Supporting documentation: Unreliable inferences about chinstrap penguin population trends: a statistical critique and reanalysis

Chris Oosthuizen, Murray Christian, Azwianewi B. Makhado, Mzabalazo Ngwenya

2023-06-19

This document reviews the analysis of Krüger (2023) (Citation: Krüger, L. (2023). Decreasing Trends of Chinstrap Penguin Breeding Colonies in a Region of Major and Ongoing Rapid Environmental Changes Suggest Population Level Vulnerability. Diversity, 15(3), 327.). The Krüger (2023) supplementary material provide much of the analysis code that is used here. Our additional analysis cautions that the analysis performed by Krüger (2023) cannot support robust inference.

Krüger (2023) reanalysis

Load packages and set plotting theme

```
# Load packages
# data summary
library(reshape2)
library(plyr)
library(dplyr)
library(tidyverse)
#plots
library(ggplot2)
library(patchwork)
library(sjPlot)
#models
library(energy)
library(optimx)
library(minqa)
library(dfoptim)
library(MCMCglmm)

# plot theme
th <- theme(axis.text=element_text(size=12, face="bold",colour="grey30"),
          axis.title=element_text(size=14,face="bold"),
          panel.grid.major = element_blank(),
          panel.grid.minor = element_blank(),
          title =element_text(size=12, face="bold",colour="black"))
```

Load and process MAPPPD data for area 48.1:

```
# Humphries et al. (2017) Mapping Application for Penguin Populations
# and Projected Dynamics (MAPPPD): data and tools for dynamic management
# and decision support. Polar Record 53 (269): 160-166 doi:10.1017/S0032247417000055

df <- read.csv(here::here("./data/mapppd AllCounts_V_4_0.csv"))

# subset chinstrap penguin
chins<-subset(df,common_name=="chinstrap penguin")

summary(as.factor(chins$common_name))
```

```
## chinstrap penguin
##              1342
```

```
summary(as.factor(chins$count_type))
```

```
## adults chicks  nests
##     91    147   1104
```

```
# use only nest counts
nests<-subset(chins,count_type=="nests")

# WCO: Select Harmony Point (Nelson Island) data
# HP = subset(nests, nests$site_id == "HARM")
# HP

# some populations had multiple counts over the same season:
# this summarises the count with the maximum nests
nestM<-ddply(na.omit(data.frame(nests)), c("season_starting","site_id"),
             summarise,
             nests=max(penguin_count),
             Lat=mean(latitude_epsg_4326),
             Lon=mean(longitude_epsg_4326))
```

Here, the na.omit function removes all rows where there are NA values (missing data). Some rows have missing information for: - the day of the count - the day and month of the count - the accuracy of the count - the vantage point (ground, boat, uav, vhr) - on 4 occasions there are no count data (NA). One can argue that counts with unknown accuracy, vantage point, or count dates should be excluded from analysis, as was done here. Alternatively, one can argue that it makes little sense to exclude counts (e.g., those with high accuracy) where the only data missing is the day on the month where the count was conducted. This is because we did not subset / select counts in any other way (e.g., data was not limited to 'accurate' counts, or those happening within a certain date limit). Thus, this paper could arguably have use more of the available count data (given what was used). It is also worth discussing whether counts with poor accuracy should have been included in analysis, and if included, what the impact of counts with poor accuracy can have on the results.

```r
# summarizing number of populations and number of counts
countsN <-ddply(nestM, c("site_id","Lat","Lon"), summarise,
                ncounts=length(nests),
                interval=(max(season_starting)-min(season_starting)))
head(countsN)
```

```
##   site_id     Lat     Lon ncounts interval
## 1    ACUN -60.761 -44.637       1        0
## 2    AILS -60.780 -44.631       2       36
## 3    AITC -62.407 -59.752       4       21
## 4    AITK -60.738 -44.525       2       35
## 5    ALCO -64.240 -61.127       2       13
## 6    AMPH -60.684 -45.339       2       36
```

```r
# summarizing number of populations and number of counts with more than 0 nests
countsN2<-ddply(subset(nestM,nests>0), c("site_id","Lat","Lon"), summarise,
                ncounts=length(nests),
                interval=(max(season_starting)-min(season_starting)))
head(countsN2)
```

```
##   site_id     Lat     Lon ncounts interval
## 1    ACUN -60.761 -44.637       1        0
## 2    AILS -60.780 -44.631       2       36
## 3    AITC -62.407 -59.752       4       21
## 4    AITK -60.738 -44.525       2       35
## 5    ALCO -64.240 -61.127       2       13
## 6    AMPH -60.684 -45.339       2       36
```

```r
summary(as.factor(countsN2$ncounts))
```

```
##   1   2   3   4   5   6   7   8   9  10  11  12  14  15  21
## 148  89  14   8   4   2   2   3   1   3   2   1   1   1   2
```

```r
npops=length(countsN2$ncounts[countsN2$ncounts>1])
npops # number of populations
```

```
## [1] 133
```

```r
nestM2<-merge(nestM,countsN) # number of counts for each population by merging

# test for Poisson distribution (Poisson M-test)
poisson.mtest(nestM2$nests[nestM2$ncounts>1 & nestM2$nests>0],R=199)
```

```
##
##  Poisson M-test
##
## data:  nestM2$nests[nestM2$ncounts > 1 & nestM2$nests > 0] replicates:  199
## M-CvM = 158.43, p-value = 0.1658
## sample estimates:
## [1] 3006.691
```

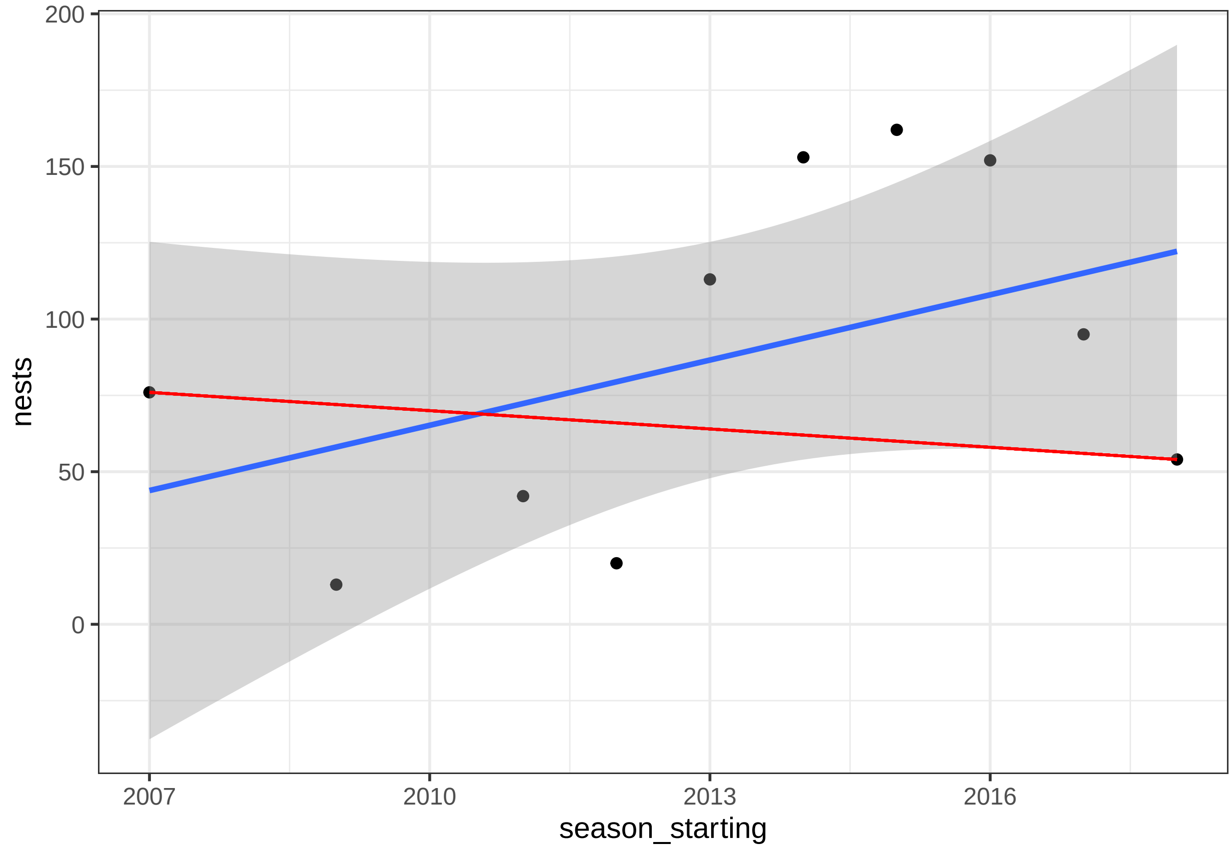
Here, the poisson.mtest is conducted on all the data (nestM2$nests[nestM2$ncounts>1 & nestM2$nests>0]. Yet, a glm is run per site. Should this test not be conducted per site, if we are conducting site-specific analysis? Regardless, we can probably just assume a Poisson distribution because counts are often Poisson distributed.

Calculate the mean slope of the decrease per site (glm)

```
# subset only populations with at least 2 counts and with any nest recorded
nestm3<-subset(nestM2,ncounts>1 & nests>0)

# calculating population level slope
slopeN <-na.omit(data.frame((nestm3 %>%
                              group_by(site_id,Lat,Lon) %>%
                              do({
                                mod=glm(nests~season_starting,family="poisson",
                                data=.)
                                data.frame(Intercept=coef(mod)[1],
                                           Slope=coef(mod)[2])}))))
```

Sanity check:

```
# WCO: aside some sites have positive glm slopes (growth) but
# negative proportion change (calculated below) e.g. site_id == "SPTR"
# Reminds me of recommendation by Hill et al. (2019) J of Crustacean Biology 39:316-322.
# "Avoid diagnosing or rejecting a multi-year trend based on a comparison of two years."

chck = subset(nestm3, nestm3$site_id == "SPTR")
chck = chck[order(chck$season_starting),]
ggplot(data = chck, aes(season_starting, nests)) +
         geom_point()+
         geom_smooth(method='lm', formula= y~x) +
         geom_segment(aes(x = min(chck$season_starting),
                          y = chck$nests[1],
                          xend = max(chck$season_starting),
                          yend = chck$nests[10]), colour = "red") +
         theme_bw()
```




```
# The model slopes are the same if the decrease is the same.
# E.g. these two sites halved in size and have the same slope (0.01925409)
# (but different intercepts)

subset(nestm3, nestm3$site_id == 'ANDE')
```

```
##    site_id     Lat     Lon season_starting nests ncounts interval
## 14    ANDE -60.757 -44.601            2019   200       2       36
## 15    ANDE -60.757 -44.601            1983   100       2       36
```

```
subset(slopeN, slopeN$site_id == 'ANDE')
```

```
##   site_id     Lat     Lon Intercept      Slope
## 6    ANDE -60.757 -44.601 -33.57569 0.01925409
```

```
subset(nestm3, nestm3$site_id == 'AILS')
```

```
##   site_id    Lat     Lon season_starting nests ncounts interval
## 2    AILS -60.78 -44.631            2019  3000       2       36
## 3    AILS -60.78 -44.631            1983  6000       2       36
```

```
subset(slopeN, slopeN$site_id == 'AILS')
```

```
##   site_id    Lat     Lon Intercept       Slope
## 1    AILS -60.78 -44.631  46.88037 -0.01925409
```

Here it is clear that there is rounding of numbers (100, 200, 3000, 6000). Rounding can contribute to uncertainty in true trends.

Some summary stats (Krüger 2023)

```
sloN <-merge(slopeN,countsN2) # number of counts for each population by merging

summary(as.factor(sloN$ncounts))
```

```
##  2  3  4  5  6  7  8  9 10 11 12 14 15 21
## 89 14  8  4  2  2  3  1  3  2  1  1  1  2
```

```
sloN$stdSlope<-sloN$Slope/sloN$interval
mean(sloN$Slope)
```

```
## [1] -0.02045084
```

```
sd(sloN$Slope)/sqrt(length(sloN$Slope)-1)
```

```
## [1] 0.007251265
```

```
mean(sloN$Slope[sloN$Slope<0])
```

```
## [1] -0.04960635
```

```
sd(sloN$Slope[sloN$Slope<0])/sqrt(length(sloN$Slope[sloN$Slope<0])-1)
```

```
## [1] 0.009966612
```

```
# number of populations
length(sloN$Slope)
```

```
## [1] 133
```

```
# number of decreasing populations
length(sloN$Slope[sloN$Slope<0])
```

```
## [1] 83
```

```
# proportion of decreasing populations
length(sloN$Slope[sloN$Slope<0])/length(sloN$Slope)
```

```
## [1] 0.6240602
```

Identify first and last counts

```
# identify year of first count
firstN<-ddply(nestM, c("site_id"), summarise,
              Ncounts=length(nests),
              season_starting=min(season_starting))

# counts on the first year
firstCount<-merge(nestM,firstN)

# identify year of last count
lastN<-ddply(nestM, c("site_id"), summarise,
             season_starting=max(season_starting))

# counts of the last year
lastCount<-merge(nestM,lastN)

summary(firstCount$Ncounts)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.000   1.000   2.000   2.278   2.000  21.000
```

```
# change names to join data frames
names(firstCount)[names(firstCount) == 'season_starting'] <- 'First'
names(firstCount)[names(firstCount) == 'nests'] <- 'FirstCount'
names(lastCount)[names(lastCount) == 'season_starting'] <- 'Last'
names(lastCount)[names(lastCount) == 'nests'] <- 'LastCount'
firlas<-merge(firstCount,lastCount,by=c("site_id","Lat","Lon")) # first and last counts
firlas<-subset(firlas,Ncounts>1) # subset only pops with more than one count
firlas$PercChange<-((firlas$LastCount/firlas$FirstCount)-1)*100 #percentual change
firlas$PercChange[is.na(as.numeric(firlas$PercChange))]<-0  # make NA = 0
Slope.Counts<-merge(firlas,sloN,by=c("site_id","Lat","Lon")) # merge slope and counts
summary(Slope.Counts$PercChange) #### percent change at the population level
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -100.00  -61.62  -23.08   11.31   28.33  900.00
```

```
sd(Slope.Counts$PercChange) /sqrt(length(Slope.Counts$PercChange)-1) # standard error
```

```
## [1] 11.2554
```

Krüger 2023 Figure 2 proportion decrease

```
#subset only decreasing populations (WCO: THIS ALSO SELECTS (ONE) STABLE POPULATION)
decr<-subset(Slope.Counts,Slope<0)
decr$YearDecr<-(-1*decr$Slope) # decrease per year
decr$PercDecr<-(-1*decr$PercChange) # absolute percent decrease

### classify range of decrease in categories ($decrCat)
```

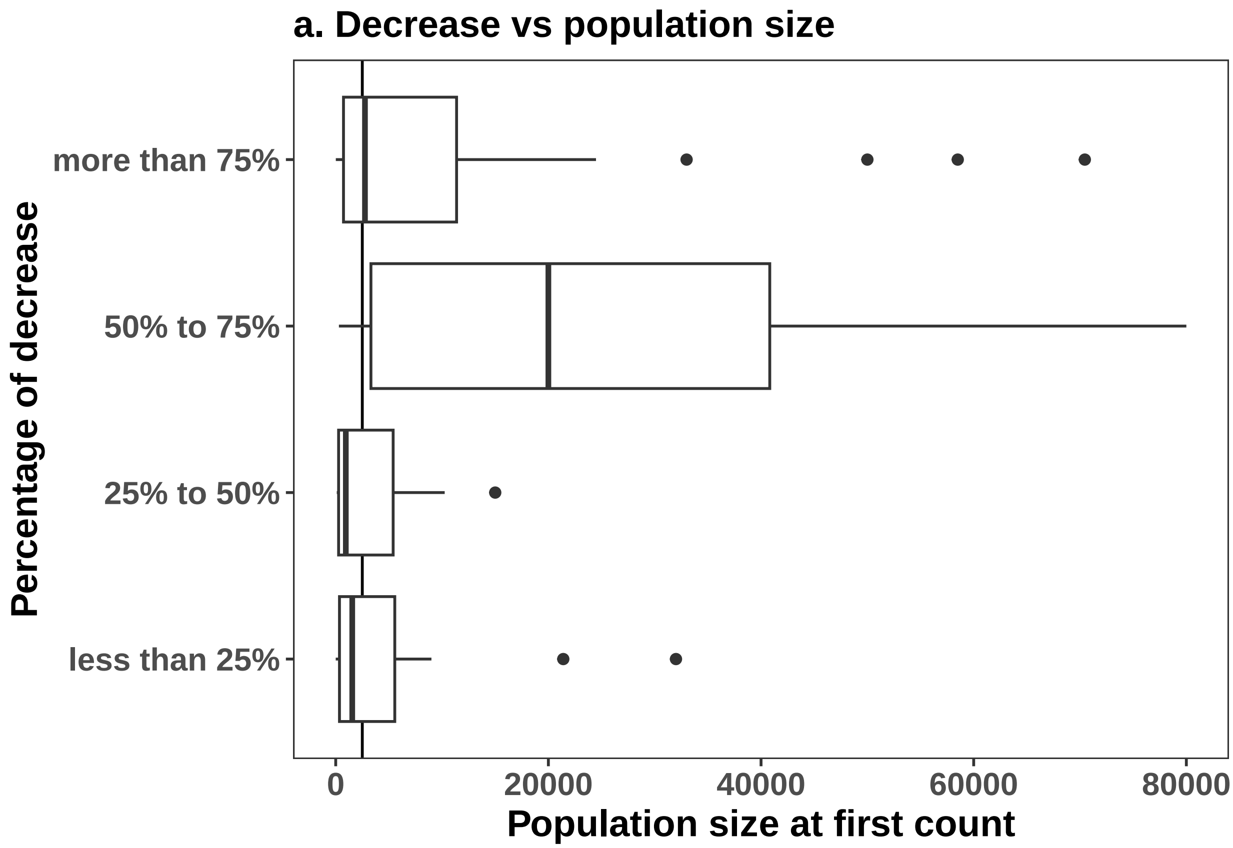
```r
decr$decrCat[decr$PercDecr<=25]<-"less than 25%"
decr$decrCat[decr$PercDecr>25 & decr$PercDecr<=50]<-"25% to 50%"
decr$decrCat[decr$PercDecr>50 & decr$PercDecr<=75]<-"50% to 75%"
# WCO: CODING ERROR / BUG. SELECTS >55 %, NOT 75%
decr$decrCat[decr$PercDecr>55]<-"more than 75%"
decr$decrCat<-factor(decr$decrCat,levels=c("less than 25%",
                                          "25% to 50%",
                                          "50% to 75%",
                                          "more than 75%")) # order of levels
n<-ddply(decr, c("decrCat"), summarise,
         N=length(FirstCount))
n
```

```
##         decrCat  N
## 1 less than 25% 19
## 2    25% to 50% 21
## 3    50% to 75%  5
## 4 more than 75% 38
```

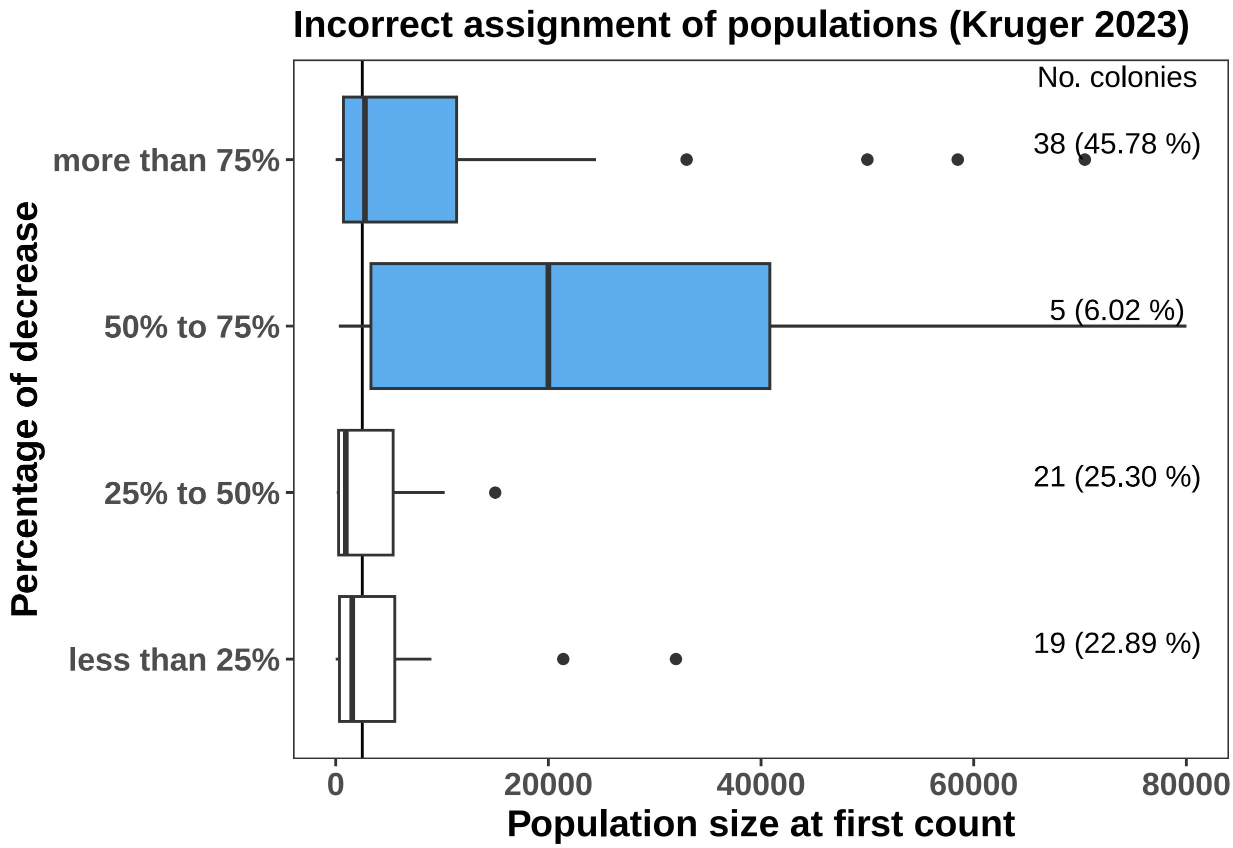
```r
sum(n$N) # check number of pops
```

```
## [1] 83
```

```r
n$perc<-n$N/83 # percentage of populations in each categories

perc_original = n$perc

#figure 2
ggplot(decr,aes(decrCat,FirstCount))+
  geom_hline(yintercept=2500)+
  geom_boxplot()+
  coord_flip()+theme_bw()+th+
  xlab("Percentage of decrease")+
  ylab("Population size at first count")+
  ggtitle(label="a. Decrease vs population size")
```




```r
fig2a = ggplot(decr,aes(decrCat,FirstCount))+
  geom_hline(yintercept=2500)+
  geom_boxplot()+
  geom_boxplot(data=decr[decr$decrCat=="more than 75%",],
                          aes(x = decrCat, y = FirstCount),fill="steelblue2")+
  geom_boxplot(data=decr[decr$decrCat=="50% to 75%",],
                          aes(x = decrCat, y = FirstCount),fill="steelblue2")+
  coord_flip()+theme_bw()+th+
  xlab("Percentage of decrease")+
  ylab("Population size at first count")+
  ggtitle(label="Incorrect assignment of populations (Kruger 2023)") +
  annotate("text", x = c(1.1, 2.1, 3.1, 4.1, 4.5),
          y = c(73500, 73500, 73500,73500,73500),
          label = c("19 (22.89 %)", "21 (25.30 %)",
                    "5 (6.02 %)", "38 (45.78 %)",
                    "No. colonies"), size=4)

fig2a
```



The above figure is incorrect as it includes population declines >55 % in the >75 % category. In other words, there are too many populations included in the >75 % category

Corrected Figure 2

```
# subset only decreasing populations (THIS ALSO SELECTS 1 STABLE POPULATION)
decr<-subset(Slope.Counts,Slope<0)
decr$YearDecr<-(-1*decr$Slope) # decrease per year
decr$PercDecr<-(-1*decr$PercChange) # absolute percent decrease
### classify range of decrease in categories ($decrCat)
decr$decrCat[decr$PercDecr<=25]<-"less than 25%"
decr$decrCat[decr$PercDecr>25 & decr$PercDecr<=50]<-"25% to 50%"
decr$decrCat[decr$PercDecr>50 & decr$PercDecr<=75]<-"50% to 75%"
# This line had a CODING ERROR / BUG. it selected >55 %, NOT 75%
decr$decrCat[decr$PercDecr>75]<-"more than 75%"
decr$decrCat<-factor(decr$decrCat,levels=c("less than 25%",
                                           "25% to 50%",
                                           "50% to 75%",
                                           "more than 75%")) # order of levels
n<-ddply(decr, c("decrCat"), summarise,
         N=length(FirstCount))
n
```

```
##         decrCat  N
## 1 less than 25% 19
## 2    25% to 50% 21
## 3    50% to 75% 26
## 4 more than 75% 17
```

