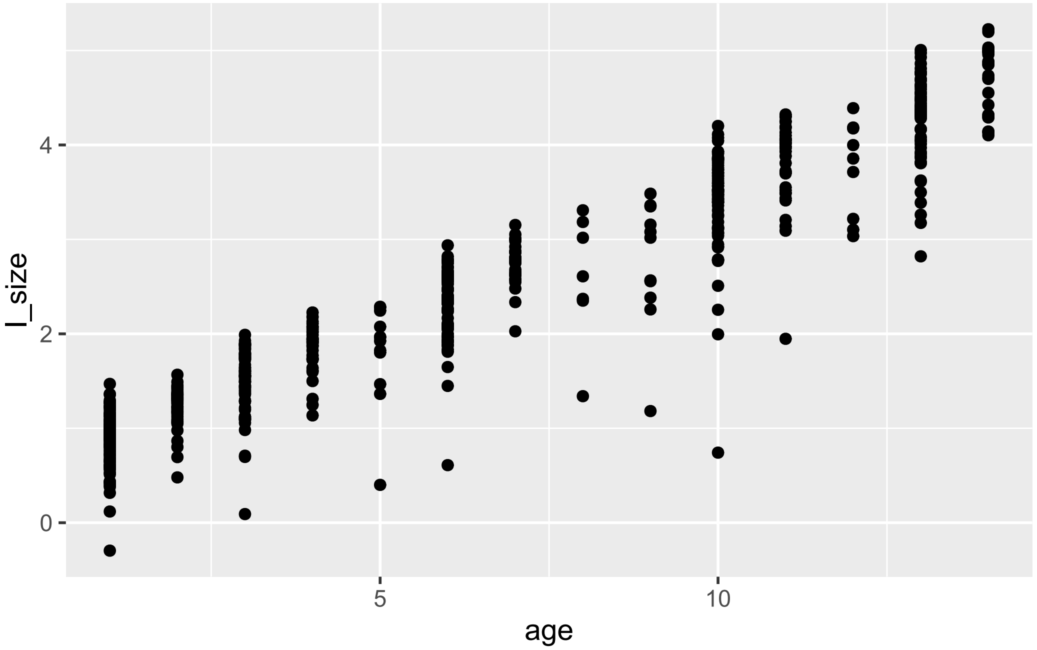
```
2 1_1  0.04398394  4.016   3
3 1_1  0.04398394  7.821   6
4 1_2 -0.01040480  3.133   1
5 1_2 -0.01040480  5.999   3
6 1_2 -0.01040480 15.602   6
```

```
growth_dat %>%
  ggplot(aes(x = age, y = size)) +
  geom_point()
```



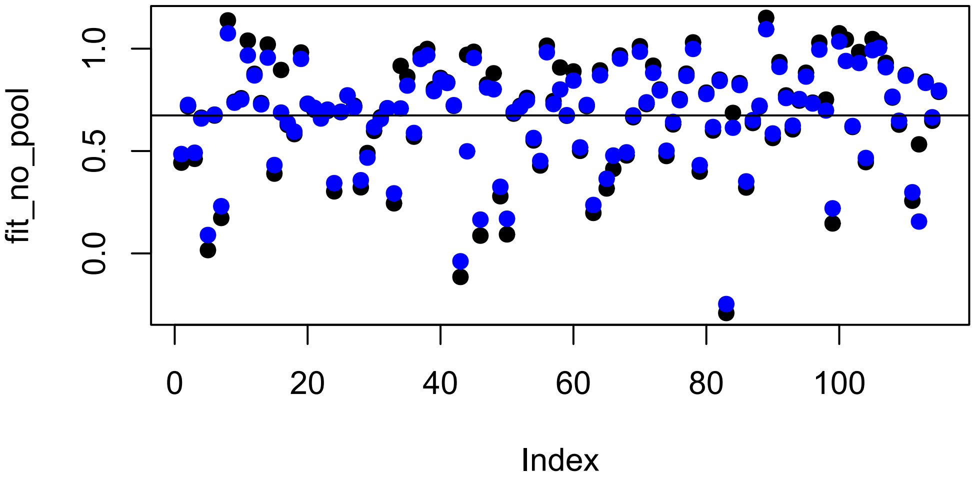
Looks like exponential growth. We could probably use a log-normal GLMM, but let's transform it for fun.

```
growth_dat$l_size <- log(growth_dat$size)

growth_dat %>%
  ggplot(aes(x = age, y = l_size)) +
  geom_point()
```



