Lab 7

Bootstrapping Linear Models

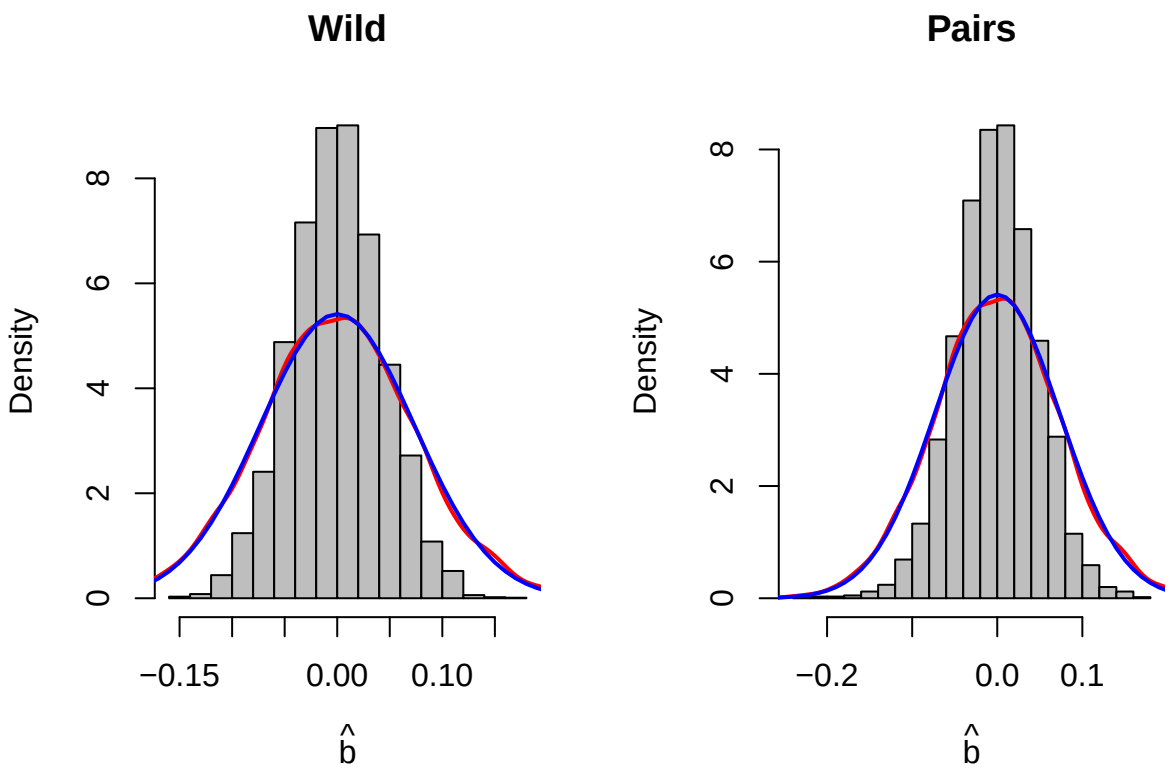
To show that bootstrapping regression estimates is a reasonable thing to do, we will compare the distribution derived from the bootstrap to the actual sampling distribution, which we can simulate. In practice, we cannot simulate the true sampling distribution, but can only get the bootstrap distribution.