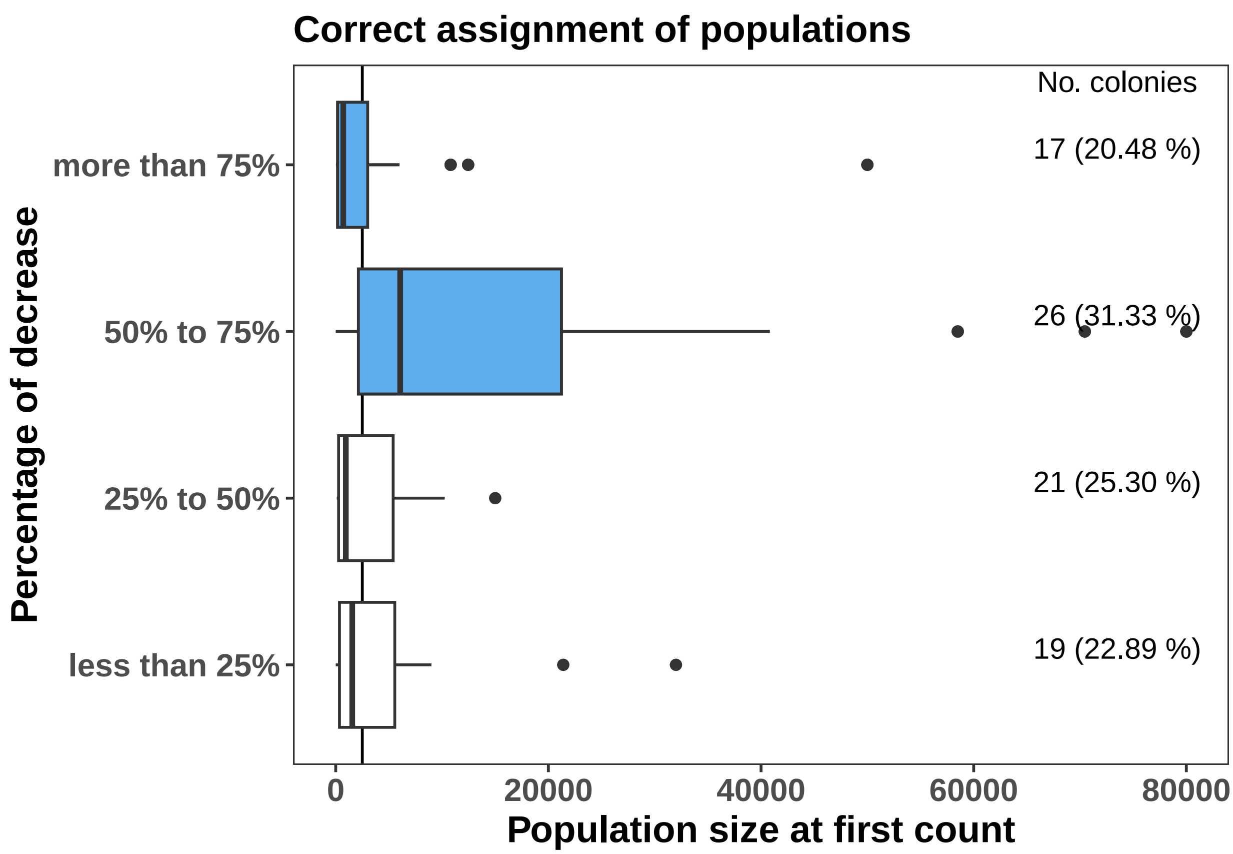
```r
#sum(n$N) # check number of pops

n$perc<-n$N/83 # percentage of populations in each categories

perc_corrected =n$perc

## original manuscript percentage of populations in each category
print(perc_original)
```

```
## [1] 0.22891566 0.25301205 0.06024096 0.45783133
```

```r
#### corrected percentage of populations in each category
perc_corrected
```

```
## [1] 0.2289157 0.2530120 0.3132530 0.2048193
```

```r
#figure 2 corrected

fig2b = ggplot(decr,aes(decrCat,FirstCount))+
  geom_hline(yintercept=2500)+
  geom_boxplot()+
  geom_boxplot(data=decr[decr$decrCat=="more than 75%",],
                          aes(x = decrCat, y = FirstCount),fill="steelblue2")+
  geom_boxplot(data=decr[decr$decrCat=="50% to 75%",],
                          aes(x = decrCat, y = FirstCount),fill="steelblue2")+
  coord_flip()+theme_bw()+th+
  xlab("Percentage of decrease")+
  ylab("Population size at first count")+
  ggtitle(label="Correct assignment of populations") +
   annotate("text", x = c(1.1, 2.1, 3.1, 4.1, 4.5),
            y = c(73500, 73500, 73500,73500,73500),
          label = c("19 (22.89 %)", "21 (25.30 %)",
                    "26 (31.33 %)", "17 (20.48 %)", "No. colonies"),
          size=4)

fig2b
```




```r
# library(cowplot)
# plot_grid(fig2a, fig2b)

## Save Plot
#  pdf("./Figure 2.pdf", useDingbats = FALSE, width = 14, height = 7)
#  plot_grid(fig2a, fig2b,
#             labels = "AUTO", scale = 0.9, vjust = 2, hjust = -4)
# dev.off()
```

Extra: plot population increases

```r
## Proportion increase
incr<-subset(Slope.Counts,Slope>=0)
incr$Yearincr<-(-1*incr$Slope) # increase per year
incr$Percincr<-(-1*incr$PercChange) # absolute percent increase

# growing populations
incr$incrCat[incr$Percincr> -25 & incr$Percincr<=0]<-"0% to 25% growth"
incr$incrCat[incr$Percincr> -50 & incr$Percincr<= -25]<-"25% to 50% growth"
incr$incrCat[incr$Percincr> -75 & incr$Percincr<= -50]<-"50% to 75% growth"
incr$incrCat[incr$Percincr<= -75] <-"75% and higher growth"
# order of levels
incr$incrCat<-factor(incr$incrCat,levels=c("0% to 25% growth",
                                          "25% to 50% growth",
```

```
                                         "50% to 75% growth",
                                         "75% and higher growth"))
n<-ddply(incr, c("incrCat"), summarise,
                  N=length(FirstCount))
n
```

```
##                  incrCat  N
## 1      0% to 25% growth 11
## 2     25% to 50% growth 10
## 3     50% to 75% growth  4
## 4 75% and higher growth 22
## 5                  <NA>  3
```

```
#sum(n$N) # check number of pops

n$perc<-n$N/(11+10+4+22) # percentage of populations in each categories
n
```

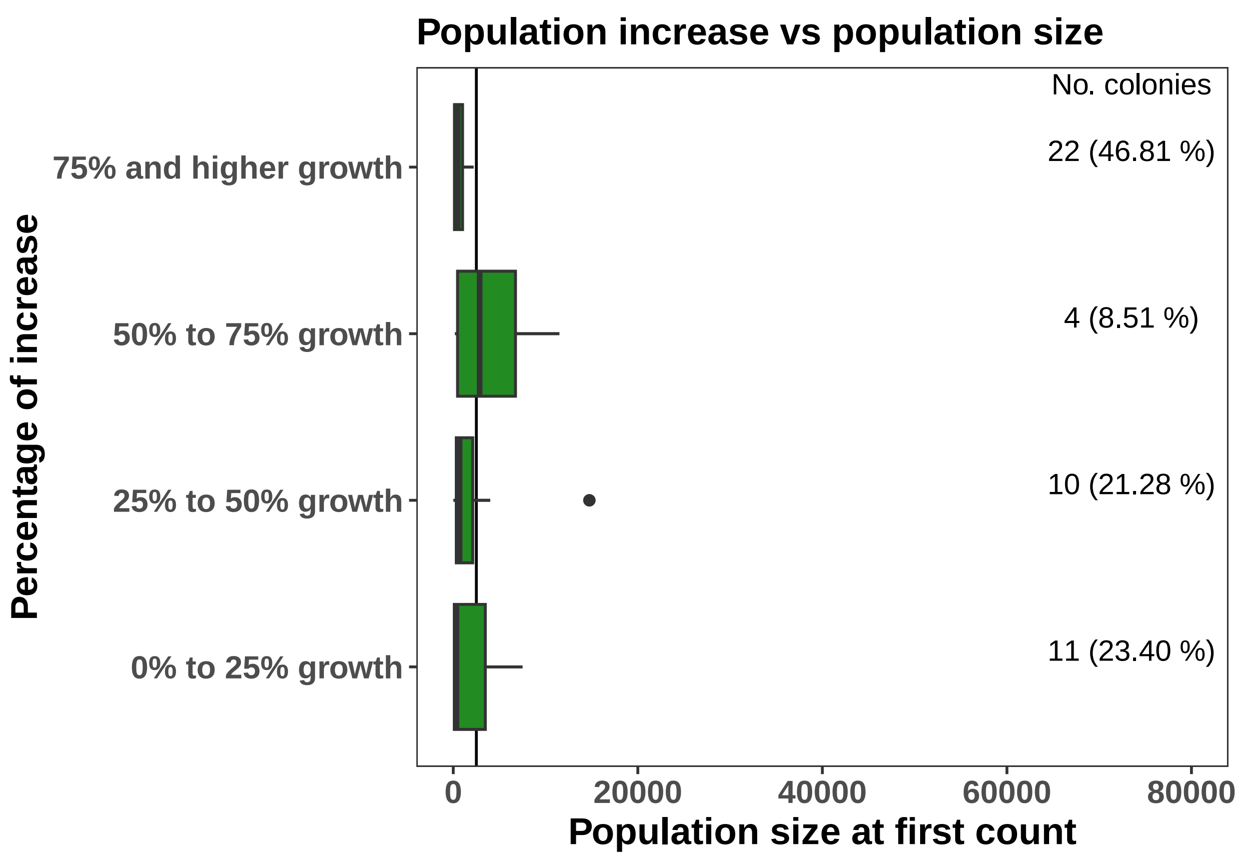
```
##                  incrCat  N       perc
## 1      0% to 25% growth 11 0.23404255
## 2     25% to 50% growth 10 0.21276596
## 3     50% to 75% growth  4 0.08510638
## 4 75% and higher growth 22 0.46808511
## 5                  <NA>  3 0.06382979
```

```
perc_corrected = n$perc
n
```

```
##                  incrCat  N       perc
## 1      0% to 25% growth 11 0.23404255
## 2     25% to 50% growth 10 0.21276596
## 3     50% to 75% growth  4 0.08510638
## 4 75% and higher growth 22 0.46808511
## 5                  <NA>  3 0.06382979
```

```
# there are 3 populations with positive slopes (thus in this data)
incr = na.omit(incr)       # that had negative last - first counts

fig2c = ggplot(incr,aes(incrCat,FirstCount))+
         geom_hline(yintercept=2500)+
         geom_boxplot()+
         geom_boxplot(data=incr[incr$incrCat=="0% to 25% growth",],
                      aes(x = incrCat, y = FirstCount),fill="forestgreen")+
           geom_boxplot(data=incr[incr$incrCat=="25% to 50% growth",],
                      aes(x = incrCat, y = FirstCount),fill="forestgreen")+
                      aes(x = incrCat, y = FirstCount),fill="forestgreen")+
           geom_boxplot(data=incr[incr$incrCat=="75% and higher growth",],
                      aes(x = incrCat, y = FirstCount),fill="forestgreen")+
          coord_flip()+theme_bw()+th+
           xlab("Percentage of increase")+
           ylab("Population size at first count")+
           ggtitle(label="Population increase vs population size")+
```

```
         ylim(0,80000)+
    annotate("text", x = c(1.1, 2.1, 3.1, 4.1, 4.5),
                y = c(73500, 73500, 73500,73500,73500),
            label = c("11 (23.40 %)", "10 (21.28 %)",
                      "4 (8.51 %)", "22 (46.81 %)", "No. colonies"),
            size=4)

fig2c
```




```
## Save Plot
 # pdf("./Figure 2d.pdf",useDingbats = FALSE, width = 14, height = 12)
 # plot_grid(fig2a, fig2b, fig2c,
 #           labels = "AUTO", scale = 0.9, vjust = 2, hjust = -4)
 # dev.off()
```

MCMCglmm mixed model data

```
nestM3<- nestm3   #populations with at least 2 counts and with any nest recorded
length(unique(nestM3$site_id))
```

```
## [1] 146
```

What is the sample size per site? Krüger (2023) reports 133 sites, but the count here is 146 unique sites. But the code gave 133 above. Why do we get both 133 and 146? Some sites have 2 counts (e.g., TAYL) but one of the counts are zero. TAYL is included in the nestm3 data frame. It remains in there because the filter (nestm3; paper's code above) is on ncounts>1 & nests>0), and nests is the variable that counts how many nest counts were made. But that variable does not condition on the counts being more than 0. 133 sites had more than one data point in nestM3.

Specify MCMCglmm mixed model prior (Krüger 2023)

```
prior<- list(R = list(V = 1, nu = 0.002),
             G = list(G1 = list(V = diag(2), nu = 0.002,
                                alpha.mu = rep(0, 2),
                                alpha.V= diag(133, 2, 2))))
```

Fit MCMCglmm mixed model (Krüger 2023)

```
mc1<-MCMCglmm(nests~season_starting, random=~us(1 + Lat):site_id, rcov=~units,
              family="poisson", mev=NULL,
              data=nestM3, start=NULL, nodes="ALL", scale=TRUE,
              nitt=13000, thin=10, burnin=3000,
              pr=T, pl=FALSE, verbose=TRUE, DIC=TRUE, singular.ok=FALSE,
              saveX=TRUE,prior=prior, saveZ=TRUE, saveXL=TRUE, slice=FALSE,
              ginverse=NULL, trunc=FALSE)
```

```
##
##                        MCMC iteration = 0
##
##  Acceptance ratio for liability set 1 = 0.000359
##
##                        MCMC iteration = 1000
##
##  Acceptance ratio for liability set 1 = 0.222497
##
##                        MCMC iteration = 2000
##
##  Acceptance ratio for liability set 1 = 0.286242
##
##                        MCMC iteration = 3000
##
##  Acceptance ratio for liability set 1 = 0.309743
##
##                        MCMC iteration = 4000
##
##  Acceptance ratio for liability set 1 = 0.280134
##
##                        MCMC iteration = 5000
##
##  Acceptance ratio for liability set 1 = 0.279695
```

```
##
##                         MCMC iteration = 6000
##
##  Acceptance ratio for liability set 1 = 0.279777
##
##                         MCMC iteration = 7000
##
##  Acceptance ratio for liability set 1 = 0.280299
##
##                         MCMC iteration = 8000
##
##  Acceptance ratio for liability set 1 = 0.279063
##
##                         MCMC iteration = 9000
##
##  Acceptance ratio for liability set 1 = 0.279420
##
##                         MCMC iteration = 10000
##
##  Acceptance ratio for liability set 1 = 0.280415
##
##                         MCMC iteration = 11000
##
##  Acceptance ratio for liability set 1 = 0.279380
##
##                         MCMC iteration = 12000
##
##  Acceptance ratio for liability set 1 = 0.279470
##
##                         MCMC iteration = 13000
##
##  Acceptance ratio for liability set 1 = 0.279603
```

```
# Note: low ESS for random effects. Random effect ESS was 25-27 in Kruger (2023),
# (see supplement), so this is not only a problem that we are encountering.

summary(mc1)
```

```
##
##  Iterations = 3001:12991
##  Thinning interval  = 10
##  Sample size  = 1000
##
##  DIC: 4789.514
##
##  G-structure:  ~us(1 + Lat):site_id
##
##                                  post.mean l-95% CI u-95% CI eff.samp
## (Intercept):(Intercept).site_id  1859.236 50.41690 4499.063    78.16
## Lat:(Intercept).site_id            30.399  1.19892   73.002    78.80
## (Intercept):Lat.site_id            30.399  1.19892   73.002    78.80
## Lat:Lat.site_id                     0.498  0.02465    1.187    79.51
##
##  R-structure:  ~units
```

```
##
##       post.mean l-95% CI u-95% CI eff.samp
## units    0.2936   0.2456   0.3388     1000
##
##  Location effects: nests ~ season_starting
##
##                 post.mean  l-95% CI  u-95% CI eff.samp  pMCMC
## (Intercept)     27.641549 20.637750 34.441221     1000 <0.001 ***
## season_starting -0.010386 -0.013840 -0.006857     1000 <0.001 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Random effect syntax: ~us fit different variances across each component in formula, plus the covariances. The linear model inside the variance function has two parameters, an intercept(1) and a regression slope associated with latitude. Each site now has an intercept and a slope specified. But slope (latitude) does not vary within site, and there is only one count per year, per site. This is not a good random effect model structure.

Sanity check:

```
# Each site only has one latitude value
uniqueLat = nestM3  %>%
  group_by(site_id) %>%
  summarise(count = n_distinct(Lat))
max(uniqueLat$count)
```

```
## [1] 1
```

```
# There are 140 unique latitudes from the 146 sites
length(unique(nestM3$Lat))
```

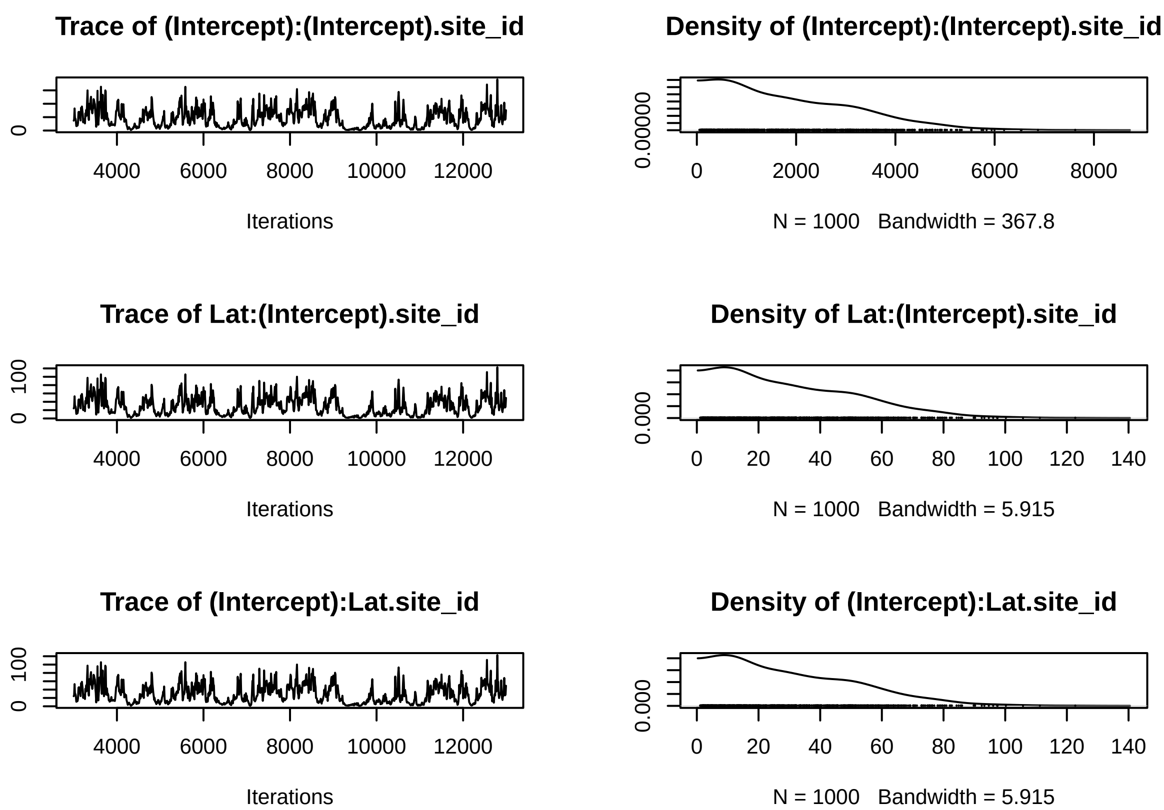
```
## [1] 140
```

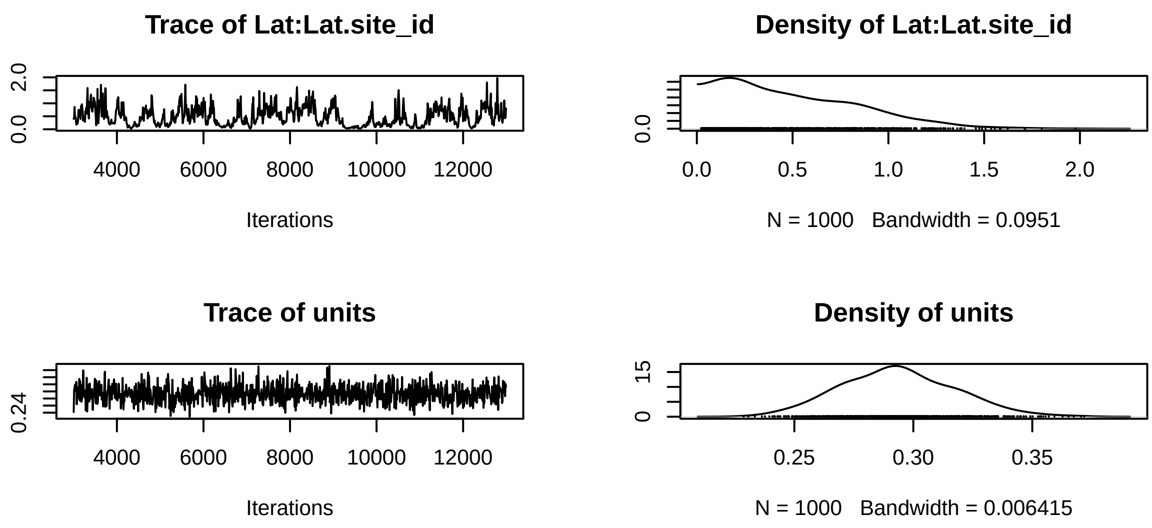
MCMCglmm model checking

It is important to evaluate the fit of the model.

```
# The samples from the posterior distribution are stored as mcmc objects,
# which can be summarized and visualized using the coda package

# from MCMC Course notes (page 60):
# Aim to store 1,000-2,000 iterations and have the autocorrelation between
# successive stored iterations less than 0.1 (page 22).

# Assessing model convergence. We do this separately for both fixed
# and random effects. The trace plot should look like a fuzzy caterpillar
# plot(mc1$Sol)
```

```
# variances of the random effects (trace plots)
plot(mc1$VCV)
```






```
# It looks like some of the variances of the random effects haven't
# mixed very well.

# what are the effective sample size for the random effects?
coda::effectiveSize(mc1$VCV)
```

```
## (Intercept):(Intercept).site_id         Lat:(Intercept).site_id
##                         78.15863                        78.80318
##         (Intercept):Lat.site_id                 Lat:Lat.site_id
##                         78.80318                        79.51024
##                            units
##                       1000.00000
```

```
# The effective sample size is very small.

k = 1 # number of fixed effects
autocorr(mc1$Sol[, 1:k])  # fine - low correlation
```

```
## , , 1
##
##                 [,1]
## Lag 0    1.000000000
## Lag 10   0.031840767
## Lag 50   0.003738654
## Lag 100  0.024877240
## Lag 500 -0.025319623
```

```r
# from MCMC Course notes (page 60):
diag(autocorr(mc1$VCV)[2, , ])   # very high autocorrelation
```

```
## (Intercept):(Intercept).site_id        Lat:(Intercept).site_id
##                      0.76272451                     0.75993577
##         (Intercept):Lat.site_id                Lat:Lat.site_id
##                      0.75993577                     0.75687200
##                           units
##                      0.02487251
```

MCMCglmm Random effects (Krüger 2023)

```r
sol<-data.frame(mc1$Sol) # random effects
# names(sol)
solm<-melt(sol,id.vars=c("X.Intercept.","season_starting"))
# head(solm)
solm$site_id<-substring(solm$variable,first=22,last=26)
```

> WCO: I found the following confusing. The code above drops all the 'Lat.site_id' (are these not the slopes?) (because 'site_id' is blank for them in solm) and keeps only the X.Intercept..site_id. The idea was to plot the slope (decrease in population size). I thought that sigma X.Intercept would be the amount of variation in intercepts between sites and sigma Lat would be the amount of variation in the regression slopes between sites. Yet, Figure 3B in Krüger 2023 looks very similar to the one produced from our own analysis (with the exception that the y-axis is from 0 to 40, rather than from 0 to 1). However, later analysis shows that they are not equivalent (see towards the end).

```r
ranef<-ddply(solm, c("site_id"), summarise,
        int=mean(value),
        intsd=sd(value),
        intse=intsd/sqrt(length(value)-1))

rlat<-merge(ranef,slopeN,by="site_id")
```

Predicting counts from mixed model (1960 to 2020) (Krüger 2023)

