Hierarchical Bayesian Modeling with `tidymodels`

```r
# Load packages
library(tidyverse)
library(tidymodels)
library(multilevelmod) # install from Github
```

```r
# Grab data and subset to the North Rocky Forest
df <- read_csv("../data/subsets/dat_small.csv")
m333 <- df %>%
  filter(province == "M333")
```

Fitting a model

```r
set.seed(37) # the best number

# HB engine, standard priors
hb_spec <- linear_reg() %>%
  set_engine("stan-glmer",
             prior_aux =  rstanarm::exponential(rate = 1),
             prior = NULL,
             prior_intercept = NULL,
             prior_covariance = rstanarm::decov(shape = 2))

# Fit the model
hb_fit <-
  hb_spec %>%
  fit(BIOLIVE_TPA ~ 1 +  forprob + (1 | subsection), # varying intercepts
      data = m333)

hb_fit
```

```
## parsnip model object
##
## Fit time:  47.2s
## stan_glmer
##  family:       gaussian [identity]
##  formula:      BIOLIVE_TPA ~ 1 + forprob + (1 | subsection)
##  observations: 3003
## ------
##             Median MAD_SD
## (Intercept) -4.7    3.8
## forprob     48.8    2.9
##
## Auxiliary parameter(s):
##       Median MAD_SD
## sigma 35.2    0.5
##
## Error terms:
##  Groups     Name        Std.Dev.
##  subsection (Intercept) 12
##  Residual               35
## Num. levels: subsection 20
##
## ------
## * For help interpreting the printed output see ?print.stanreg
## * For info on the priors used see ?prior_summary.stanreg
```

```
hb_stan <- hb_fit$fit$stanfit
print(hb_stan)
```

