Model-Averaged-Bootstrap-CI-(hopefully)-Final

Cole

October 1, 2019

Step 1: Load Data, Run Full Models & Dredge

```
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 3.5.3
```

```
## -- Attaching packages ---------------------------------------------------------------
-------- tidyverse 1.2.1 --
```

```
## v ggplot2 3.1.1     v purrr   0.3.2
## v tibble  2.1.3     v dplyr   0.8.1
## v tidyr   0.8.3     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.4.0
```

```
## Warning: package 'ggplot2' was built under R version 3.5.3
```

```
## Warning: package 'tibble' was built under R version 3.5.3
```

```
## Warning: package 'tidyr' was built under R version 3.5.3
```

```
## Warning: package 'readr' was built under R version 3.5.3
```

```
## Warning: package 'purrr' was built under R version 3.5.3
```

```
## Warning: package 'dplyr' was built under R version 3.5.3
```

```
## Warning: package 'stringr' was built under R version 3.5.3
```

```
## Warning: package 'forcats' was built under R version 3.5.3
```

```
## -- Conflicts ------------------------------------------------------------------------
-- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(glmmTMB)
```

```
## Warning in checkMatrixPackageVersion(): Package version inconsistency detected.
## TMB was built with Matrix version 1.2.14
## Current Matrix version is 1.2.17
## Please re-install 'TMB' from source using install.packages('TMB', type = 'source') or ask CRA
N for a binary version of 'TMB' matching CRAN's 'Matrix' package
```

```
library(ggeffects)
```

```
## Warning: package 'ggeffects' was built under R version 3.5.3
```

```
library(DHARMa)
```

```
## Warning: package 'DHARMa' was built under R version 3.5.3
```

```
library(MuMIn)
```

```
## Warning: package 'MuMIn' was built under R version 3.5.3
```

```
library(cowplot)
```

```
## Warning: package 'cowplot' was built under R version 3.5.3
```

```
##
## ************************************************************
```

```
## Note: As of version 1.0.0, cowplot does not change the
```

```
##   default ggplot2 theme anymore. To recover the previous
```

```
##   behavior, execute:
##   theme_set(theme_cowplot())
```

```
## ************************************************************
```

```
##
## Attaching package: 'cowplot'
```

```
## The following object is masked from 'package:ggeffects':
##
##     get_title
```

```
library(AICcmodavg)
```

```
## Warning: package 'AICcmodavg' was built under R version 3.5.3
```

```
##
## Attaching package: 'AICcmodavg'
```

```
## The following objects are masked from 'package:MuMIn':
##
##     AICc, DIC, importance
```

```
library(latexpdf)
```

```
## Warning: package 'latexpdf' was built under R version 3.5.2
```

```
mainlice <- read_csv("C:/Users/brookson/Salmon_Work/SalmonWork-master/Hakai_lice_data_CB_edits.csv")

#make vars into factors
mainlice$year <- as.factor(mainlice$year);mainlice$collection <- as.factor(mainlice$collection)

#get the necessary model selection things:
lepmod.crossed <- glmmTMB(all.leps ~ spp * site.region + spp * year +
                            site.region * year + (1 | collection),
                          data = mainlice, family=nbinom2)
calmod.crossed <- glmmTMB(all.cal ~ spp * site.region + spp * year +
                            site.region * year + (1 | collection),
                          data = mainlice, family=nbinom2)
lepmod.crossed_dredge = MuMIn::dredge(lepmod.crossed, subset = (`cond(site.region)` && `cond(year)`))
calmod.crossed_dredge = MuMIn::dredge(calmod.crossed, subset = (`cond(site.region)` && `cond(year)`))
```

Step 2: Get the Data Into Appropriate Subgroups Based on Hierarchical Structure

```
#so the goal here is to bootstrap the data (parametric) by resampling the hierarchical levels, t
hen
#run the model averaging process with the new data and use that to get our model-averaged CI's.