```r
# construct an hypothetical dataframe to generate the populations estimaters
years<-data.frame(season_starting=c(1960:2020))
pops<-data.frame(site_id=countsN2$site_id[countsN2$ncounts>1],
                 Lat=countsN2$Lat[countsN2$ncounts>1])
popy<-merge(pops,years)
popy$nests<-c(0) ### MCMCglmm needs a column with the response variable
popypred<-data.frame(predict(mc1,newdata=popy,type="response",
                             marginal=mc1$Random$formula,
                             interval="prediction", posterior="mean"))
popy$fit<-popypred$fit
```

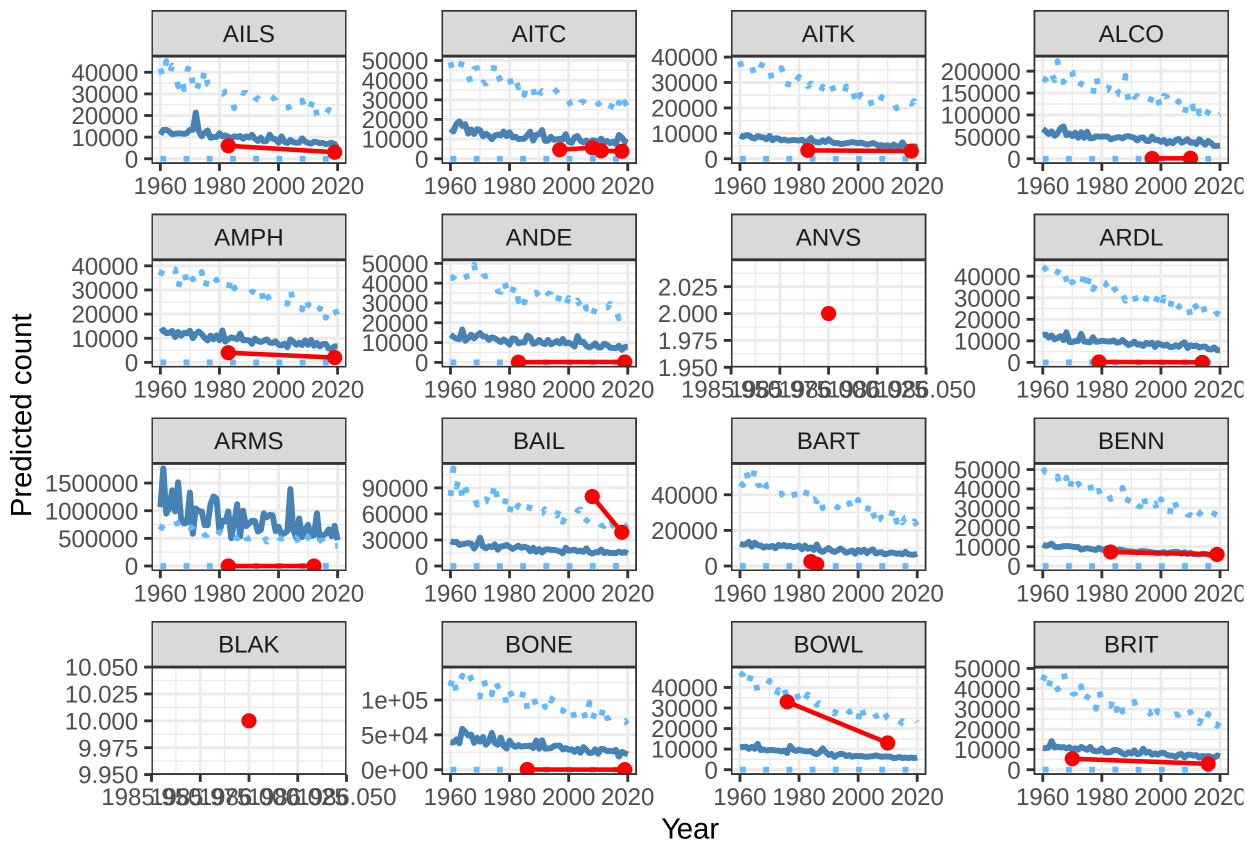
The prediction above contains a high degree of uncertainty, which was ignored. The uncertainty is the lwr and upr columns, which is the Highest Posterior Density intervals, I believe, from coda::HPDinterval. https://rdrr.io/cran/MCMCglmm/src/R/predict.MCMCglmm.R*

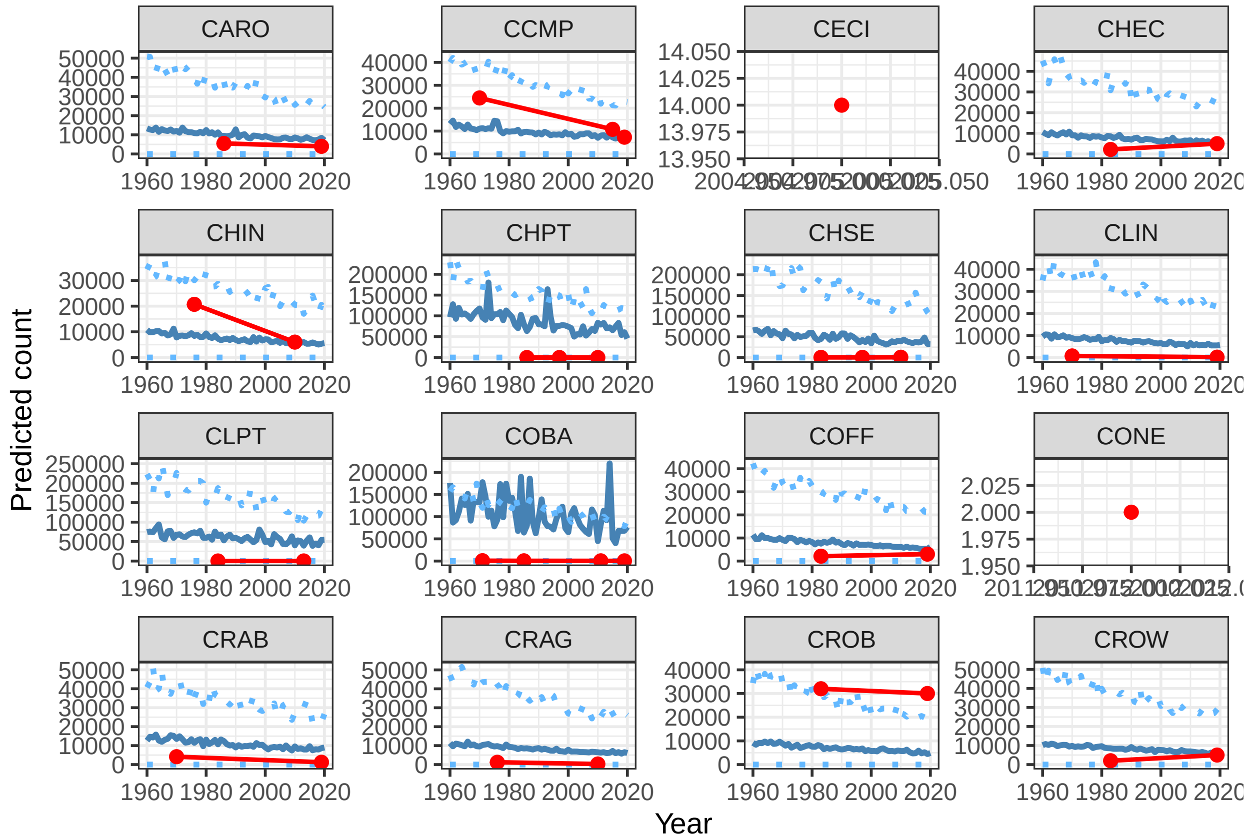
Here, the syntax marginal=mc1*Random*formula was used. This means random effects were marginalized (see simulation study).

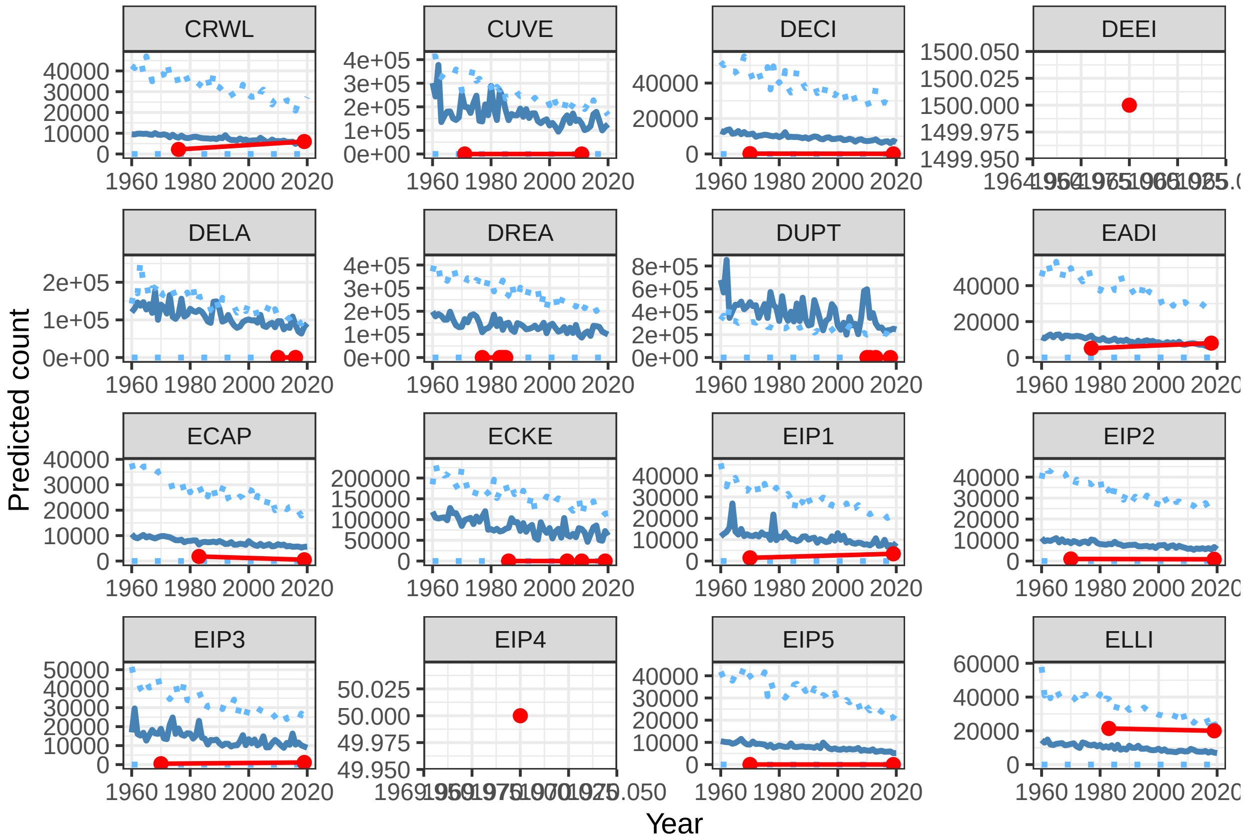
How accurate are the predictions relative to observed data?

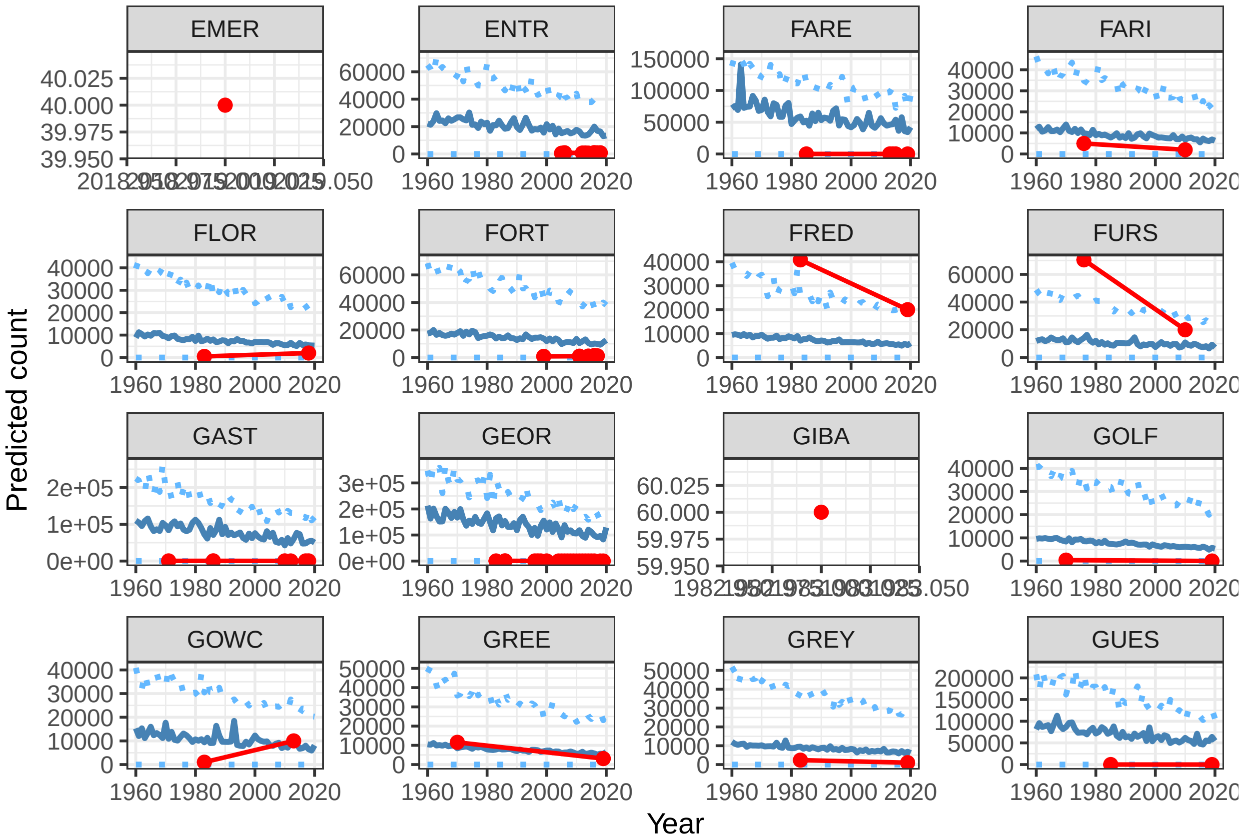
```
# Plot the observed data against predicted data, per site.

# ids = factor(unique(popy$site_id))
#
# for(i in ids) {
#   temp1 = popy %>%
#       dplyr::filter(site_id == i)
#   temp2 = nestm3 %>%
#     dplyr::filter(site_id == i)
#
# temp =  ggplot(data = temp1,
#          aes(x = season_starting, y = fit)) +
#     geom_point() + geom_line() +
#     scale_y_continuous(limits = c(0, max(temp1$fit)))+
#     geom_point(data = temp2, aes(season_starting, y = nests),
#     color = "red", cex = 2) +
#     geom_line(data = temp2, aes(season_starting, y = nests),
#     color = "red") +
#     ggtitle(i) +
#     theme_bw()
#
# ggsave(temp, file=paste0("./figures/plot_", i,".png"),
# width = 14, height = 10, units = "cm")
# }


# Add lower and upper prediction intervals to the data used for inference
popy$lwr<-popypred$lwr
popy$upr<-popypred$upr

# Plot the observed data against predicted data, per site.
library(ggforce)

required_n_pages = round(133/16)+1

for(i in 1:required_n_pages){

print(ggplot(data = popy) +
    geom_line(aes(x = season_starting, y = fit), col = "steelblue",linewidth=1.04) +
    geom_line(aes(x = season_starting, y = lwr), col = "steelblue1",
              linetype="dotted", linewidth = 1.02) +
```

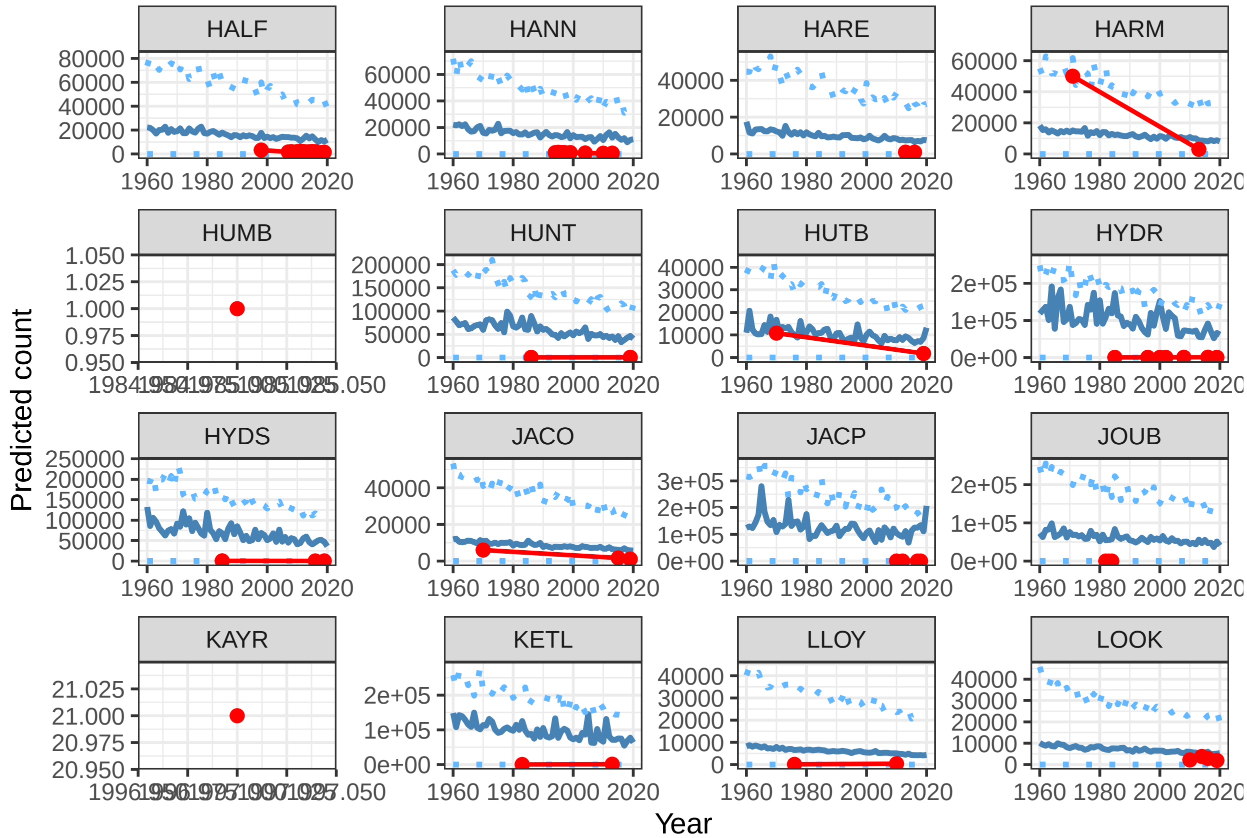
```r
  geom_line(aes(x = season_starting, y = upr), col = "steelblue1",
            linetype="dotted",linewidth = 1.02) +
  geom_point(data = nestm3, aes(season_starting, y = nests),
             color = "red", cex = 2) +
  geom_line(data = nestm3, aes(season_starting, y = nests),
            color = "red",size=0.8) +
  theme_bw() +
  xlab("Year") +
  ylab("Predicted count") +
#  theme(strip.text = element_text(size = 1.5)) +
  facet_wrap_paginate(~ site_id, ncol = 4, nrow = 4,
                      page = i,
                      scales = 'free'))}
```

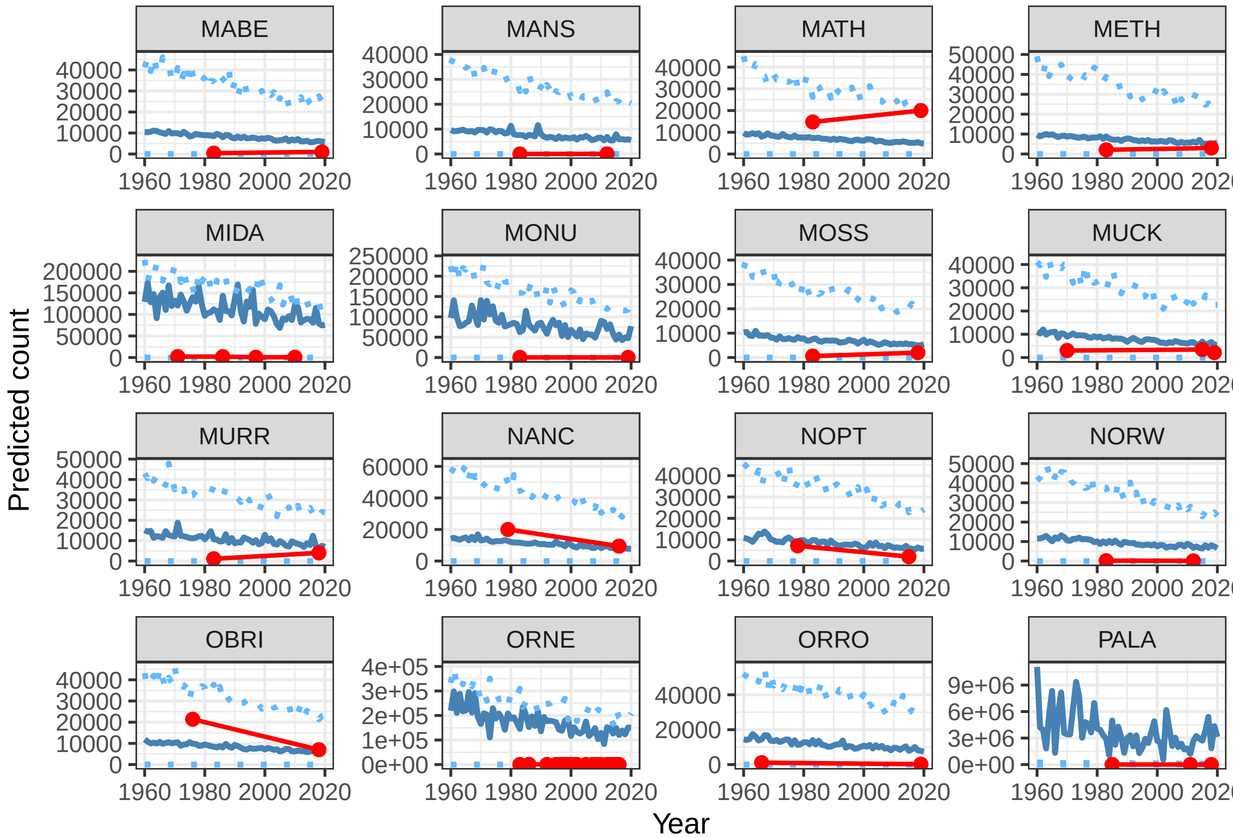


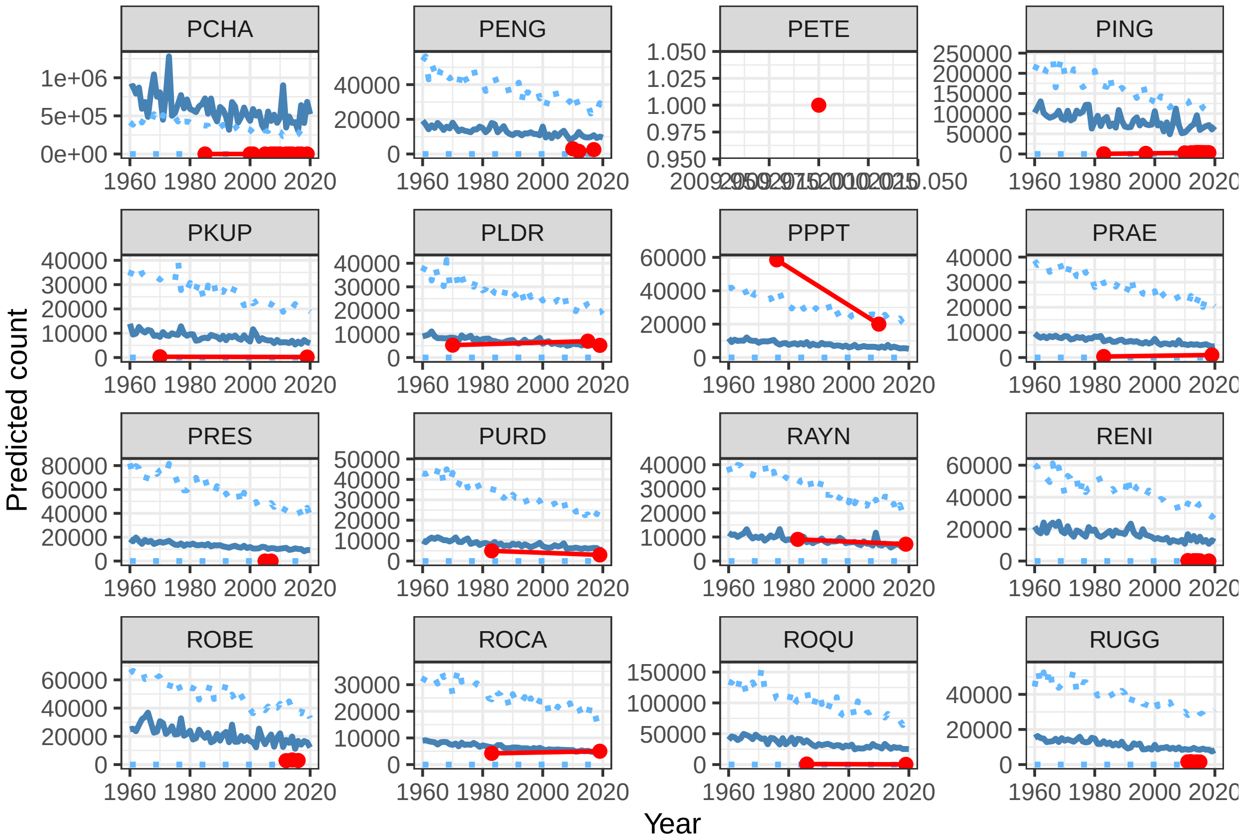

CARO
CCMP
CECI
CHEC
CHIN
CHPT
CHSE
CLIN
CLPT
COBA
COFF
CONE
CRAB
CRAG
CROB
CROW
Predicted count
Year


CRWL
CUVE
DECI
DEEI
DELA
DREA
DUPT
EADI
ECAP
ECKE
EIP1
EIP2
EIP3
EIP4
EIP5
ELLI
Predicted count
Year

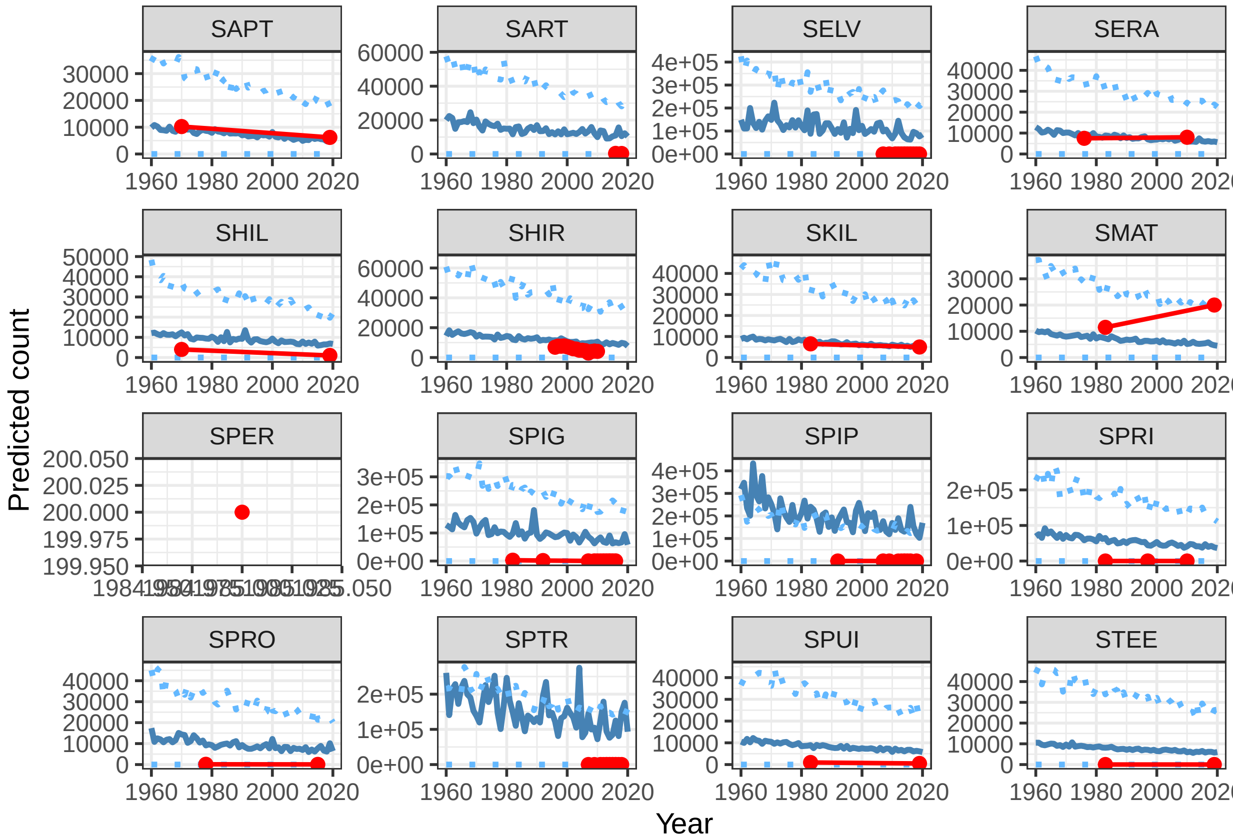

EMER
ENTR
FARE
FARI
FLOR
FORT
FRED
FURS
GAST
GEOR
GIBA
GOLF
GOWC
GREE
GREY
GUES
Predicted count
Year

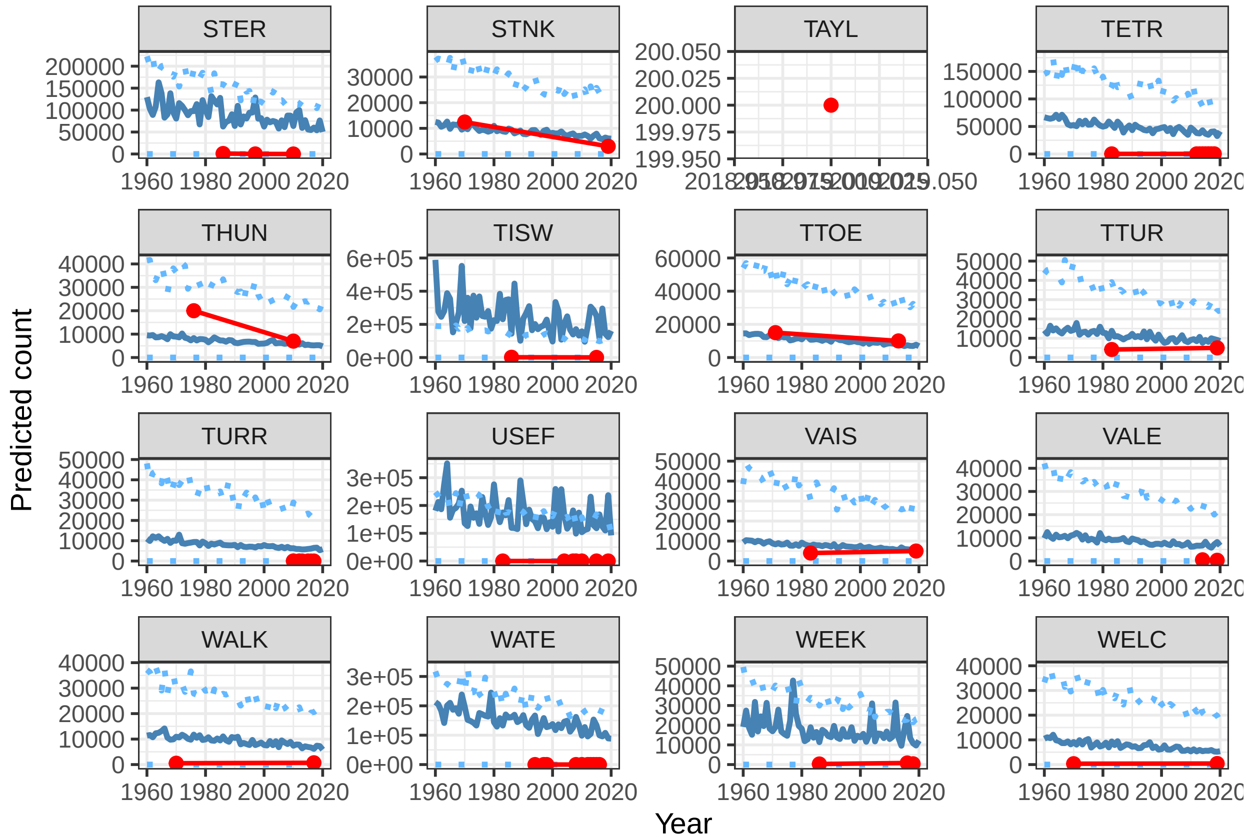

HALF
HANN
HARE
HARM
HUMB
HUNT
HUTB
HYDR
HYDS
JACO
JACP
JOUB
KAYR
KETL
LLOY
LOOK
Predicted count
Year


MABE
MANS
MATH
METH
MIDA
MONU
MOSS
MUCK
MURR
NANC
NOPT
NORW
OBRI
ORNE
ORRO
PALA
Predicted count
Year



PCHA
PENG
PETE
PING
PKUP
PLDR
PPPT
PRAE
PRES
PURD
RAYN
RENI
ROBE
ROCA
ROQU
RUGG
Predicted count
Year


SAPT
SART
SELV
SERA
SHIL
SHIR
SKIL
SMAT
SPER
SPIG
SPIP
SPRI
SPRO
SPTR
SPUI
STEE
Predicted count
Year

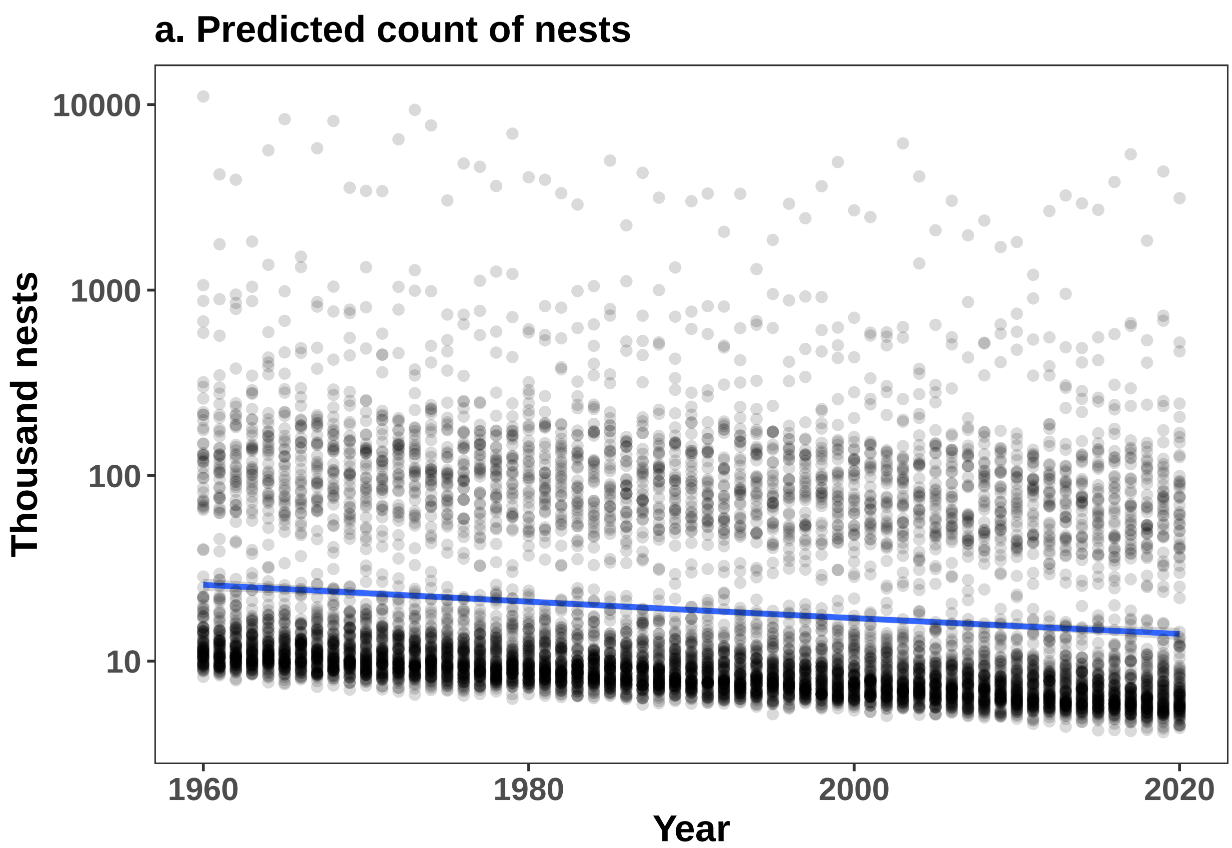