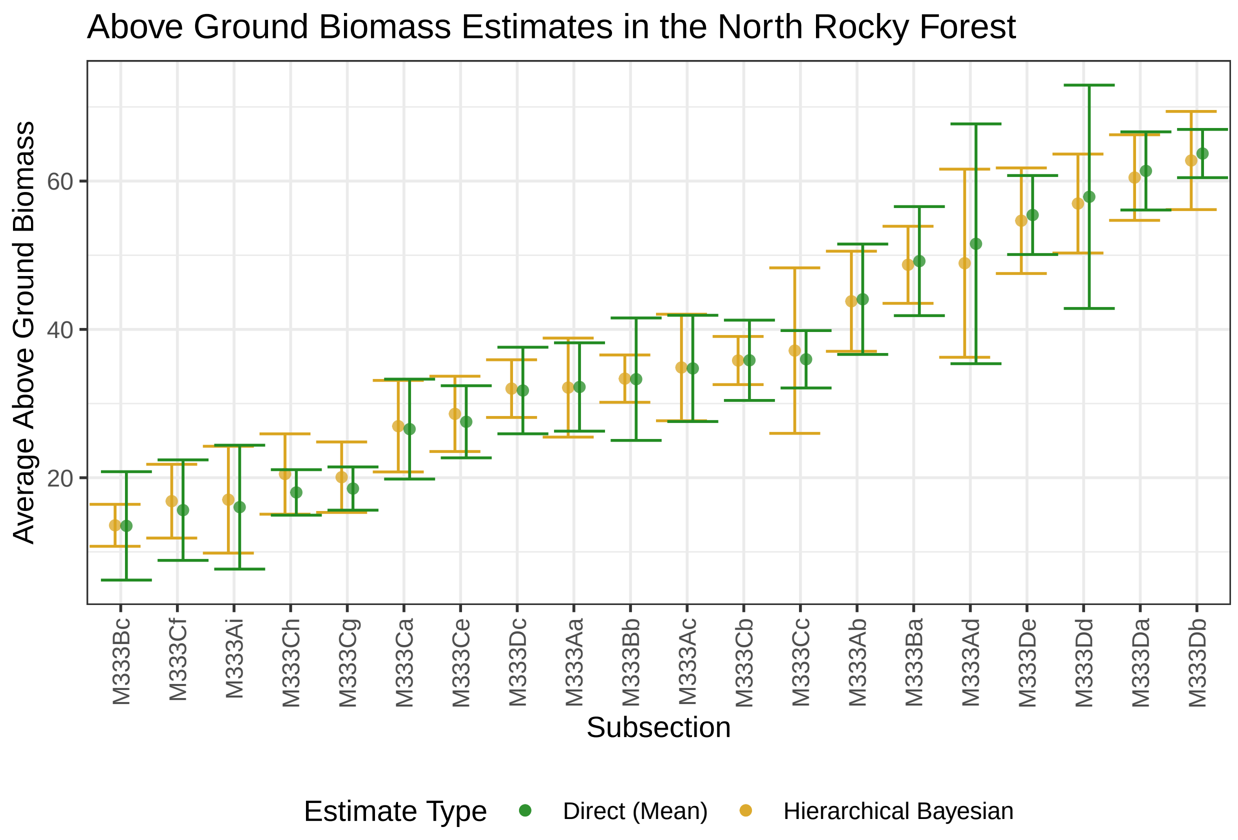
```
## Inference for Stan model: continuous.
## 4 chains, each with iter=2000; warmup=1000; thin=1;
## post-warmup draws per chain=1000, total post-warmup draws=4000.
##
##                                                   mean se_mean    sd      2.5%
## (Intercept)                                      -4.65    0.12  3.76    -11.88
## forprob                                          48.80    0.05  2.84     43.26
## b[(Intercept) subsection:M333Aa]                  1.33    0.12  4.19     -6.56
## b[(Intercept) subsection:M333Ab]                  2.42    0.11  4.55     -6.57
## b[(Intercept) subsection:M333Ac]                 -2.16    0.11  4.22    -10.56
## b[(Intercept) subsection:M333Ad]                  7.05    0.14  6.28     -4.85
## b[(Intercept) subsection:M333Ai]                 -4.00    0.12  5.71    -15.24
## b[(Intercept) subsection:M333Ba]                  8.29    0.12  3.57      1.61
## b[(Intercept) subsection:M333Bb]                 -6.30    0.12  3.35    -12.84
## b[(Intercept) subsection:M333Bc]                 -2.83    0.12  3.68    -10.15
## b[(Intercept) subsection:M333Ca]                 -4.72    0.12  4.59    -13.79
## b[(Intercept) subsection:M333Cb]                 -2.92    0.11  3.42     -9.63
## b[(Intercept) subsection:M333Cc]                 -3.13    0.12  6.14    -15.24
## b[(Intercept) subsection:M333Ce]                -10.28    0.12  4.56    -19.16
## b[(Intercept) subsection:M333Cf]                -10.10    0.12  4.72    -19.43
## b[(Intercept) subsection:M333Cg]                -12.50    0.12  4.68    -21.69
## b[(Intercept) subsection:M333Ch]                -17.21    0.12  4.73    -26.62
## b[(Intercept) subsection:M333Da]                 18.22    0.12  3.63     11.33
## b[(Intercept) subsection:M333Db]                 19.45    0.12  3.62     12.63
## b[(Intercept) subsection:M333Dc]                 -8.06    0.12  3.66    -15.06
## b[(Intercept) subsection:M333Dd]                 14.33    0.12  3.76      7.29
## b[(Intercept) subsection:M333De]                 11.62    0.12  3.83      4.26
## b[(Intercept) subsection:_NEW_subsection]        -0.08    0.19 11.93    -23.81
## sigma                                            35.17    0.01  0.45     34.31
## Sigma[subsection:(Intercept),(Intercept)]       145.38    2.00 56.22     69.42
## mean_PPD                                         38.57    0.01  0.91     36.77
## log-posterior                                -15033.76    0.17  4.68 -15043.54
##                                                    25%       50%       75%
## (Intercept)                                      -7.18     -4.74     -2.12
## forprob                                          46.86     48.83     50.73
## b[(Intercept) subsection:M333Aa]                 -1.41      1.36      4.08
## b[(Intercept) subsection:M333Ab]                 -0.66      2.37      5.40
## b[(Intercept) subsection:M333Ac]                 -4.92     -2.17      0.63
## b[(Intercept) subsection:M333Ad]                  2.77      7.01     10.93
## b[(Intercept) subsection:M333Ai]                 -7.82     -3.98     -0.19
## b[(Intercept) subsection:M333Ba]                  5.86      8.15     10.68
## b[(Intercept) subsection:M333Bb]                 -8.50     -6.31     -4.04
## b[(Intercept) subsection:M333Bc]                 -5.32     -2.80     -0.34
## b[(Intercept) subsection:M333Ca]              -7.81     -4.72     -1.63
## b[(Intercept) subsection:M333Cb]              -5.23     -2.96     -0.73
## b[(Intercept) subsection:M333Cc]              -7.30     -3.08      1.00
## b[(Intercept) subsection:M333Ce]             -13.36    -10.27     -7.19
## b[(Intercept) subsection:M333Cf]             -13.23    -10.06     -6.86
## b[(Intercept) subsection:M333Cg]             -15.52    -12.44     -9.37
## b[(Intercept) subsection:M333Ch]             -20.36    -17.11    -13.95
## b[(Intercept) subsection:M333Da]              15.82     18.19     20.63
## b[(Intercept) subsection:M333Db]              17.05     19.45     21.83
## b[(Intercept) subsection:M333Dc]             -10.52     -8.10     -5.63
## b[(Intercept) subsection:M333Dd]              11.81     14.29     16.70
## b[(Intercept) subsection:M333De]               9.07     11.57     14.08
## b[(Intercept) subsection:_NEW_subsection]     -7.86     -0.15      7.17
## sigma                                         34.87     35.17     35.48
## Sigma[subsection:(Intercept),(Intercept)]    105.18    133.77    174.41
## mean_PPD                                      37.98     38.58     39.19
## log-posterior                             -15036.85 -15033.45 -15030.44
##                                               97.5% n_eff Rhat
## (Intercept)                                    2.84   913 1.00
## forprob                                       54.46  3845 1.00
## b[(Intercept) subsection:M333Aa]               9.30  1277 1.00
## b[(Intercept) subsection:M333Ab]              11.65  1676 1.00
## b[(Intercept) subsection:M333Ac]               6.20  1506 1.00
## b[(Intercept) subsection:M333Ad]              20.05  2147 1.00
## b[(Intercept) subsection:M333Ai]               7.43  2192 1.00
## b[(Intercept) subsection:M333Ba]              15.55   958 1.00
## b[(Intercept) subsection:M333Bb]               0.23   798 1.00
## b[(Intercept) subsection:M333Bc]               4.20   952 1.00
## b[(Intercept) subsection:M333Ca]               4.10  1499 1.00
## b[(Intercept) subsection:M333Cb]               3.93   898 1.00
## b[(Intercept) subsection:M333Cc]               8.97  2643 1.00
## b[(Intercept) subsection:M333Ce]              -1.41  1520 1.00
## b[(Intercept) subsection:M333Cf]              -1.13  1500 1.00
## b[(Intercept) subsection:M333Cg]              -3.50  1501 1.00
## b[(Intercept) subsection:M333Ch]              -8.32  1565 1.00
## b[(Intercept) subsection:M333Da]              25.31   968 1.00
## b[(Intercept) subsection:M333Db]              26.68   966 1.00
## b[(Intercept) subsection:M333Dc]              -0.84   884 1.00
## b[(Intercept) subsection:M333Dd]              21.98  1041 1.00
## b[(Intercept) subsection:M333De]              19.10  1101 1.00
## b[(Intercept) subsection:_NEW_subsection]     24.30  3872 1.00
## sigma                                         36.09  4129 1.00
## Sigma[subsection:(Intercept),(Intercept)]    285.27   788 1.01
## mean_PPD                                      40.33  3991 1.00
## log-posterior                             -15025.48   760 1.01
##
## Samples were drawn using NUTS(diag_e) at Sun Oct 25 09:47:20 2020.
## For each parameter, n_eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor on split chains (at
## convergence, Rhat=1).
```

```
# Let's plot the means
hb_fit_df <- data.frame(
  fitted = hb_fit$fit$fitted.values,
  true = m333$BIOLIVE_TPA,
  subsection = m333$subsection
  )

# Load sds
conf <- read_csv("sd_df.csv")
```

```
## Warning: Missing column names filled in: 'X1' [1]
```