Settings where the model based sampling distribution holds

We first show that bootstrapping linear models can indeed be a reasonable thing to do when the data satisfies all our assumptions. In particular, we can see that as n grows larger, the bootstrap distribution grows more and more similar to the distribution of the test statistic we would get from simulations as well as the model based sampling distribution. We can see that the empirical bootstrap and wild bootstrap both work reasonably well. The simulated sampling distribution should be more or less the same each time you run the code below. However, the bootstrap distribution depends on the observed data, it will change each time you run the code. Run it several times to see how the bootstrap distribution changes from run to run. Then, try increasing $n = 20, 100, 400$. In the plots, the bootstrap distribution is shown with the histogram, the true sampling distribution is shown in red, and the model based sampling distribution is shown in blue.

```
## Homoscedastic linear model
# Fixed Covariates
# Using Wild Bootstrap

sim.size <- 5000
b <- 1
n <- 20
# Fixed Design
x <- seq(0, 10, by = 10/(n-1))
y <- 1 + b * x + rnorm(n)

## We are interested in the estimated coefficient
mod <- lm(y~ x)
observed.stat <- summary(mod)$coef[2, 1]


### Approximate sampling distribution using the Wild and empirical bootstrap
rec.boot <- matrix(0, sim.size, 2)
for(i in 1:sim.size){

  # Wild Bootstrap
  y.boot.wild <- mod$fitted + mod$residuals * rnorm(n)

  # Calculate the statistic for the bootstrap sample
  rec.boot[i, 1] <- summary(lm(y.boot.wild ~ x))$coeff[2,1] - observed.stat


  # Pairs Bootstrap
  ind <- sample(n, replace = T)
```

```r
  x.boot.emp <- x[ind]
  y.boot.emp <- y[ind]

  # Calculate the statistic for the bootstrap sample
  rec.boot[i, 2] <- summary(lm(y.boot.emp ~ x.boot.emp))$coeff[2,1] - observed.stat


}



rec.sim <- rep(0, sim.size)
for(i in 1:sim.size){

  ## Don't re-draw X
  x.sim <- x
  y.sim <- 1 + b * x.sim + rnorm(n)
  rec.sim[i] <- summary(lm(y.sim ~ x.sim))$coeff[2,1] - b

}


par(mfrow = c(1, 2))
### Bootstrapped sampling distribution in gray histogram
hist(rec.boot[,1], freq = F, col = "gray", breaks = 15, main = "Wild", xlab = expression(hat(b)))
### Simulated Sampling Distribution in red
lines(density(rec.sim), col = "red", lwd = 2)
### Model Based Sampling Distribution in Blue
lines(seq(-5, 5, by = .01), dnorm(seq(-5, 5, by = .01), mean = 0, sd = sqrt(1 / sum((x - mean(x))^2))),


### Bootstrapped sampling distribution in gray histogram
hist(rec.boot[,2], freq = F, col = "gray", breaks = 15, main = "Pairs", xlab =  expression(hat(b)))
### Simulated Sampling Distribution in red
lines(density(rec.sim), col = "red", lwd = 2)
### Model Based Sampling Distribution in Blue
lines(seq(-5, 5, by = .01), dnorm(seq(-5, 5, by = .01), mean = 0, sd = sqrt(1 / sum((x - mean(x))^2))),
```

