Penguins Assignment

Iaon Gulyas (51903899), Luc Thiery (51903906)

2024-01-23

Contents

(1) Data Exploration

Starting with the EDA, we are taking a look at the data itself. We have 4 columns, from which the "bill_length" is the feature column that we are trying to predict in this experiment. We can see that two columns are non-numeric and two are numeric. This will require further preprocessing steps that will allow us to work with the data in a suitable way.

```
head(penguins)
```

```
##   species    sex bill_depth bill_length
## 1  Adelie   male       18.7        39.1
## 2  Adelie female       17.4        39.5
## 3  Adelie female       18.0        40.3
## 5  Adelie female       19.3        36.7
## 6  Adelie   male       20.6        39.3
## 7  Adelie female       17.8        38.9
```

An important step is to check whether we have any missing values. In this case, we do not.

```
na_count <-sapply(data, function(y) sum(length(which(is.na(y)))))
```

```
## Warning in is.na(y): is.na() applied to non-(list or vector) of type 'symbol'
## Warning in is.na(y): is.na() applied to non-(list or vector) of type 'language'
## Warning in is.na(y): is.na() applied to non-(list or vector) of type 'language'
## Warning in is.na(y): is.na() applied to non-(list or vector) of type 'symbol'
## Warning in is.na(y): is.na() applied to non-(list or vector) of type 'language'
```

```
data.frame(na_count)
```

```
##           na_count
## ...              0
```

```
## list            0
## package         0
## lib.loc         0
## verbose         0
## envir           0
## overwrite       0
##                 0
```

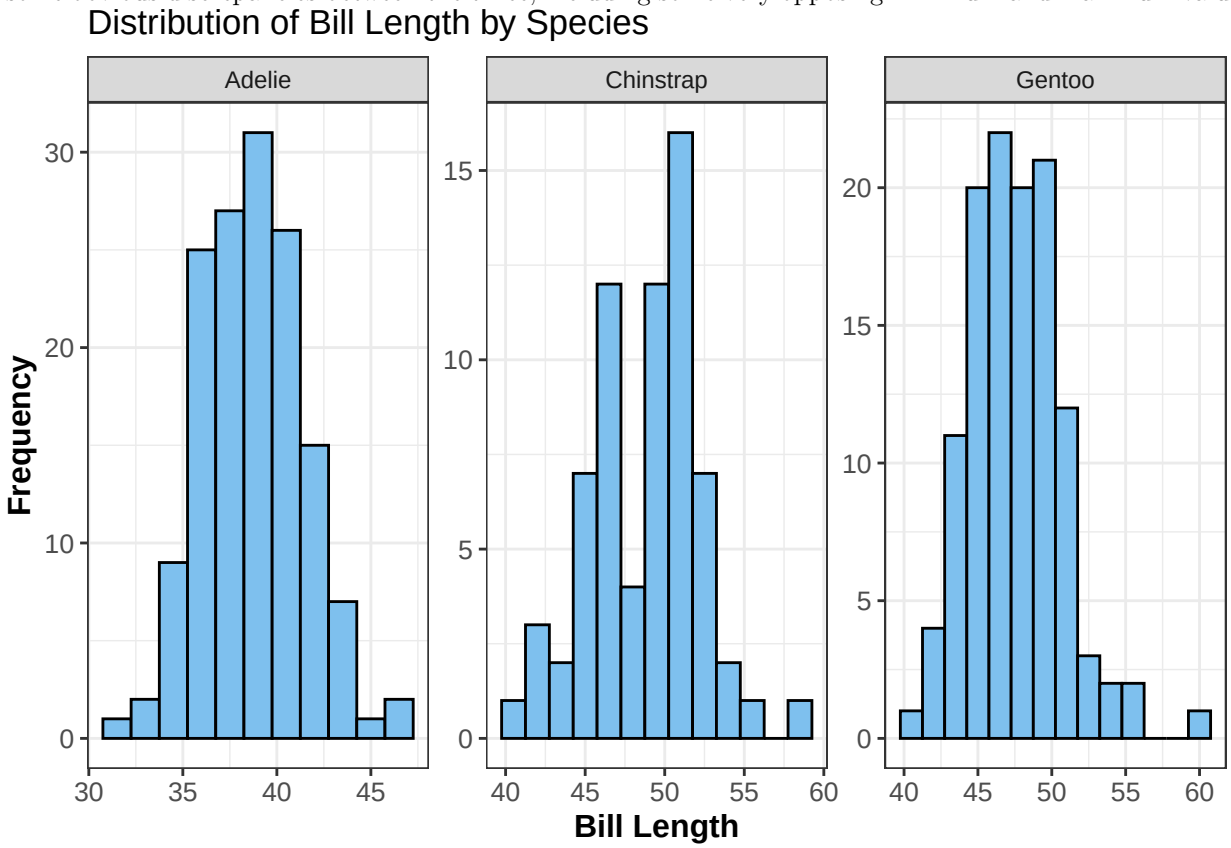
Before prediction anything on our data, we need to take a look at the current basic statistical information about our data. Of course, this only applies on the numerical columns.

```
describe(penguins)
```

```
##             vars   n  mean   sd median trimmed  mad  min  max range  skew
## species*       1 333  1.92 0.89    2.0    1.90 1.48  1.0  3.0   2.0  0.16
## sex*           2 333  1.50 0.50    2.0    1.51 0.00  1.0  2.0   1.0 -0.02
## bill_depth     3 333 17.16 1.97   17.3   17.19 2.22 13.1 21.5   8.4 -0.15
## bill_length    4 333 43.99 5.47   44.5   43.98 6.97 32.1 59.6  27.5  0.04
##             kurtosis   se
## species*       -1.72 0.05
## sex*           -2.01 0.03
## bill_depth     -0.91 0.11
## bill_length    -0.90 0.30
```