It looks nice and linear now! So let's do some modeling. It is probably a bad idea to completely pool this data since multiple measures were taken for each individual. So we will use a multilevel model (partial pooling) to get at the variation between individuals. Let's fit no-pool model as well and compare the intercepts

```r
growth_dat$id <- factor(growth_dat$id)
mod_p_pool <- lmer(l_size ~ 1 + age + (1 + age|id), data = growth_dat)
mod_no_pool <- lm(l_size ~ -1 + age + id + id*age, data = growth_dat)
fit_no_pool <- as.numeric(coef(mod_no_pool)[2:116])
fit_p_pool <- coef(mod_p_pool)$id[,1]
plot(fit_no_pool, pch = 19)
points(fit_p_pool, col = "blue", ylab = "Intercept", pch = 19)
abline(h = fixef(mod_p_pool)[1])
```



Notice how, in general, the further the group intercepts get from the mean, the more the partially pooled estimate is shrunk toward the mean!

Adding a group level predictor to the intercept

So this is doing what we want, shrinking the intercepts toward the group mean to help overfitting and use as much information as we can reasonably use. But there is a variable we haven't considered yet. Seed size is likely a major factor in the size of a seedling at time 0 (i.e. the intercept). Let's try a varying intercepts model where the intercept is a function of seed size. We want a model that looks like:

$$\begin{aligned} size_i &\sim Normal(\mu_i, \sigma_{size}) \\ \mu_i &= \alpha_{j[i]} + rate \times age_i \\ \alpha_j &\sim Normal\left(\overline{\alpha} + \gamma_\alpha \times seedsize_j, \sigma_\alpha\right) \end{aligned}$$

How do we get this into `lmer` though? Well, if we remember that this form of the model of α_j is just a re-writing of the form:

$$\begin{aligned} \alpha_j &= \overline{\alpha} + \gamma_\alpha \times seedsize_j + \eta_j \\ \eta_j &\sim Nomral(0, \sigma_\alpha) \end{aligned}$$

We can plug this definition of α_j into the level one equation:

$$\mu_i = \left(\overline{\alpha} + \gamma_\alpha \times seedsize_j + \eta_j\right) + rate \times age_i$$

So, in `lmer` it becomes: `lmer(l_size ~ 1 + seed_size + age + (1|id))`. Let's try it out.