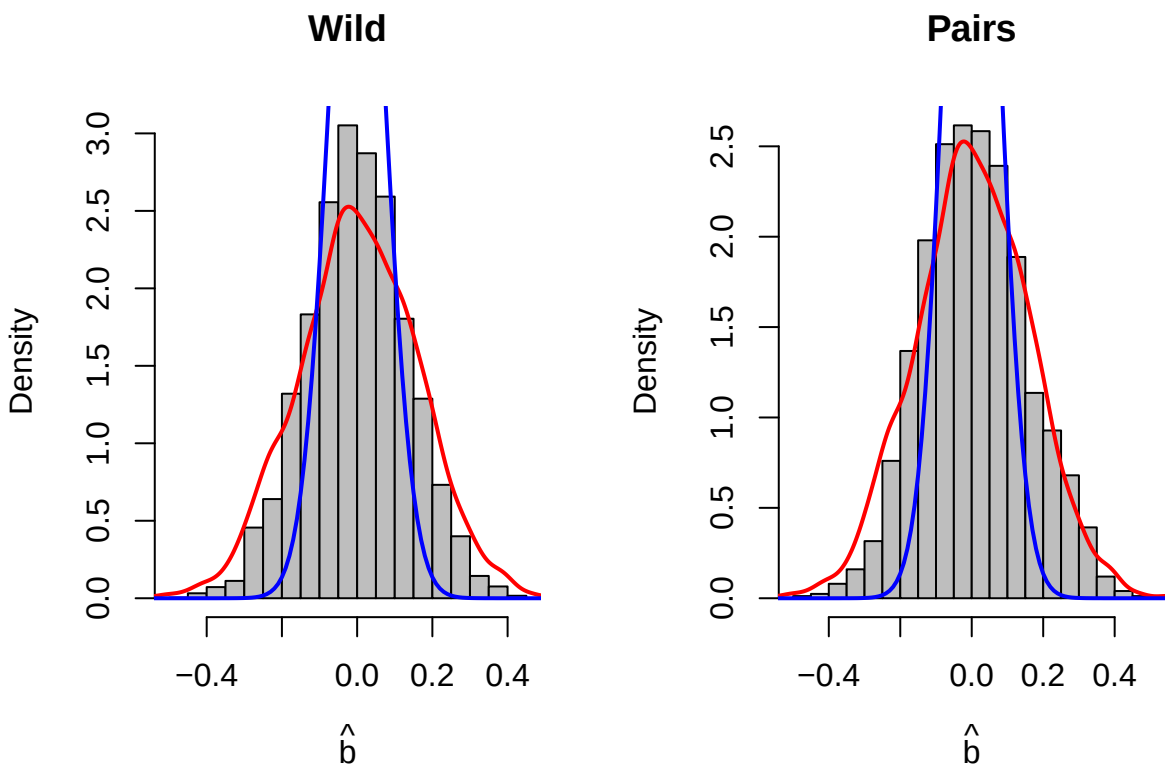


Settings where the model based sampling distribution does not hold

In the settings above, all our assumptions hold, so the model based sampling distribution also holds. However, when the homoscedasticity assumption does not hold, the model based sampling distribution is no longer correct. As we will see though, the bootstrap distribution can still approximate the true sampling distribution (shown in red) well. When n is small, that the histogram (bootstrap) may not be so close to the red (true sampling distribution). However, as n increases, the histogram (bootstrap) generally agrees with the red (true sampling distribution), but the the blue (model based) does not. Try this with multiple times with $n = 20, 100, 400$.

```
## Heteroscedastic linear model
# Fixed Covariates
# Using Wild Bootstrap

sim.size <- 5000
b <- 1
n <- 20
# Fixed Design
x <- seq(0, 10, by = 10/(n-1))
y <- 1 + b * x + rnorm(n, sd = x / 3)

## We are interested in the estimated coefficient
mod <- lm(y~ x)
observed.stat <- summary(mod)$coef[2, 1]


### Approximate sampling distribution using the Wild and empirical bootstrap
rec.boot <- matrix(0, sim.size, 2)
for(i in 1:sim.size){
```

```r
  # Wild Bootstrap
  y.boot.wild <- mod$fitted + mod$residuals * rnorm(n)

  # Calculate the statistic for the bootstrap sample
  rec.boot[i, 1] <- summary(lm(y.boot.wild ~ x))$coeff[2,1] - observed.stat


  # Pairs Bootstrap
  ind <- sample(n, replace = T)
  x.boot.emp <- x[ind]
  y.boot.emp <- y[ind]

  # Calculate the statistic for the bootstrap sample
  rec.boot[i, 2] <- summary(lm(y.boot.emp ~ x.boot.emp))$coeff[2,1] - observed.stat


}



rec.sim <- rep(0, sim.size)
for(i in 1:sim.size){

  # don't redraw X
  x.sim <- x
  y.sim <- 1 + b * x.sim + rnorm(n, sd = x / 3)
  rec.sim[i] <- summary(lm(y.sim ~ x.sim))$coeff[2,1] - b

}


par(mfrow = c(1, 2))
### Bootstrapped sampling distribution in gray histogram
hist(rec.boot[,1], freq = F, col = "gray", breaks = 15, main = "Wild", xlab = expression(hat(b)))
### Simulated Sampling Distribution in red
lines(density(rec.sim), col = "red", lwd = 2)
### Model Based Sampling Distribution in Blue
lines(seq(-5, 5, by = .01), dnorm(seq(-5, 5, by = .01), mean = 0, sd = sqrt(1 / sum((x - mean(x))^2))),


### Bootstrapped sampling distribution in gray histogram
hist(rec.boot[,2], freq = F, col = "gray", breaks = 15, main = "Pairs", xlab =  expression(hat(b)))
### Simulated Sampling Distribution in red
lines(density(rec.sim), col = "red", lwd = 2)
### Model Based Sampling Distribution in Blue
lines(seq(-5, 5, by = .01), dnorm(seq(-5, 5, by = .01), mean = 0, sd = sqrt(1 / sum((x - mean(x))^2))),
```