Data Visualization

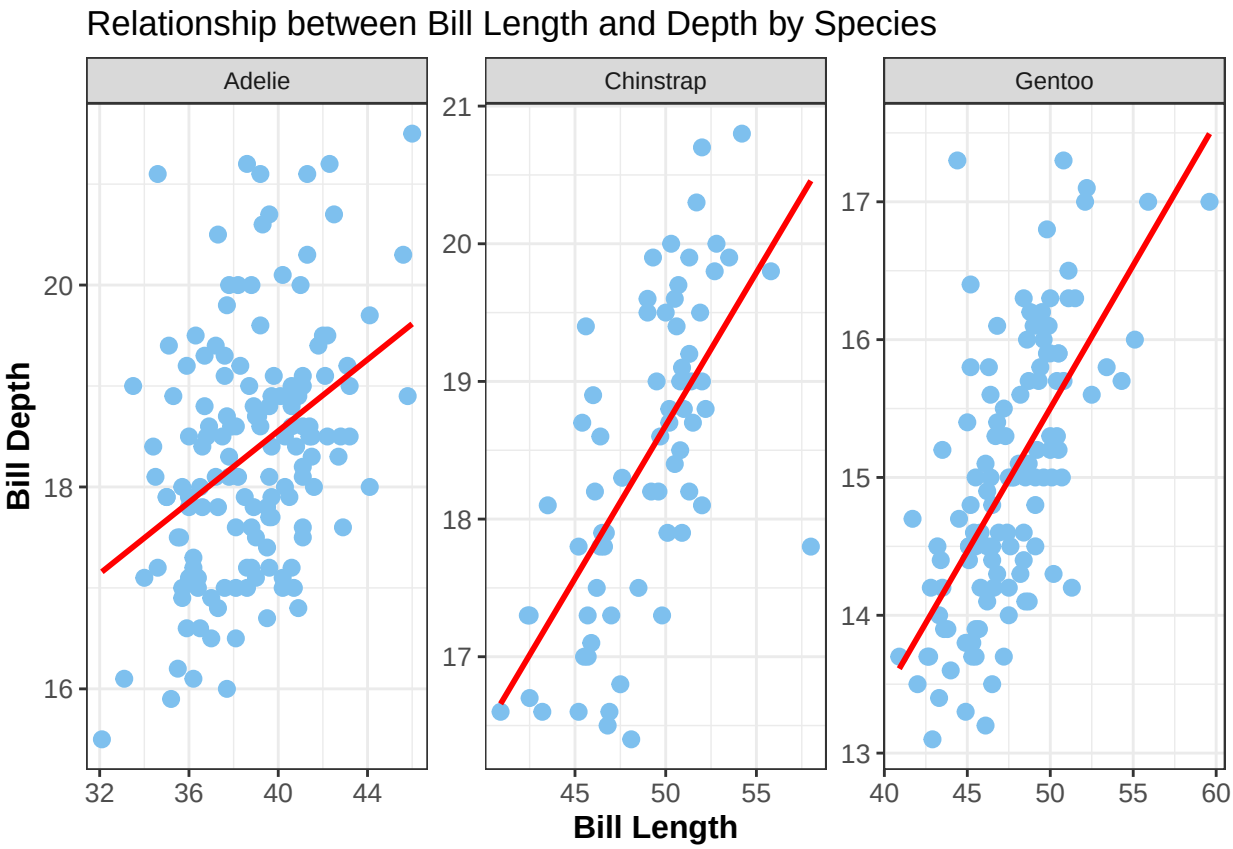
In this step, we would like to take a closer look at our data using visualizations. Given the structure of the data, we would like to take a look at the distribution of the bill lengths for the 3 penguin species. We can see some obvious discrepancies between the three, including some very opposing minimum and maximum values.



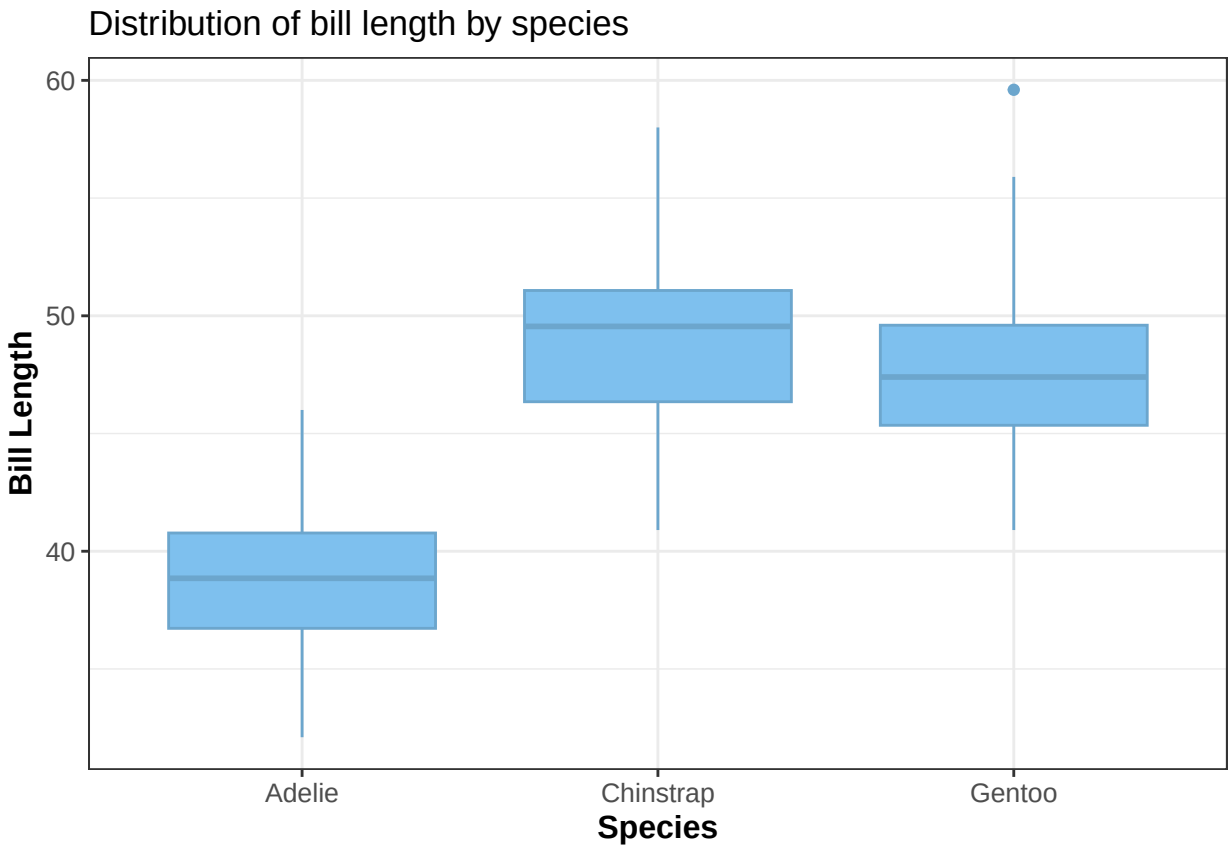
In order to check the correlation between the second numerical feature of our data, the bill_depth, we have created some scatterplots that are also depicting the 3 penguins species, but this time with a look into the correlation between the measurements of the bill. We can observe a certain linearity, which we have underlined using the red line. The longer the bill, the higher the depth.