```
# In this plot, I would like error bars on the HB estimate. however, since the model fits
# to the plot level and then I summarized these means, I am not sure what a true error
# bar would look like.
hb_fit_df %>%
  group_by(subsection) %>%
  summarize(mean_fit = mean(fitted),
            mean_true = mean(true)) %>%
  left_join(conf, by = c("subsection" = "id")) %>%
  mutate(subsection = fct_reorder(subsection, mean_true)) %>%
  ggplot(aes(x = subsection,
             y = mean_fit)) +
  geom_point(aes(color = "goldenrod"),
             alpha = 0.75,
             position = position_nudge(x = -0.1)) +
  geom_point(
    aes(y = mean_true, color = "forestgreen"),
    alpha = 0.75,
    position = position_nudge(x = 0.1)
  ) +
  geom_errorbar(
    mapping = aes(
      ymin = mean_fit - 1.96 * bootstrap_sd,
      ymax = mean_fit + 1.96 * bootstrap_sd
    ),
    position = position_nudge(x = -0.1),
    color = "goldenrod"
  ) +
  geom_errorbar(
    mapping = aes(
      ymin = mean_true - 1.96 * direct_sd,
      ymax = mean_true + 1.96 * direct_sd
    ),
    position = position_nudge(x = 0.1),
    color = "forestgreen"
  ) +
  theme_bw() +
  theme(axis.text.x = element_text(
    angle = 90,
    vjust = 0.5,
    hjust = 1
    ),
    legend.position = "bottom") +
  labs(
    x = "Subsection",
    y = "Average Above Ground Biomass",
    title = "Above Ground Biomass Estimates in the North Rocky Forest"
  ) +
  scale_color_manual(name = 'Estimate Type',
                     values =c('goldenrod'='goldenrod',
                               'forestgreen'='forestgreen'),
                     labels = c('Direct (Mean)','Hierarchical Bayesian'),
                     guide = "legend")
```



```
# m333 %>%
#   group_by(subsection) %>%
#   arrange(subsection) %>%
#   summarize(n = n()) %>% View()
```

Bootstrap

```
set.seed(13)
m333_nested <- m333 %>%
  mutate(id = subsection) %>%
  group_by(subsection) %>%
  nest()
means <- data.frame()
for (j in 1:length(unique(m333$subsection))) {
```

```r
  for (i in 1:1000) {
    means[j, i] <- sample_n(
      m333_nested[[2]][[j]],
      size = length(m333_nested[[2]][[j]]$BIOLIVE_TPA),
      replace = TRUE
    ) %>%
      summarize(mean = mean(BIOLIVE_TPA)) %>%
      select(mean)
  }
}
library(matrixStats)
sds <- means %>%
  as.matrix() %>%
  rowSds()


ordered_subsections <- m333 %>%
  arrange(subsection) %>%
  select(subsection) %>%
  unique() %>%
  unlist()

set.seed(utf8ToInt("this is gonna take awhile"))
boots <- list()
fit <- list()
mean_df <- list()
final <- data.frame()

for(i in 1:1000){
for(j in 1:length(unique(m333$subsection))) {
    boots[[j]] <- sample_n(
      m333_nested[[2]][[j]],
      size = length(m333_nested[[2]][[j]]$BIOLIVE_TPA),
      replace = TRUE
    )
    boots_df <- bind_rows(boots)
    }

    fit[[i]] <- hb_spec %>%
      fit(BIOLIVE_TPA ~ 1 +  forprob + (1 | id),
          data = boots_df)

    mean_df[[i]] <- data.frame(fitted = fit[[i]]$fit$fitted.values,
                               subsection = boots_df$id)

    final[1:20, i] <- mean_df[[i]] %>%
      group_by(subsection) %>%
      summarize(mean = mean(fitted)) %>%
      select(mean)

    if (i %% 50 == 1) {
      print(i)
    }
```

```r
}

boot_sds <- final %>%
  as.matrix() %>%
  rowSds()

sd_df <- data.frame(
  id = ordered_subsections,
  direct_sd = sds,
  bootstrap_sd = boot_sds
)

saveRDS(sd_df, file = "sd_df.rds")
```

Fit freq model, split data, compare test MSEs

```r
set.seed(1)
m333_test <- m333 %>%
  sample_frac(0.25)
m333_train <- m333 %>%
  anti_join(m333_test)

freq_spec <- linear_reg() %>%
  set_engine("lmer")

freq_fit <- freq_spec %>%
    fit(BIOLIVE_TPA ~ 1 +  forprob + (1 | subsection), # varying intercepts
      data = m333_train)

hb_fit_train <-
  hb_spec %>%
  fit(BIOLIVE_TPA ~ 1 +  forprob + (1 | subsection), # varying intercepts
      data = m333_train)



results_test <- hb_fit_train %>%
  predict(new_data = m333_test) %>%
  mutate(
    truth = m333_test$BIOLIVE_TPA,
    model = "hb"
  ) %>%
  bind_rows(
    freq_fit %>%
  predict(new_data = m333_test) %>%
  mutate(
    truth = m333_test$BIOLIVE_TPA,
    model = "freq"
  )
  )

results_test %>%
  group_by(model) %>%
  rmse(truth = truth,
       estimate = .pred)
```

```
## # A tibble: 2 x 4
##   model .metric .estimator .estimate
##   <chr> <chr>   <chr>          <dbl>
## 1 freq  rmse    standard        34.6
## 2 hb    rmse    standard        34.6
```

```
# tidyposterior::perf_mod() will likely be helpful comparing models
```