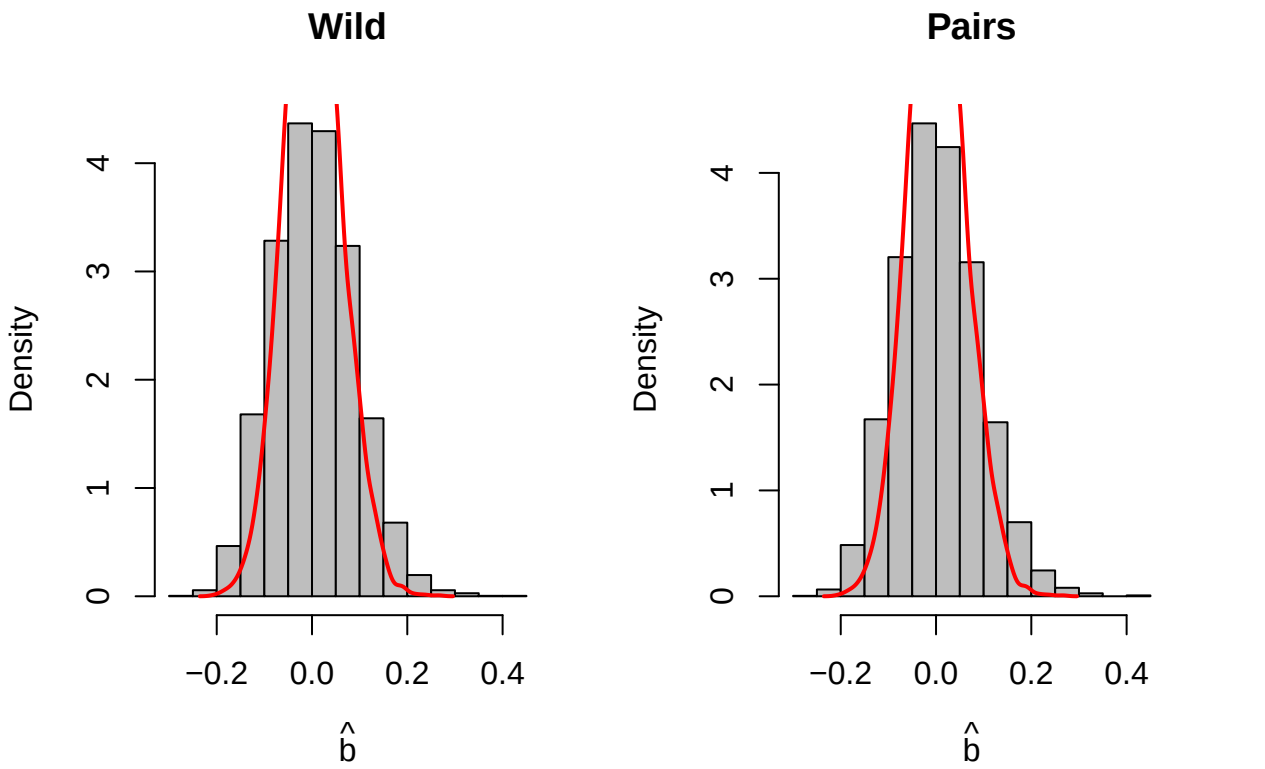


Non-standard quantities

Finally, we can see that the bootstrap approximates the sampling distribution of quantities for which the sampling distribution would be hard compute. In particular, we will look at the sampling distribution of $\hat{b}_1/\hat{b}_2$.

```
## Homoscedastic linear model
# Fixed Covariates
# Using Wild Bootstrap

sim.size <- 5000
b <- c(1, 2)
n <- 100
# Fixed Design
x <- mvtnorm::rmvnorm(n, sigma = matrix(c(1, .2, .2, 1), byrow = T, 2, 2))
y <- 1 + x %*% b + rnorm(n)

mod <- lm(y ~ x)
## We are interested in the estimated coefficient
observed.stat <- mod$coef[2] / mod$coef[3]


### Approximate sampling distribution using the Wild and empirical bootstrap
rec.boot <- matrix(0, sim.size, 2)
for(i in 1:sim.size){

  # Wild Bootstrap
  y.boot.wild <- mod$fitted + mod$residuals * rnorm(n)

  mod.wild <- lm(y.boot.wild ~ x)
```

```r
  # Calculate the statistic for the bootstrap sample
  rec.boot[i, 1] <- mod.wild$coeff[2] / mod.wild$coeff[3]  - observed.stat


  # Pairs Bootstrap
  ind <- sample(n, replace = T)
  x.boot.emp <- x[ind, ]
  y.boot.emp <- y[ind]

  mod.emp <- lm(y.boot.emp ~ x.boot.emp)
  # Calculate the statistic for the bootstrap sample
  rec.boot[i, 2] <- mod.emp$coeff[2] / mod.emp$coeff[3] - observed.stat


}



rec.sim <- rep(0, sim.size)
for(i in 1:sim.size){

  ## Don't re-draw
  y.sim <- 1 + x %*% b + rnorm(n)
  mod.sim <- lm(y.sim ~ x)
  rec.sim[i] <- mod.sim$coefficients[2] / mod.sim$coefficients[3] - (b[1] / b[2])

}


par(mfrow = c(1, 2))
### Bootstrapped sampling distribution in gray histogram
hist(rec.boot[,1], freq = F, col = "gray", breaks = 15, main = "Wild", xlab = expression(hat(b)))
### Simulated Sampling Distribution in red
lines(density(rec.sim), col = "red", lwd = 2)
### Model Based Sampling Distribution in Blue
#lines(seq(-5, 5, by = .01), dnorm(seq(-5, 5, by = .01), mean = 0, sd = sqrt(1 / sum((x - mean(x))^2)))


### Bootstrapped sampling distribution in gray histogram
hist(rec.boot[,2], freq = F, col = "gray", breaks = 15, main = "Pairs", xlab =  expression(hat(b)))
### Simulated Sampling Distribution in red
lines(density(rec.sim), col = "red", lwd = 2)
```




```
### Model Based Sampling Distribution in Blue
```

