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

Chris Oosthuizen, Murray Christian, Azwianewi Makhado, Mzabalazo Ngwenya

2023-10-05

Contents

1 Krüger (2023) reanalysis

This script provides a reanalysis of chinstrap penguin population trends, in the context 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.).

- To reduce extrapolation beyond the range of observed data, this revised analysis restricted predictions of population trends between 1980 and 2019.
- This script evaluated the 30-year population change between 1990 and 2019 at all sites (n = 71) with two or more counts (with accuracy < 5) over a period of at least 10 years between 1980 and 2019.

2 Load packages and set plotting theme

```
# Load packages
library(tidyverse)
library(MCMCglmm)
library(scales)
library(ggforce)

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

3 Load and process MAPPPD data for area 48.1 and 48.2:

```
# 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 data to chinstrap penguins only
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
```

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

```
## [1] 1104   15
```

```
# subset to cammlr_region 48.1 and 48.2
nests <- subset(nests,cammlr_region =="48.1" | cammlr_region =="48.2")
dim(nests)
```

```
## [1] 1103   15
```

```
# remove the most uncertain counts (could be very inaccurate - an order of magnitude)
# This is a choice we made for the current analysis.
# Comment this line of code out to run analysis including uncertain counts
nests <- subset(nests, accuracy < 5)
dim(nests)
```

```
## [1] 944  15
```

```
nests = subset(nests, nests$season_starting> 1979)

which(colSums(is.na(nests))>0)
```

```
##   day month
##     7     8
```

```
# some populations had multiple counts over the same season:
# this code summarises the count with the maximum nests
nestM = nests %>%
          group_by(site_id, season_starting) %>%
          slice(which.max(penguin_count)) %>%      # take only maximum counts
          dplyr::rename(Lat = latitude_epsg_4326,
                      Lon = longitude_epsg_4326,
                      nests = penguin_count) %>%
          dplyr::select(season_starting, site_id, nests, Lat, Lon)

dim(nestM)
```

```
## [1] 764   5
```

4 Processed data summary

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

```
##   site_id     Lat     Lon ncounts minseason maxseason interval
## 1    ACUN -60.761 -44.637       2      1983      2004       21
## 2    AILS -60.780 -44.631       1      1983      1983        0
## 3    AITC -62.407 -59.752       4      1997      2018       21
## 4    AITK -60.738 -44.525       1      1983      1983        0
## 5    ALCO -64.240 -61.127       3      1989      2010       21
## 6    AMPH -60.684 -45.339       1      1983      1983        0
```

```
summary(as.factor(countsN$ncounts)) # most populations are only counted once
```

```
##   1   2   3   4   5   6   7   8   9  10  11  12  15  16  22  23  24  28  30  31
## 152  35  13   7   4   5   1   4   2   5   2   2   1   1   2   1   1   2   1   1
##  33
##   1
```

```
npops=length(countsN$ncounts[countsN$ncounts>1])
npops # number of populations with more than 2 counts
```

```
## [1] 91
```

```
nestM2 <- merge(nestM,countsN) # add number of counts for each population to nestM2

# Subset to sites with more than 1 count:
nestm3 = subset(nestM2, ncounts>1)
nestm3 = subset(nestm3, nestm3$interval > 9)
#nestm3 = subset(nestm3, nestm3$minseason < 2005 & nestm3$maxseason > 2004)

nestM3 <- nestm3

countspersite = nestM3 %>%
      group_by(site_id) %>%
      summarise(counts = mean(ncounts))