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

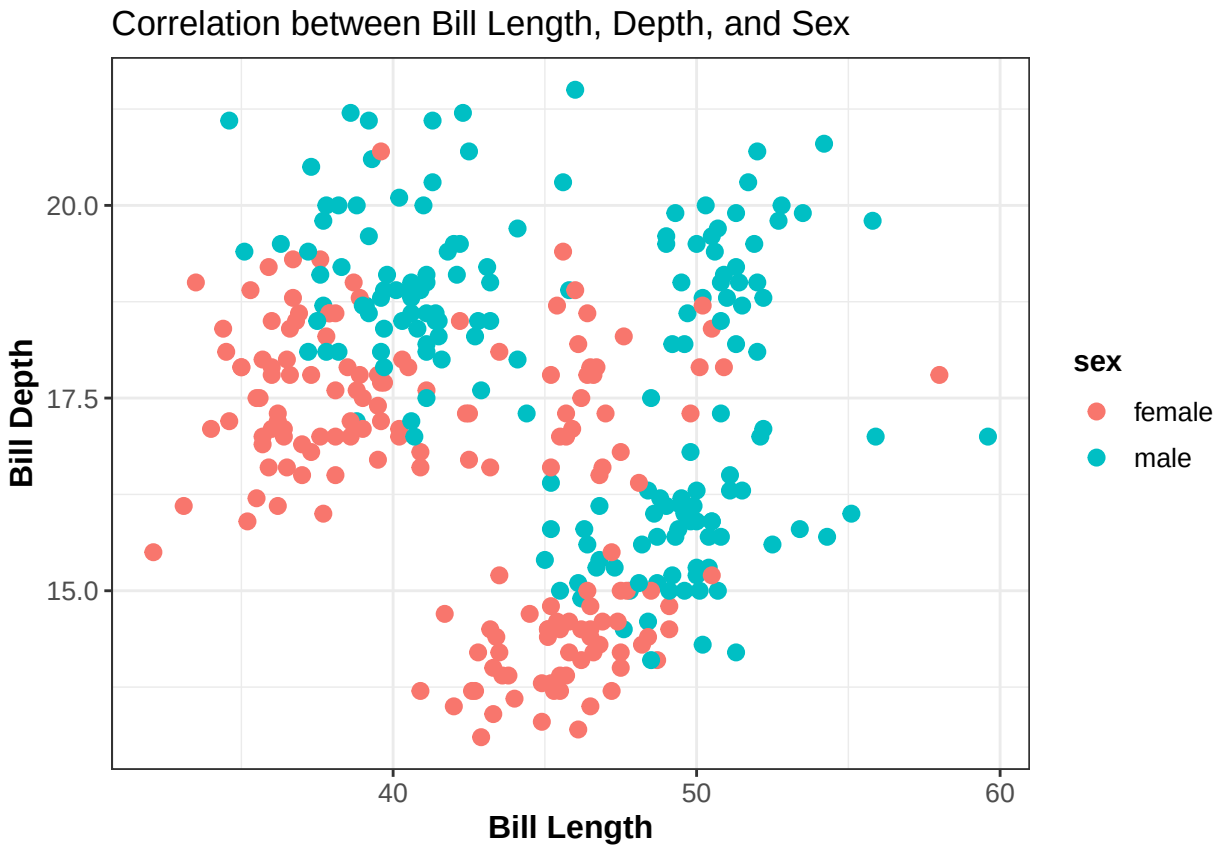


When observing the distribution of the bill length by species, we can tell that the Chinstrap has the generally the lenghtiest bills, while an outlier for the Gentoo shows the longest bill and the Adelie is at the bottom of this



hierarchy.

After comparing exclusively the characteristics of the bill based on the species, we would like to take a look and the distribution based on the sex of the penguins. We can notice a slight cluster formation, the female species tending to have shorther bill length and an average bill depth, while the male counterparts are generally having longer bills and a pretty evenly distributed bill depth.



(2) Modelling

```
# For Separate Model
data_separate <- list(
  N = nrow(penguins),
  K = length(unique(penguins$species)),
  bill_length = penguins$bill_length,
  species = as.integer(factor(penguins$species)),
  bill_depth = penguins$bill_depth,
  penguin_sex = penguins$sex
)

# For Hierarchical Model
data_hierarchical <- list(
  N = nrow(penguins),
  K = length(unique(penguins$species)),
  bill_length = penguins$bill_length,
  species = as.integer(factor(penguins$species)),
  bill_depth = penguins$bill_depth,
  penguin_sex = penguins$sex
)
```

Loading the two RSTAN models, one for the non-hierarhical model and one for the hierarchical one.

```
# Load Stan models
separate_model <- stan_model("~/Documents/TU Wien - Data Science MSc/WS23/Bayesian Statistics/Assignmen
```

```
## Trying to compile a simple C file
```

```
## Running /Library/Frameworks/R.framework/Resources/bin/R CMD SHLIB foo.c
## using C compiler: 'Apple clang version 15.0.0 (clang-1500.1.0.2.5)'
## using SDK: 'MacOSX14.2.sdk'
## clang -arch arm64 -I"/Library/Frameworks/R.framework/Resources/include" -DNDEBUG   -I"/Library/Framew
## In file included from <built-in>:1:
## In file included from /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/library/StanHeade
## In file included from /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/library/RcppEigen/
## In file included from /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/library/RcppEigen/
## /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/library/RcppEigen/include/Eigen/src/Core
## namespace Eigen {
## ^
## /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/library/RcppEigen/include/Eigen/src/Core
## namespace Eigen {
##                 ^
##                 ;
## In file included from <built-in>:1:
## In file included from /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/library/StanHeade
## In file included from /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/library/RcppEigen/
## /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/library/RcppEigen/include/Eigen/Core:96
## #include <complex>
##          ^~~~~~~~~
## 3 errors generated.
## make: *** [foo.o] Error 1
```

```
hierarchical_model <- stan_model("~/Documents/TU Wien - Data Science MSc/WS23/Bayesian Statistics/Assig
```

```
## Trying to compile a simple C file
```

```
## Running /Library/Frameworks/R.framework/Resources/bin/R CMD SHLIB foo.c
## using C compiler: 'Apple clang version 15.0.0 (clang-1500.1.0.2.5)'
## using SDK: 'MacOSX14.2.sdk'
## clang -arch arm64 -I"/Library/Frameworks/R.framework/Resources/include" -DNDEBUG   -I"/Library/Framew
## In file included from <built-in>:1:
## In file included from /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/library/StanHeade
## In file included from /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/library/RcppEigen/
## In file included from /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/library/RcppEigen/
## /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/library/RcppEigen/include/Eigen/src/Core
## namespace Eigen {
## ^
## /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/library/RcppEigen/include/Eigen/src/Core
## namespace Eigen {
##                 ^
##                 ;
## In file included from <built-in>:1:
## In file included from /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/library/StanHeade
## In file included from /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/library/RcppEigen/
## /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/library/RcppEigen/include/Eigen/Core:96
## #include <complex>
##          ^~~~~~~~~
## 3 errors generated.
## make: *** [foo.o] Error 1
```

```
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 1).
```

```
## Chain 1:
## Chain 1: Gradient evaluation took 4.4e-05 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0.44 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 1: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 1: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 1: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 1: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 1: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 1: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 1: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 1: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 1: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 1: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 1: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 1.794 seconds (Warm-up)
## Chain 1:                1.261 seconds (Sampling)
## Chain 1:                3.055 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 1.6e-05 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.16 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 2: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 2: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 2: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 2: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 2: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 2: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 2: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 2: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 2: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 2: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 2: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 1.739 seconds (Warm-up)
## Chain 2:                1.396 seconds (Sampling)
## Chain 2:                3.135 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 1.5e-05 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.15 seconds.
## Chain 3: Adjust your expectations accordingly!
```

```
## Chain 3:
## Chain 3:
## Chain 3: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 3: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 3: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 3: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 3: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 3: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 3: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 3: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 3: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 3: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 3: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 3: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 1.632 seconds (Warm-up)
## Chain 3:                1.325 seconds (Sampling)
## Chain 3:                2.957 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 1.5e-05 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0.15 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 4: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 4: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 4: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 4: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 4: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 4: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 4: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 4: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 4: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 4: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 4: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 1.812 seconds (Warm-up)
## Chain 4:                1.393 seconds (Sampling)
## Chain 4:                3.205 seconds (Total)
## Chain 4:

##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 6.3e-05 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0.63 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 2000 [  0%]  (Warmup)
```

```
## Chain 1: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 1: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 1: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 1: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 1: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 1: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 1: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 1: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 1: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 1: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 1: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 4.29 seconds (Warm-up)
## Chain 1:                4.712 seconds (Sampling)
## Chain 1:                9.002 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 2.1e-05 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.21 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 2: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 2: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 2: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 2: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 2: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 2: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 2: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 2: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 2: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 2: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 2: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 4.415 seconds (Warm-up)
## Chain 2:                3.758 seconds (Sampling)
## Chain 2:                8.173 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 2.1e-05 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.21 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 3: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 3: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 3: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 3: Iteration:  800 / 2000 [ 40%]  (Warmup)
```

```
## Chain 3: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 3: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 3: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 3: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 3: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 3: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 3: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 4.233 seconds (Warm-up)
## Chain 3:                4.183 seconds (Sampling)
## Chain 3:                8.416 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 2.1e-05 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0.21 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 4: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 4: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 4: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 4: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 4: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 4: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 4: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 4: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 4: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 4: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 4: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 4.56 seconds (Warm-up)
## Chain 4:                4.6 seconds (Sampling)
## Chain 4:                9.16 seconds (Total)
## Chain 4:
```