 Essentially,
#we're wrapping our esitmator of mu in a function and bootstrapping it

bootlice =  mainlice %>%
  select(all.cal, all.leps, spp, site.region, collection, year, ufn)

sock2015D <- bootlice %>% filter(spp == 'SO' & year == '2015' & site.region == 'D')
sock2015J <- bootlice %>% filter(spp == 'SO' & year == '2015' & site.region == 'J')
sock2016D <- bootlice %>% filter(spp == 'SO' & year == '2016' & site.region == 'D')
sock2016J <- bootlice %>% filter(spp == 'SO' & year == '2016' & site.region == 'J')
sock2017D <- bootlice %>% filter(spp == 'SO' & year == '2017' & site.region == 'D')
sock2017J <- bootlice %>% filter(spp == 'SO' & year == '2017' & site.region == 'J')
sock2018D <- bootlice %>% filter(spp == 'SO' & year == '2018' & site.region == 'D')
sock2018J <- bootlice %>% filter(spp == 'SO' & year == '2018' & site.region == 'J')

chum2015D <- bootlice %>% filter(spp == 'CU' & year == '2015' & site.region == 'D')
chum2015J <- bootlice %>% filter(spp == 'CU' & year == '2015' & site.region == 'J')
chum2016D <- bootlice %>% filter(spp == 'CU' & year == '2016' & site.region == 'D')
chum2016J <- bootlice %>% filter(spp == 'CU' & year == '2016' & site.region == 'J')
chum2017D <- bootlice %>% filter(spp == 'CU' & year == '2017' & site.region == 'D')
chum2017J <- bootlice %>% filter(spp == 'CU' & year == '2017' & site.region == 'J')
chum2018D <- bootlice %>% filter(spp == 'CU' & year == '2018' & site.region == 'D')
chum2018J <- bootlice %>% filter(spp == 'CU' & year == '2018' & site.region == 'J')

pink2015D <- bootlice %>% filter(spp == 'PI' & year == '2015' & site.region == 'D')
pink2015J <- bootlice %>% filter(spp == 'PI' & year == '2015' & site.region == 'J')
pink2016D <- bootlice %>% filter(spp == 'PI' & year == '2016' & site.region == 'D')
pink2016J <- bootlice %>% filter(spp == 'PI' & year == '2016' & site.region == 'J')
pink2017D <- bootlice %>% filter(spp == 'PI' & year == '2017' & site.region == 'D')
pink2017J <- bootlice %>% filter(spp == 'PI' & year == '2017' & site.region == 'J')
pink2018D <- bootlice %>% filter(spp == 'PI' & year == '2018' & site.region == 'D')
pink2018J <- bootlice %>% filter(spp == 'PI' & year == '2018' & site.region == 'J')
```

Step 3: Bootstrap! Resample the Data, Then Run the Models, Model-Average, and Store the Result

```
bootintervalcal = matrix(nrow = 24, ncol = 1000)
bootintervallep = matrix(nrow = 24, ncol = 1000)


pb = txtProgressBar(min = 0, max = 1000, initial = 0)
start_time <- Sys.time()
for(i in 1:1000) {
  sock2015Dboot = matrix(nrow = nrow(sock2015D), ncol = 7)
  n = 1
  for(k in unique(sock2015D$collection)) { #for each collection
    #resample the observations in the collection with replacement for the number of rows in that
given collection
    res <- as.matrix(sock2015D %>%
                       group_by(collection) %>%
                       filter(collection == k) %>%
                       sample_n(., n(), replace = TRUE))
    rows = c(n:(n+nrow(res)-1))
    sock2015Dboot[rows,] = res
    n = n + nrow(res)
  }
  sock2015Jboot = matrix(nrow = nrow(sock2015J), ncol = 7)
  n = 1
  for(k in unique(sock2015J$collection)) { #for each collection
    #resample the observations in the collection with replacement for the number of rows in that
given collection
    res <- as.matrix(sock2015J %>%
                       group_by(collection) %>%
                       filter(collection == k) %>%
                       sample_n(., n(), replace = TRUE))
    rows = c(n:(n+nrow(res)-1))
    sock2015Jboot[rows,] = res
    n = n + nrow(res)
  }
  sock2016Dboot = matrix(nrow = nrow(sock2016D), ncol = 7)
  n = 1
  for(k in unique(sock2016D$collection)) { #for each collection
    #resample the observations in the collection with replacement for the number of rows in that
given collection
    res <- as.matrix(sock2016D %>%
                       group_by(collection) %>%
                       filter(collection == k) %>%
                       sample_n(., n(), replace = TRUE))
    rows = c(n:(n+nrow(res)-1))
    sock2016Dboot[rows,] = res
    n = n + nrow(res)
  }
  sock2016Jboot = matrix(nrow = nrow(sock2016J), ncol = 7)
  n = 1
  for(k in unique(sock2016J$collection)) { #for each collection
    #resample the observations in the collection with replacement for the number of rows in that
given collection
    res <- as.matrix(sock2016J %>%
                       group_by(collection) %>%
```

```
                      filter(collection == k) %>%
                      sample_n(., n(), replace = TRUE))
    rows = c(n:(n+nrow(res)-1))
    sock2016Jboot[rows,] = res
    n = n + nrow(res)
  }
  sock2017Dboot = matrix(nrow = nrow(sock2017D), ncol = 7)
  n = 1
  for(k in unique(sock2017D$collection)) { #for each collection
    #resample the observations in the collection with replacement for the number of rows in that
given collection
    res <- as.matrix(sock2017D %>%
                      group_by(collection) %>%
                      filter(collection == k) %>%
                      sample_n(., n(), replace = TRUE))
    rows = c(n:(n+nrow(res)-1))
    sock2017Dboot[rows,] = res
    n = n + nrow(res)
  }
  sock2017Jboot = matrix(nrow = nrow(sock2017J), ncol = 7)
  n = 1
  for(k in unique(sock2017J$collection)) { #for each collection
    #resample the observations in the collection with replacement for the number of rows in that