summary(countspersite$counts)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   2.000   2.000   4.000   7.887  10.000  33.000
```

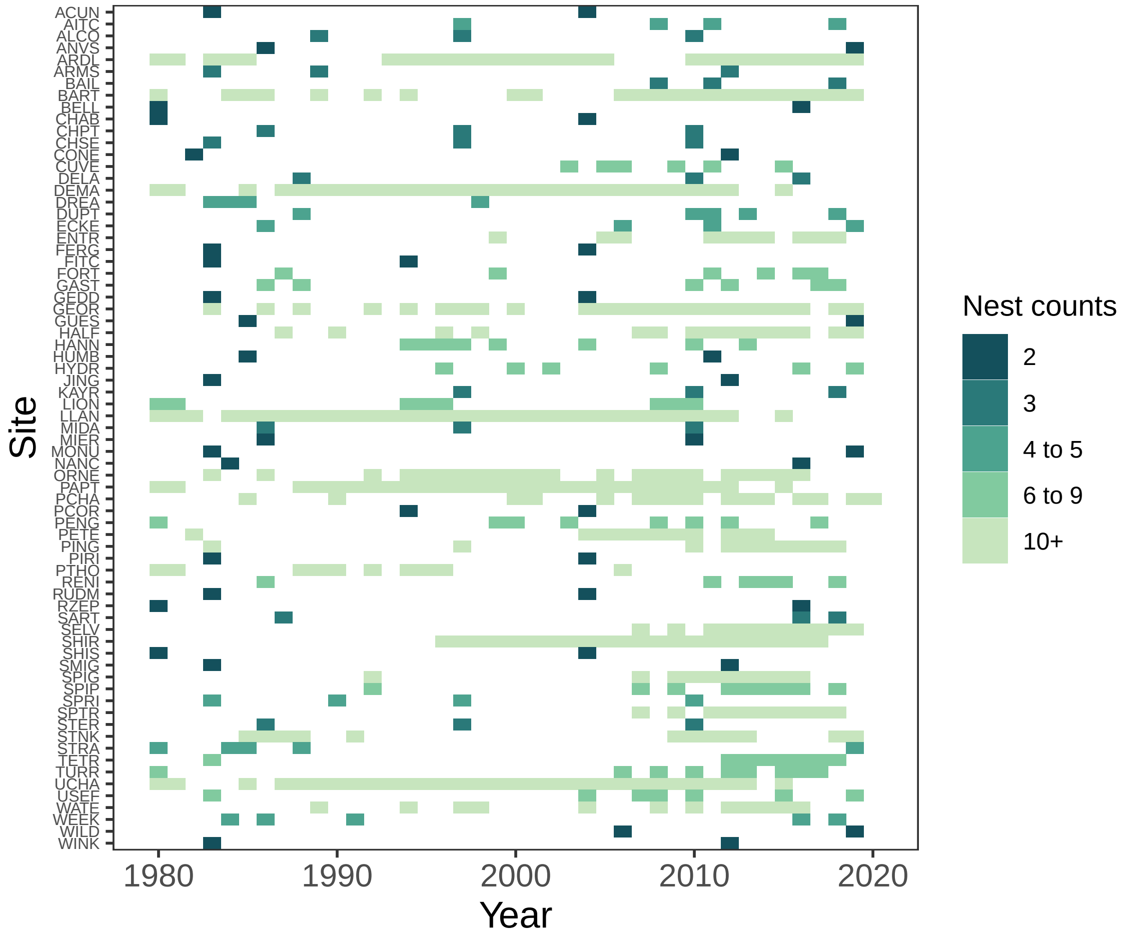
5 Oosthuizen et al data distribution figures