```
# Blue solid lines are the predicted abundance (posterior mean) used by Krüger (2023)
# to predict regional declines. Light blue dots are the 95 % Highest Posterior Density
# interval for this prediction. Red points are the observed counts
# (connected with a red line).
```

Figure 3A (Krüger 2023)

```
#p1<-plot_model(mc1,type="emm",terms="season_starting[all]",show.values = T,
#                 ci.lvl=0.9999)+
#   theme_bw()+th+xlim(1960,2020)+xlab("Year")+ylab("Nests")+
#   ggtitle(label="a. Predicted count of nests") ### plot directly from the model
#
# p1

p1v2<-ggplot(popy,aes(season_starting,fit/1000))+
  geom_smooth()+
  geom_point(alpha=0.15)+xlab("Year")+
  theme_bw()+th+ylab("Thousand nests")+
  ggtitle(label="a. Predicted count of nests")+
  scale_y_log10() # plot from the predicted fit

p1v2
```

```
## `geom_smooth()` using method = 'gam' and formula = 'y ~ s(x, bs = "cs")'
```



This figure plots all the individual site level predictions. It cannot be sensible given the poor model fit and predictions. On some model runs the output look similar to that of Kruger (2023) in other model runs the y-axis scale is larger (e.g. to 1e+05).

Figure 3B (Krüger 2023)

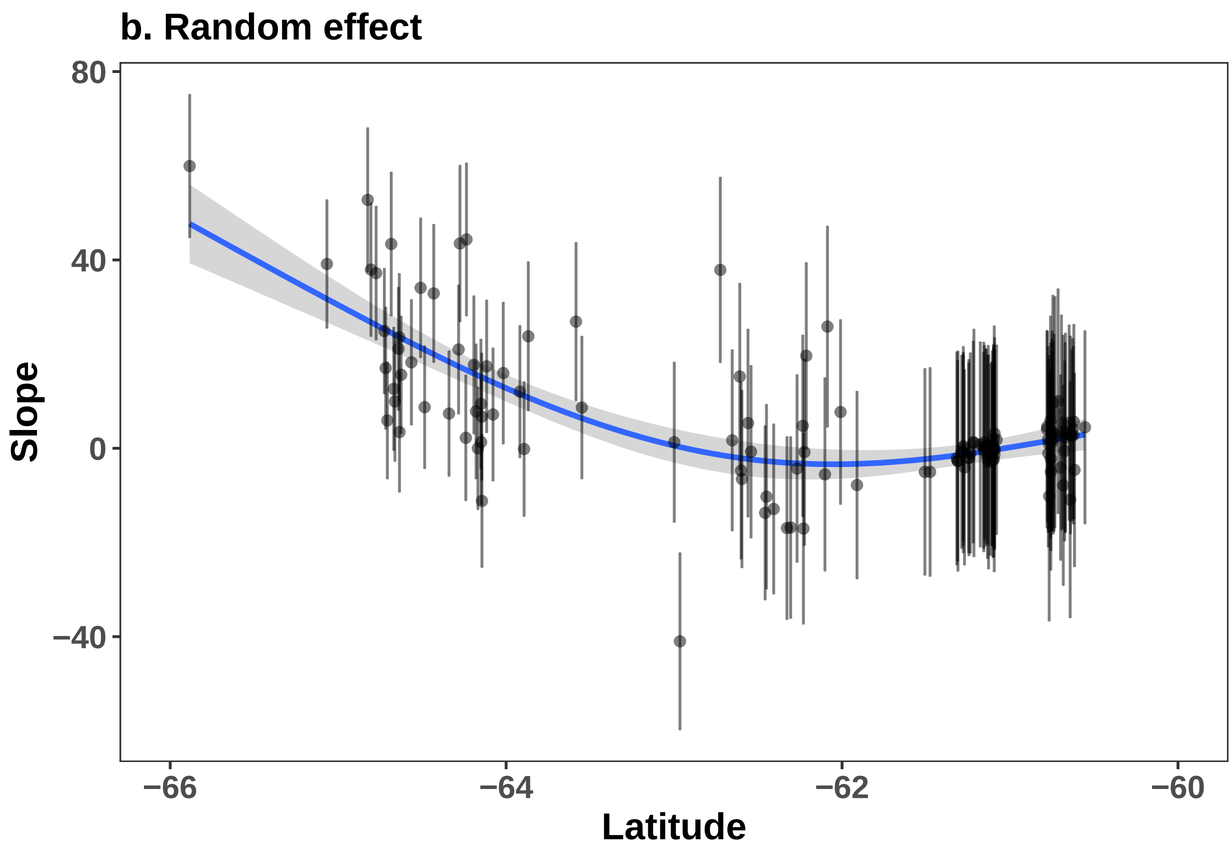
This plots the MCMCglmm intercept(?) - it is labelled "int". But the paper legend says slope (which is what we are interested in). This figure makes use of a very poor fitting model (mc1), but initally the output looks similar to that from our own analysis. That is because both plots latitude on the x-axis - so the distibution of points on the x-axis are the same. The sites vary a lot on the y-axis. This plot does not represent changes in population rate of change.

The error bar is calculated as sd/2. The paper caption refers to 'standard deviation' But why divide the standard deviation by 2?

```
p2<- ggplot(subset(rlat,Lat>(-67)),aes(Lat,int))+
  stat_smooth(method="gam",formula=y~s(x,k=2))+
  geom_errorbar(aes(ymin=int-(intsd/2),ymax=int+(intsd/2)),alpha=0.5)+
  geom_point(alpha=0.5)+
  theme_bw()+th+
  ggtitle(label="b. Random effect")+
```

```
  ylab("Slope")+xlim(-66,-60)+
  xlab("Latitude")

p2
```

```
## Warning in smooth.construct.tp.smooth.spec(object, dk$data, dk$knots): basis dimension, k, increased
```




```
# ps: as results are based on randomization
# expect slight differences every time you run the model
# but the trends are consistent everytime
# lagged analysis to determine how much pops have decreased
```

Population change in 3 generations

```
library(lubridate)
library(tidyr)
#library(tidyquant)
library(dplyr)
library(broom)
library(purrr)
library(stringr)
library(knitr)
```

```r
#library(timetk)

# Use library(xts) instead, below:

head(popy)
```

```
##   site_id     Lat season_starting nests       fit lwr    upr
## 1    AILS -60.780            1960     0 11247.305   1  41709
## 2    AITC -62.407            1960     0 15180.183   1  49091
## 3    AITK -60.738            1960     0  9203.926   5  38362
## 4    ALCO -64.240            1960     0 68090.844   0 183991
## 5    AMPH -60.684            1960     0 12532.167   1  35973
## 6    ANDE -60.757            1960     0 13991.602   0  42464
```

```r
popT<-ddply(popy, c("season_starting"), summarise,
            tot=sum(fit), ### total population
            mean=mean(fit)) ### mean population

# create a time stamp for year
popT$TS<-(as.POSIXct(strptime(paste(popT$season_starting,c("01-01"),sep="-"),
                              format="%Y-%m-%d" ,tz="GMT")) )
# create a time stamp for year
popy$TS<-(as.POSIXct(strptime(paste(popy$season_starting,c("01-01"),sep="-"),
                              format="%Y-%m-%d" ,tz="GMT")) )

mts<-xts::xts(popT$tot,order.by=popT$TS) # create a temporal data frame

# create a lag data frame
mlag<-((data.frame(year=popT$season_starting,mts %>%
                     xts::lag.xts(k = c(0,27,28,29,30)))))
mlag
```

```
##            year     lag0   lag27   lag28   lag29   lag30
## 1960-01-01 1960 19994395      NA      NA      NA      NA
## 1961-01-01 1961 13212475      NA      NA      NA      NA
## 1962-01-01 1962 12169578      NA      NA      NA      NA
## 1963-01-01 1963  9840806      NA      NA      NA      NA
## 1964-01-01 1964 14151603      NA      NA      NA      NA
## 1965-01-01 1965 16064435      NA      NA      NA      NA
## 1966-01-01 1966  9371126      NA      NA      NA      NA
## 1967-01-01 1967 13515490      NA      NA      NA      NA
## 1968-01-01 1968 15920406      NA      NA      NA      NA
## 1969-01-01 1969 11130980      NA      NA      NA      NA
## 1970-01-01 1970 11206689      NA      NA      NA      NA
## 1971-01-01 1971 10262036      NA      NA      NA      NA
## 1972-01-01 1972 13885008      NA      NA      NA      NA
## 1973-01-01 1973 17305139      NA      NA      NA      NA
## 1974-01-01 1974 14766263      NA      NA      NA      NA
## 1975-01-01 1975  9843819      NA      NA      NA      NA
## 1976-01-01 1976 11559295      NA      NA      NA      NA
## 1977-01-01 1977 12147044      NA      NA      NA      NA
## 1978-01-01 1978 10766492      NA      NA      NA      NA
```

```
## 1979-01-01 1979 14205934       NA       NA       NA       NA
## 1980-01-01 1980 10758585       NA       NA       NA       NA
## 1981-01-01 1981 10607751       NA       NA       NA       NA
## 1982-01-01 1982  9718237       NA       NA       NA       NA
## 1983-01-01 1983  9478053       NA       NA       NA       NA
## 1984-01-01 1984  7584680       NA       NA       NA       NA
## 1985-01-01 1985 11663767       NA       NA       NA       NA
## 1986-01-01 1986  8423110       NA       NA       NA       NA
## 1987-01-01 1987 10503745 19994395       NA       NA       NA
## 1988-01-01 1988  9539356 13212475 19994395       NA       NA
## 1989-01-01 1989  7299467 12169578 13212475 19994395       NA
## 1990-01-01 1990  9093154  9840806 12169578 13212475 19994395
## 1991-01-01 1991  9281439 14151603  9840806 12169578 13212475
## 1992-01-01 1992  8168403 16064435 14151603  9840806 12169578
## 1993-01-01 1993  8998680  9371126 16064435 14151603  9840806
## 1994-01-01 1994  7313702 13515490  9371126 16064435 14151603
## 1995-01-01 1995  7562148 15920406 13515490  9371126 16064435
## 1996-01-01 1996  8520174 11130980 15920406 13515490  9371126
## 1997-01-01 1997  8065673 11206689 11130980 15920406 13515490
## 1998-01-01 1998  9847905 10262036 11206689 11130980 15920406
## 1999-01-01 1999 10442679 13885008 10262036 11206689 11130980
## 2000-01-01 2000  7973303 17305139 13885008 10262036 11206689
## 2001-01-01 2001  8007181 14766263 17305139 13885008 10262036
## 2002-01-01 2002  5783291  9843819 14766263 17305139 13885008
## 2003-01-01 2003 11330933 11559295  9843819 14766263 17305139
## 2004-01-01 2004 10232773 12147044 11559295  9843819 14766263
## 2005-01-01 2005  7082553 10766492 12147044 11559295  9843819
## 2006-01-01 2006  8000893 14205934 10766492 12147044 11559295
## 2007-01-01 2007  7047888 10758585 14205934 10766492 12147044
## 2008-01-01 2008  7346759 10607751 10758585 14205934 10766492
## 2009-01-01 2009  6852312  9718237 10607751 10758585 14205934
## 2010-01-01 2010  7005599  9478053  9718237 10607751 10758585
## 2011-01-01 2011  6491008  7584680  9478053  9718237 10607751
## 2012-01-01 2012  7600251 11663767  7584680  9478053  9718237
## 2013-01-01 2013  8582740  8423110 11663767  7584680  9478053
## 2014-01-01 2014  7716648 10503745  8423110 11663767  7584680
## 2015-01-01 2015  7462203  9539356 10503745  8423110 11663767
## 2016-01-01 2016  8397280  7299467  9539356 10503745  8423110
## 2017-01-01 2017 10387723  9093154  7299467  9539356 10503745
## 2018-01-01 2018  6169821  9281439  9093154  7299467  9539356
## 2019-01-01 2019  9374071  8168403  9281439  9093154  7299467
## 2020-01-01 2020  7652306  8998680  8168403  9281439  9093154
```

```r
# proportional change for all lags
mlag$ch3<-(mlag$lag0/mlag$lag27)-1
mlag$ch4<-(mlag$lag0/mlag$lag28)-1
mlag$ch5<-(mlag$lag0/mlag$lag29)-1
mlag$ch6<-(mlag$lag0/mlag$lag30)-1

mlags<-data.frame(year=mlag$year,mlag[7:10])

chm<-na.omit(melt(mlags,id.vars="year"))
summary(chm$value)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -0.6658 -0.3972 -0.2922 -0.2597 -0.1201  0.4231
```

```
quantile(chm$value,probs=0.95)
```

```
##       95%
## 0.1123939
```

```
quantile(chm$value,probs=0.05)
```

```
##         5%
## -0.5383713
```

```
mean(chm$value)
```

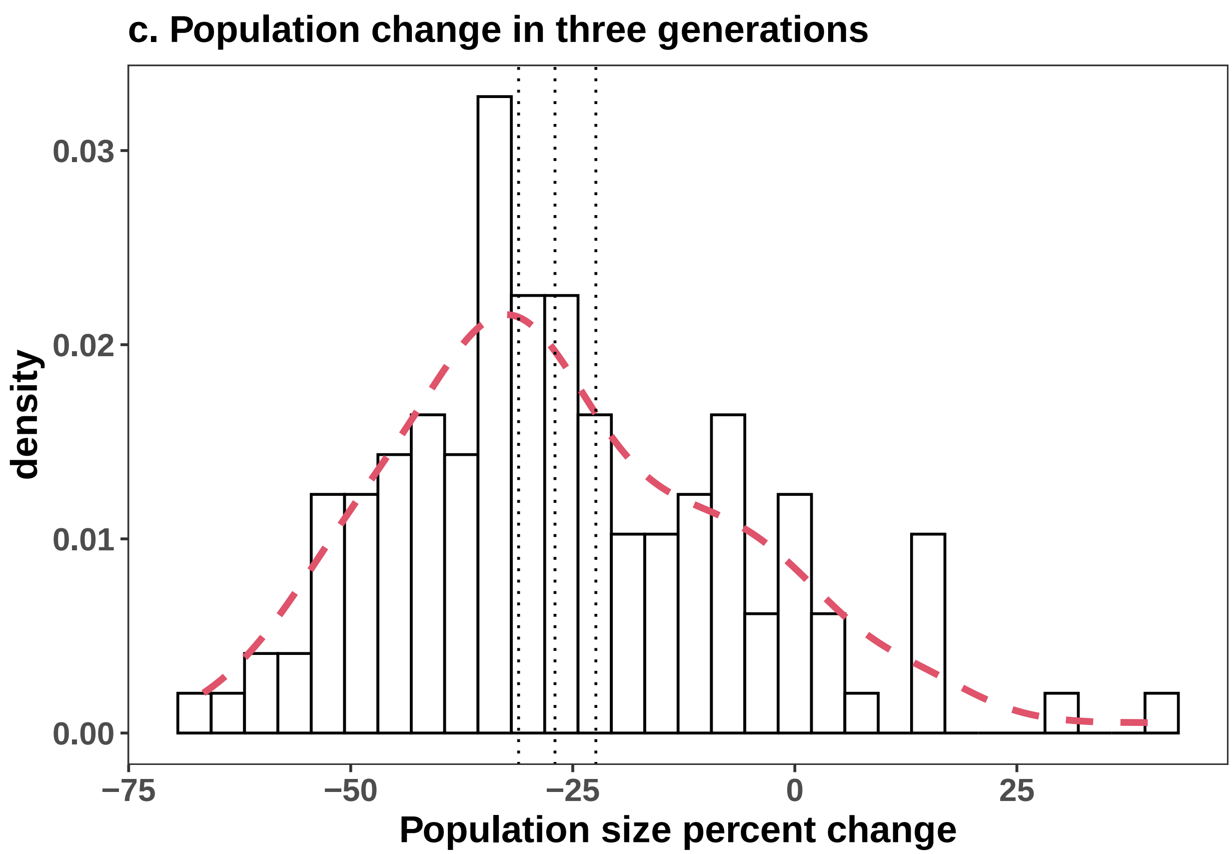
```
## [1] -0.2596547
```

```
sd(chm$value)
```

```
## [1] 0.1997553
```

```
p3<-ggplot(chm,aes(value*100))+
  geom_histogram(aes(y = ..density..), colour = 1, fill = "white") +
  geom_density(lwd = 1.2, linetype = 2,colour = 2)+
  theme_bw()+th+
  geom_vline(xintercept = c(-22.4,-27.0,-31.1),linetype="dotted")+
  xlab("Population size percent change")+
  ggtitle(label="c. Population change in three generations")

p3
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
# p1v2/p2/p3
```

Analysis for Oosthuizen et al. (current)

Fit a better GLMM

```
# head(nestM3)

# How is this model different to Kruger (2023)?
# 1) We used a different model specification for fixed and random effects
# 2) We z-standardized the covariates before running the model
# 3) We used longer mcmc chains
# 4) When predicting from the fitted model, we did not marginalise the random effects

# Covariates should be standardized.
nestM3$Zseason_starting = scale(nestM3$season_starting)
nestM3$ZLat = scale(nestM3$Lat)
```

```
mc2 <- MCMCglmm(nests ~ Zseason_starting * ZLat,
                random=~us(1 + Zseason_starting):site_id,
                rcov=~units,
              family="poisson", mev=NULL,
```

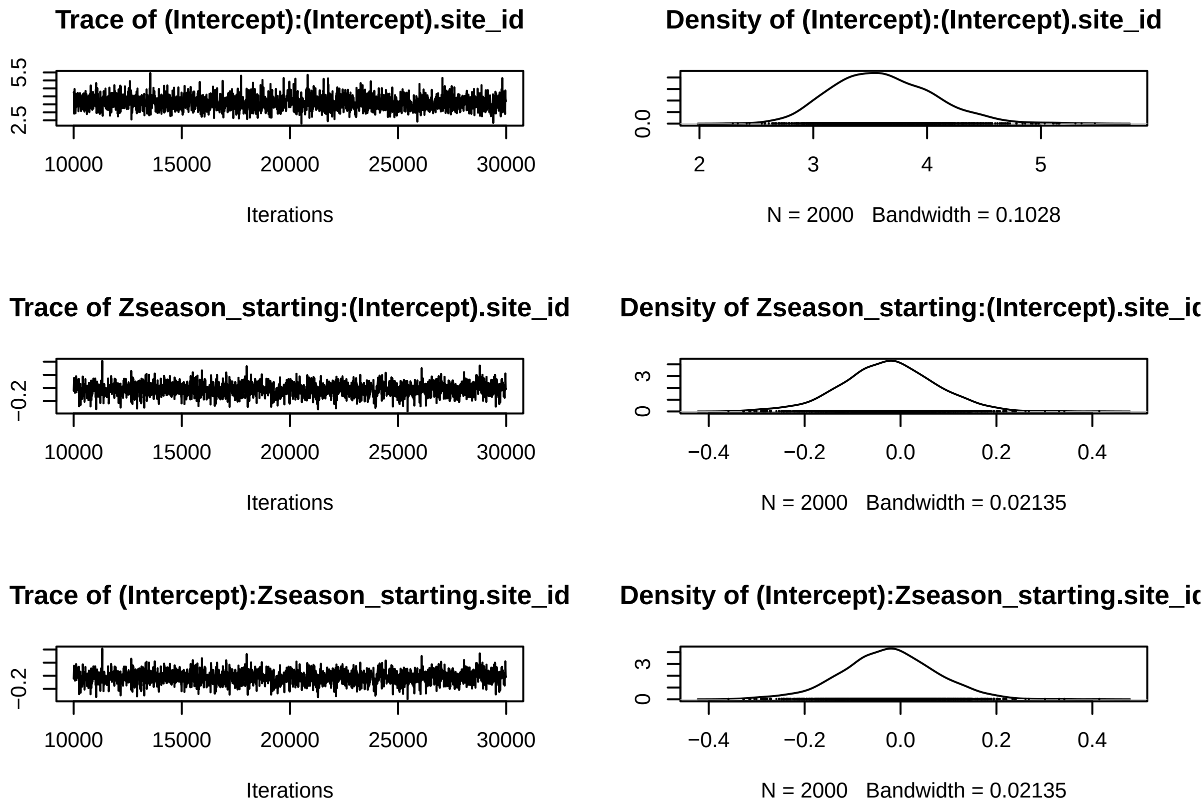
```
             data=nestM3,start=NULL, nodes="ALL", scale=TRUE,
             nitt=30000, thin=10, burnin=10000, pr=T,
             pl=FALSE, verbose=TRUE, DIC=TRUE, singular.ok=FALSE, saveX=TRUE,
             prior=prior, saveZ=TRUE, saveXL=TRUE, slice=FALSE,
             ginverse=NULL, trunc=FALSE)
```

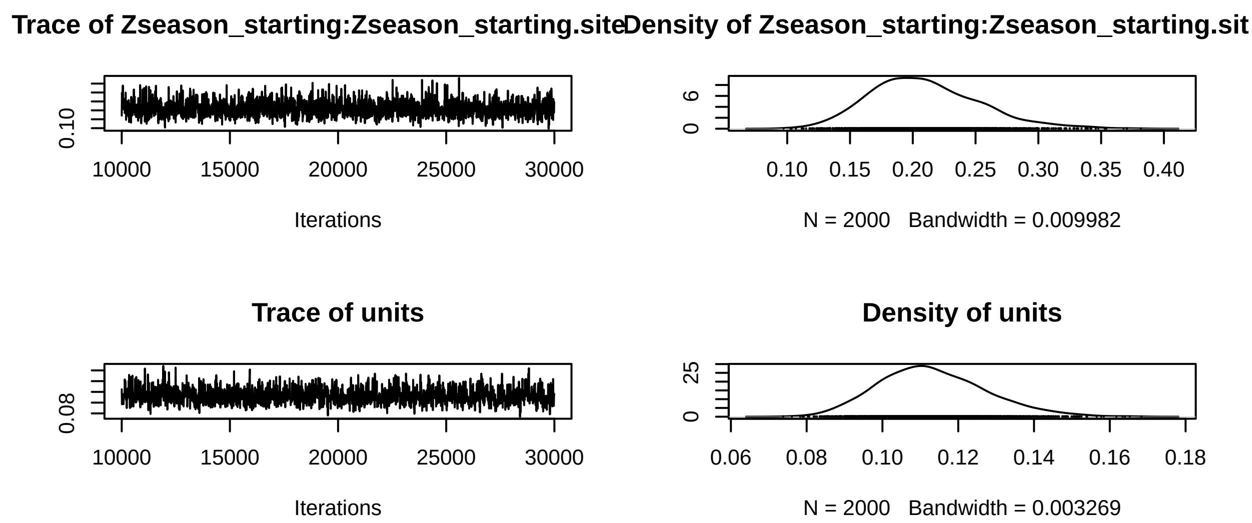
```
summary(mc2)
```