Bootstrapping Linear Models

The R package `lmboot` can do the bootstrap with a single function (instead of the for loop we used above). We'll examine the housing price data set again, and fit a model which predicts the log(price) given the log(area), bedrooms, bathrooms, whether there is a garage, and quality.

```
# install.packages("lmboot")
library("lmboot")
library("lmtest")
```

```
## Loading required package: zoo
```

```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
library("sandwich")


fileName <- "https://raw.githubusercontent.com/ysamwang/btry6020_sp22/main/lectureData/estate.csv"
housing_data <- read.csv(fileName)
names(housing_data)
```

```
##  [1] "id"      "price"   "area"    "bed"     "bath"    "ac"      "garage"
##  [8] "pool"    "year"    "quality" "style"   "lot"     "highway"
```

```r
mod <- lm(log(price) ~ log(area) + bed + bath + garage + quality,
          data = housing_data)


summary(mod)
```

```
##
## Call:
## lm(formula = log(price) ~ log(area) + bed + bath + garage + quality,
##     data = housing_data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.45576 -0.11130 -0.01898  0.11071  0.66293
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)    7.160399   0.373898  19.151  < 2e-16 ***
## log(area)      0.695670   0.050503  13.775  < 2e-16 ***
## bed            0.001827   0.010254   0.178 0.858647
## bath           0.050975   0.013234   3.852 0.000132 ***
## garage         0.064141   0.015402   4.164 3.66e-05 ***
## qualitylow    -0.473414   0.040721 -11.626  < 2e-16 ***
## qualitymedium -0.350370   0.030252 -11.582  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1847 on 515 degrees of freedom
## Multiple R-squared:  0.8191, Adjusted R-squared:  0.817
## F-statistic: 388.7 on 6 and 515 DF,  p-value: < 2.2e-16
```