```
# 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))
```

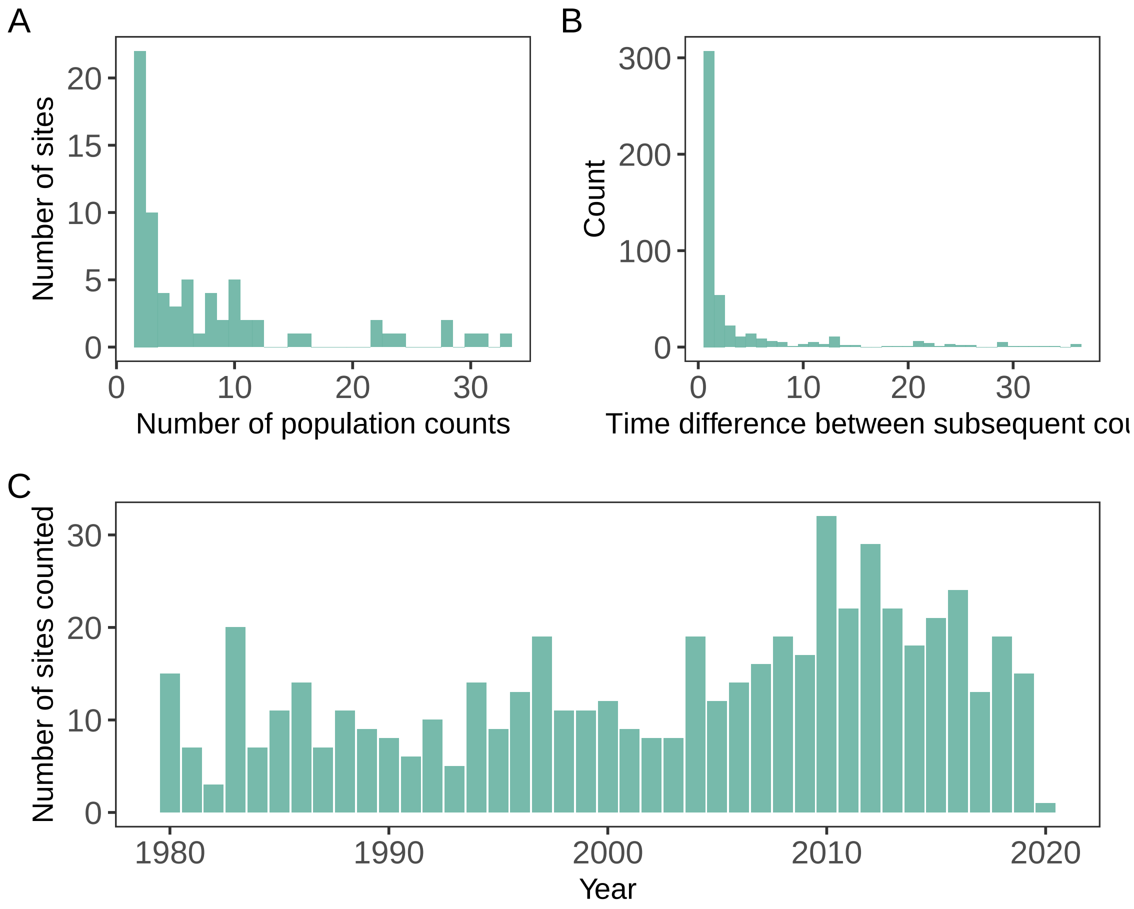
```
## [1] 71
```

```
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())
```

```r
# samplesize.plot

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

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

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

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
```



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



6 Analysis for Oosthuizen et al. (current)

6.1 Fit a better GLMM

```
# 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)

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))))
```

```
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=F, DIC=TRUE, singular.ok=FALSE, saveX=TRUE,
              prior=prior, saveZ=TRUE, saveXL=TRUE, slice=FALSE,
              ginverse=NULL, trunc=FALSE)
```