The training of the models (using 4 chains, 2000 iterations each) was not time consuming, moving seamingly fast on our local machine. The summary of the models looks as follows: - even though a at a very fine margin, the non-hierarchical model seems to perform better and to be more stable overall - the hierarchical model has some performance drops when looking at the mean - a very early sign of overfitting maybe

```
# Summary for Separate Model
summary(fit_separate)
```

```
## $summary
##                       mean     se_mean         sd         2.5%          25%
## alpha[1]        21.4986868 0.060912139 2.95405889   15.6226472   19.5352259
## alpha[2]        11.3336313 0.094634844 4.39511402    2.8077154    8.3830259
## alpha[3]        15.2820517 0.071347962 3.28601612    9.2290725   13.0002948
## beta_depth[1]    0.9440605 0.003306628 0.16056872    0.6227403    0.8377994
## beta_depth[2]    2.0353088 0.005149058 0.23852009    1.5762575    1.8688330
## beta_depth[3]    2.1525888 0.004744964 0.21846004    1.7085831    2.0006142
## sigma            2.4476467 0.001792030 0.09612967    2.2679157    2.3817025
```

```
## lp__          -471.4974134 0.049451574 1.95926997 -476.3143175 -472.5265431
##                        50%         75%       97.5%    n_eff      Rhat
## alpha[1]        21.5772828   23.452084   27.421835 2351.964 1.0010895
## alpha[2]        11.2458677   14.360119   19.801550 2156.940 1.0000199
## alpha[3]        15.1496010   17.546948   21.897387 2121.174 1.0023292
## beta_depth[1]    0.9416797    1.050197    1.270151 2358.040 1.0011273
## beta_depth[2]    2.0410558    2.192370    2.498633 2145.825 0.9999635
## beta_depth[3]    2.1606191    2.304545    2.562634 2119.718 1.0021899
## sigma            2.4458454    2.509788    2.647994 2877.559 1.0001874
## lp__          -471.1408638 -470.051141 -468.755216 1569.742 1.0002628
##
## $c_summary
## , , chains = chain:1
##
##               stats
## parameter               mean         sd         2.5%         25%          50%
##   alpha[1]        21.5642177 2.99816737   15.5070248   19.722867   21.6577034
##   alpha[2]        11.5429233 4.22967025    3.4036359    8.512592   11.4412586
##   alpha[3]        15.3215141 3.34997101    8.9339897   13.052180   15.2372404
##   beta_depth[1]    0.9406531 0.16299563    0.6181507    0.837141    0.9362451
##   beta_depth[2]    2.0237752 0.22952603    1.6023601    1.867570    2.0308847
##   beta_depth[3]    2.1498607 0.22254039    1.7191817    1.990042    2.1572476
##   sigma            2.4458815 0.09437802    2.2776593    2.379103    2.4398760
##   lp__          -471.5595952 1.99135593 -476.2749010 -472.663343 -471.2400347
##               stats
## parameter               75%       97.5%
##   alpha[1]        23.488694   27.460635
##   alpha[2]        14.449280   19.400665
##   alpha[3]        17.748126   21.860661
##   beta_depth[1]    1.039104    1.277109
##   beta_depth[2]    2.180649    2.461569
##   beta_depth[3]    2.299658    2.563062
##   sigma            2.503844    2.649876
##   lp__          -470.011330 -468.733661
##
## , , chains = chain:2
##
##               stats
## parameter               mean         sd         2.5%          25%          50%
##   alpha[1]        21.4540521 2.77640499   16.2808971   19.5095912   21.5772828
##   alpha[2]        11.4134810 4.31217408    3.1819150    8.4603388   11.3768707
##   alpha[3]        15.0705957 3.18619009    9.2284681   12.8194408   14.9019930
##   beta_depth[1]    0.9461551 0.15076271    0.6278871    0.8518204    0.9432623
##   beta_depth[2]    2.0311861 0.23400625    1.5881475    1.8604501    2.0332586
##   beta_depth[3]    2.1663410 0.21188362    1.7294535    2.0337398    2.1770813
##   sigma            2.4486655 0.09709685    2.2585844    2.3847328    2.4482768
##   lp__          -471.4054575 1.91186407 -476.2674353 -472.2912166 -471.0223702
##               stats
## parameter               75%       97.5%
##   alpha[1]        23.218159   27.253956
##   alpha[2]        14.557221   19.266354
##   alpha[3]        17.144094   21.773254
##   beta_depth[1]    1.051843    1.224130
##   beta_depth[2]    2.192294    2.482648
```

```
##   beta_depth[3]    2.316506    2.562716
##   sigma            2.514166    2.638184
##   lp__          -470.045376 -468.768921
##
## , , chains = chain:3
##
##               stats
## parameter              mean         sd         2.5%          25%          50%
##   alpha[1]        21.588625 2.94276495   15.5833604   19.6989546   21.539254
##   alpha[2]        11.118796 4.71346545    2.1475353    7.9258051   10.891867
##   alpha[3]        15.489966 3.29993930    9.2624503   13.1966292   15.446768
##   beta_depth[1]    0.939465 0.15986998    0.6180637    0.8342394    0.943668
##   beta_depth[2]    2.046443 0.25580299    1.5579516    1.8677224    2.054336
##   beta_depth[3]    2.139324 0.21958046    1.6867611    1.9868676    2.138030
##   sigma            2.449407 0.09489529    2.2675397    2.3849066    2.448596
##   lp__          -471.514873 1.95638791 -476.4615027 -472.6201003 -471.161285
##               stats
## parameter              75%       97.5%
##   alpha[1]        23.498201   27.515070
##   alpha[2]        14.366407   20.311908
##   alpha[3]        17.734624   22.193043
##   beta_depth[1]    1.043322    1.271091
##   beta_depth[2]    2.217016    2.542664
##   beta_depth[3]    2.291891    2.551787
##   sigma            2.507994    2.642168
##   lp__          -470.088732 -468.752377
##
## , , chains = chain:4
##
##               stats
## parameter              mean         sd        2.5%          25%          50%
##   alpha[1]        21.3878523 3.08999632   15.442302   19.3064222   21.4570420
##   alpha[2]        11.2593252 4.30377148    2.693144    8.5932061   11.1379732
##   alpha[3]        15.2461315 3.29688358    9.304324   12.9543379   15.0960753
##   beta_depth[1]    0.9499686 0.16816423    0.632516    0.8278406    0.9455708
##   beta_depth[2]    2.0398309 0.23359615    1.598656    1.8810606    2.0469453
##   beta_depth[3]    2.1548300 0.21915765    1.708568    2.0152244    2.1628162
##   sigma            2.4466329 0.09819873    2.268940    2.3746204    2.4448270
##   lp__          -471.5097278 1.97623755 -476.315693 -472.4963338 -471.1527584
##               stats
## parameter              75%       97.5%
##   alpha[1]        23.629323   27.227542
##   alpha[2]        14.145360   19.817612
##   alpha[3]        17.418674   21.835153
##   beta_depth[1]    1.067472    1.280625
##   beta_depth[2]    2.180941    2.508566
##   beta_depth[3]    2.307725    2.560518
##   sigma            2.511291    2.653970
##   lp__          -470.048660 -468.739525
```

```
# Summary for Hierarchical Model
summary(fit_hierarchical)
```

```
## $summary
##                              mean     se_mean         sd         2.5%
```