```
##
##  Iterations = 10001:29991
##  Thinning interval  = 10
##  Sample size  = 2000
##
##  DIC: 4792.062
##
##  G-structure:  ~us(1 + Zseason_starting):site_id
##
##                                             post.mean l-95% CI u-95% CI eff.samp
## (Intercept):(Intercept).site_id               3.6112   2.8466   4.5155     2000
## Zseason_starting:(Intercept).site_id         -0.0245  -0.2166   0.1764     1433
## (Intercept):Zseason_starting.site_id         -0.0245  -0.2166   0.1764     1433
## Zseason_starting:Zseason_starting.site_id     0.2096   0.1248   0.2940     2000
##
##  R-structure:  ~units
##
##       post.mean l-95% CI u-95% CI eff.samp
## units    0.1133  0.08659   0.1413     1465
##
##  Location effects: nests ~ Zseason_starting * ZLat
##
##                        post.mean l-95% CI u-95% CI eff.samp  pMCMC
## (Intercept)              5.81684  5.46729  6.14416     1320 <5e-04 ***
## Zseason_starting        -0.18069 -0.27842 -0.06831     2000  0.002 **
## ZLat                     1.42719  1.09686  1.73423     2000 <5e-04 ***
## Zseason_starting:ZLat    0.05033 -0.04286  0.14509     1725  0.306
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

MCMCglmm diagnostics for mc2

```
# Assessing model convergence. We do this separately for both fixed
# and random effects. The trace plot should look like a fuzzy caterpillar
# plot(mc2$Sol)

# variances of the random effects - shows good mixing
plot(mc2$VCV)
```


Trace of (Intercept):(Intercept).site_id
2.5
5.5
10000
15000
20000
25000
30000
Iterations
Density of (Intercept):(Intercept).site_id
0.0
2
3
4
5
N = 2000 Bandwidth = 0.1028
Trace of Zseason_starting:(Intercept).site_id
−0.2
10000
15000
20000
25000
30000
Iterations
Density of Zseason_starting:(Intercept).site_ic
0
3
−0.4
−0.2
0.0
0.2
0.4
N = 2000 Bandwidth = 0.02135
Trace of (Intercept):Zseason_starting.site_id
−0.2
10000
15000
20000
25000
30000
Iterations
Density of (Intercept):Zseason_starting.site_ic
0
3
−0.4
−0.2
0.0
0.2
0.4
N = 2000 Bandwidth = 0.02135




```r
# what are the effective sample size for the random effects?
coda::effectiveSize(mc2$VCV)
```

```
##           (Intercept):(Intercept).site_id
##                                  2000.000
##      Zseason_starting:(Intercept).site_id
##                                  1432.780
##      (Intercept):Zseason_starting.site_id
##                                  1432.780
## Zseason_starting:Zseason_starting.site_id
##                                  2000.000
##                                     units
##                                  1465.188
```

```r
# The effective sample size is large

# from MCMC Course notes (page 60):
diag(autocorr(mc2$VCV)[2, , ])    # low autocorrelation
```

```
##           (Intercept):(Intercept).site_id
##                               0.007545644
##      Zseason_starting:(Intercept).site_id
##                               0.078267025
##      (Intercept):Zseason_starting.site_id
##                               0.078267025
```

```
## Zseason_starting:Zseason_starting.site_id
##                               0.031064450
##                                     units
##                               0.154094402
```

Predict using MCMCglmm mc2

```r
# construct an hypothetical dataframe to predict to

# need to predict to z-standardized variables
Z1 = dplyr::select(nestM3, season_starting, Lat)
Z2 <- scale(Z1)
attr(Z2,"scaled:center")
```

```
## season_starting             Lat
##      2003.32985       -62.96523
```

```r
attr(Z2,"scaled:scale")
```

```
## season_starting             Lat
##       15.685180        1.600454
```

```r
ave_ss = attr(Z2,"scaled:center")[[1]]
ave_lat = attr(Z2,"scaled:center")[[2]]

sd_ss = attr(Z2,"scaled:scale")[[1]]
sd_lat = attr(Z2,"scaled:scale")[[2]]

years<-data.frame(season_starting=c(1960:2020))
pops<-data.frame(site_id=countsN2$site_id[countsN2$ncounts>1],
                 Lat=countsN2$Lat[countsN2$ncounts>1])
popy<-merge(pops,years)
popy$nests<-c(0) ### MCMCglmm needs a column with the response variable

popy$Zseason_starting = (popy$season_starting - ave_ss)/sd_ss
popy$ZLat = (popy$Lat - ave_lat)/sd_lat

head(popy)
```

```
##   site_id     Lat season_starting nests Zseason_starting       ZLat
## 1    AILS -60.780            1960     0        -2.762471  1.3653784
## 2    AITC -62.407            1960     0        -2.762471  0.3487919
## 3    AITK -60.738            1960     0        -2.762471  1.3916209
## 4    ALCO -64.240            1960     0        -2.762471 -0.7965080
## 5    AMPH -60.684            1960     0        -2.762471  1.4253614
## 6    ANDE -60.757            1960     0        -2.762471  1.3797493
```

```r
popypred <- data.frame(predict(mc2,
                               newdata=popy,
                               type="response",
```

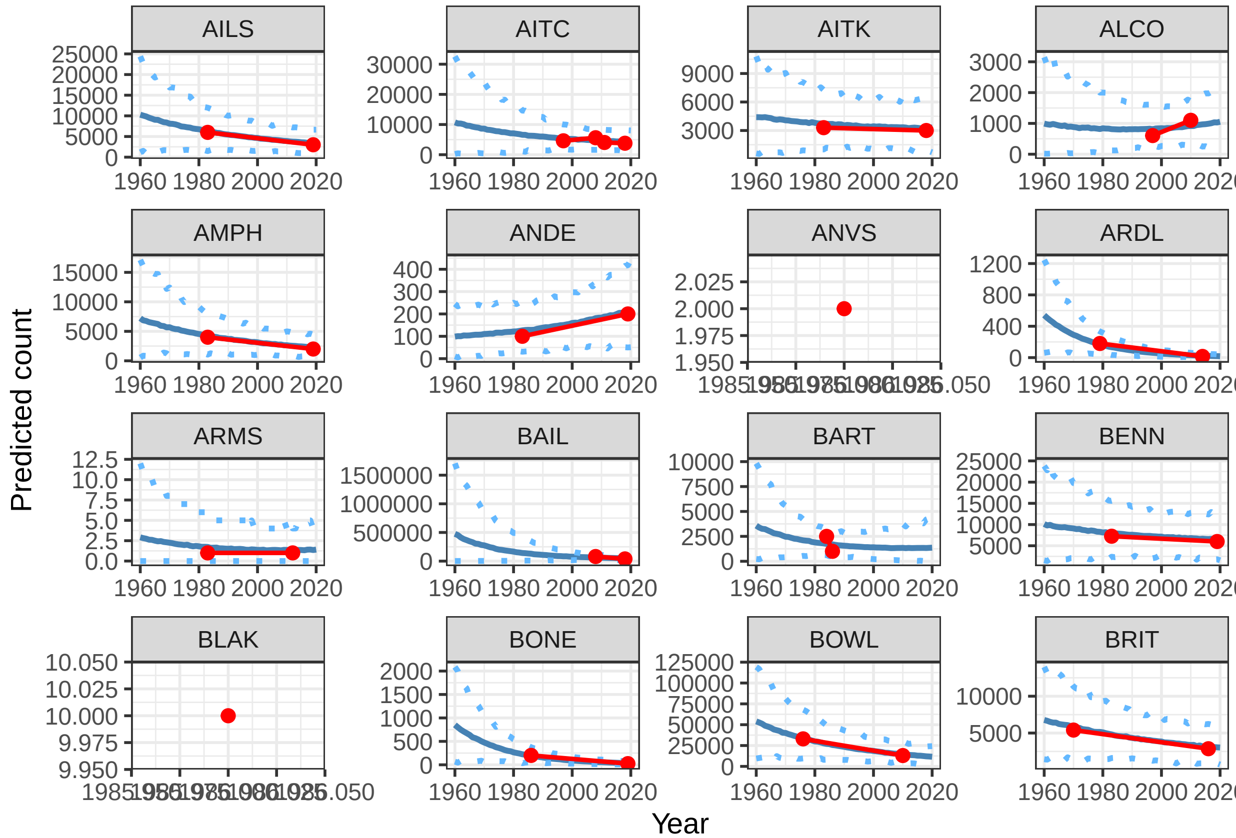
```
                              marginal=NULL,      # crucial, and not default code.
                              interval="prediction",
                              posterior="all"))

head(popypred)
```

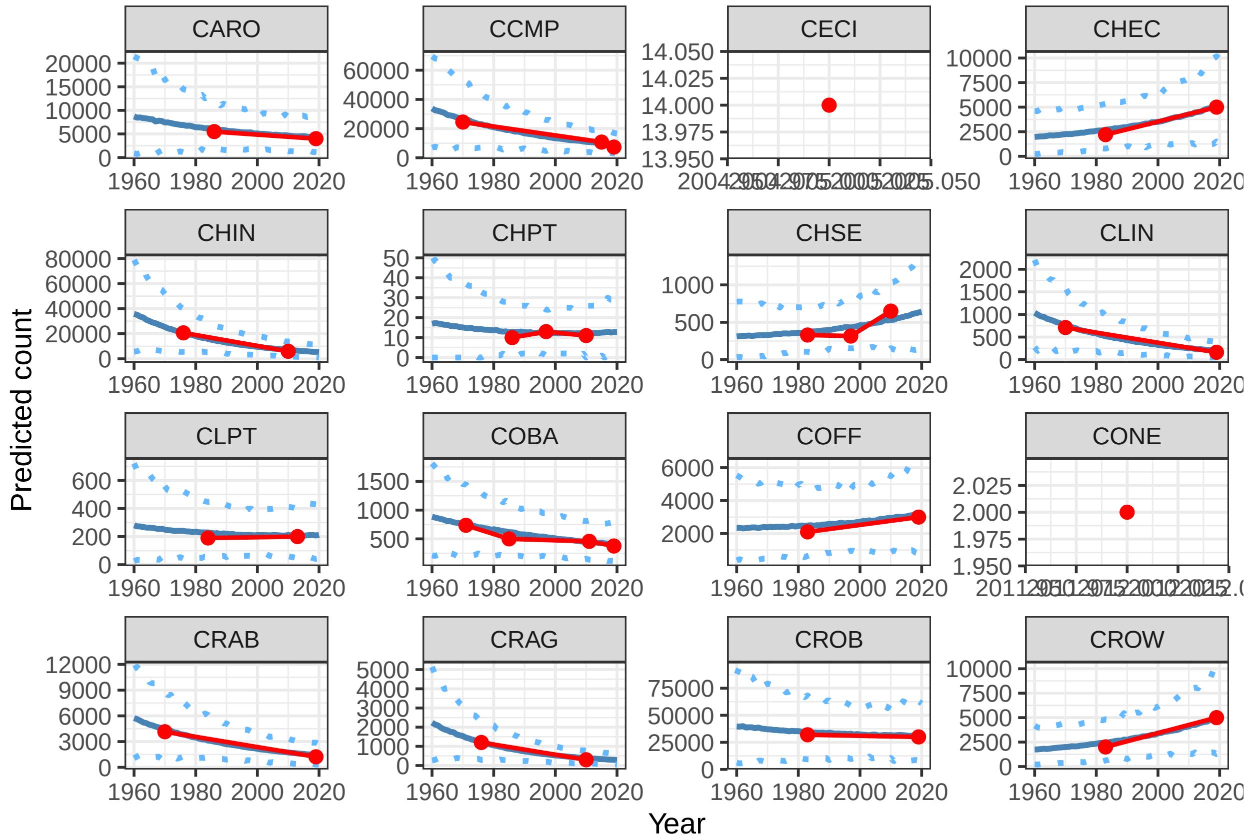
```
##         fit  lwr   upr
## 1 10280.171 1180 24406
## 2 10720.798  229 32606
## 3  4404.931  589 10807
## 4   993.215   18  3148
## 5  7220.764  560 17115
## 6    98.630   10   245
```

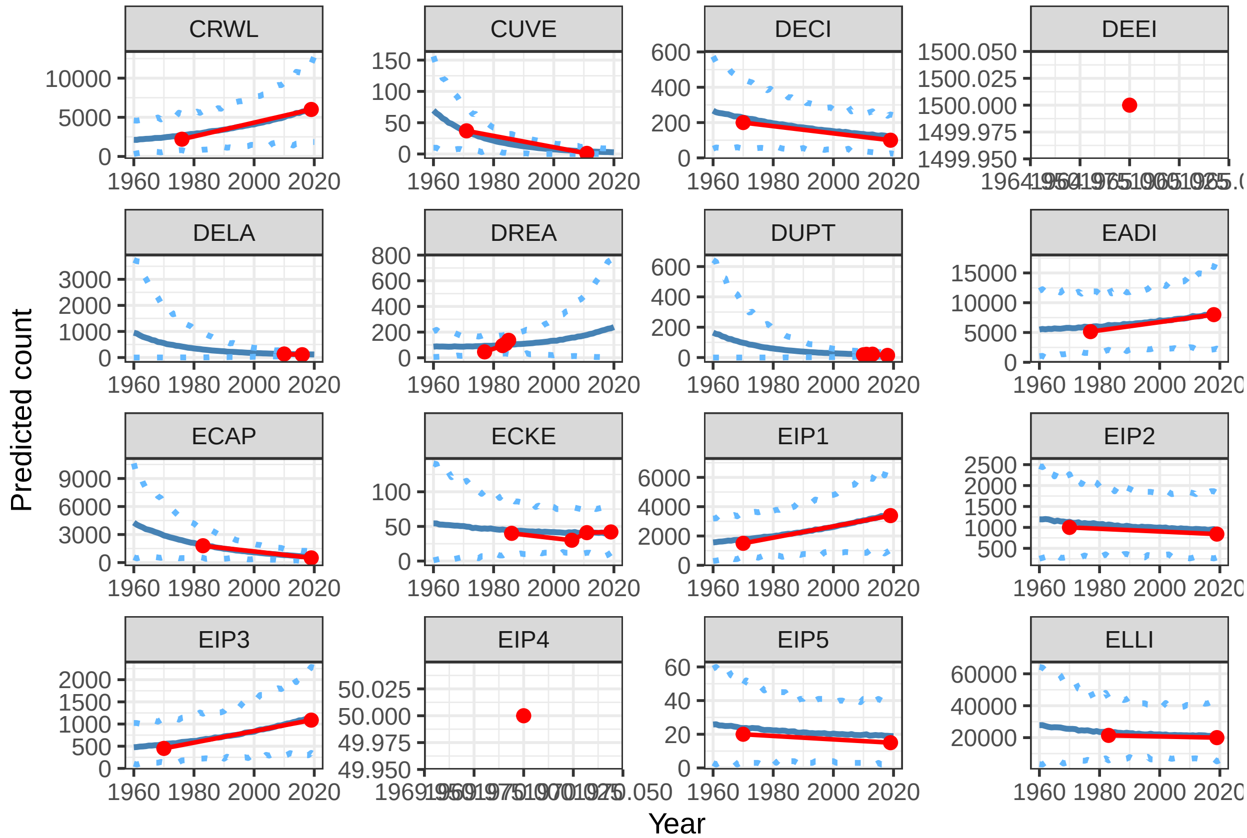
```
popy$Zfit = popypred$fit
popy$Zlwr = popypred$lwr
popy$Zupr = popypred$upr

## How accurate are the predictions relative to observed data?
```

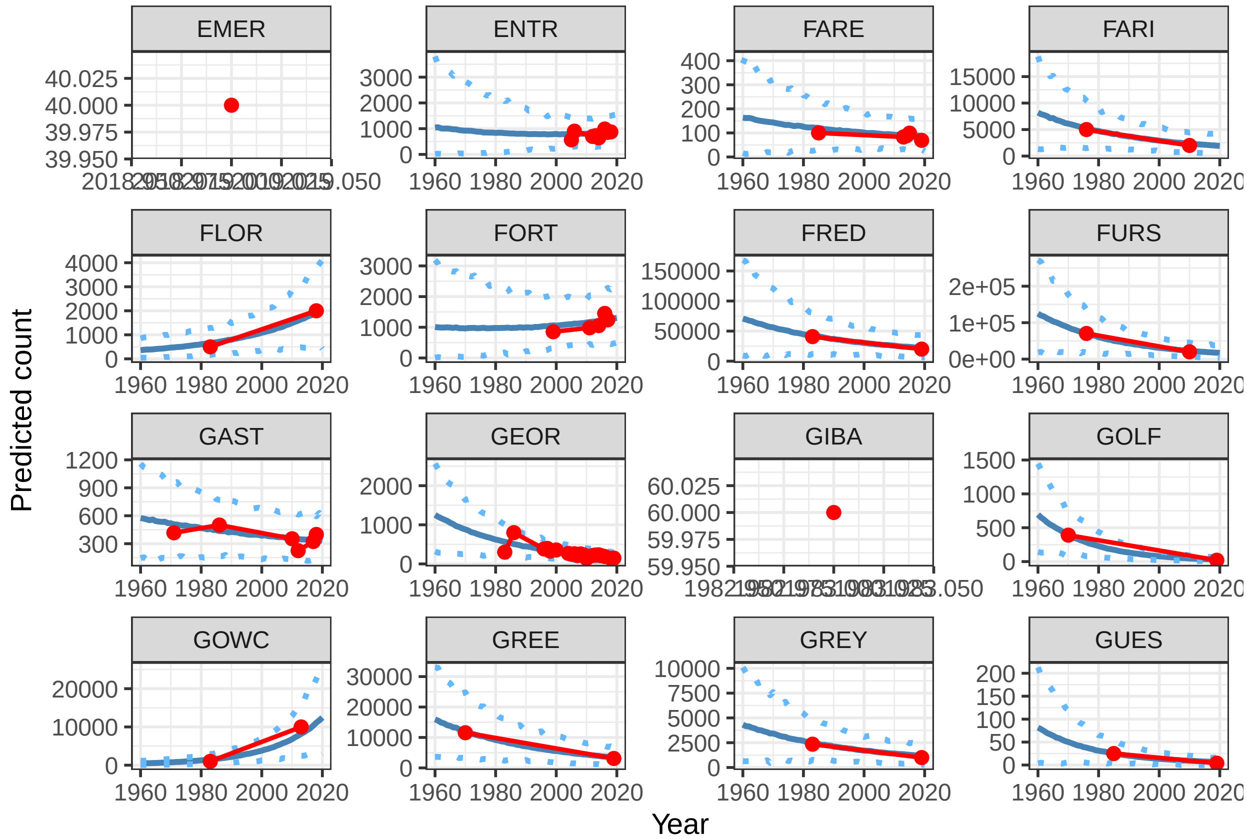
Conditional model predictions

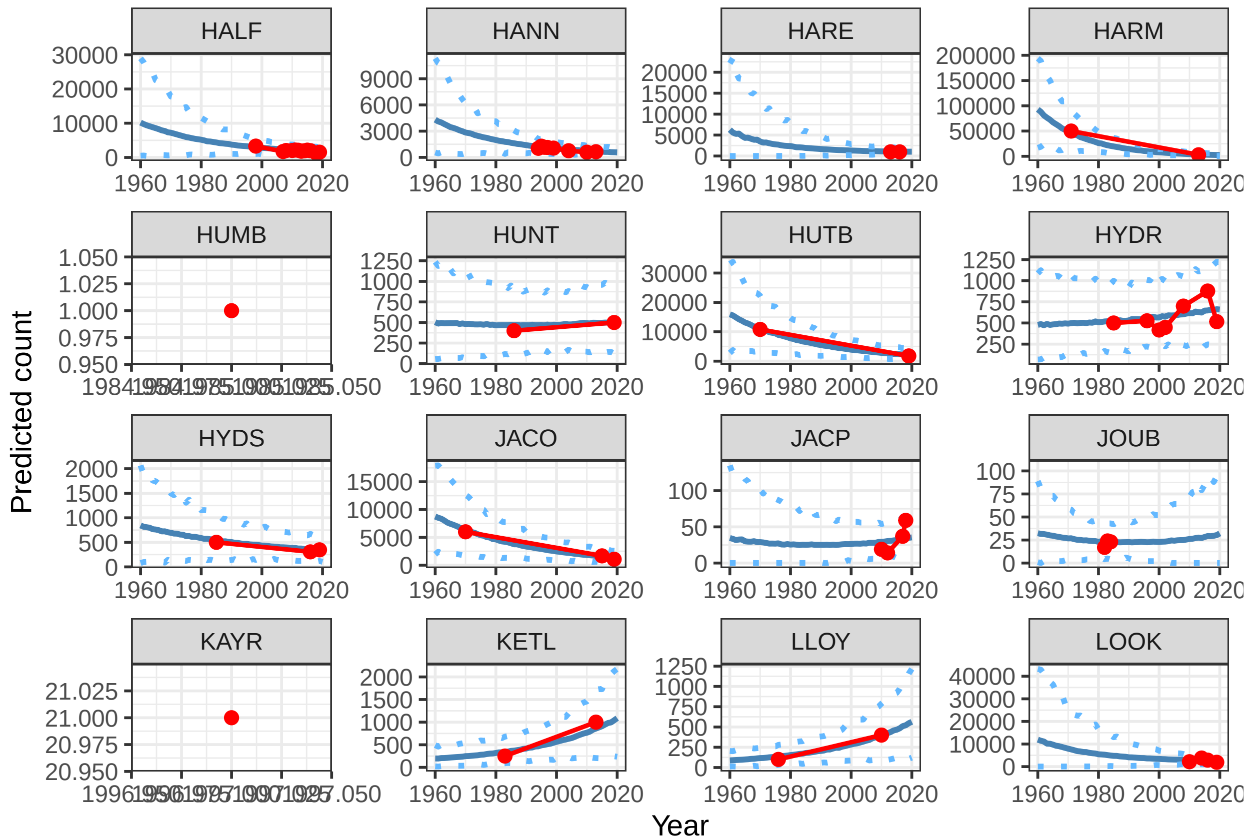
```
required_n_pages = round(133/16)+1

for(i in 1:required_n_pages){

print(ggplot(data = popy) +
    geom_line(aes(x = season_starting, y = Zfit),
              col = "steelblue", linewidth=1.04) +
    geom_line(aes(x = season_starting, y = Zlwr),
              col = "steelblue1", linetype="dotted", linewidth = 1.02) +
    geom_line(aes(x = season_starting, y = Zupr),
              col = "steelblue1", linetype="dotted", linewidth=1.02) +
    geom_point(data = nestm3, aes(season_starting, y = nests),
               color = "red", cex = 2) +
    geom_line(data = nestm3, aes(season_starting, y = nests),
              color = "red",size=0.8) +
    theme_bw() +
    xlab("Year") +
    ylab("Predicted count") +
  #  theme(strip.text = element_text(size = 1.5)) +
    facet_wrap_paginate(~ site_id, ncol = 4, nrow = 4,
                        page = i,
                        scales = 'free'))}
```