```
summary(mc2)
```

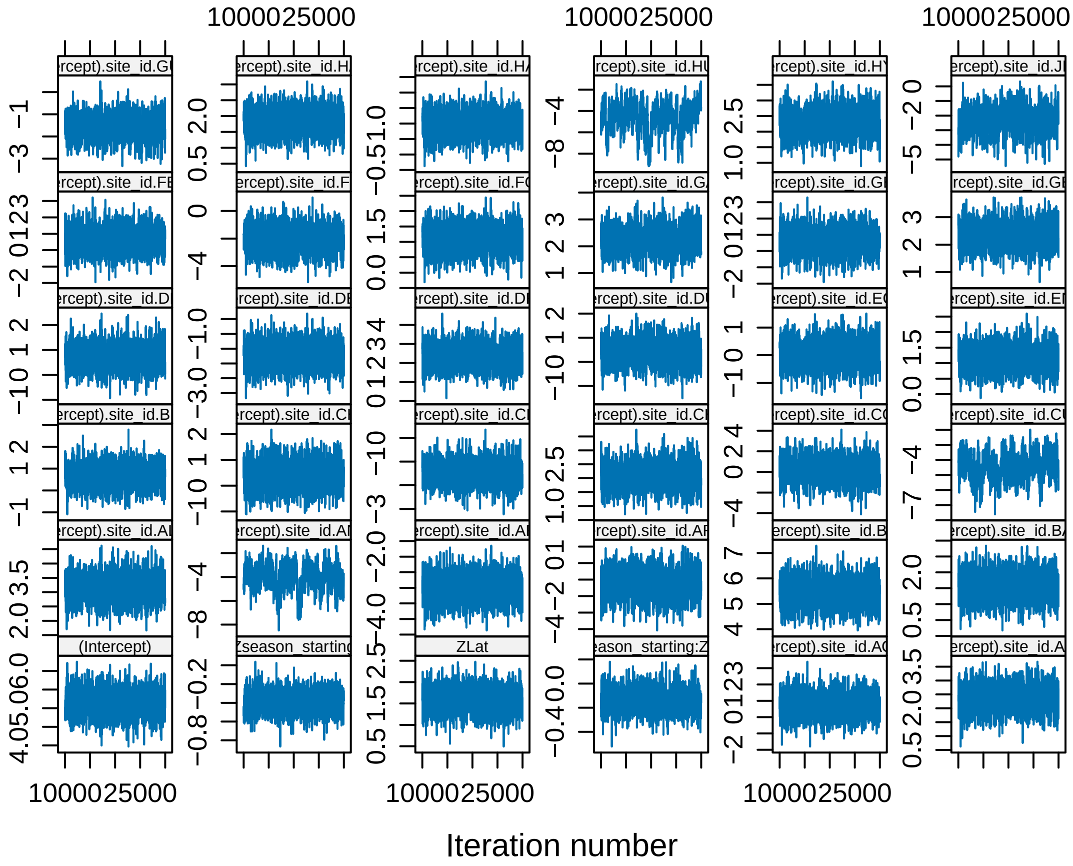
```
##
##  Iterations = 10001:29991
##  Thinning interval  = 10
##  Sample size  = 2000
##
##  DIC: 5098.358
##
##  G-structure:  ~us(1 + Zseason_starting):site_id
##
##                                           post.mean l-95% CI u-95% CI eff.samp
## (Intercept):(Intercept).site_id              7.2646   4.9001   10.029    704.4
## Zseason_starting:(Intercept).site_id         1.3796   0.7324    2.185    606.9
## (Intercept):Zseason_starting.site_id         1.3796   0.7324    2.185    606.9
## Zseason_starting:Zseason_starting.site_id    0.7137   0.4535    1.013    890.8
##
##  R-structure:  ~units
##
##       post.mean l-95% CI u-95% CI eff.samp
## units    0.1136  0.09322   