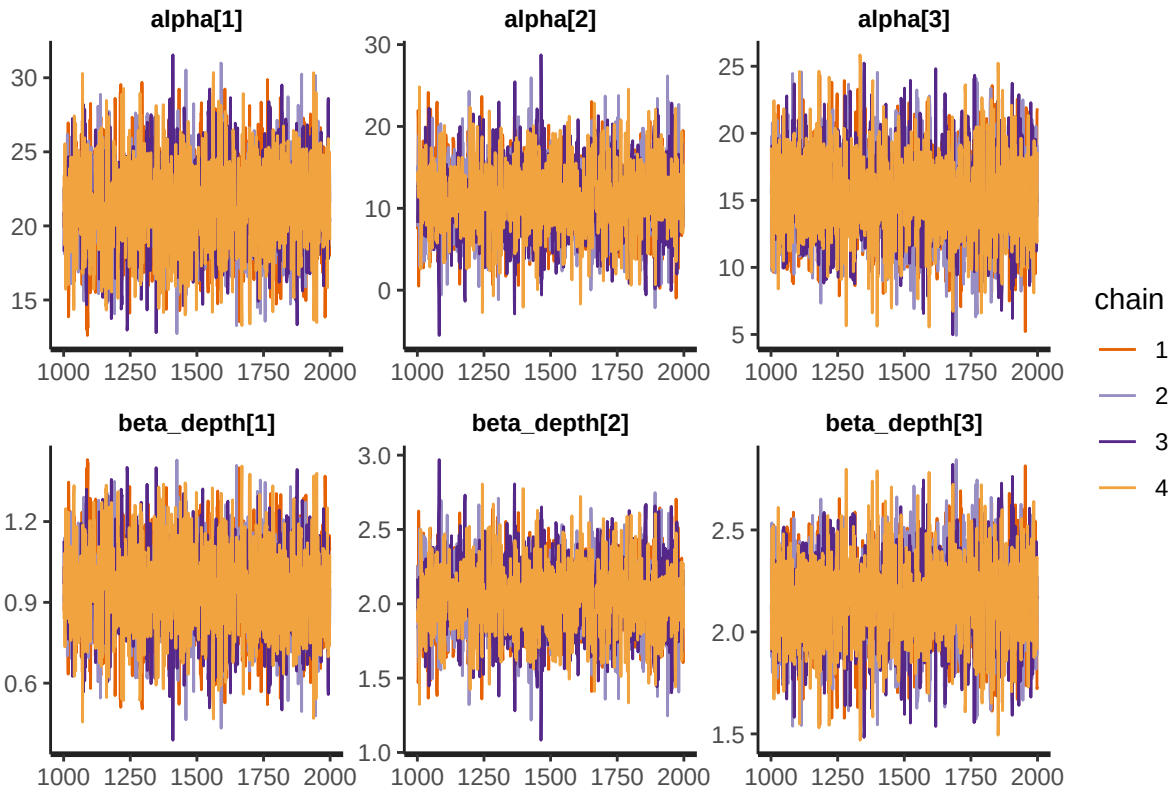
```
## alpha                  18.09986482 0.044552041 2.37353744   13.5214034
## beta_depth              1.53737810 0.007838941 0.30567567    0.9323265
## sigma                   2.45286154 0.001635918 0.09587845    2.2721929
## alpha_species[1]        1.29576544 0.033637971 1.80054717   -2.1659143
## alpha_species[2]       -0.56065884 0.032021774 1.83043177   -4.1348839
## alpha_species[3]       -0.04570985 0.031509762 1.83234813   -3.8443469
## beta_depth_species[1]  -0.47881337 0.007377103 0.29007007   -1.0394345
## beta_depth_species[2]   0.16219685 0.007251388 0.28924075   -0.4095633
## beta_depth_species[3]   0.43033329 0.007508746 0.29193698   -0.1466001
## lp__                 -471.08941451 0.053228683 2.13081001 -476.0491304
##                               25%           50%          75%         97.5%
## alpha                  16.50443316   18.01201881   19.6420055   23.02469611
## beta_depth              1.33085165    1.54311774    1.7414971    2.10990282
## sigma                   2.38736669    2.45046876    2.5137139    2.65038101
## alpha_species[1]        0.04530332    1.32786410    2.5412958    4.78643050
## alpha_species[2]       -1.78103799   -0.55477515    0.6292554    3.10062927
## alpha_species[3]       -1.29491777   -0.01604679    1.1799932    3.45953372
## beta_depth_species[1]  -0.67071016   -0.48015538   -0.2863965    0.08458792
## beta_depth_species[2]  -0.02865150    0.16154643    0.3497188    0.72808172
## beta_depth_species[3]   0.23340750    0.42602017    0.6355760    0.98961394
## lp__                 -472.29328213 -470.78898309 -469.5090719 -468.01397997
##                         n_eff      Rhat
## alpha                2838.291 1.0004828
## beta_depth           1520.572 1.0000398
## sigma                3434.936 1.0005364
## alpha_species[1]     2865.164 1.0007914
## alpha_species[2]     3267.505 1.0000425
## alpha_species[3]     3381.627 0.9995373
## beta_depth_species[1] 1546.088 0.9998725
## beta_depth_species[2] 1591.024 1.0006707
## beta_depth_species[3] 1511.623 1.0005475
## lp__                 1602.500 0.9996657
##
## $c_summary
## , , chains = chain:1
##
##                       stats
## parameter                    mean         sd         2.5%           25%
##   alpha                18.08019500 2.31780187   13.5881892   16.50399363
##   beta_depth            1.53415029 0.30550094    0.9323438    1.33516809
##   sigma                 2.44874305 0.09992487    2.2455388    2.38451756
##   alpha_species[1]      1.26019932 1.77032432   -2.0934851    0.02653058
##   alpha_species[2]     -0.48580173 1.81821802   -3.9296145   -1.62030237
##   alpha_species[3]     -0.02348265 1.78090511   -3.5871870   -1.23133026
##   beta_depth_species[1] -0.47264617 0.29327723   -1.0446756   -0.65918097
##   beta_depth_species[2]  0.16205991 0.29137293   -0.4006551   -0.03370117
##   beta_depth_species[3]  0.43389400 0.29748769   -0.1684971    0.23772102
##   lp__                -471.16111929 2.21719535 -476.4944043 -472.33906584
##                       stats
## parameter                    50%          75%        97.5%
##   alpha                18.03230511   19.6335759   22.6757473
##   beta_depth            1.53640107    1.7392315    2.1195352
##   sigma                 2.44748023    2.5106097    2.6597364
##   alpha_species[1]      1.27843355    2.5060778    4.7517186
```

```
##   alpha_species[2]        -0.58291254    0.6545633    3.3184285
##   alpha_species[3]        -0.03913186    1.1536762    3.4492704
##   beta_depth_species[1]   -0.47449164   -0.2756829    0.0845953
##   beta_depth_species[2]    0.16853001    0.3483414    0.7161219
##   beta_depth_species[3]    0.43263778    0.6517907    0.9886525
##   lp__                  -470.80426258 -469.5270218 -468.0724788
##
## , , chains = chain:2
##
##                        stats
## parameter                      mean         sd         2.5%           25%
##   alpha                   18.10263449 2.32088661   13.6855188  1.652634e+01
##   beta_depth               1.54615450 0.30392194    0.9766352  1.329433e+00
##   sigma                    2.45640999 0.09203177    2.2861118  2.391078e+00
##   alpha_species[1]         1.27193909 1.80714102   -2.1646679 -6.510161e-03
##   alpha_species[2]        -0.57678860 1.88506571   -4.1997384 -1.877564e+00
##   alpha_species[3]        -0.02927105 1.86944035   -3.8664461 -1.386637e+00
##   beta_depth_species[1]   -0.48619407 0.28511004   -1.0302724 -6.753611e-01
##   beta_depth_species[2]    0.15444057 0.28602306   -0.3826835 -4.592753e-02
##   beta_depth_species[3]    0.41987586 0.29473867   -0.1562649  2.178306e-01
##   lp__                  -471.01924031 1.99005592 -475.4205882 -4.721910e+02
##                        stats
## parameter                        50%          75%        97.5%
##   alpha                   17.97515177   19.6440714   22.9730977
##   beta_depth               1.55503060    1.7459716    2.1123195
##   sigma                    2.45407299    2.5187376    2.6365533
##   alpha_species[1]         1.29825295    2.5412958    4.6721451
##   alpha_species[2]        -0.53068207    0.6322117    3.0250578
##   alpha_species[3]         0.03530629    1.1753564    3.6964569
##   beta_depth_species[1]   -0.48052091   -0.2973658    0.0721882
##   beta_depth_species[2]    0.15084934    0.3439368    0.7144397
##   beta_depth_species[3]    0.41518617    0.6334906    0.9592933
##   lp__                  -470.79865457 -469.5345061 -467.9491040
##
## , , chains = chain:3
##
##                        stats
## parameter                      mean        sd         2.5%           25%
##   alpha                   18.1819998 2.4505182   13.5510083   16.54394568
##   beta_depth               1.5338695 0.3093256    0.9229026    1.32088494
##   sigma                    2.4496002 0.0961939    2.2748549    2.38165044
##   alpha_species[1]         1.2911521 1.7259417   -2.0926041    0.15926418
##   alpha_species[2]        -0.5409008 1.8651031   -4.0692907   -1.77812960
##   alpha_species[3]        -0.1177333 1.8568652   -3.7729354   -1.37037785
##   beta_depth_species[1]   -0.4799636 0.2929629   -1.0394587   -0.67839722
##   beta_depth_species[2]    0.1606290 0.2948921   -0.4309391   -0.02765816
##   beta_depth_species[3]    0.4335605 0.2890774   -0.1032509    0.23255949
##   lp__                  -471.1091862 2.0850575 -476.0124562 -472.38502290
##                        stats
## parameter                        50%          75%        97.5%
##   alpha                   18.1429345   19.7271038   23.23094709
##   beta_depth               1.5354891    1.7401188    2.09070672
##   sigma                    2.4489840    2.5106842    2.65035761
##   alpha_species[1]         1.3256417    2.4311492    4.88666883
```