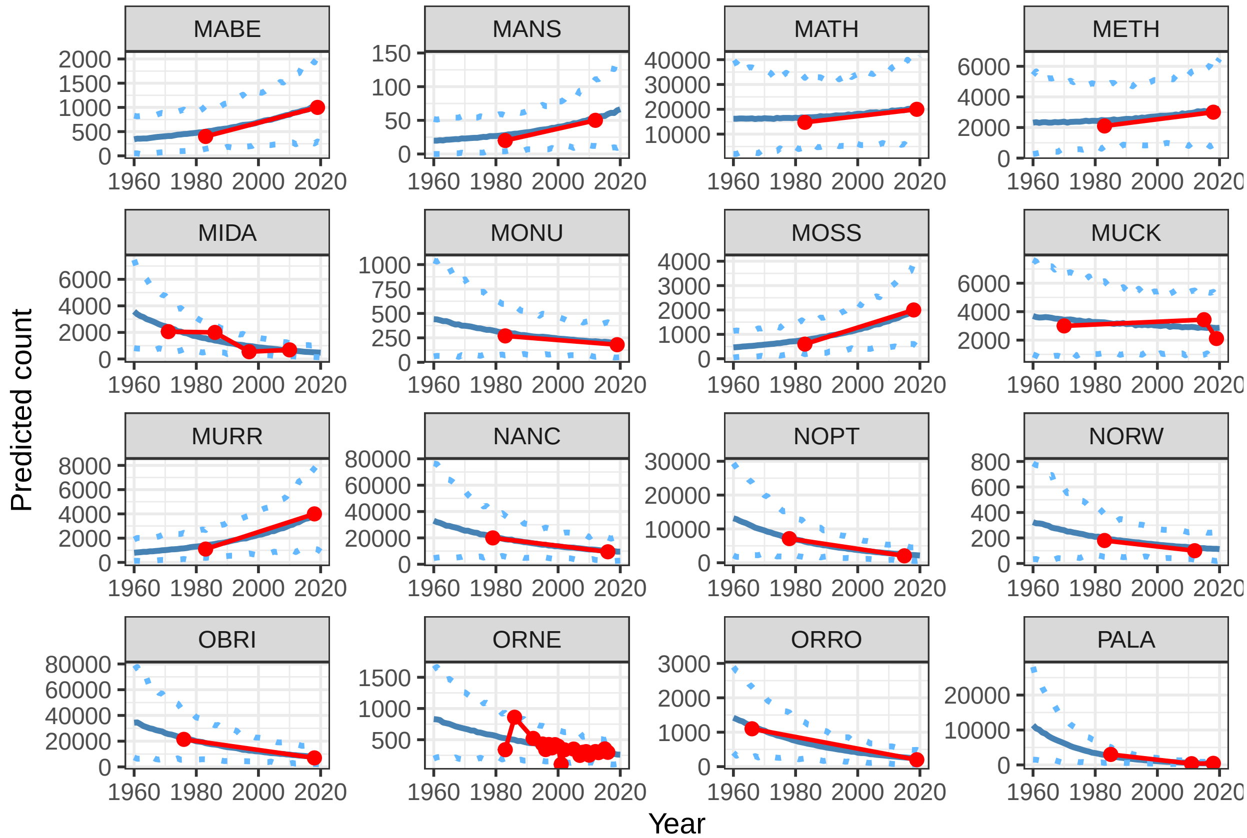

AILS
AITC
AITK
ALCO
AMPH
ANDE
ANVS
ARDL
ARMS
BAIL
BART
BENN
BLAK
BONE
BOWL
BRIT
Predicted count
Year

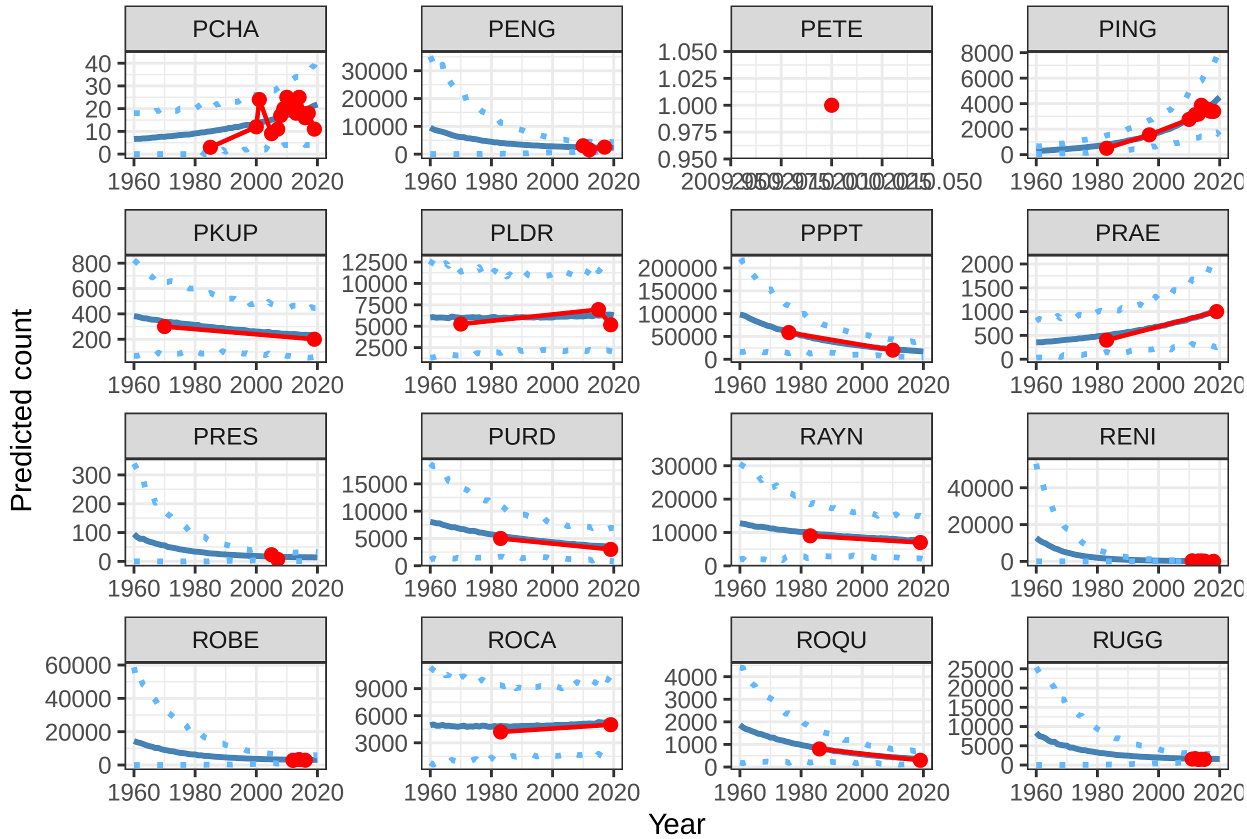

CARO
CCMP
CECI
CHEC
CHIN
CHPT
CHSE
CLIN
CLPT
COBA
COFF
CONE
CRAB
CRAG
CROB
CROW
Predicted count
Year

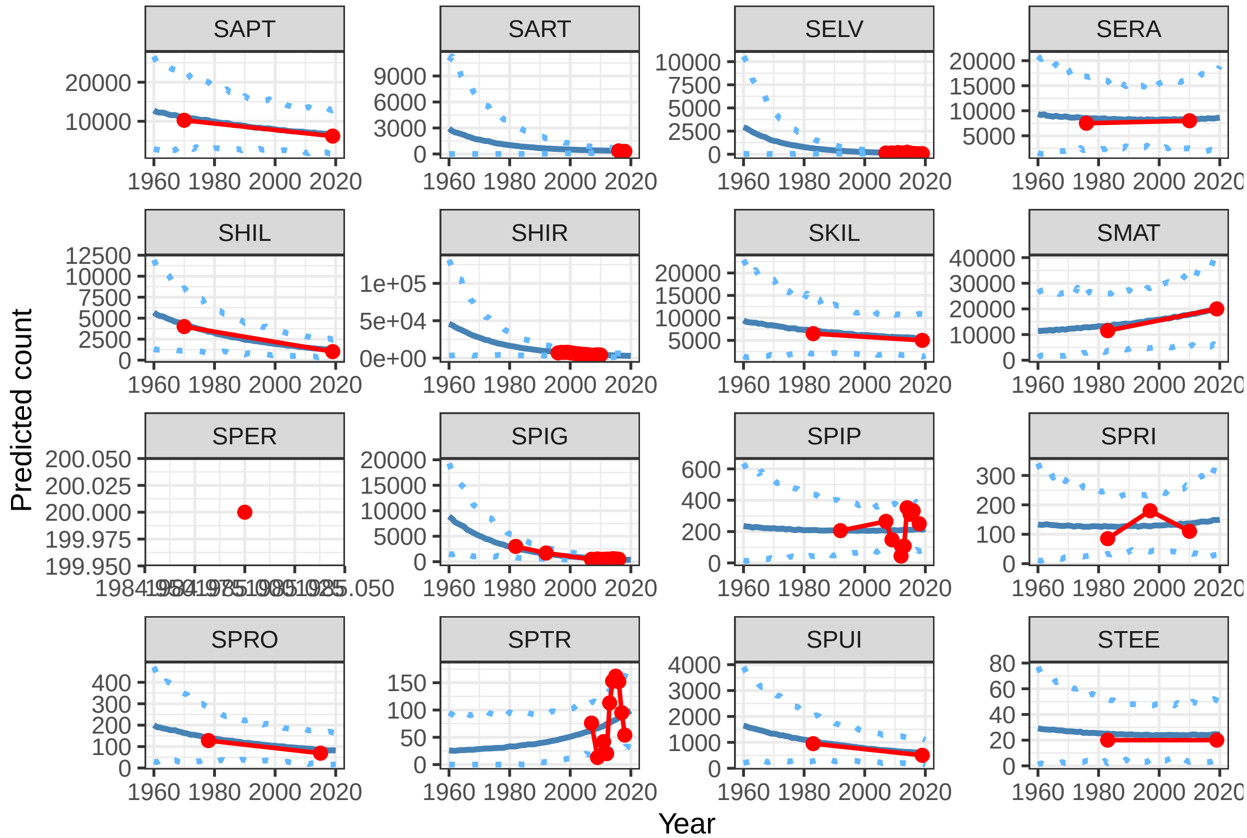

CRWL
CUVE
DECI
DEEI
DELA
DREA
DUPT
EADI
ECAP
ECKE
EIP1
EIP2
EIP3
EIP4
EIP5
ELLI
Predicted count
Year

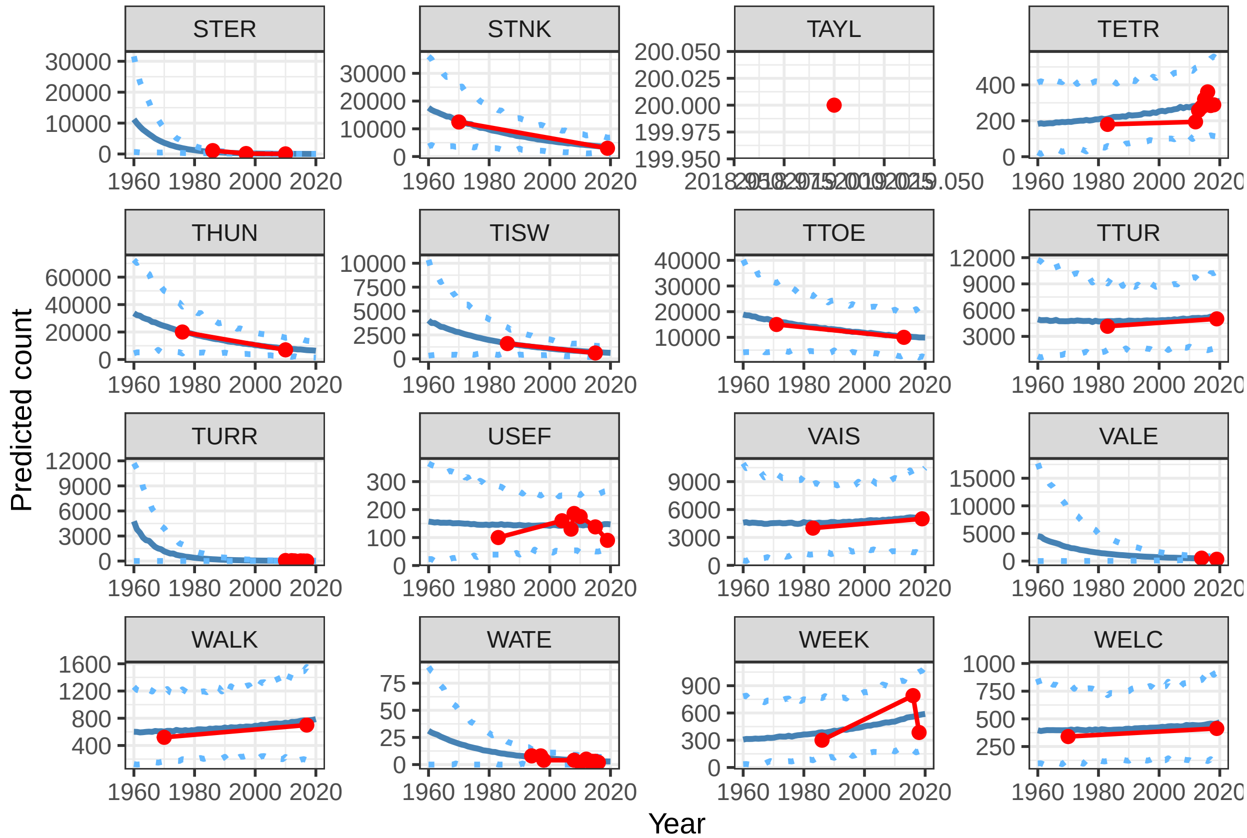

EMER
ENTR
FARE
FARI
FLOR
FORT
FRED
FURS
GAST
GEOR
GIBA
GOLF
GOWC
GREE
GREY
GUES
Predicted count
Year


HALF
HANN
HARE
HARM
HUMB
HUNT
HUTB
HYDR
HYDS
JACO
JACP
JOUB
KAYR
KETL
LLOY
LOOK
Predicted count
Year


MABE
MANS
MATH
METH
MIDA
MONU
MOSS
MUCK
MURR
NANC
NOPT
NORW
OBRI
ORNE
ORRO
PALA
Predicted count
Year
1960 1980 2000 2020


PCHA
PENG
PETE
PING
PKUP
PLDR
PPPT
PRAE
PRES
PURD
RAYN
RENI
ROBE
ROCA
ROQU
RUGG
Predicted count
Year


SAPT
SART
SELV
SERA
SHIL
SHIR
SKIL
SMAT
SPER
SPIG
SPIP
SPRI
SPRO
SPTR
SPUI
STEE
Predicted count
Year

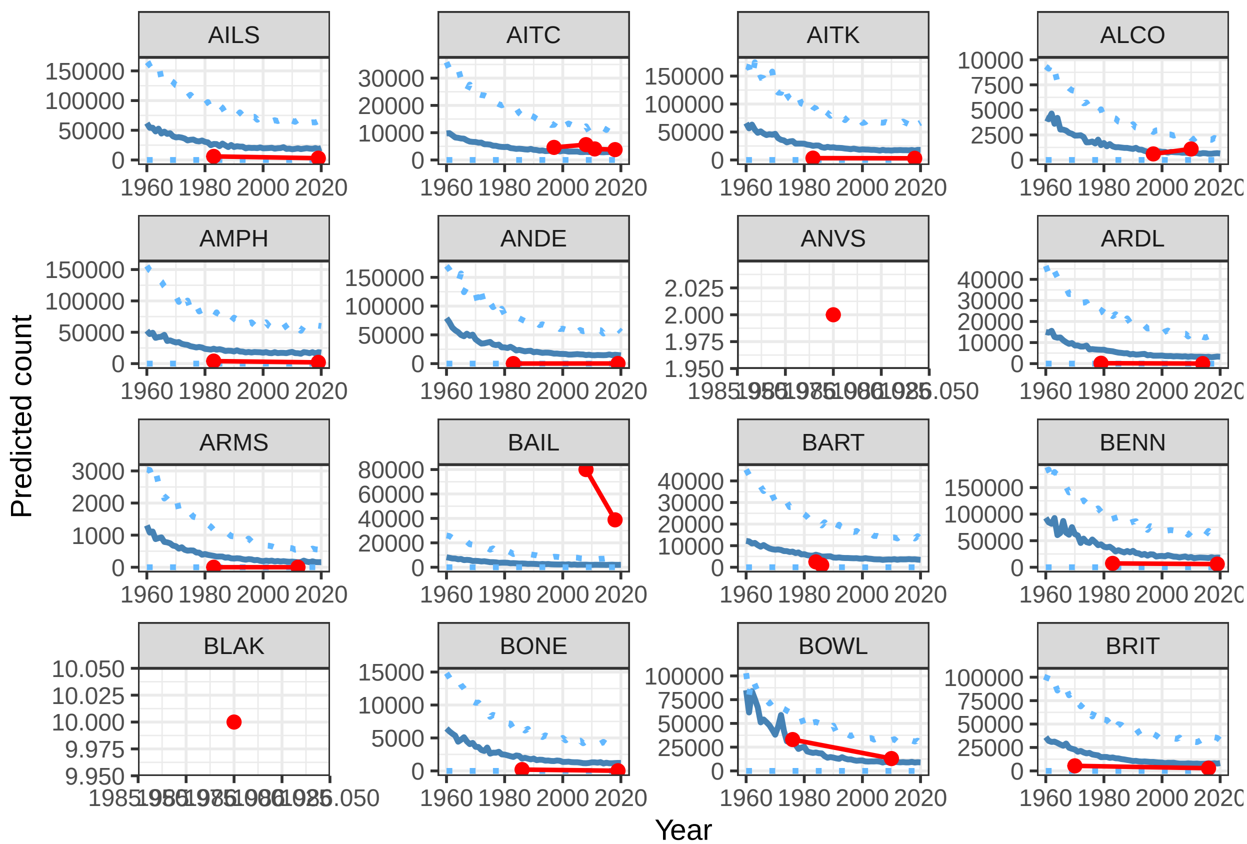


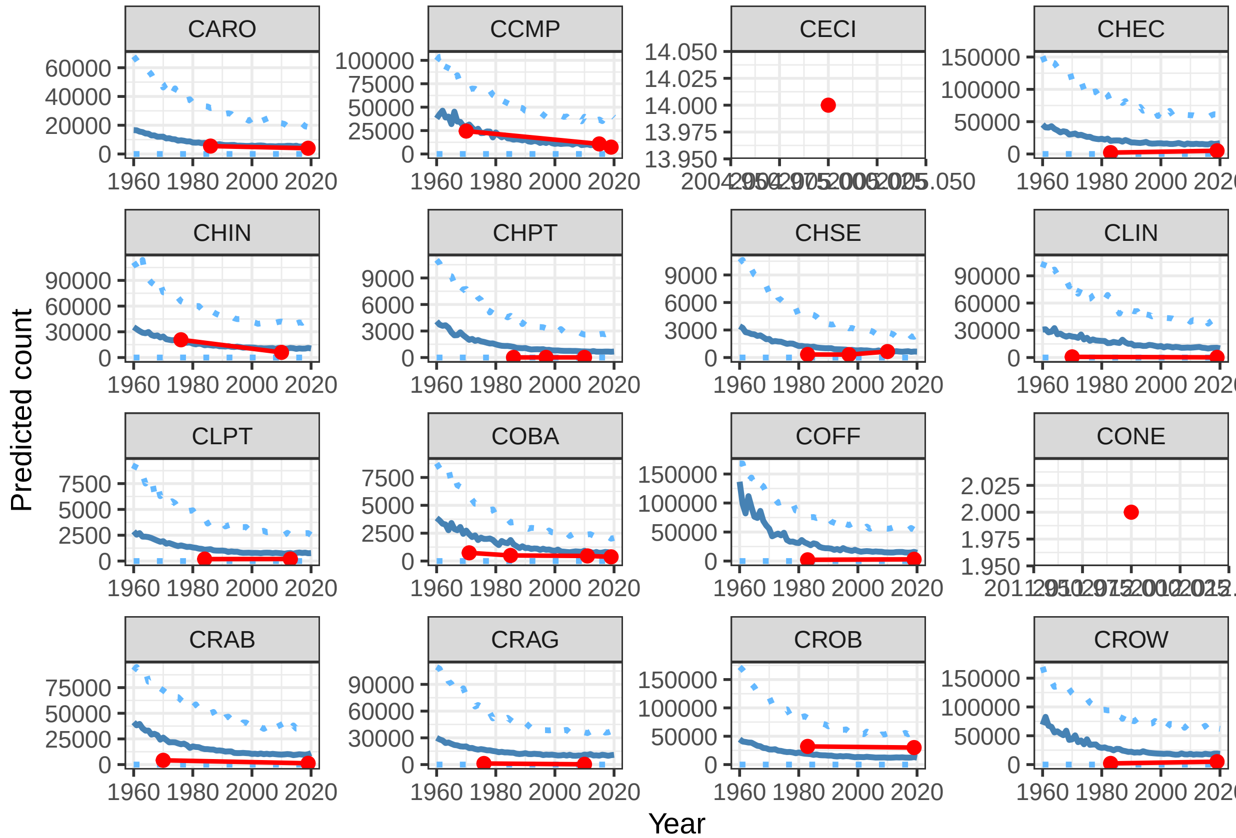

```
# Predictions are good, although back-predicting to 1960 is extrapolation
# (there are only 2 counts prior to 1970) so uncertainty (prediction intervals)
# is high.
```

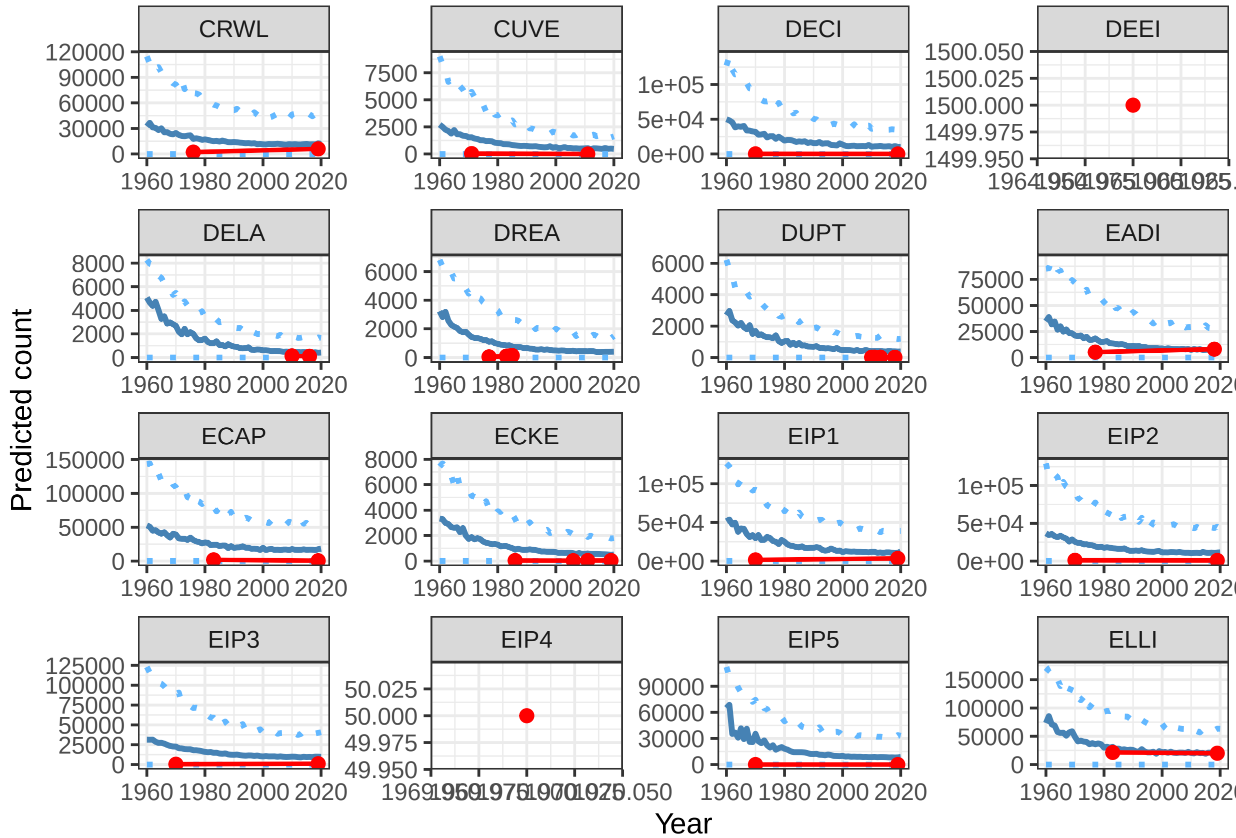
Marginal model predictions

```
# This predicts the population average response - i.e., a similar prediction at all sites.
# If you are adding up the individual sites, as Kruger 2023 did to calculate popT,
# then you are adding predictions of the average response every time.


popypred_marg <- data.frame(predict(mc2,
                             newdata=popy,
                             type="response",
                             #marginal=NULL,      # crucial, and not default code.
                             marginal=~us(1 + Zseason_starting):site_id,
                             interval="prediction",
                             posterior="all"))

popy$Zfit_marg = popypred_marg$fit
popy$Zlwr_marg = popypred_marg$lwr
popy$Zupr_marg = popypred_marg$upr

## How accurate are the predictions relative to observed data?
```

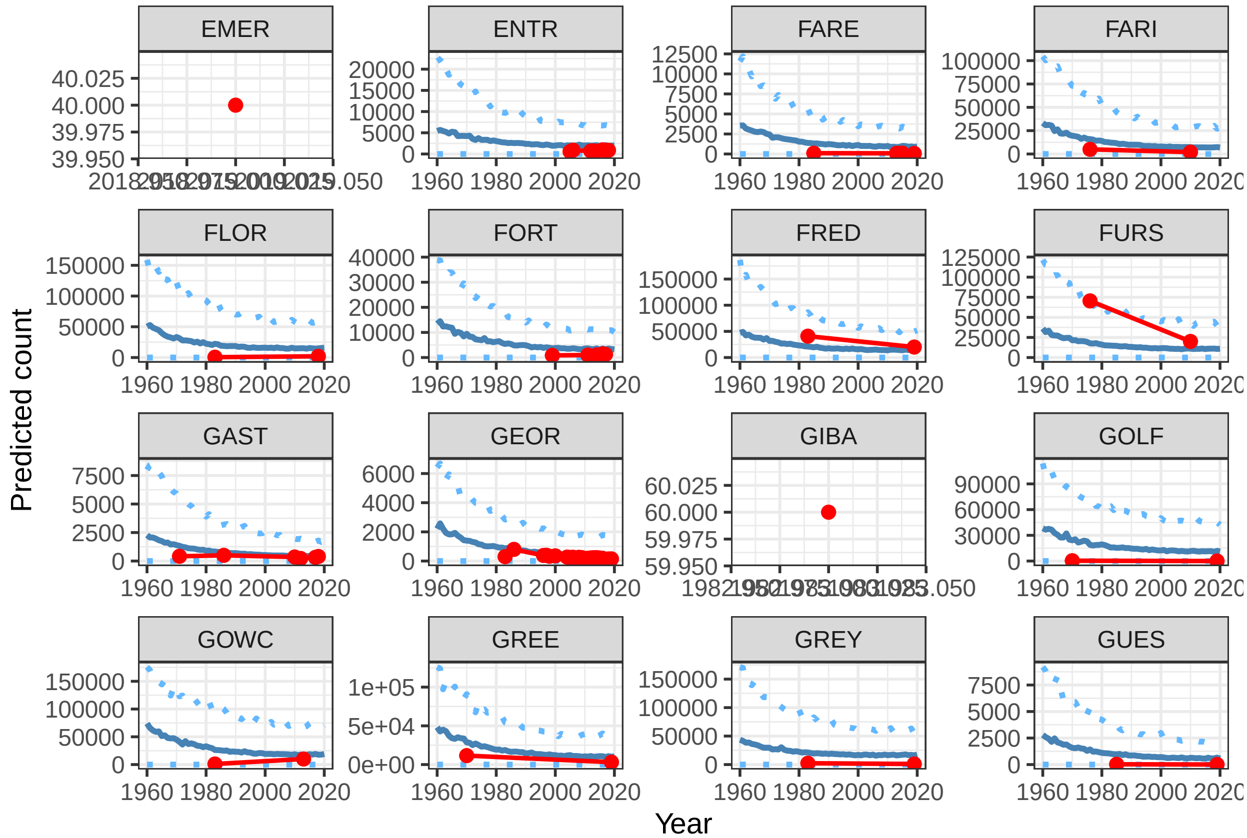
```r
required_n_pages = round(133/16)+1

for(i in 1:required_n_pages){

print(ggplot(data = popy) +
    geom_line(aes(x = season_starting, y = Zfit_marg),
              col = "steelblue", linewidth=1.04) +
    geom_line(aes(x = season_starting, y = Zlwr_marg),
              col = "steelblue1", linetype="dotted", linewidth = 1.02) +
    geom_line(aes(x = season_starting, y = Zupr_marg),
              col = "steelblue1", linetype="dotted", linewidth=1.02) +
    geom_point(data = nestm3, aes(season_starting, y = nests),
               color = "red", cex = 2) +
    geom_line(data = nestm3, aes(season_starting, y = nests),
              color = "red",size=0.8) +
    theme_bw() +
    xlab("Year") +
    ylab("Predicted count") +
  #  theme(strip.text = element_text(size = 1.5)) +
    facet_wrap_paginate(~ site_id, ncol = 4, nrow = 4,
                        page = i,
                        scales = 'free'))}
```