0.1354     1402
##
##  Location effects: nests ~ Zseason_starting * ZLat
##
##                       post.mean l-95% CI u-95% CI eff.samp  pMCMC
## (Intercept)             5.14951  4.46730  5.76721     2000 <5e-04 ***
## Zseason_starting       -0.39111 -0.62038 -0.19254     1470  0.002 **
## ZLat                    1.57571  1.05132  2.11820     1744 <5e-04 ***
## Zseason_starting:ZLat  -0.13224 -0.31196  0.04182     1066  0.142
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
```

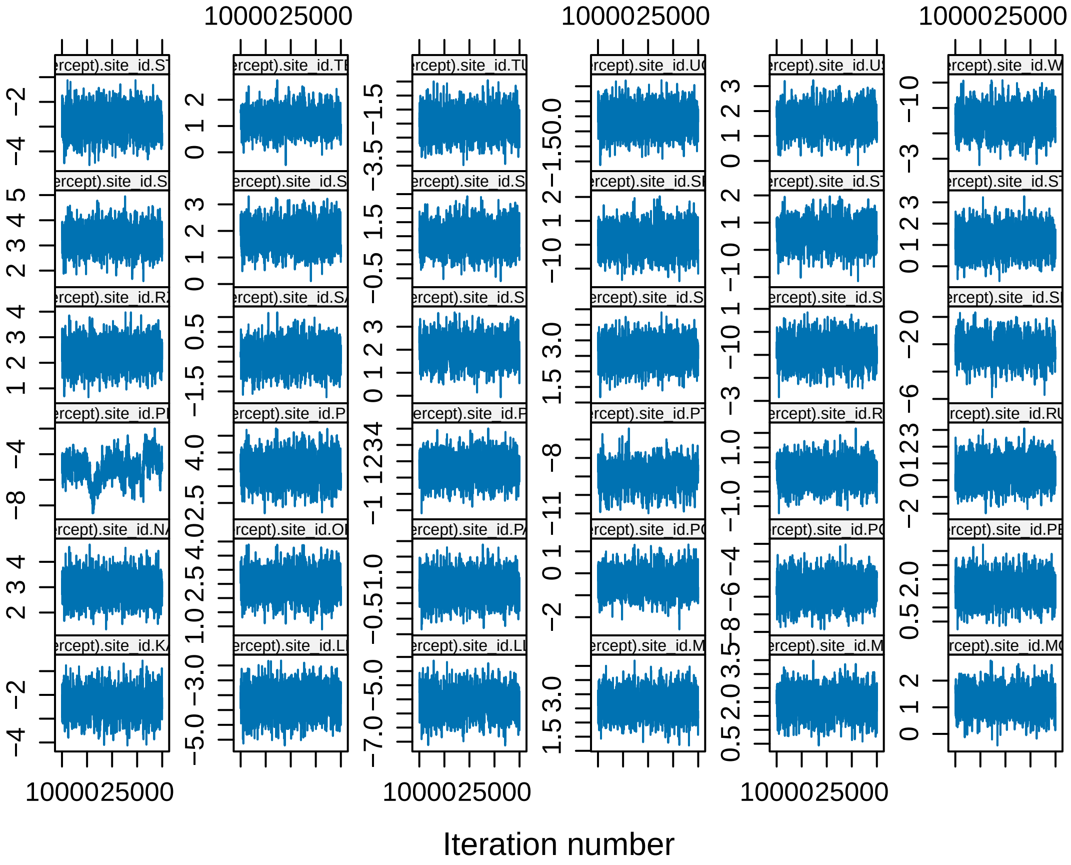
6.2 MCMCglmm diagnostics for mc2

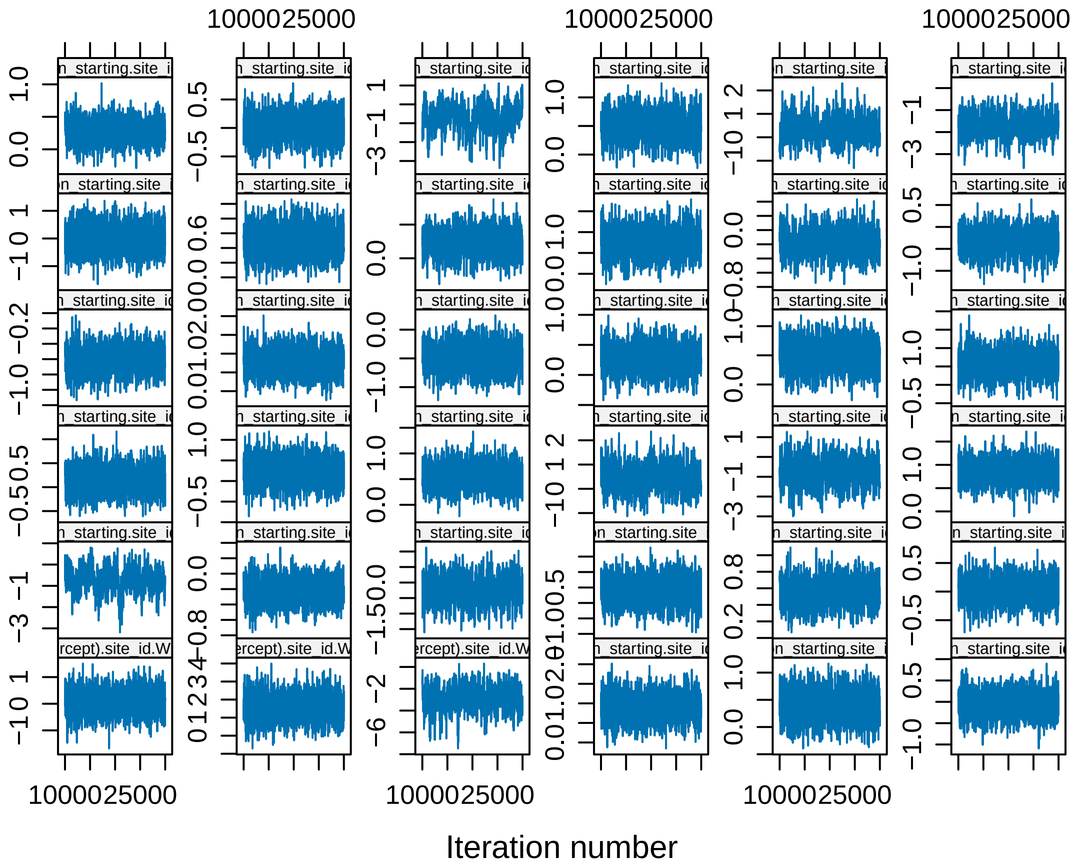
```
# Assessing model convergence. We do this separately for both fixed
# and random effects. The trace plot should look like a fuzzy caterpillar