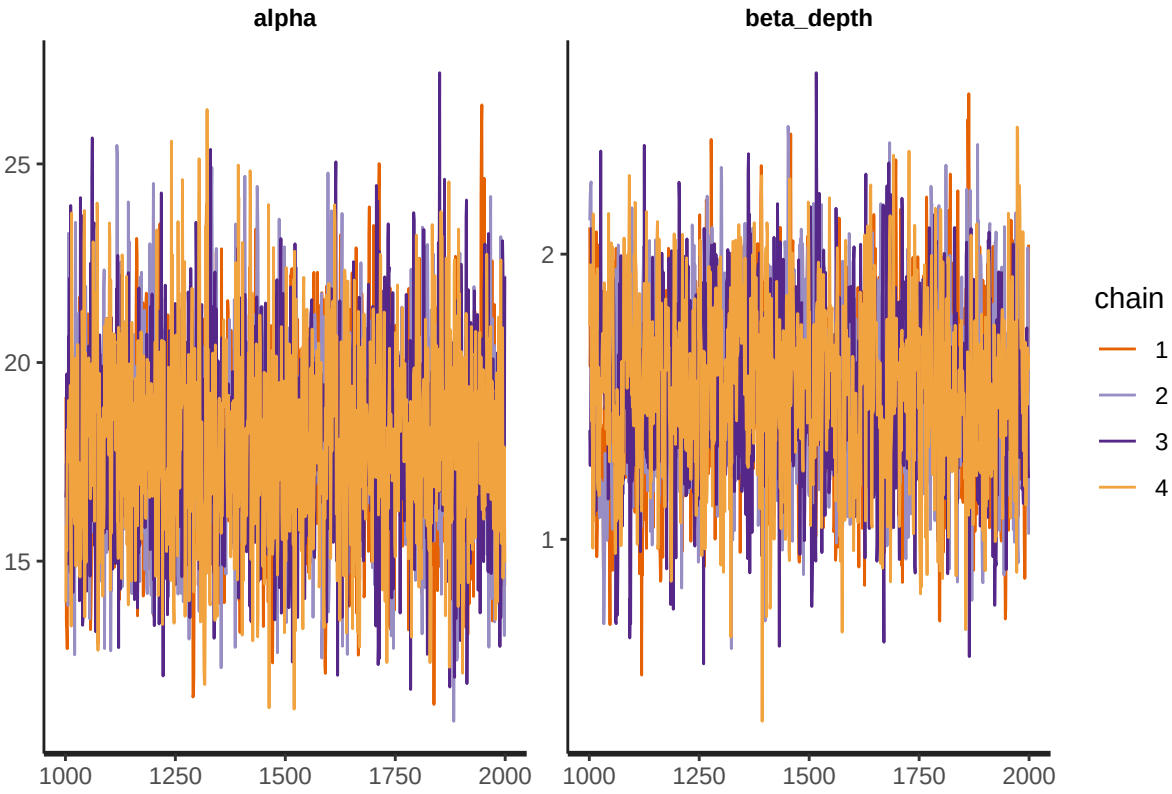
```
##   alpha_species[2]        -0.4856427    0.7164787    3.23843777
##   alpha_species[3]        -0.1631894    1.2308598    3.40093736
##   beta_depth_species[1]   -0.4818330   -0.2888954    0.08510919
##   beta_depth_species[2]    0.1615788    0.3552302    0.74351101
##   beta_depth_species[3]    0.4229377    0.6346310    1.03249115
##   lp__                  -470.8412462 -469.5956013 -468.07666965
##
## , , chains = chain:4
##
##                       stats
## parameter                      mean         sd         2.5%           25%
##   alpha                   18.03463003 2.40342889   13.4059465   16.38280018
##   beta_depth               1.53533813 0.30421270    0.9220274    1.33944524
##   sigma                    2.45669294 0.09505418    2.2822881    2.38976743
##   alpha_species[1]         1.35977123 1.89551518   -2.2589904    0.03207427
##   alpha_species[2]        -0.63914424 1.74973081   -4.4436904   -1.79559819
##   alpha_species[3]        -0.01235244 1.82171101   -4.0052276   -1.20147034
##   beta_depth_species[1]   -0.47644967 0.28911797   -1.0365532   -0.66776975
##   beta_depth_species[2]    0.17165792 0.28472984   -0.4089164   -0.01124812
##   beta_depth_species[3]    0.43400277 0.28650150   -0.1325079    0.24222142
##   lp__                  -471.06811225 2.22269559 -476.1803075 -472.30118929
##                       stats
## parameter                        50%          75%         97.5%
##   alpha                   17.98741547   19.5805739   23.19474712
##   beta_depth               1.54868871    1.7374678    2.10604548
##   sigma                    2.45547168    2.5169902    2.65717904
##   alpha_species[1]         1.40198941    2.6455294    4.94516184
##   alpha_species[2]        -0.60306428    0.4798287    2.67328682
##   alpha_species[3]         0.05278868    1.1711383    3.29504852
##   beta_depth_species[1]   -0.48084061   -0.2850413    0.09701047
##   beta_depth_species[2]    0.16127786    0.3545262    0.75615923
##   beta_depth_species[3]    0.43421919    0.6238175    0.99264102
##   lp__                  -470.69771817 -469.4522035 -467.99115349
```