given collection
    res <- as.matrix(sock2017J %>%
                      group_by(collection) %>%
                      filter(collection == k) %>%
                      sample_n(., n(), replace = TRUE))
    rows = c(n:(n+nrow(res)-1))
    sock2017Jboot[rows,] = res
    n = n + nrow(res)
  }
  sock2018Dboot = matrix(nrow = nrow(sock2018D), ncol = 7)
  n = 1
  for(k in unique(sock2018D$collection)) { #for each collection
    #resample the observations in the collection with replacement for the number of rows in that
given collection
    res <- as.matrix(sock2018D %>%
                      group_by(collection) %>%
                      filter(collection == k) %>%
                      sample_n(., n(), replace = TRUE))
    rows = c(n:(n+nrow(res)-1))
    sock2018Dboot[rows,] = res
    n = n + nrow(res)
  }
  sock2018Jboot = matrix(nrow = nrow(sock2018J), ncol = 7)
  n = 1
  for(k in unique(sock2018J$collection)) { #for each collection
    #resample the observations in the collection with replacement for the number of rows in that
given collection
    res <- as.matrix(sock2018J %>%
                      group_by(collection) %>%
                      filter(collection == k) %>%
                      sample_n(., n(), replace = TRUE))
```

```
    rows = c(n:(n+nrow(res)-1))
    sock2018Jboot[rows,] = res
    n = n + nrow(res)
  }

  #chum

  chum2015Dboot = matrix(nrow = nrow(chum2015D), ncol = 7)
  n = 1
  for(k in unique(chum2015D$collection)) { #for each collection
    #resample the observations in the collection with replacement for the number of rows in that
given collection
    res <- as.matrix(chum2015D %>%
                       group_by(collection) %>%
                       filter(collection == k) %>%
                       sample_n(., n(), replace = TRUE))
    rows = c(n:(n+nrow(res)-1))
    chum2015Dboot[rows,] = res
    n = n + nrow(res)
  }
  chum2015Jboot = matrix(nrow = nrow(chum2015J), ncol = 7)
  n = 1
  for(k in unique(chum2015J$collection)) { #for each collection
    #resample the observations in the collection with replacement for the number of rows in that
given collection
    res <- as.matrix(chum2015J %>%
                       group_by(collection) %>%
                       filter(collection == k) %>%
                       sample_n(., n(), replace = TRUE))
    rows = c(n:(n+nrow(res)-1))
    chum2015Jboot[rows,] = res
    n = n + nrow(res)
  }
  chum2016Dboot = matrix(nrow = nrow(chum2016D), ncol = 7)
  n = 1
  for(k in unique(chum2016D$collection)) { #for each collection
    #resample the observations in the collection with replacement for the number of rows in that
given collection
    res <- as.matrix(chum2016D %>%
                       group_by(collection) %>%
                       filter(collection == k) %>%
                       sample_n(., n(), replace = TRUE))
    rows = c(n:(n+nrow(res)-1))
    chum2016Dboot[rows,] = res
    n = n + nrow(res)
  }
  chum2016Jboot = matrix(nrow = nrow(chum2016J), ncol = 7)
  n = 1
  for(k in unique(chum2016J$collection)) { #for each collection
    #resample the observations in the collection with replacement for the number of rows in that
given collection
    res <- as.matrix(chum2016J %>%
                       group_by(collection) %>%
                       filter(collection == k) %>%
```

```
                      sample_n(., n(), replace = TRUE))
    rows = c(n:(n+nrow(res)-1))
    chum2016Jboot[rows,] = res
    n = n + nrow(res)
  }
  chum2017Dboot = matrix(nrow = nrow(chum2017D), ncol = 7)
  n = 1
  for(k in unique(chum2017D$collection)) { #for each collection
    #resample the observations in the collection with replacement for the number of rows in that
given collection
    res <- as.matrix(chum2017D %>%
                      group_by(collection) %>%
                      filter(collection == k) %>%
                      sample_n(., n(), replace = TRUE))
    rows = c(n:(n+nrow(res)-1))
    chum2017Dboot[rows,] = res
    n = n + nrow(res)
  }
  chum2017Jboot = matrix(nrow = nrow(chum2017J), ncol = 7)
  n = 1
  for(k in unique(chum2017J$collection)) { #for each collection
    #resample the observations in the collection with replacement for the number of rows in that
given collection
    res <- as.matrix(chum2017J %>%
                      group_by(collection) %>%
                      filter(collection == k) %>%
                      sample_n(., n(), replace = TRUE))
    rows = c(n:(n+nrow(res)-1))
    chum2017Jboot[rows,] = res
    n = n + nrow(res)
  }
  chum2018Dboot = matrix(nrow = nrow(chum2018D), ncol = 7)
  n = 1
  for(k in unique(chum2018D$collection)) { #for each collection
    #resample the observations in the collection with replacement for the number of rows in that
given collection
    res <- as.matrix(chum2018D %>%
                      group_by(collection) %>%
                      filter(collection == k) %>%
                      sample_n(., n(), replace = TRUE))
    rows = c(n:(n+nrow(res)-1))
    chum2018Dboot[rows,] = res
    n = n + nrow(res)
  }
  chum2018Jboot = matrix(nrow = nrow(chum2018J), ncol = 7)
  n = 1
  for(k in unique(chum2018J$collection)) { #for each collection
    #resample the observations in the collection with replacement for the number of rows in that
given collection