# effective sample size
coda::effectiveSize(mc2$VCV)
```

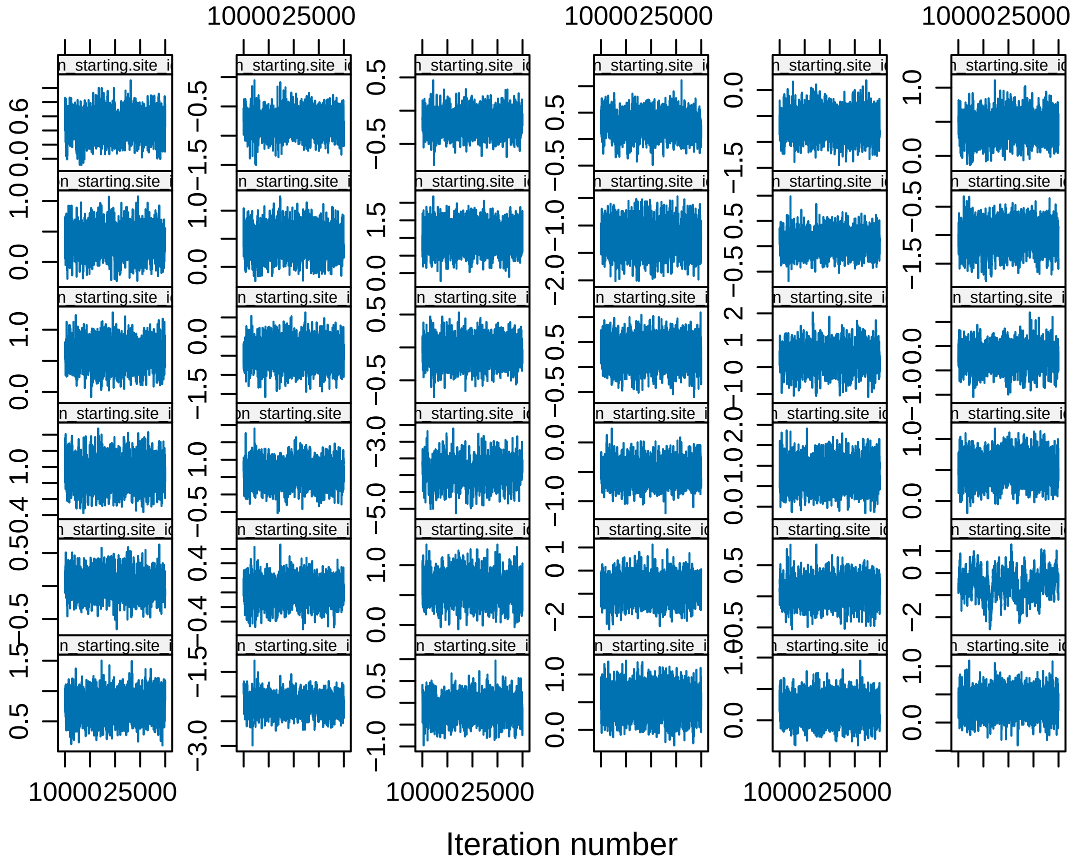
```
##           (Intercept):(Intercept).site_id
##                                    704.4337
##       Zseason_starting:(Intercept).site_id
##                                    606.8866
##       (Intercept):Zseason_starting.site_id
##                                    606.8866
## Zseason_starting:Zseason_starting.site_id
##                                    890.8188
##                                       units
##                                   1402.4433
```

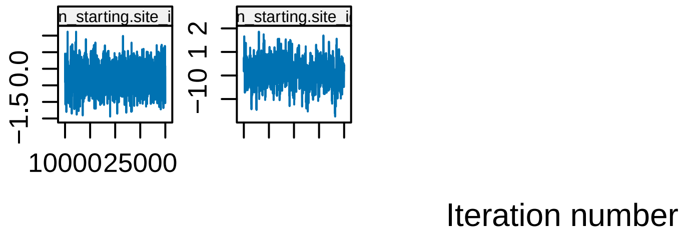
```
# check that the mcmc chain is mixing well - should be "white noise"
lattice::xyplot(as.mcmc(mc2$Sol), layout=c(6,6), par.strip.text=list(cex=0.5))
```



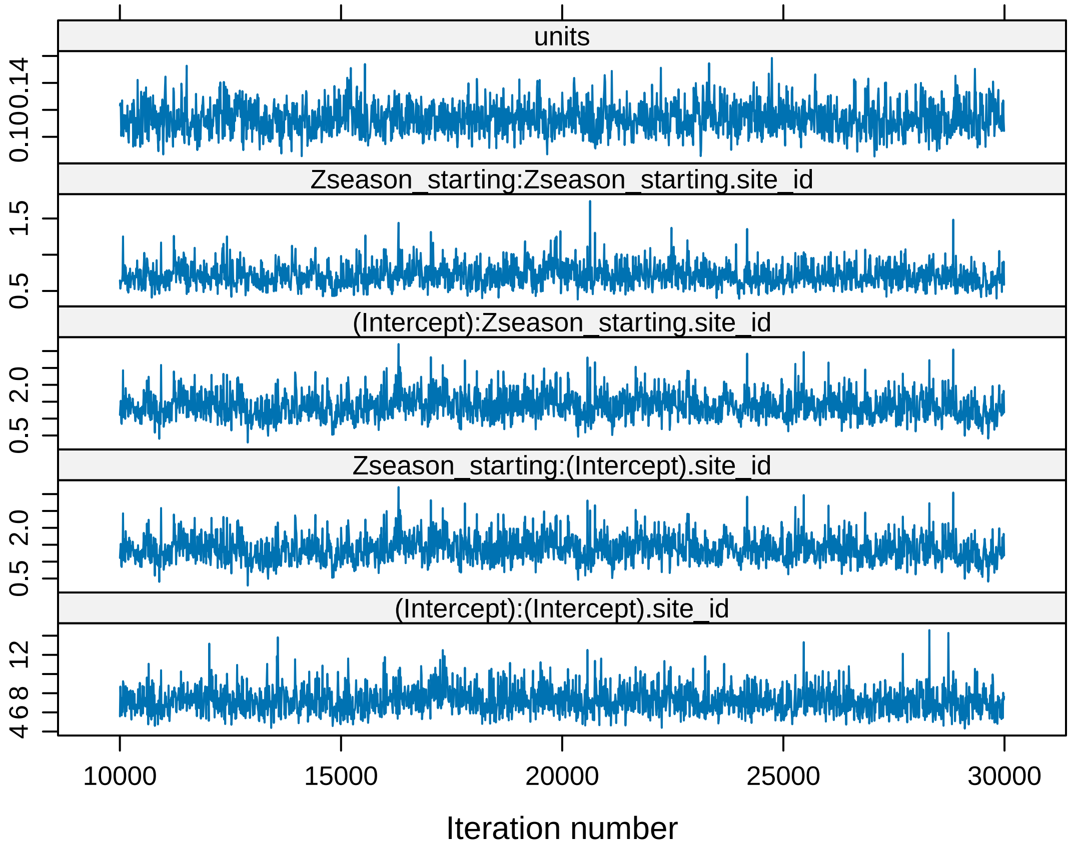

ercept).site_id.ST
ercept).site_id.TI
ercept).site_id.TU
ercept).site_id.UG
ercept).site_id.US
ercept).site_id.W
ercept).site_id.S
ercept).site_id.S
ercept).site_id.S
ercept).site_id.SI
ercept).site_id.ST
ercept).site_id.ST
ercept).site_id.RA
ercept).site_id.SA
ercept).site_id.S
ercept).site_id.S
ercept).site_id.S
ercept).site_id.S
ercept).site_id.PI
ercept).site_id.P
ercept).site_id.F
ercept).site_id.PT
ercept).site_id.R
ercept).site_id.RU
ercept).site_id.NA
ercept).site_id.OI
ercept).site_id.PA
ercept).site_id.PC
ercept).site_id.PC
ercept).site_id.PE
ercept).site_id.KA
ercept).site_id.LI
ercept).site_id.LI
ercept).site_id.M
ercept).site_id.M
ercept).site_id.MO
1000025000
Iteration number


1000025000
n_starting.site_i
n_starting.site_id
rcept).site_id.W
ercept).site_id.W
Iteration number


1000025000
n_starting.site_i
Iteration number