By taking a look at the traceplots, we can confirm the above, the non hierarchical model showing a better performance at some points during the fitting.

```
traceplot(fit_separate, pars = c("alpha", "beta_depth"))
```

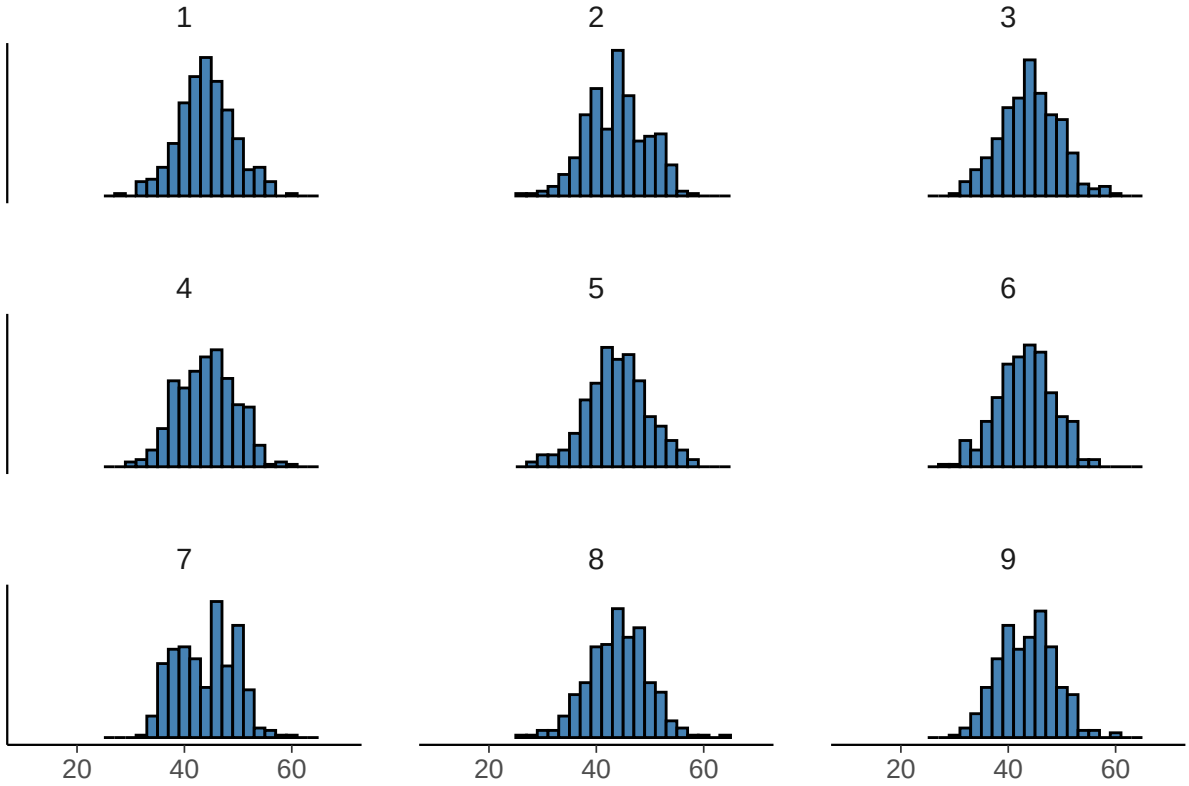



```
traceplot(fit_hierarchical, pars = c("alpha", "beta_depth"))
```