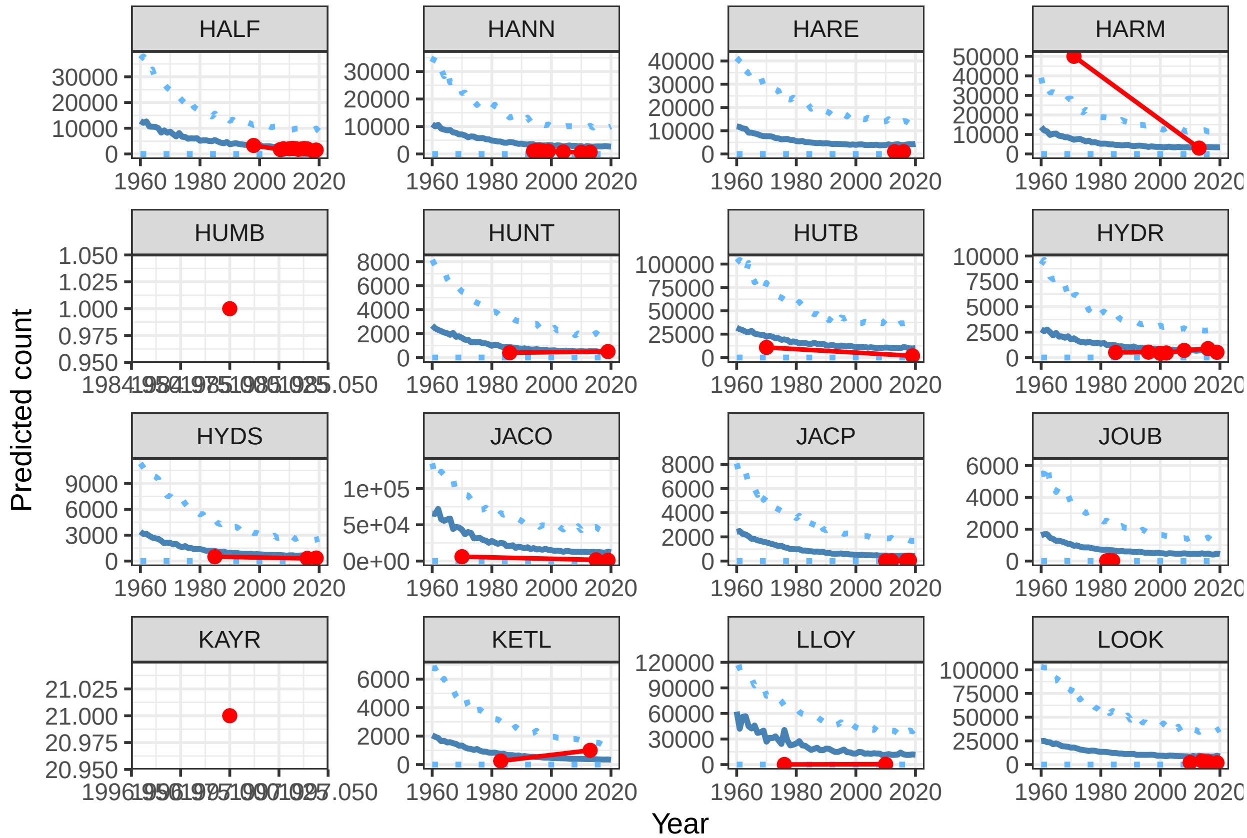


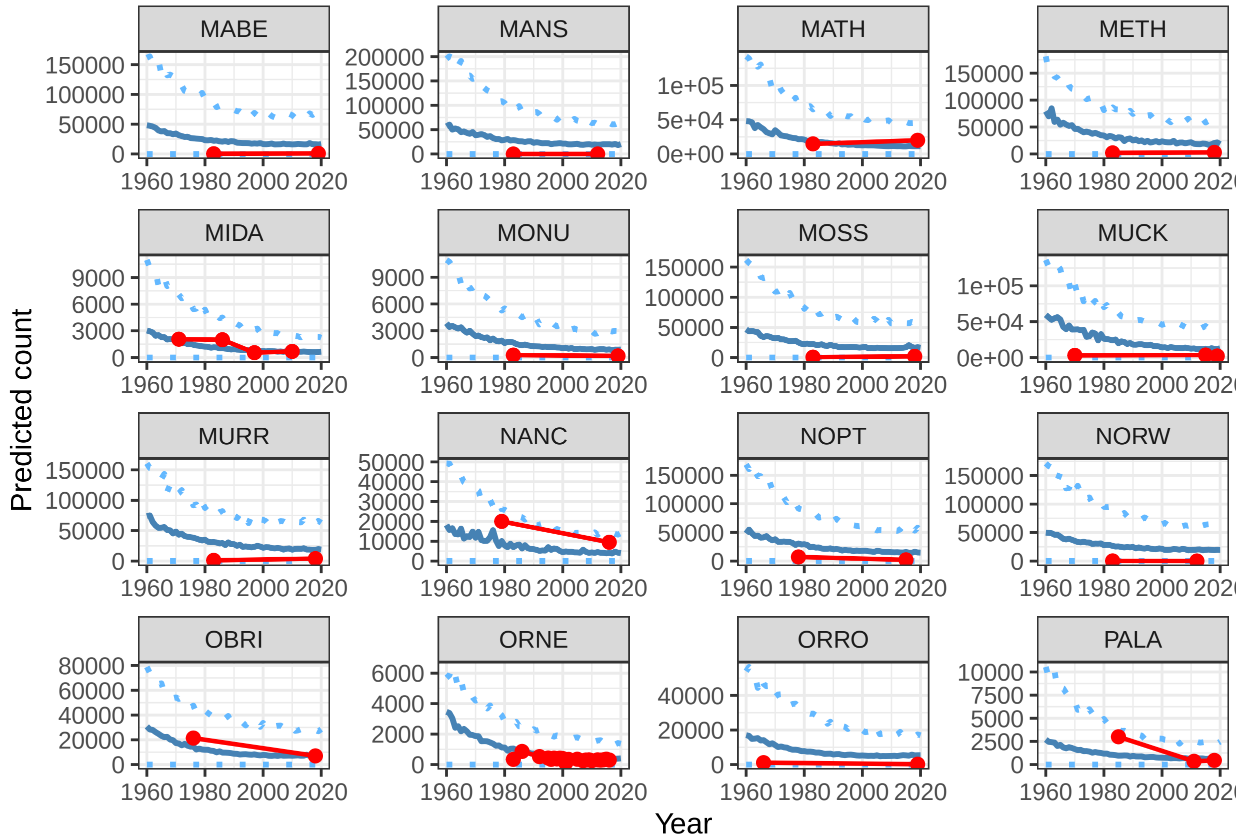

CARO
CCMP
CECI
CHEC
CHIN
CHPT
CHSE
CLIN
CLPT
COBA
COFF
CONE
CRAB
CRAG
CROB
CROW
Predicted count
Year



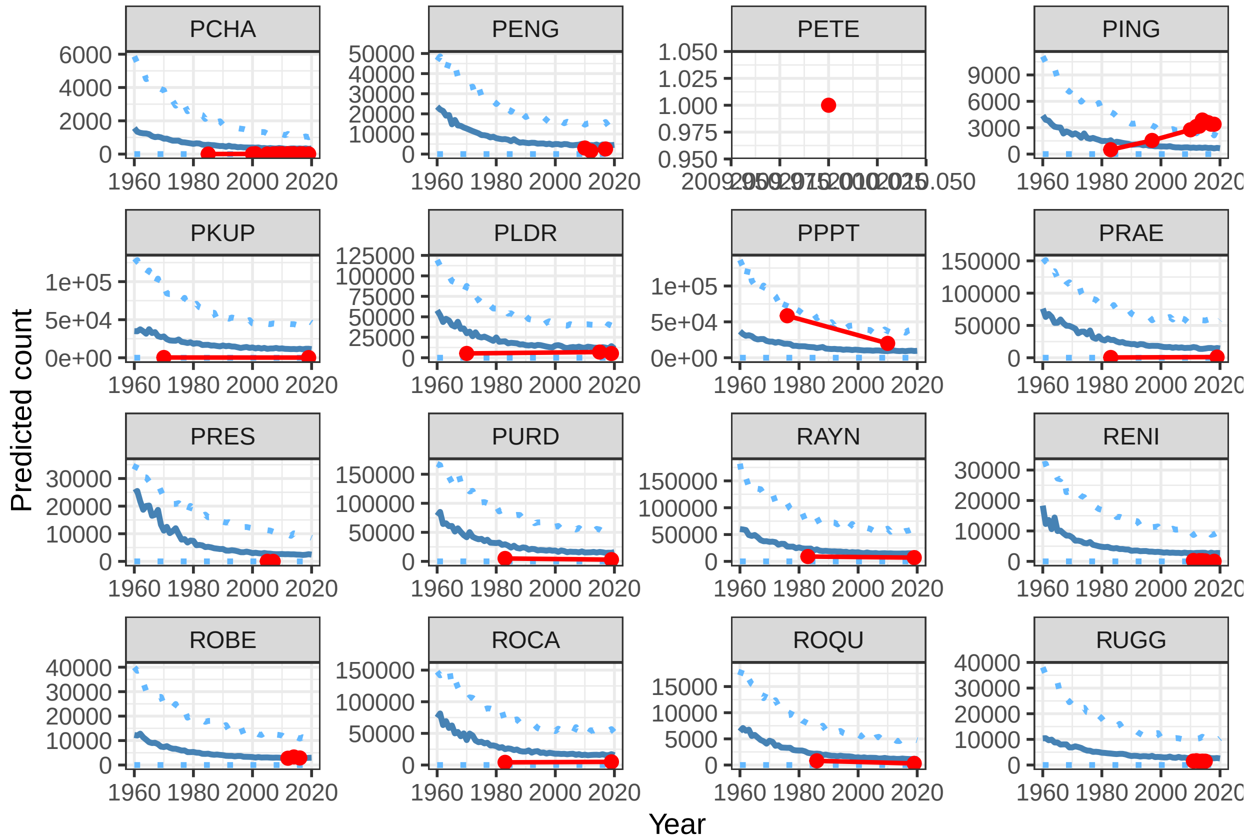
CRWL
CUVE
DECI
DEEI
DELA
DREA
DUPT
EADI
ECAP
ECKE
EIP1
EIP2
EIP3
EIP4
EIP5
ELLI
Predicted count
Year


EMER
ENTR
FARE
FARI
FLOR
FORT
FRED
FURS
GAST
GEOR
GIBA
GOLF
GOWC
GREE
GREY
GUES
Predicted count
Year

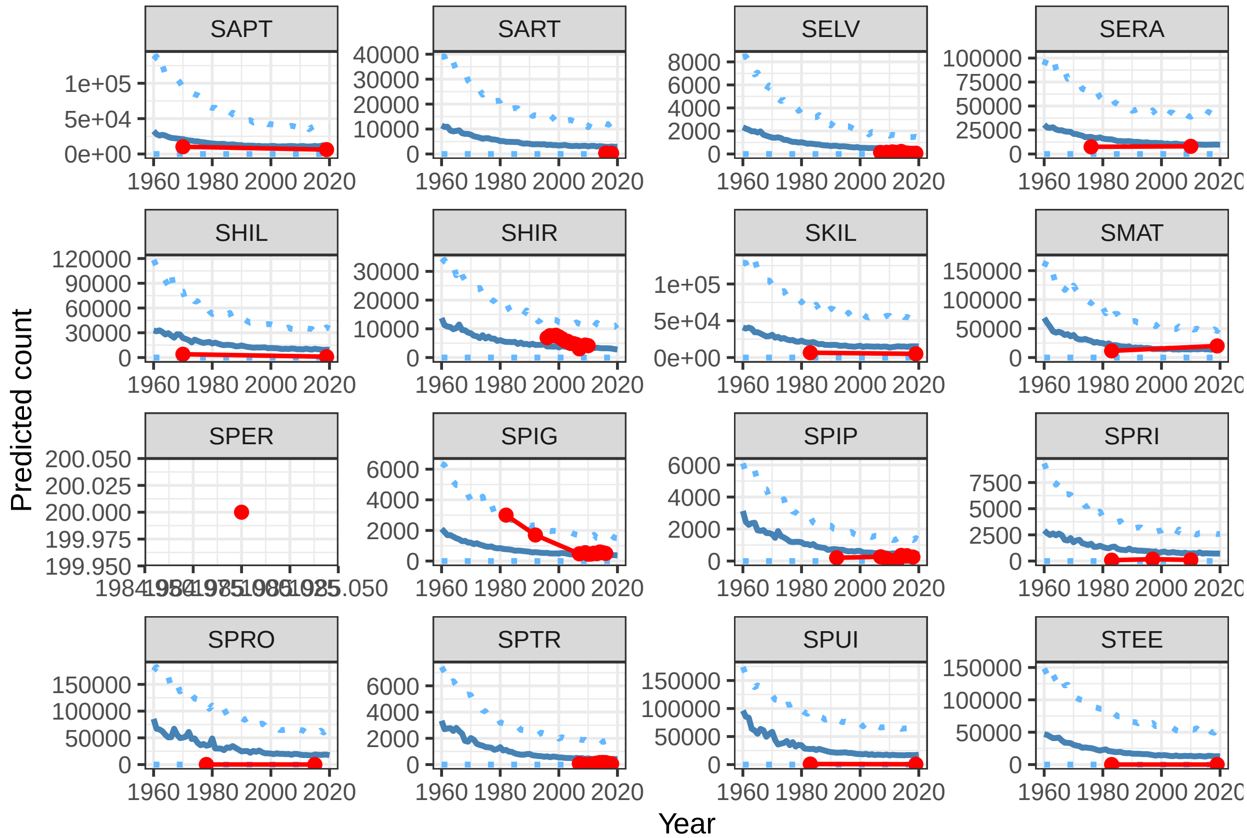

HALF
HANN
HARE
HARM
HUMB
HUNT
HUTB
HYDR
HYDS
JACO
JACP
JOUB
KAYR
KETL
LLOY
LOOK
Predicted count
Year
1960 1980 2000 2020

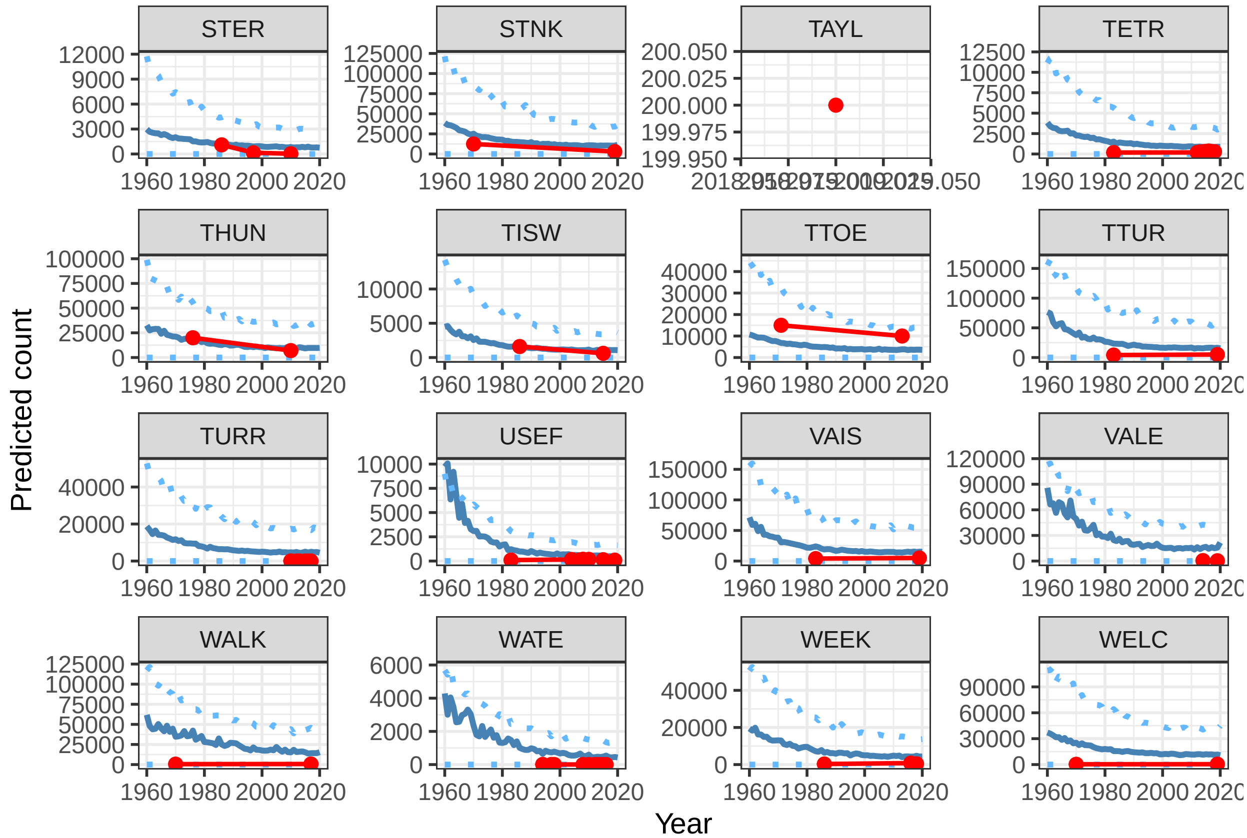


Predicted count
MABE
MANS
MATH
METH
MIDA
MONU
MOSS
MUCK
MURR
NANC
NOPT
NORW
OBRI
ORNE
ORRO
PALA
1960 1980 2000 2020
Year



PCHA
PENG
PETE
PING
PKUP
PLDR
PPPT
PRAE
PRES
PURD
RAYN
RENI
ROBE
ROCA
ROQU
RUGG
Predicted count
Year


SAPT
SART
SELV
SERA
SHIL
SHIR
SKIL
SMAT
SPER
SPIG
SPIP
SPRI
SPRO
SPTR
SPUI
STEE
Predicted count
Year



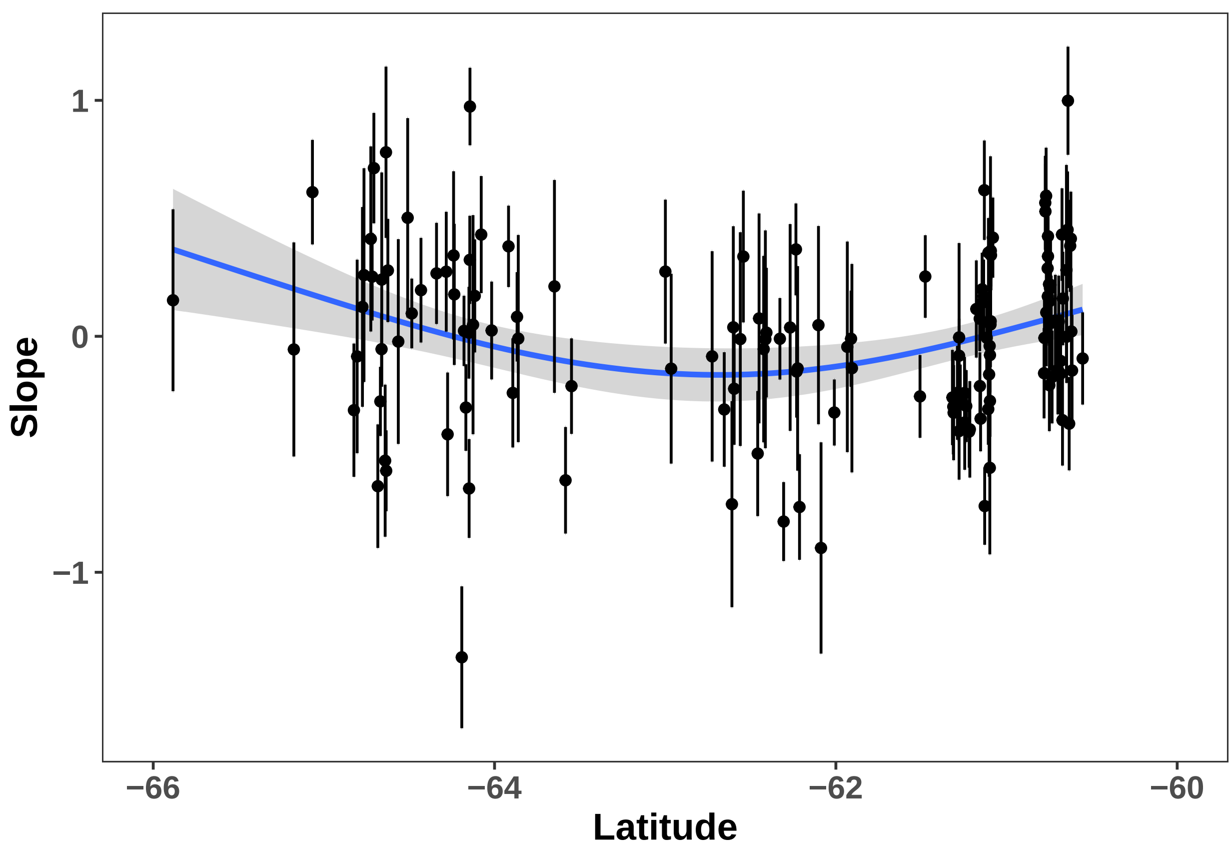

```r
# extract random effects from MCMCglmm
# https://stackoverflow.com/questions/64562052/extract-random-effects-from-mcmcglmm
library(broom.mixed)
re = tidy(mc2, effects="ran_vals")
unique(re$group)
```

```
## [1] "site_id"
```

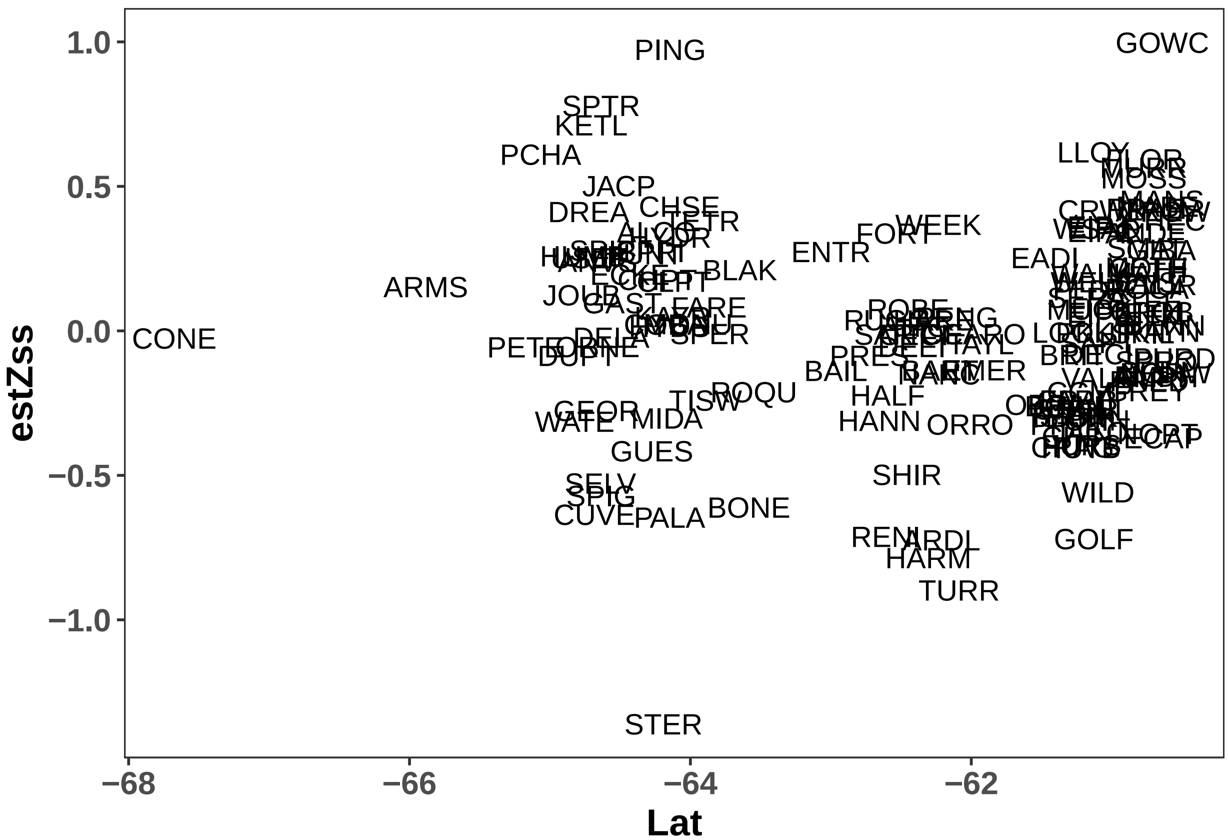
```r
re = re %>%
    dplyr::select(-group, -effect) %>%
    pivot_wider(names_from = term, values_from = c(estimate, std.error))

head(re)
```

```
## # A tibble: 6 x 5
##   level `estimate_(Intercept)` estimate_Zseason_starting `std.error_(Intercept)`
##   <chr>                  <dbl>                     <dbl>                   <dbl>
## 1 AILS                  0.528                    -0.156                    0.317
## 2 AITC                  2.12                      0.0156                   0.247
## 3 AITK                  0.231                     0.0571                   0.328
## 4 ALCO                  1.98                      0.343                    0.345
## 5 AMPH                  0.0666                   -0.158                    0.331
## 6 ANDE                 -2.76                      0.339                    0.331
## # i 1 more variable: std.error_Zseason_starting <dbl>
```