    res <- as.matrix(chum2018J %>%
                      group_by(collection) %>%
                      filter(collection == k) %>%
                      sample_n(., n(), replace = TRUE))
    rows = c(n:(n+nrow(res)-1))
```

```
    chum2018Jboot[rows,] = res
    n = n + nrow(res)
  }

  #pink

  pink2015Dboot = matrix(nrow = nrow(pink2015D), ncol = 7)
  n = 1
  for(k in unique(pink2015D$collection)) { #for each collection
    #resample the observations in the collection with replacement for the number of rows in that
given collection
    res <- as.matrix(pink2015D %>%
                       group_by(collection) %>%
                       filter(collection == k) %>%
                       sample_n(., n(), replace = TRUE))
    rows = c(n:(n+nrow(res)-1))
    pink2015Dboot[rows,] = res
    n = n + nrow(res)
  }
  pink2015Jboot = matrix(nrow = nrow(pink2015J), ncol = 7)
  n = 1
  for(k in unique(pink2015J$collection)) { #for each collection
    #resample the observations in the collection with replacement for the number of rows in that
given collection
    res <- as.matrix(pink2015J %>%
                       group_by(collection) %>%
                       filter(collection == k) %>%
                       sample_n(., n(), replace = TRUE))
    rows = c(n:(n+nrow(res)-1))
    pink2015Jboot[rows,] = res
    n = n + nrow(res)
  }
  pink2016Dboot = matrix(nrow = nrow(pink2016D), ncol = 7)
  n = 1
  for(k in unique(pink2016D$collection)) { #for each collection
    #resample the observations in the collection with replacement for the number of rows in that
given collection
    res <- as.matrix(pink2016D %>%
                       group_by(collection) %>%
                       filter(collection == k) %>%
                       sample_n(., n(), replace = TRUE))
    rows = c(n:(n+nrow(res)-1))
    pink2016Dboot[rows,] = res
    n = n + nrow(res)
  }
  pink2016Jboot = matrix(nrow = nrow(pink2016J), ncol = 7)
  n = 1
  for(k in unique(pink2016J$collection)) { #for each collection
    #resample the observations in the collection with replacement for the number of rows in that
given collection
    res <- as.matrix(pink2016J %>%
                       group_by(collection) %>%
                       filter(collection == k) %>%
                       sample_n(., n(), replace = TRUE))
```

```
    rows = c(n:(n+nrow(res)-1))
    pink2016Jboot[rows,] = res
    n = n + nrow(res)
  }
  pink2017Dboot = matrix(nrow = nrow(pink2017D), ncol = 7)
  n = 1
  for(k in unique(pink2017D$collection)) { #for each collection
    #resample the observations in the collection with replacement for the number of rows in that
given collection
    res <- as.matrix(pink2017D %>%
                       group_by(collection) %>%
                       filter(collection == k) %>%
                       sample_n(., n(), replace = TRUE))
    rows = c(n:(n+nrow(res)-1))
    pink2017Dboot[rows,] = res
    n = n + nrow(res)
  }
  pink2017Jboot = matrix(nrow = nrow(pink2017J), ncol = 7)
  n = 1
  for(k in unique(pink2017J$collection)) { #for each collection
    #resample the observations in the collection with replacement for the number of rows in that
given collection
    res <- as.matrix(pink2017J %>%
                       group_by(collection) %>%
                       filter(collection == k) %>%
                       sample_n(., n(), replace = TRUE))
    rows = c(n:(n+nrow(res)-1))
    pink2017Jboot[rows,] = res
    n = n + nrow(res)
  }
  pink2018Dboot = matrix(nrow = nrow(pink2018D), ncol = 7)
  n = 1
  for(k in unique(pink2018D$collection)) { #for each collection
    #resample the observations in the collection with replacement for the number of rows in that
given collection
    res <- as.matrix(pink2018D %>%
                       group_by(collection) %>%
                       filter(collection == k) %>%
                       sample_n(., n(), replace = TRUE))
    rows = c(n:(n+nrow(res)-1))
    pink2018Dboot[rows,] = res
    n = n + nrow(res)
  }
  pink2018Jboot = matrix(nrow = nrow(pink2018J), ncol = 7)
  n = 1
  for(k in unique(pink2018J$collection)) { #for each collection
    #resample the observations in the collection with replacement for the number of rows in that
given collection
    res <- as.matrix(pink2018J %>%
                       group_by(collection) %>%
                       filter(collection == k) %>%
                       sample_n(., n(), replace = TRUE))
    rows = c(n:(n+nrow(res)-1))
    pink2018Jboot[rows,] = res
```

```
    n = n + nrow(res)
  }

  #bind the matrices so we can have our resampled dataframe
  bootdata = data.frame(rbind(sock2015Dboot,sock2015Jboot,sock2016Dboot,sock2016Jboot,sock2017Db
oot,sock2017Jboot,sock2018Dboot,sock2018Jboot,
                  chum2015Dboot,chum2015Jboot,chum2016Dboot,chum2016Jboot,chum2017Dboot,chum2017
Jboot,chum2018Dboot,chum2018Jboot,
                  pink2015Dboot,pink2015Jboot,pink2016Dboot,pink2016Jboot,pink2017Dboot,pink2017
Jboot,pink2018Dboot,pink2018Jboot)) %>%
    rename(all.cal = X1, all.leps = X2, spp = X3, site.region = X4, collection = X5, year = X6,
 ufn = X7)
  bootdata$all.cal = as.integer(as.character(bootdata$all.cal))
  bootdata$all.leps = as.integer(as.character(bootdata$all.leps))

  #now run our set of models
  #omited the last two models in each set since their weights were 0.