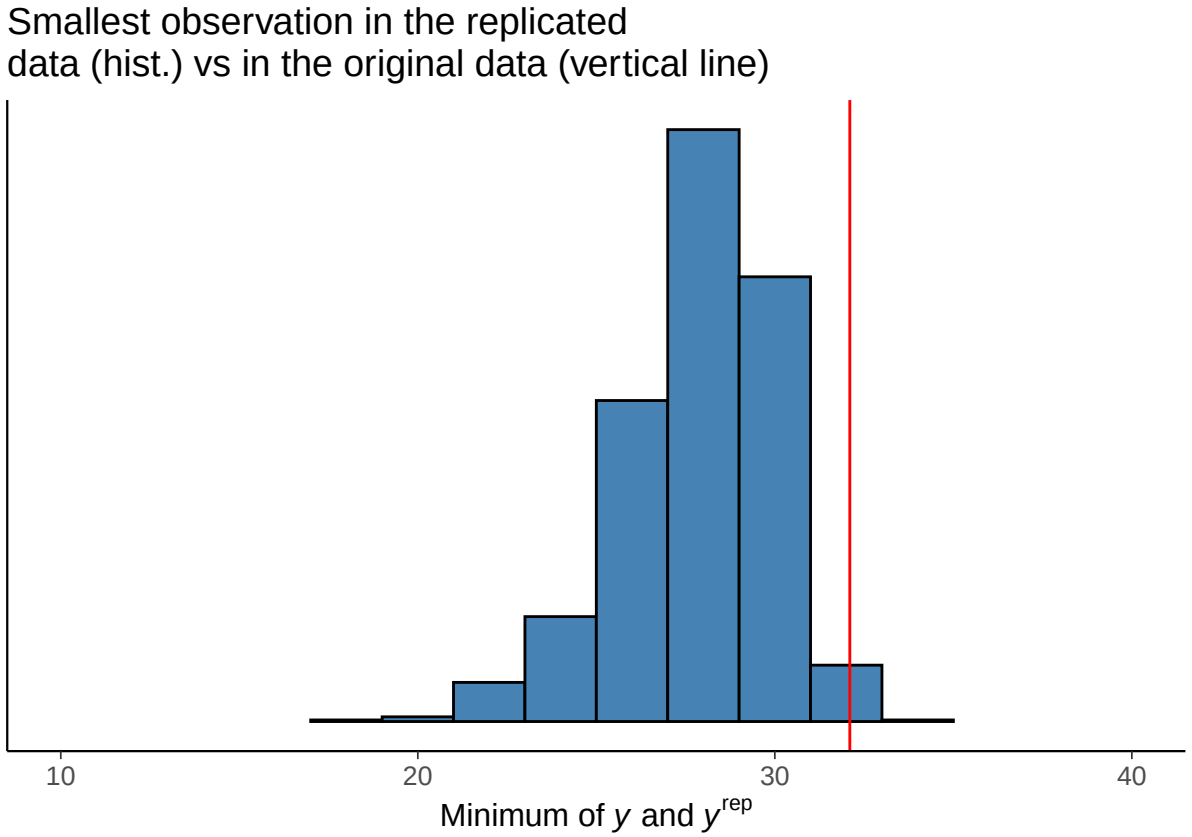
(3) Model Checking

When checking our model, we took 9 random datasets from the PDD, each consisting of the same amount of observations as the original dataset. We can see some discrepancies when looking at the subplots, the even distribution fading away in some cases completely.



When checking the smallest observation in the replicated data (this time x1000) and in the original data, we notice that the positive skew of the replicated data (also the highest minimum amount), coincides with the minimum amount of the original data.

```
yrep1000 <- replicate(1000, rt(n, n-1)*sqrt(1+1/n)*s+my) %>%
  as.data.frame()
# the minimum value over 1000 replicates
minvals <- data.frame(x = sapply(yrep1000, min))
#' Plot test statistic for the data and the replicated data sets
title1 <- 'Smallest observation in the replicated
data (hist.) vs in the original data (vertical line)'
ggplot(data = minvals) +
  geom_histogram(aes(x = x), fill = 'steelblue',
                 color = 'black', binwidth = 2) +
  geom_vline(aes(xintercept = min(x)), data = data.frame(x = y),
             color = 'red') +
  coord_cartesian(xlim = c(10, 40)) +
  labs(x = TeX('Minimum of \\textit{y} and \\textit{y}$^{rep}$'),
       y = '', title = title1) +
  scale_y_continuous(breaks=NULL)
```



(6) Discussion

The performance of a model is really impacted by the quality and quantity of the data, and at the same time on the approache that are being take when preprocessing the data. In some cases, CV might be a really good tool, but it might also lead to overfitting. A simple train/test split might lead to misguiding results (statistically and mathematically great results, but practically biased). This is an ongoing issue in statistical data analysis which has to be handled with care.

(7) Conclusion

Overall, while writing this assignment, we have found at times that the topics were a bit complicated and we should've started working on it earlier. Probably a common mistake between us, students, is underestimating the amount of effort an assignment requires - not only, but also completely going over the lecture materials and trying to fully understand the topic.

As for the assignment itself, the intially adorable penguin topic has become a bit of a headache, especially over the last part of the assignment. But, overall, a nice dataset to work with, pretty simple and very clean (when it comes to EDA). The modelling was a bit more complex, as it was also expected to be.

(8) AI Disclosure

AI has been used in this project for the following: - explanations of topics when unclear - basically creting prompts that would "simplify" some information that we have found complicated at times (especially regarding the posterior predictive analysis) - our stan code was not rendering at one point so we have checked bits of it (where the error would occur) to fix the syntax issue - to add visually pleasing elements to our plots

GitHub

https://github.com/BubuGly/bayest_statistics_TU_Wien_WS23