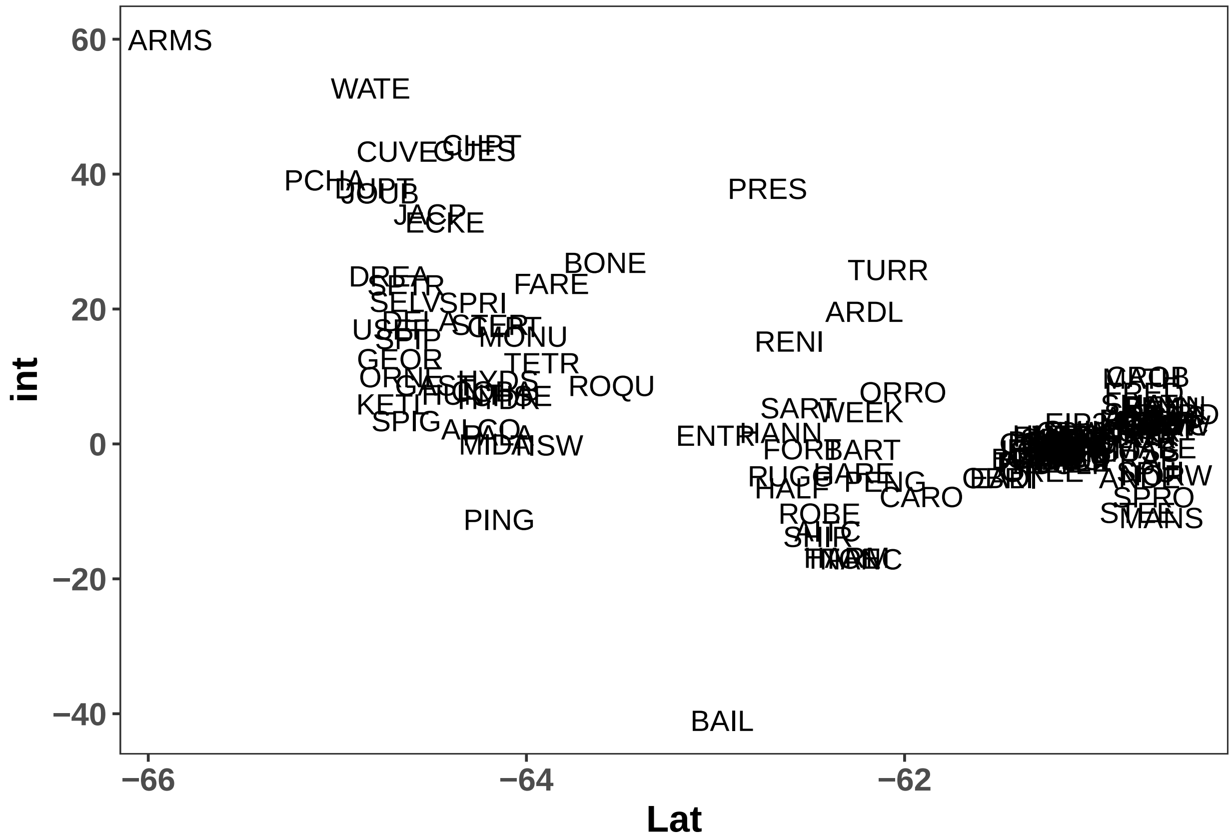
```r
# estimate_(Intercept) is related to the initial population size
# estimate_Zseason_starting is the slope of population increase (+)
# or decrease (-)

names(re) = c("site_id", "est_int", "estZss",
              "se_int", "seZss")
# add latitude
nestM3_lat = dplyr::select(nestM3, Lat, site_id) %>%
             dplyr::distinct(site_id, Lat)

re = left_join(re, nestM3_lat, by = "site_id")

# plot relationship between slope and latitude

ggplot(data = re, aes(x = Lat, y = estZss))+
  stat_smooth(method="gam",formula=y~s(x,k=2))+
  # geom_smooth(method='lm', formula= y~x)+
  geom_point()+
  geom_errorbar(aes(ymin=estZss-seZss,
                    ymax=estZss+seZss))+
  theme_bw()+th+
  ylab("Slope")+xlim(-66,-60)+
  xlab("Latitude")
```



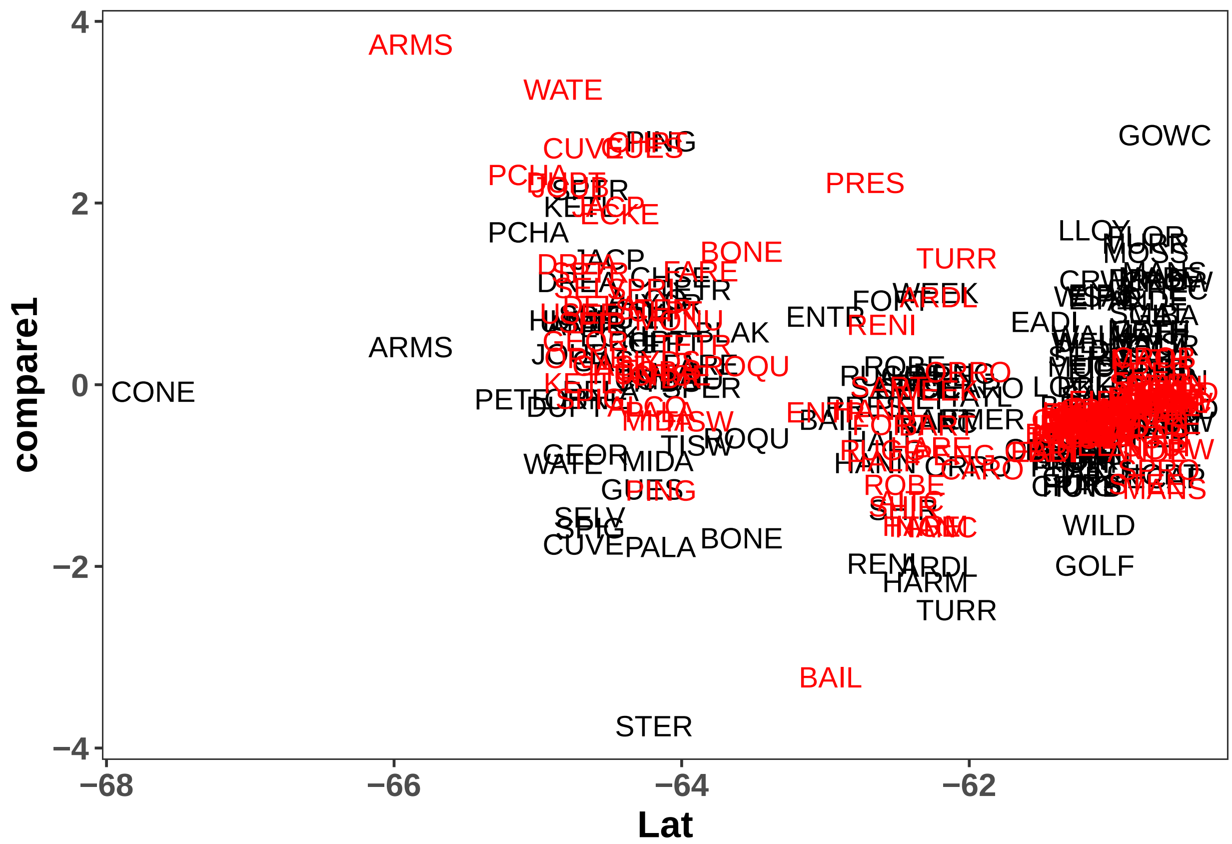
```r
ggplot(re, aes(x= Lat, y= estZss)) +
  geom_text(aes(label=site_id)) + theme_bw()+th
```




```r
# Compare this figure with Figure 3B from Kruger 2023, plotted here
ggplot(subset(rlat,Lat>(-67)),
       aes(Lat,int)) +
  geom_text(aes(label=site_id))  + theme_bw()+th
```




```
# Can we try plot them on the same graph to see if there is a similar trend between results?
rlat$compare1 = scale(rlat$int)
re$compare1 = scale(re$estZss)

# Plot standardized slopes from this analysis and Kruger 2023
# Not sure if this is useful?
ggplot(re, aes(x= Lat, y= compare1)) +
  geom_text(aes(label=site_id)) +
  geom_text(data = rlat, aes(x = Lat, y = compare1,label=site_id),
  col = "red") + theme_bw()  +th
```




```
compare = left_join(rlat, re, by = 'site_id')
cor(compare$estZss, compare$int, method = c("spearman"))
```

```
## [1] 0.08462982
```

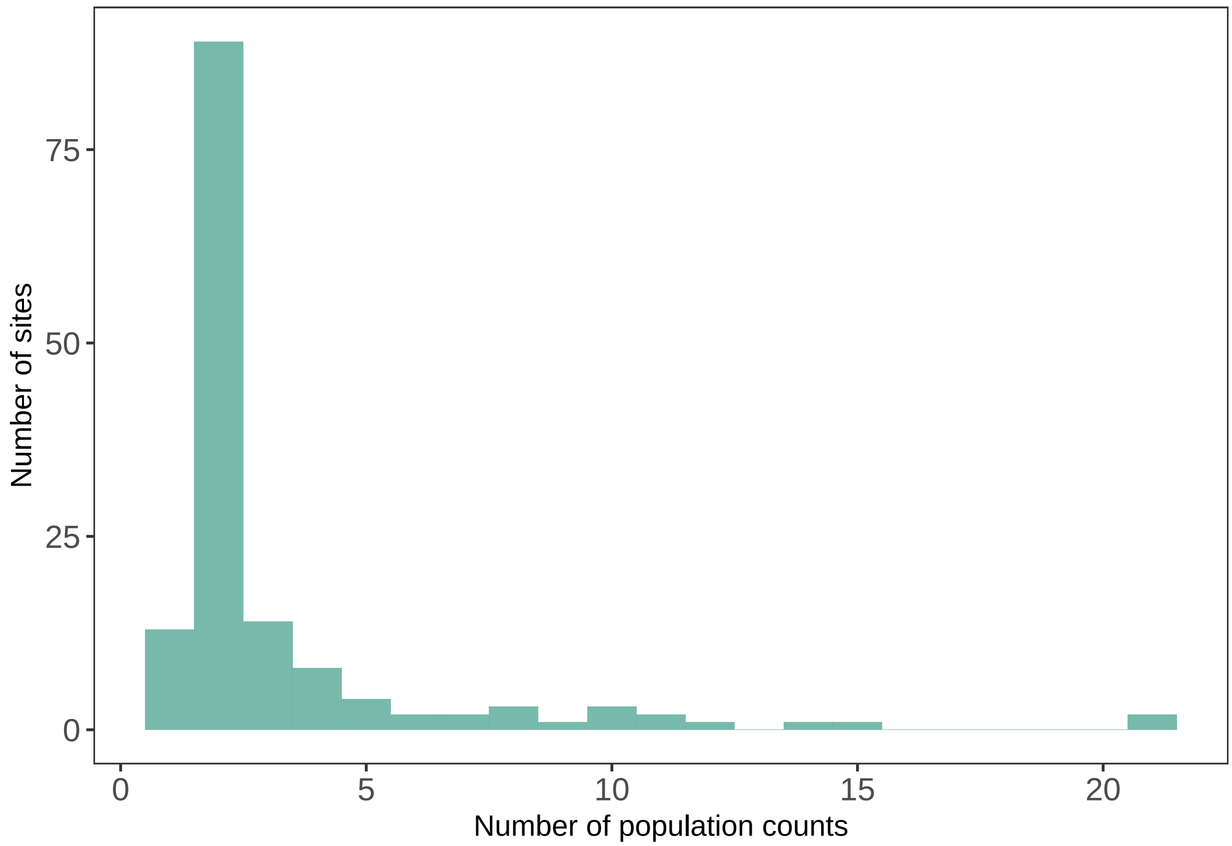
Oosthuizen et al data distribution figures

```
# This shows that there are some 1-count sites in the data being analysed
# (n = 146, not n = 133)
samplesize = nestM3 %>% group_by(site_id, ncounts) %>% tally()
length(unique(nestM3$site_id))
```

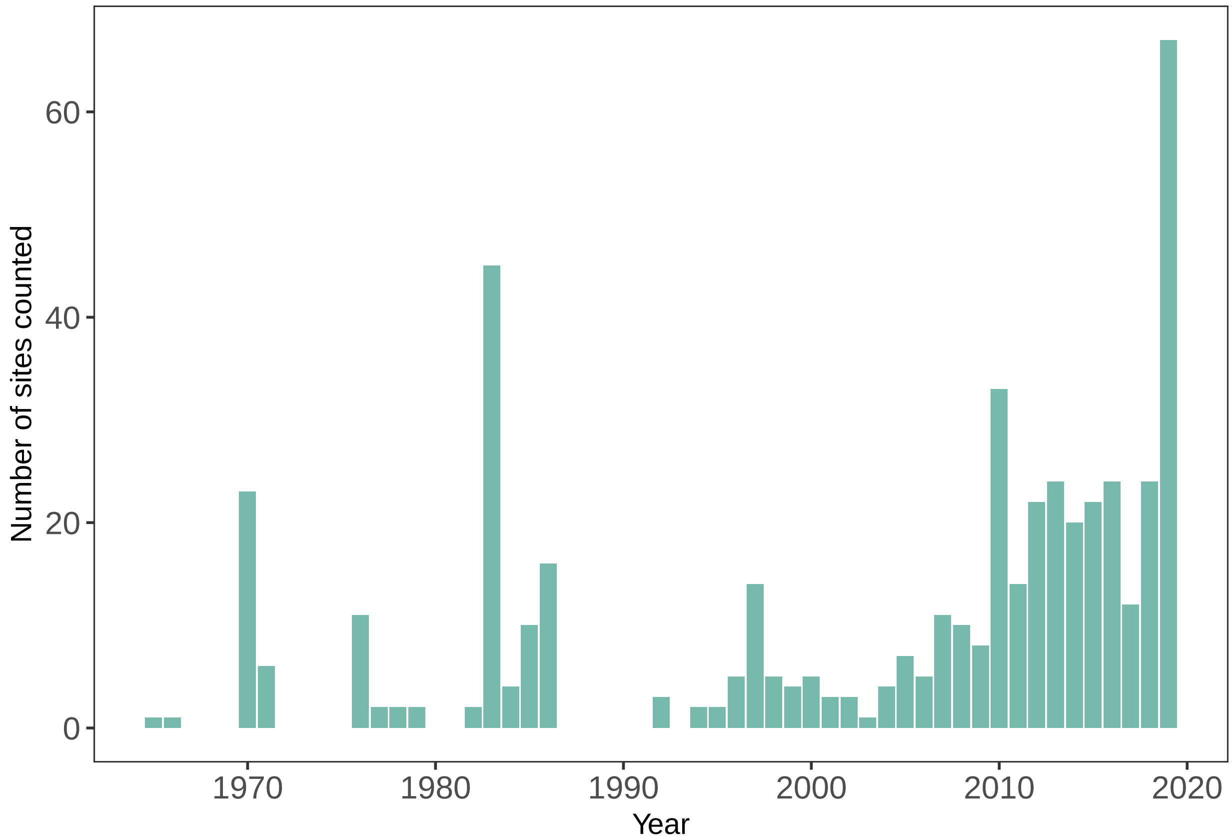
```
## [1] 146
```

```
samplesize.plot <- samplesize %>%
  ggplot(aes(x=n)) +
  geom_histogram(binwidth=1, fill="#69b3a2",  alpha=0.9) +
  theme_bw()+
  ylab("Number of sites")+
  xlab("Number of population counts") +
  theme(axis.text=element_text(size=12),
        panel.grid.major = element_blank(),
```

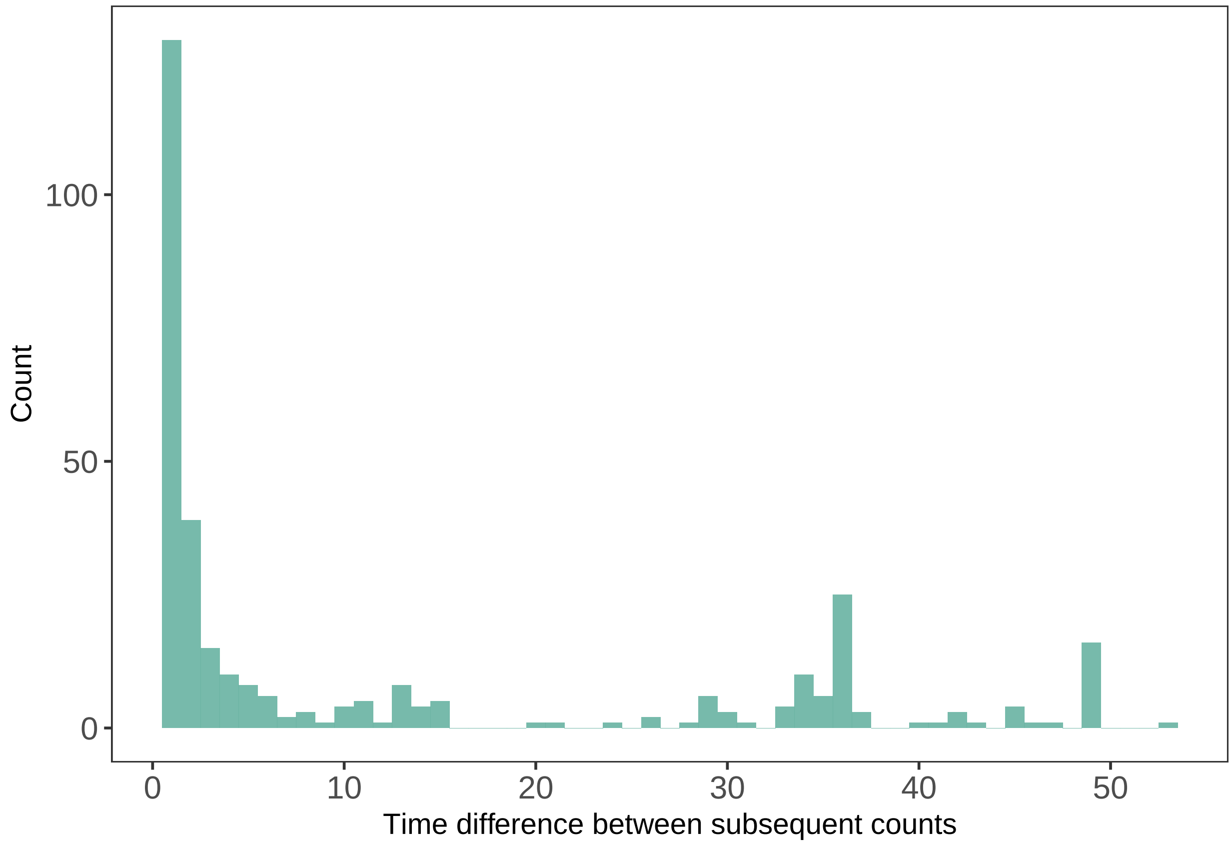
```
       panel.grid.minor = element_blank())

samplesize.plot
```




```
## Save Plot
# pdf("./Figure samplesize.pdf",
#     useDingbats = FALSE, width = 4, height = 4)
# samplesize.plot
# dev.off()

samplesizeYear = nestM3 %>% group_by(season_starting) %>% tally()
#samplesizeYear

samplesizeYear.plot = samplesizeYear %>%
  ggplot(aes(x=season_starting, y = n)) +
  geom_bar(stat = "identity", fill="#69b3a2",  alpha=0.9) +
  theme_bw() +
  ylab("Number of sites counted")+
  xlab("Year") +
  theme(axis.text=element_text(size=12),
        panel.grid.major = element_blank(),
        panel.grid.minor = element_blank())+
  scale_x_continuous(breaks = seq(1960, 2020, by = 10))

samplesizeYear.plot
```

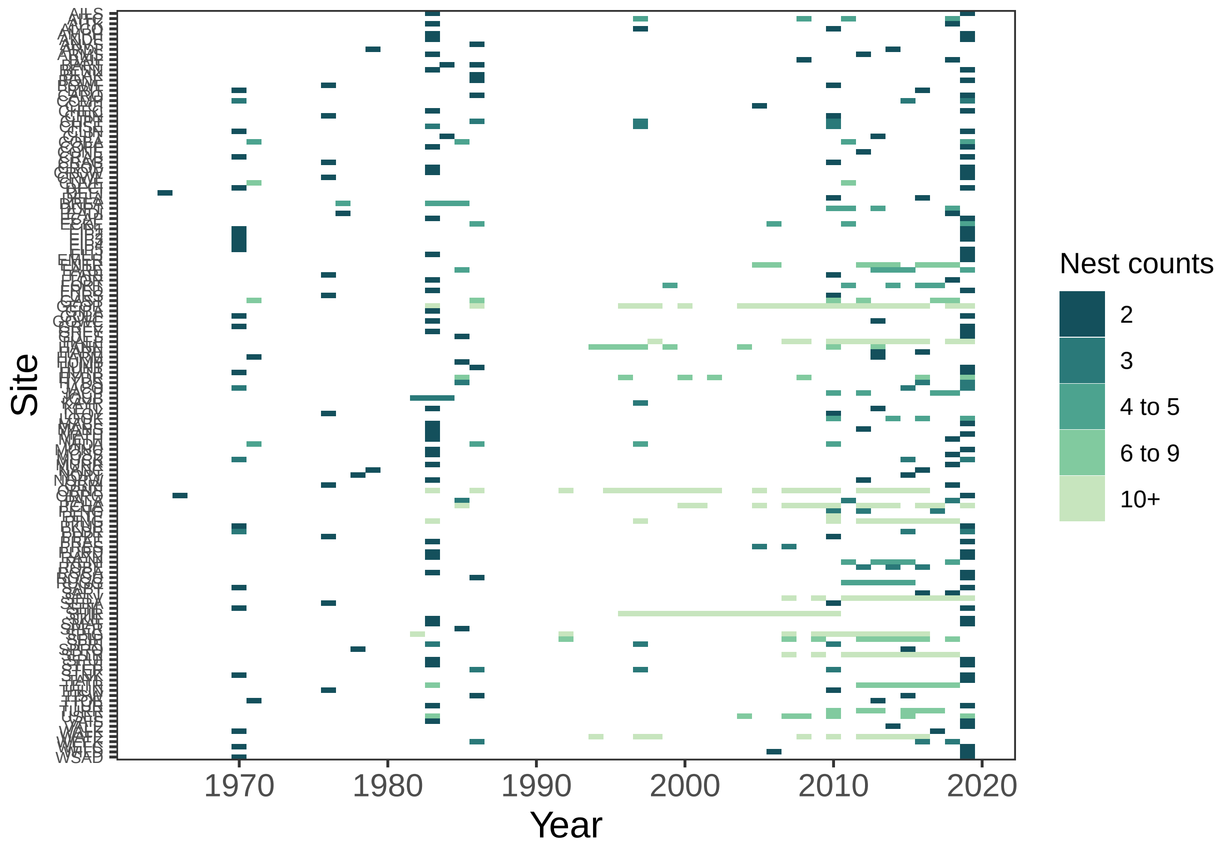



```r
## Save Plot
# pdf("./Figure samplesizeYear.pdf",
#     useDingbats = FALSE, width = 6, height = 4)
# samplesizeYear.plot
# dev.off()

# time between counts per site
diff = nestm3 %>%
  arrange(site_id, season_starting) %>%
  dplyr::group_by(site_id) %>%
  mutate(time.difference = season_starting - lag(season_starting))
#diff

diff.plot = diff %>%
  ggplot(aes(x=time.difference)) +
  geom_histogram(binwidth=1, fill="#69b3a2",  alpha=0.9) +
  theme_bw()+
  ylab("Count")+
  xlab("Time difference between subsequent counts") +
  theme(axis.text=element_text(size=12),
        panel.grid.major = element_blank(),
        panel.grid.minor = element_blank())+
  scale_x_continuous(breaks = seq(0, 50, by = 10))

diff.plot
```

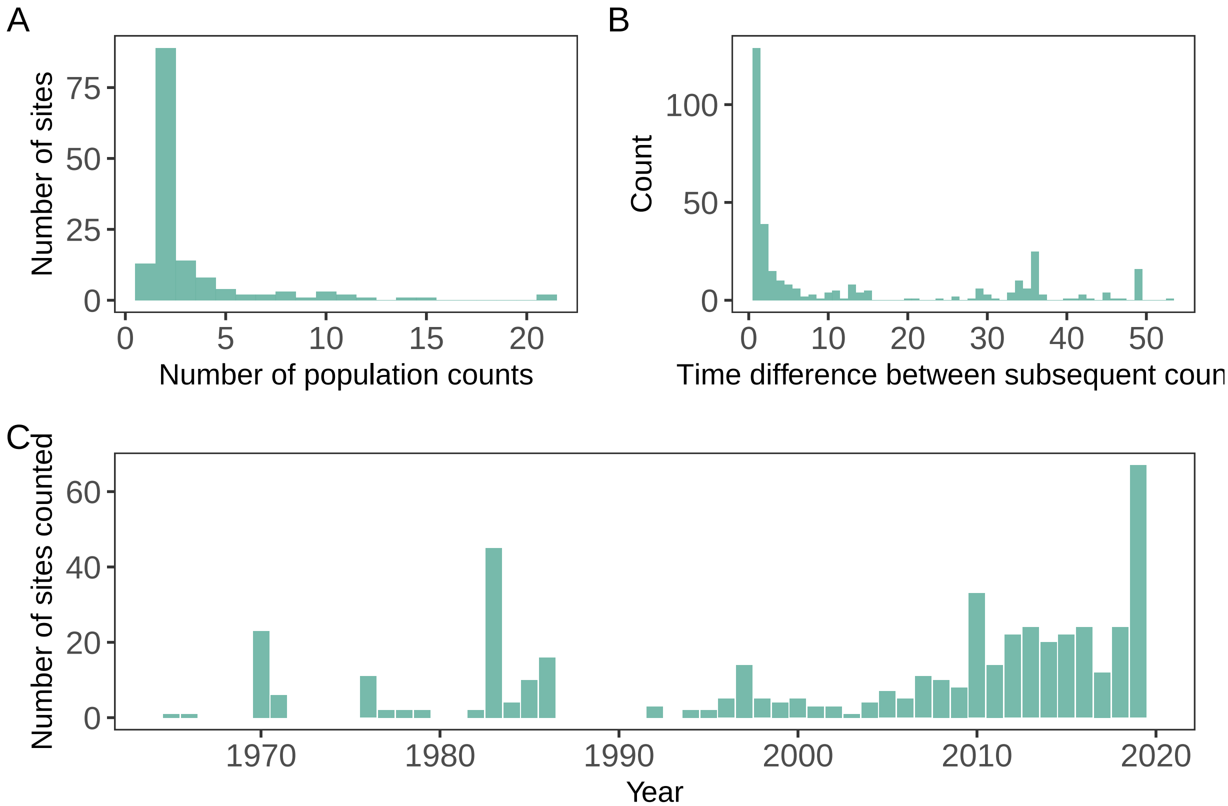



```
## Save Plot
# pdf("./Figure timedifferance.pdf",
#     useDingbats = FALSE, width = 6, height = 4)
# diff.plot
# dev.off()


library(colorspace)
library(scales)

nestm3$countbreaks = cut(nestm3$ncounts, c(0, 2, 3, 5, 9, Inf))

heat = ggplot(nestm3, aes(x = as.numeric(season_starting),
                   y = site_id,
                   fill= cut(ncounts, c(0, 2, 3, 5, 9, Inf),
                             labels = c('2','3','4 to 5','6 to 9','10+')))) +
  geom_tile() +
  scale_fill_discrete_sequential(palette = "BluGrn", rev = F)+
  guides(fill=guide_legend(title="Nest counts")) +
  theme_bw()+
  ylab("Site")+
  xlab("Year") +
  theme(axis.text.x=element_text(size=12),
        axis.title.x=element_text(size=14),
        axis.text.y = element_text(size = 6),
        axis.title.y=element_text(size=14),
```

```
        panel.grid.major = element_blank(),
        panel.grid.minor = element_blank())+
  scale_x_continuous(breaks = seq(1960, 2020, by = 10))+
  scale_y_discrete(limits=rev)

heat
```




```
## Save Plot
# pdf("./Figure samplesize_heat.pdf",
#     useDingbats = FALSE, width = 7, height = 8)
# heat
# dev.off()

library(patchwork)
combinedfig = (samplesize.plot | diff.plot) / samplesizeYear.plot +
  plot_layout(nrow = 2, widths = c(1, 3)) +
  plot_annotation(tag_levels = 'A')
combinedfig
```




```
## Save Plot
# pdf("./Figure combined.pdf",
#     useDingbats = FALSE, width = 8, height = 6)
# combinedfig
# dev.off()
```