  lep1 = glmmTMB(all.leps ~ site.region + year + spp +
                   site.region * year +
                   (1 | collection), data = bootdata, family = nbinom2)
  lep2 = glmmTMB(all.leps ~ site.region + year + spp +
                   spp * year + site.region * year +
                   (1 | collection), data = bootdata, family = nbinom2)
  lep3 = glmmTMB(all.leps ~ site.region + year + spp +
                   spp * site.region + site.region * year +
                   (1 | collection), data = bootdata, family = nbinom2)
  lep4 = glmmTMB(all.leps ~ site.region + year + spp +
                   spp * site.region + spp * year + site.region * year +
                   (1 | collection), data = bootdata, family = nbinom2)
  lep5 = glmmTMB(all.leps ~ site.region + year + spp +
                   (1 | collection), data = bootdata, family = nbinom2)
  lep6 = glmmTMB(all.leps ~ site.region + year + spp +
                   spp * year +
                   (1 | collection), data = bootdata, family = nbinom2)
  lep7 = glmmTMB(all.leps ~ site.region + year + spp +
                   spp * site.region +
                   (1 | collection), data = bootdata, family = nbinom2)
  lep8 = glmmTMB(all.leps ~ site.region + year + spp +
                   spp * site.region + spp * year +
                   (1 | collection), data = bootdata, family = nbinom2)

  cal1 = glmmTMB(all.cal ~ site.region + year + spp +
                   site.region * year + site.region * spp +
                   (1 | collection), data = bootdata, family = nbinom2)
  cal2 = glmmTMB(all.cal ~ site.region + year + spp +
                   site.region * year +
                   (1 | collection), data = bootdata, family = nbinom2)
  cal3 = glmmTMB(all.cal ~ site.region + year + spp +
                   spp * site.region + site.region * year + spp * year +
                   (1 | collection), data = bootdata, family = nbinom2)
  cal4 = glmmTMB(all.cal ~ site.region + year + spp +
                   spp * year + site.region * year +
                   (1 | collection), data = bootdata, family = nbinom2)
  cal5 = glmmTMB(all.cal ~ site.region + year + spp +
```

```
                  (1 | collection), data = bootdata, family = nbinom2)
  cal6 = glmmTMB(all.cal ~ site.region + year + spp +
                  site.region * year +
                  (1 | collection), data = bootdata, family = nbinom2)
  cal7 = glmmTMB(all.cal ~ site.region + year + spp +
                  spp * year +
                  (1 | collection), data = bootdata, family = nbinom2)
  cal8 = glmmTMB(all.cal ~ site.region + year + spp +
                  spp * site.region + spp * year +
                  (1 | collection), data = bootdata, family = nbinom2)

  #get the predictions of the estiamtes
  lep1pred <- ggpredict(lep1, terms = c('spp', 'year', 'site.region'))
  lep2pred <- ggpredict(lep2, terms = c('spp', 'year', 'site.region'))
  lep3pred <- ggpredict(lep3, terms = c('spp', 'year', 'site.region'))
  lep4pred <- ggpredict(lep4, terms = c('spp', 'year', 'site.region'))
  lep5pred <- ggpredict(lep5, terms = c('spp', 'year', 'site.region'))