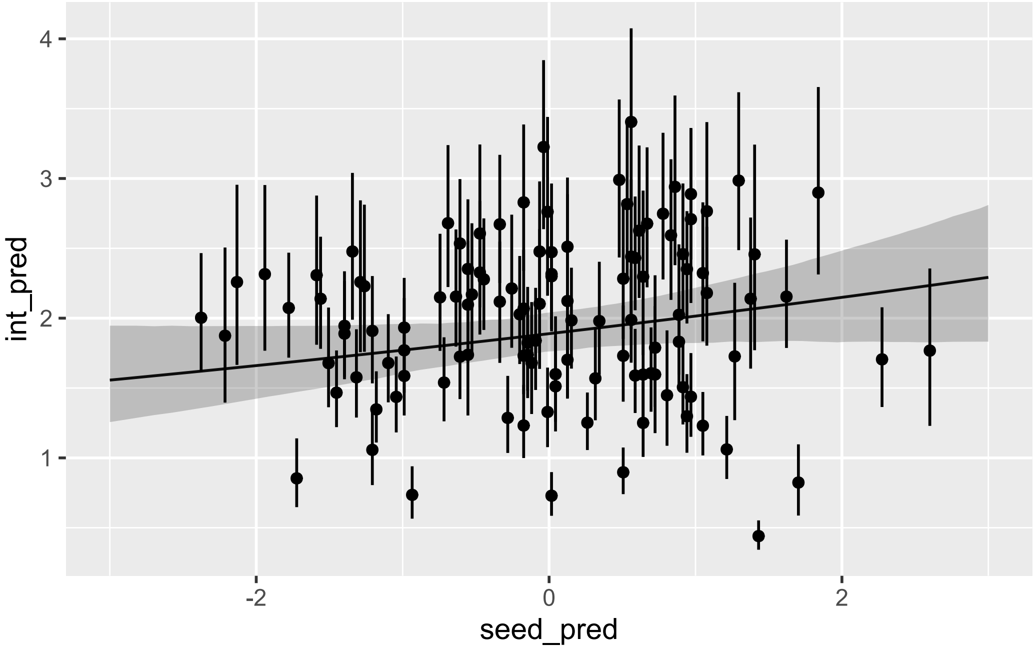
```
mod_seed_int <- lmer(l_size ~ 1 + seed_size + age + (1|id), data = growth_dat)
display(mod_seed_int)
```

```
lmer(formula = l_size ~ 1 + seed_size + age + (1 | id), data = growth_dat)
            coef.est coef.se
(Intercept) 0.63     0.04
seed_size   0.07     0.03
age         0.28     0.00

Error terms:
 Groups   Name        Std.Dev.
 id       (Intercept) 0.36
 Residual             0.22
---
number of obs: 562, groups: id, 115
AIC = 203.8, DIC = 153.2
deviance = 173.5
```

Let’s plot the model for the group level intercepts! This might take some **wrangling**:

```
seed_size <- unique(growth_dat[,c("id", "seed_size")])
sims <- sim(mod_seed_int, 1000)
rans <- t(sims@ranef$id[,,1])
ints <- matrix(nrow = 115, ncol = 1000)
for(i in 1:1000){
    ints[,i] <- exp(rans[,i] + sims@fixef[i,1] + seed_size$seed_size *
sims@fixef[i,2])
}

ints <- data.frame(ints)
z <- data.frame(mu = apply(ints, 1, mean), upr = apply(ints, 1, quantile, .975),
lwr = apply(ints, 1, quantile, .025), seed_size = seed_size$seed_size, id =
seed_size$id)

seed_pred <- seq(-3,3,l = 100)
int_pred <- int_upr <- int_lwr <- c()
for(i in 1:100){
  temp <- sims@fixef[,1] + sims@fixef[,2] * seed_pred[i]
  int_pred[i] <- exp(mean(temp))
  int_upr[i] <- exp(quantile(temp, .975))
  int_lwr[i] <- exp(quantile(temp, .025))
}

data.frame(seed_pred, int_pred, int_upr, int_lwr) %>%
  ggplot(aes(x = seed_pred, y = int_pred)) +
  geom_line() +
  geom_ribbon(aes(x = seed_pred, ymax = int_upr, ymin = int_lwr), alpha = .25)
```

```
+
  geom_point(data = z, aes(x = seed_size, y = mu)) +
  geom_errorbar(data = z, aes(x = seed_size, ymax = upr, ymin = lwr), inherit.aes
= F,
                width = 0)
```



Adding group level predictors to the slope: or, group level predictors as interactions. How to add group level predictors to the slope using `lmer` is not intuitive–to me at least. But I think it is helpful to view adding these predictors as creating an interaction between the group level predictor, and the individual level predictor. This helps me think about statistical interactions. I.E. the "main effect" is the value of the slope on the individual level predictor when the group level predictor = 0; the slope on the interaction adjusts the slope for each group. Let's build this up.