We can now form confidence intervals for each of the parameters using a bootstrap procedure. We can see that in this case, each of the procedures we use to create confidence intervals aren't too dissimilar. This should hopefully be reassuring that our model assumptions aren't too unreasonable.

```r
## Gives list of parameter estimates from each empirical bootstrap
# (or paired boostrap) resample
paired.output <- paired.boot(log(price) ~ log(area) + bed + bath + garage + quality,
          data = housing_data, B = 5000)
## Gives list of parameter estimates from each wild bootstrap resample
wild.output <- wild.boot(log(price) ~ log(area) + bed + bath + garage + quality,
          data = housing_data, B = 5000)


## Apply takes a function (FUN) and applies it to each row or column
## of a matrix
## MAR = 1, applies the function to each row
## MAR = 2, applies the function to each column

## Get standard deviation of each column (which corresponds to the bootstrap estimates)
paired.se <- apply(paired.output$bootEstParam, MAR = 2, FUN = sd)
wild.se <- apply(wild.output$bootEstParam, MAR = 2, FUN = sd)

## For the percentile method get .025 and .975 quantiles of each column, but we
## subtract the estimated values
```

```
paired.pct <- apply(t(paired.output$bootEstParam) - mod$coefficients,
                    MAR = 1, FUN = quantile, prob = c(.025, .975))
wild.pct <- apply(t(wild.output$bootEstParam) - mod$coefficients,
                  MAR = 1, FUN = quantile, prob = c(.025, .975))

## Form the confidence intervals using the normal approximation
# but SE's estimated via bootstrap
cbind(mod$coef -1.96 * paired.se, mod$coef + 1.96 * paired.se)
```

```
##                       [,1]        [,2]
## (Intercept)    6.37261054  7.94818809
## log(area)      0.58644956  0.80489075
## bed           -0.02315526  0.02680933
## bath           0.01865593  0.08329436
## garage         0.02933643  0.09894506
## qualitylow    -0.56992859 -0.37689867
## qualitymedium -0.42397654 -0.27676378
```

```
cbind(mod$coef -1.96 * wild.se, mod$coef + 1.96 * wild.se)
```

```
##                       [,1]        [,2]
## (Intercept)    6.37829839  7.94250024
## log(area)      0.58738274  0.80395757
## bed           -0.02321490  0.02686897
## bath           0.01913437  0.08281592
## garage         0.03000300  0.09827849
## qualitylow    -0.57012759 -0.37669967
## qualitymedium -0.42434117 -0.27639916
```

```
# Percentile bootstrap
mod$coefficients - t(paired.pct[2:1, ])
```

```
##                      97.5%       2.5%
## (Intercept)    6.36417971  7.96050083
## log(area)      0.58337358  0.80743349
## bed           -0.02447773  0.02627465
## bath           0.01650245  0.08115089
## garage         0.02724734  0.09727066
## qualitylow    -0.57295788 -0.38165733
## qualitymedium -0.42502756 -0.27704360
```

```
mod$coefficients - t(wild.pct[2:1, ])
```

```
##                      97.5%       2.5%
## (Intercept)    6.37157627  7.94460352
## log(area)      0.58853048  0.80468941
## bed           -0.02357258  0.02744388
## bath           0.01890978  0.08306457
## garage         0.02988036  0.09854777
## qualitylow    -0.57115914 -0.37645536
## qualitymedium -0.42575729 -0.27699826
```

```
# Model based confidence interval
coefci(mod, level = .95)
```

```
##                     2.5 %     97.5 %
## (Intercept)    6.42584727  7.89495136
```

```
## log(area)      0.59645341  0.79488690
## bed           -0.01831687  0.02197094
## bath           0.02497504  0.07697524
## garage         0.03388163  0.09439986
## qualitylow    -0.55341409 -0.39341317
## qualitymedium -0.40980286 -0.29093747
```

```
# Robust confidence intervals
coefci(mod, level = .95, vcov. = vcovHC(mod, type = "HC3"))
```

```
##                     2.5 %      97.5 %
## (Intercept)    6.35774837  7.96305026
## log(area)      0.58416815  0.80717215
## bed           -0.02377117  0.02742524
## bath           0.01777536  0.08417493
## garage         0.02891852  0.09936298
## qualitylow    -0.57206169 -0.37476557
## qualitymedium -0.42573251 -0.27500781
```

Mixed Effects Models

To examine mixed effects models, we will explore ecology data from "Integrative modelling reveals mechanisms linking productivity and plant species richness" by Grace et. al. (2016, Nature). In particular, the authors are interested in modeling the relationship between the producitivity of a particular plot of land and the species richness of that plot of land. Take a moment to briefly skim the paper (or abstract).

We can fit a model to predict $\log(richness)$ from $\log(productivity)$ and $\log(totalbiomass)$.

```
library(lme4)
```

```
## Loading required package: Matrix
```

```
fileName <- "https://raw.githubusercontent.com/ysamwang/btry6020_sp22/main/lectureData/grace_plot_level
dat <- read.csv(fileName)

# Remove Missing Data
# Generally, we want to be careful about the data we remove
# As this may bias our estimates if the missingness is important
dat <- na.omit(dat)
dim(dat)
```

```
## [1] 1126   19
```

```
# Fit a linear model which disregards the site structure
mod <- lm(ln.prich~ ln.ptotmass + ln.pprod, data = dat)
summary(mod)
```

```
##
## Call:
## lm(formula = ln.prich ~ ln.ptotmass + ln.pprod, data = dat)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.65561 -0.29794 -0.03006  0.31573  1.19624
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.71071    0.10316   26.28   <2e-16 ***
## ln.ptotmass -0.39818    0.03143  -12.67   <2e-16 ***
## ln.pprod     0.39260    0.03406   11.53   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4781 on 1123 degrees of freedom
## Multiple R-squared:  0.1269, Adjusted R-squared:  0.1253
## F-statistic: 81.58 on 2 and 1123 DF,  p-value: < 2.2e-16
```

Questions

- Check the model we fit above. Do the linear model assumptions seem to hold?
- In their data set, they have recorded characteristics about 1,126 plots of land, and each of those plots are situated in 1 of 39 different sites. What suspicions does this raise about the independence assumption?