  lep6pred <- ggpredict(lep6, terms = c('spp', 'year', 'site.region'))
  lep7pred <- ggpredict(lep7, terms = c('spp', 'year', 'site.region'))
  lep8pred <- ggpredict(lep8, terms = c('spp', 'year', 'site.region'))

  cal1pred <- ggpredict(cal1, terms = c('spp', 'year', 'site.region'))
  cal2pred <- ggpredict(cal2, terms = c('spp', 'year', 'site.region'))
  cal3pred <- ggpredict(cal3, terms = c('spp', 'year', 'site.region'))
  cal4pred <- ggpredict(cal4, terms = c('spp', 'year', 'site.region'))
  cal5pred <- ggpredict(cal5, terms = c('spp', 'year', 'site.region'))
  cal6pred <- ggpredict(cal6, terms = c('spp', 'year', 'site.region'))
  cal7pred <- ggpredict(cal7, terms = c('spp', 'year', 'site.region'))
  cal8pred <- ggpredict(cal8, terms = c('spp', 'year', 'site.region'))

  ###start by getting them all in one dataframe with the weights

  #pull the predicted values from each one
  lepallpred = data.frame(cbind(lep1pred$predicted, lep2pred$predicted, lep3pred$predicted, lep4
pred$predicted,
                                lep5pred$predicted, lep6pred$predicted, lep7pred$predicted, lep8
pred$predicted)) %>%
    rename(lep1 = X1, lep2 = X2, lep3 = X3, lep4 = X4, lep5 = X5, lep6 = X6, lep7 = X7, lep8 = X
8)
  calallpred = data.frame(cbind(cal1pred$predicted, cal2pred$predicted, cal3pred$predicted, cal4
pred$predicted,
                                cal5pred$predicted, cal6pred$predicted, cal7pred$predicted, cal8
pred$predicted)) %>%
    rename(cal1 = X1, cal2 = X2, cal3 = X3, cal4 = X4, cal5 = X5, cal6 = X6, cal7 = X7, cal8 = X
8)

  #add the weights from the model selection object
  lepallpred = lepallpred %>%
    mutate(w1 = rep(lepmod.crossed_dredge$weight[1], nrow(lepallpred)),
           w2 = rep(lepmod.crossed_dredge$weight[2], nrow(lepallpred)),
           w3 = rep(lepmod.crossed_dredge$weight[3], nrow(lepallpred)),
           w4 = rep(lepmod.crossed_dredge$weight[4], nrow(lepallpred)),
           w5 = rep(lepmod.crossed_dredge$weight[5], nrow(lepallpred)),
           w6 = rep(lepmod.crossed_dredge$weight[6], nrow(lepallpred)),
```

```
          w7 = rep(lepmod.crossed_dredge$weight[7], nrow(lepallpred)),
          w8 = rep(lepmod.crossed_dredge$weight[8], nrow(lepallpred)))

  calallpred = calallpred %>%
    mutate(w1 = rep(calmod.crossed_dredge$weight[1], nrow(calallpred)),
           w2 = rep(calmod.crossed_dredge$weight[2], nrow(calallpred)),
           w3 = rep(calmod.crossed_dredge$weight[3], nrow(calallpred)),
           w4 = rep(calmod.crossed_dredge$weight[4], nrow(calallpred)),
           w5 = rep(calmod.crossed_dredge$weight[5], nrow(calallpred)),
           w6 = rep(calmod.crossed_dredge$weight[6], nrow(calallpred)),
           w7 = rep(calmod.crossed_dredge$weight[7], nrow(calallpred)),
           w8 = rep(calmod.crossed_dredge$weight[8], nrow(calallpred)))

  #now make averaged predictions!
  lepallpred = lepallpred %>%
    mutate(lep1w = lep1*w1, lep2w = lep2*w2, lep3w = lep3*w3, lep4w = lep4*w4, lep5w = lep5*w5,
 lep6w = lep6*w6, lep7w = lep7*w7, lep8w = lep8*w8) %>%
    mutate(avg = lep1w + lep2w + lep3w + lep4w + lep5w + lep6w + lep7w + lep8w)