$$y_i \sim Normal(\mu_i, \sigma_y)$$
$$\mu_i = \alpha_{j[i]} + \beta_{j[i]} x_i$$
$$\begin{bmatrix} \alpha_j \\ \beta_j \end{bmatrix} = MVNormal\left(\begin{bmatrix} \overline{\alpha} + \gamma_\alpha u_j \\ \overline{\beta} + \gamma_\beta u_j \end{bmatrix}, \Sigma \right)$$

So, now we have our individual level predictor, x and our group level predictor, u. To see how we can use `lmer` to do this, we can stick our linear models for the intercept and slope into the single level model:

$$\begin{aligned}\mu_i &= \alpha_{j[i]} + \beta_{j[i]}x_i \\ &= \left(\overline{\alpha} + \gamma_\alpha u_{i[j]}\right) + \left(\overline{\beta} + \gamma_\beta u_{j[i]}\right)x_i \\ &= \overline{\alpha} + \gamma_\alpha u_{j[i]} + \overline{\beta}x_i + \gamma_\beta u_{j[i]}x_i\end{aligned}$$

This gives us the formula we can use: `lmer(y ~ 1 + u + x + u*x + (1 + x|group))`. Pretty cool! Let's fit this model:

```
mod_var_ints_slopes <- lmer(l_size ~ 1 + seed_size + age + seed_size*age + (1 +
age|id), data = growth_dat)
display(mod_var_ints_slopes)
```

```
lmer(formula = l_size ~ 1 + seed_size + age + seed_size * age +
    (1 + age | id), data = growth_dat)
              coef.est coef.se
(Intercept)   0.67     0.02
seed_size     0.07     0.02
age           0.27     0.00
seed_size:age 0.00     0.00

Error terms:
 Groups   Name        Std.Dev. Corr
 id       (Intercept) 0.25
          age         0.05     0.07
 Residual             0.11
---
number of obs: 562, groups: id, 115
AIC = -150, DIC = -223.7
deviance = -194.9
```

Let's sort out this output in terms of our varying intercepts, varying slopes model with predictors. First a reminder of the model: $$ \begin{align*} y_i &\sim Normal(\mu_i, \sigma_y)\\ \mu_i &= \alpha_{j[i]} + \beta_{j[i]} x_i\\ \begin{bmatrix} \alpha_j \\ \beta_j \end{bmatrix} &\sim MVN(\begin{bmatrix} \overline{\alpha} + \gamma_{\alpha} u_j \\ \overline{\beta} + \gamma_{\beta} u_j\end{bmatrix}, \Sigma)\\ \Sigma &= \begin{bmatrix} \sigma_{\alpha}^2 \space \space \rho\sigma_{\alpha}\sigma_{\beta} \\ \rho\sigma_{\alpha}\sigma_{\beta} \space \space \sigma_{\beta}^2\end{bmatrix} \end{align*} $$ Let's fill it in with our model output:

$$ \begin{align*} \log(size_i) &\sim Normal(\space\mu_i,\space .11)\\ \mu_i &= \alpha_{j[i]} + \beta_{j[i]} x_i\\ \begin{bmatrix} \alpha_j \\ \beta_j \end{bmatrix} &\sim MVN(\begin{bmatrix} .67 + .07 \times seedsize_j \\ .27 + 0\times seedsize_j\end{bmatrix}, \Sigma)\\ \Sigma &= \begin{bmatrix} .25^2 \space \space \space .07\times .25\times .05 \\ .07\times .25\times .05 \space \space \space .05^2\end{bmatrix} \end{align*} $$

And that's our multilevel model. We have slopes and intercepts that vary by individual plant and we have group level predictors that let us potentially get better resolution on what drives variation between groups, which is something that we can't do without partial pooling!