We can account for potential dependence across site using either a fixed effect or random effect.

```
# Fit model with a fixed effect for each site
mod.fe <- lm(ln.prich~ ln.ptotmass + ln.pprod + psitecode, data = dat)
```

```r
summary(mod.fe)
```

```
##
## Call:
## lm(formula = ln.prich ~ ln.ptotmass + ln.pprod + psitecode, data = dat)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.29766 -0.11458  0.00714  0.13130  0.96619
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)            2.78415    0.11533  24.141  < 2e-16 ***
## ln.ptotmass           -0.09531    0.03351  -2.845  0.00453 **
## ln.pprod               0.07471    0.03147   2.374  0.01777 *
## psitecodeazi.cn        0.94139    0.05598  16.818  < 2e-16 ***
## psitecodebarta.us     -0.07117    0.05678  -1.253  0.21031
## psitecodebldr.us      -0.25827    0.06323  -4.085 4.74e-05 ***
## psitecodebnch.us      -0.30758    0.05791  -5.312 1.32e-07 ***
## psitecodebogong.au     0.25557    0.05615   4.551 5.94e-06 ***
## psitecodebttr.us       0.06195    0.05625   1.101  0.27103
## psitecodeburrawan.au -0.24077    0.05836  -4.125 3.99e-05 ***
## psitecodecbgb.us      -0.40084    0.05601  -7.157 1.51e-12 ***
## psitecodecdpt.us      -0.14031    0.05828  -2.408  0.01622 *
## psitecodederr.au      -0.18760    0.06328  -2.964  0.00310 **
## psitecodeelliot.us    -0.07176    0.06633  -1.082  0.27957
## psitecodefnly.us       0.42346    0.06909   6.129 1.24e-09 ***
## psitecodegilb.za       0.53764    0.05858   9.178  < 2e-16 ***
## psitecodeglac.us      -0.61854    0.05684 -10.883  < 2e-16 ***
## psitecodehall.us      -0.43247    0.05548  -7.795 1.50e-14 ***
## psitecodehart.us      -0.17462    0.07500  -2.328  0.02008 *
## psitecodehnvr.us      -0.24388    0.06077  -4.013 6.40e-05 ***
## psitecodekiny.au       0.73608    0.06118  12.032  < 2e-16 ***
## psitecodelook.us      -0.50987    0.06095  -8.365  < 2e-16 ***
## psitecodemtca.au       0.03770    0.06150   0.613  0.54003
## psitecodepape.de      -1.08446    0.07830 -13.851  < 2e-16 ***
## psitecodesage.us      -0.18666    0.05918  -3.154  0.00165 **
## psitecodesava.us      -0.16555    0.09647  -1.716  0.08645 .
## psitecodesedg.us      -0.64198    0.05755 -11.155  < 2e-16 ***
## psitecodesereng.tz    -0.06220    0.05798  -1.073  0.28358
## psitecodesevi.us      -0.47813    0.06430  -7.436 2.10e-13 ***
## psitecodesgs.us       -0.77978    0.06289 -12.400  < 2e-16 ***
## psitecodeshps.us       0.15890    0.08143   1.951  0.05126 .
## psitecodesier.us      -0.33218    0.05701  -5.826 7.46e-09 ***
## psitecodesmith.us      0.48177    0.05586   8.624  < 2e-16 ***
## psitecodespin.us      -0.55550    0.05500 -10.099  < 2e-16 ***
## psitecodesumm.za       0.77600    0.05825  13.322  < 2e-16 ***
## psitecodetemple.us     0.15023    0.06154   2.441  0.01480 *
## psitecodetrel.us      -0.94309    0.05647 -16.701  < 2e-16 ***
## psitecodetyso.us      -0.42089    0.05369  -7.840 1.07e-14 ***
## psitecodeukul.za       0.47461    0.05776   8.217 5.89e-16 ***
## psitecodeunc.us       -0.24733    0.05567  -4.442 9.80e-06 ***
## psitecodevalm.ch       0.59403    0.05791  10.259  < 2e-16 ***
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2121 on 1085 degrees of freedom
## Multiple R-squared:  0.834,  Adjusted R-squared:  0.8278
## F-statistic: 136.2 on 40 and 1085 DF,  p-value: < 2.2e-16
```

```
# Fit a model with a random intercept for each site
mod.re <- lmer(ln.prich~ ln.ptotmass + ln.pprod + (1 | psitecode), data = dat)
summary(mod.re)
```

```
## Linear mixed model fit by REML ['lmerMod']
## Formula: ln.prich ~ ln.ptotmass + ln.pprod + (1 | psitecode)
##    Data: dat
##
## REML criterion at convergence: -93.8
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -6.1282 -0.5461  0.0378  0.6084  4.5653
##
## Random effects:
##  Groups    Name        Variance Std.Dev.
##  psitecode (Intercept) 0.2189   0.4679
##  Residual              0.0450   0.2121
## Number of obs: 1126, groups:  psitecode, 39
##
## Fixed effects:
##             Estimate Std. Error t value
## (Intercept)  2.69872    0.12028  22.437
## ln.ptotmass -0.10650    0.03289  -3.238
## ln.pprod     0.08389    0.03097   2.709
##
## Correlation of Fixed Effects:
##             (Intr) ln.ptt
## ln.ptotmass -0.374
## ln.pprod    -0.017 -0.867
```

```
# We can get confidence intervals for each of the slope estimates
confint(mod.re)
```

```
## Computing profile confidence intervals ...
```

```
##                   2.5 %      97.5 %
## .sig01       0.37415098  0.58771282
## .sigma       0.20333976  0.22117992
## (Intercept)  2.46309359  2.93397981
## ln.ptotmass -0.17141642 -0.04215786
## ln.pprod     0.02331589  0.14494742
```

```
# We can also get the estimated random effects for each site
ranef(mod.re)
```

```
## $psitecode
##             (Intercept)
## amcamp.us    0.10261368
## azi.cn       1.03383497
## barta.us     0.03371104
```

```
## bldr.us     -0.15230536
## bnch.us     -0.20479023
## bogong.au    0.35234873
## bttr.us      0.16327849
## burrawan.au -0.14180528
## cbgb.us     -0.28766459
## cdpt.us     -0.04533922
## derr.au     -0.09192849
## elliot.us    0.02578124
## fnly.us      0.51849865
## gilb.za      0.63008097
## glac.us     -0.51210127
## hall.us     -0.32405896
## hart.us     -0.07575354
## hnvr.us     -0.14154704
## kiny.au      0.82650953
## look.us     -0.40621308
## mtca.au      0.13178204
## pape.de     -0.96212697
## sage.us     -0.08518304
## sava.us     -0.04513374
## sedg.us     -0.53139534
## sereng.tz    0.03483095
## sevi.us     -0.38166503
## sgs.us      -0.67989210
## shps.us      0.25154825
## sier.us     -0.22937899
## smith.us     0.57748265
## spin.us     -0.45075544
## summ.za      0.86697827
## temple.us    0.24495449
## trel.us     -0.83388755
## tyso.us     -0.32093065
## ukul.za      0.56826414
## unc.us      -0.14527656
## valm.ch      0.68663440
##
## with conditional variances for "psitecode"
```

```
# Suppose we want to test whether or not ln.pprod is associated with
# ln.prich after accounting for total mass and the random intercepts
# We can do that with the anova command as before
mod.re.null <- lmer(ln.prich~ ln.ptotmass + (1 | psitecode), data = dat)
anova(mod.re.null, mod.re)
```