  calallpred = calallpred %>%
    mutate(cal1w = cal1*w1, cal2w = cal2*w2, cal3w = cal3*w3, cal4w = cal4*w4, cal5w = cal5*w5,
 cal6w = cal6*w6, cal7w = cal7*w7, cal8w = cal8*w8) %>%
    mutate(avg = cal1w + cal2w + cal3w + cal4w + cal5w + cal6w + cal7w + cal8w)

  #keep just the averaged predictions and the relevant grouping info
  lepavgpred = lepallpred %>%
    select(avg) %>%
    mutate(sal = lep1pred$x, reg = lep1pred$facet, yr = lep1pred$group)
  lepavgpred$sal = factor(lepavgpred$sal, levels = c(1, 2, 3), labels = c('CU', 'PI', 'SO'))

  calavgpred = calallpred %>%
    select(avg) %>%
    mutate(sal = cal1pred$x, reg = cal1pred$facet, yr = cal1pred$group)
  calavgpred$sal = factor(calavgpred$sal, levels = c(1, 2, 3), labels = c('CU', 'PI', 'SO'))

  bootintervalcal[,i] = calavgpred$avg
  bootintervallep[,i] = lepavgpred$avg

  setTxtProgressBar(pb,i)
}
end_time <- Sys.time()
end_time - start_time

boot_int_cal = data.frame(bootintervalcal)
boot_int_lep = data.frame(bootintervallep)
write_csv(boot_int_cal, 'boot_int_cal.csv')
write_csv(boot_int_lep, 'boot_int_lep.csv')
```

Step 4: Pull the Data, Sort Them, and Get the Percentiles of Interest

```
#pull the data
interval_cal_long = read_csv('boot_int_cal_long.csv')
interval_lep_long = read_csv('boot_int_lep_long.csv')

#name the columns, sort them, and transpose them
names_lep = lepavgpred %>%
  unite(., col = 'names', sal:yr, sep = '_')
names_cal = calavgpred %>%
  unite(., col = 'names', sal:yr, sep = '_')

names_lep = as.vector(names_lep$names)
names_cal = as.vector(names_cal$names)

colnames(interval_cal_long) = names_cal
colnames(interval_lep_long) = names_lep

interval_cal_long_sorted <- apply(interval_cal_long,2,sort,decreasing=F)
interval_lep_long_sorted <- apply(interval_lep_long,2,sort,decreasing=F)

upci_cal = interval_cal_long_sorted[995, ]
loci_cal = interval_cal_long_sorted[5, ]
upci_lep = interval_lep_long_sorted[995, ]
loci_lep = interval_lep_long_sorted[5, ]

#put the up and lo CI's into the df's
calavgpred$conf.high = upci_cal
calavgpred$conf.low = loci_cal
lepavgpred$conf.high = upci_lep
lepavgpred$conf.low = loci_lep
write.csv(calavgpred, 'cal_plotting.csv')
write.csv(lepavgpred, 'lep_plotting.csv')
```

Step 5: Effects Plots

```
lepavgpred = read_csv('lep_plotting.csv')
```

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

```
## Parsed with column specification:
## cols(
##   X1 = col_double(),
##   avg = col_double(),
##   sal = col_character(),
##   reg = col_character(),
##   yr = col_double(),
##   conf.high = col_double(),
##   conf.low = col_double()
## )
```

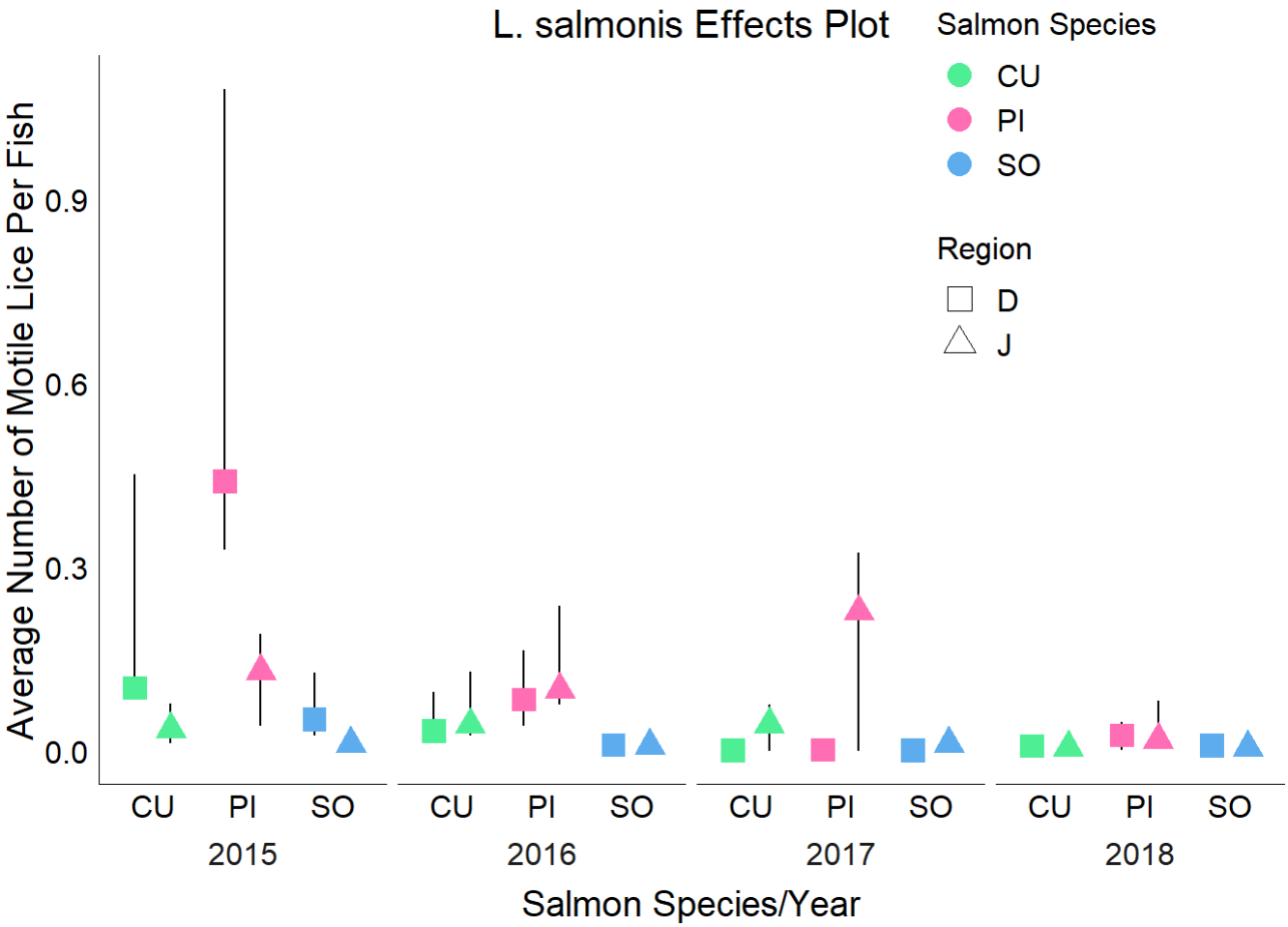
```
calavgpred = read_csv('cal_plotting.csv')
```

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

```
## Parsed with column specification:
## cols(
##   X1 = col_double(),
##   avg = col_double(),
##   sal = col_character(),
##   reg = col_character(),
##   yr = col_double(),
##   conf.high = col_double(),
##   conf.low = col_double()
## )
```

```
lepavgpred = lepavgpred %>%
  select(-X1)
calavgpred = calavgpred %>%
  select(-X1)
#Make the plots
fte_theme1 <- function(){