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

```
## Data: dat
## Models:
## mod.re.null: ln.prich ~ ln.ptotmass + (1 | psitecode)
## mod.re: ln.prich ~ ln.ptotmass + ln.pprod + (1 | psitecode)
##             npar     AIC     BIC logLik deviance  Chisq Df Pr(>Chisq)
## mod.re.null    4 -93.315 -73.210 50.658  -101.31
## mod.re         5 -98.657 -73.524 54.328  -108.66 7.3411  1    0.00674 **
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Questions

- How should we interpret each of the fixed effects in mod.fe?
- Compare the estimates of the models which account for site with the estimates of the model which naively does not account for site.

Suppose we think that each site may have it's own random intercept, as well as it's own random slope for total mass. We can fit that model using the following code

```
# Fit a model with a random intercept and slopes for each site
mod.re2 <- lmer(ln.prich~ ln.pprod + (1 + ln.pprod | psitecode), data = dat)
summary(mod.re2)
```

```
## Linear mixed model fit by REML ['lmerMod']
## Formula: ln.prich ~ ln.pprod + (1 + ln.pprod | psitecode)
##    Data: dat
##
## REML criterion at convergence: -96.9
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -6.1949 -0.5419  0.0576  0.6099  4.6042
##
## Random effects:
##  Groups    Name        Variance Std.Dev. Corr
##  psitecode (Intercept) 0.216538 0.46534
##            ln.pprod    0.005633 0.07505  -0.41
##  Residual              0.044440 0.21081
## Number of obs: 1126, groups:  psitecode, 39
##
## Fixed effects:
##              Estimate Std. Error t value
## (Intercept)  2.557165   0.117449  21.773
## ln.pprod    -0.002827   0.020730  -0.136
##
## Correlation of Fixed Effects:
##          (Intr)
## ln.pprod -0.778
```

Questions

- How would we interpret the random slopes in this context?
- Do you think it makes sense to include random slopes?