  color.background = 'white'
  color.grid.major = 'black'
  color.axis.text = 'black'
  color.axis.title = 'black'
  color.title = 'black'
  theme_bw(base_size = 9) +
    theme(panel.background = element_rect(fill=color.background,color = color.background)) +
    theme(plot.background = element_rect(fill = color.background, color = color.background)) +
    theme(panel.border = element_blank()) +
    theme(panel.grid.major = element_blank()) +
    theme(panel.grid.minor = element_blank()) +
    theme(axis.ticks = element_blank()) +
    theme(plot.title = element_text(color = color.title, size = 15, vjust = 1.25)) +
    theme(axis.text.x = element_text(size = 12, color = color.axis.text)) +
    theme(axis.text.y = element_text(size = 12, color = color.axis.text)) +
    theme(axis.title.x = element_text(size = 14, color = color.axis.title, vjust = 0)) +
    theme(axis.title.y = element_text(size = 14, color = color.axis.title, vjust = 1.25)) +
    theme(plot.title = element_text(hjust = 0.5)) +
    theme(axis.line.x = element_line(color="black", size = 0.15),
          axis.line.y = element_line(color="black", size = 0.15)) +
    theme(strip.background = element_blank(),
          strip.placement = 'outside',
          strip.text = element_text(size = 12))+
    theme(legend.position = c(0.8,0.82),
          legend.text = element_text(size = 12),
          legend.title = element_text(size = 12))
}
```

```
leg_title <- 'Salmon Species'
lepsmodplot_avg <- lepavgpred %>%
  group_by(., yr,sal,reg) %>%
  ggplot(aes(x = sal, y = avg, colour = sal, shape = reg)) +
  scale_shape_manual(values = c(15,17)) +
  geom_errorbar(aes(ymin=conf.low, ymax = conf.high,width = 0),
                position = position_dodge(width = 0.8),colour = 'Black')+
  geom_point(size = 4,position = position_dodge(width = 0.8)) +
  facet_wrap(~yr,nrow=1,strip.position = "bottom")+
  theme(strip.background = element_blank(), strip.placement = "outside") +
  scale_color_manual(leg_title,values=c('seagreen2', 'hotpink1', 'steelblue2'))+
  labs(title = "L. salmonis Effects Plot", x = 'Salmon Species/Year',
       y = 'Average Number of Motile Lice Per Fish') +
  guides(shape = guide_legend(title = 'Region', override.aes = list(shape = c(0,2)), type = 'b'
)) +
  fte_theme1()
lepsmodplot_avg
```



```
calmodplot_avg <- calavgpred %>%
  group_by(., yr,sal,reg) %>%
  ggplot(aes(x = sal, y = avg, colour = sal, shape = reg)) +
  scale_shape_manual(values = c(15,17)) +
  geom_errorbar(aes(ymin=conf.low, ymax = conf.high,width = 0), position = position_dodge(width
 = 0.8),colour = 'Black')+
  geom_point(size = 4,position = position_dodge(width = 0.8)) +
  facet_wrap(~yr,nrow=1,strip.position = "bottom")+
  theme(strip.background = element_blank(), strip.placement = "outside") +
  scale_color_manual(leg_title,values=c('seagreen2', 'hotpink1', 'steelblue2'))+
  labs(title = "C. clemensi Effects Plot", x = 'Salmon Species/Year', y = 'Average Number of Mot
ile Lice Per Fish') +
  guides(shape = guide_legend(title = 'Region', override.aes = list(shape = c(0,2)), type = 'b'
)) +
  fte_theme1()
calmodplot_avg
```