```
# the variance components
lattice::xyplot(as.mcmc(mc2$VCV), par.strip.text=list(cex=0.8))
```



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

```
##            (Intercept):(Intercept).site_id
##                                 0.04111137
##      Zseason_starting:(Intercept).site_id
##                                 0.14014311
##      (Intercept):Zseason_starting.site_id
##                                 0.14014311
## Zseason_starting:Zseason_starting.site_id
##                                 0.13846975
##                                      units
##                                 0.17538329
```

6.3 Predict using MCMCglmm mc2

```
# construct an hypothetical dataframe to predict to

# need to predict to z-standardized variables
Z1 = dplyr::select(nestM3, season_starting, Lat)
```

```r
Z2 <- scale(Z1)
attr(Z2,"scaled:center")
```

```
## season_starting             Lat
##      2002.78393       -63.18634
```

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

```
## season_starting             Lat
##       11.475949        1.290311
```

```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(1980:2019)) # extrapolate to 1980

pops<-data.frame(site_id=countsN$site_id[countsN$ncounts>1],
                 Lat=countsN$Lat[countsN$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    ACUN -60.761            1980     0        -1.985363  1.8796575
## 2    AITC -62.407            1980     0        -1.985363  0.6039961
## 3    ALCO -64.240            1980     0        -1.985363 -0.8165916
## 4    ANVS -64.661            1980     0        -1.985363 -1.1428695
## 5    ARDL -62.213            1980     0        -1.985363  0.7543475
## 6    ARMS -65.884            1980     0        -1.985363 -2.0907030
```

```r
# Don't extrapolate more than X years
first_last_season = nestM3 %>%
        dplyr::group_by(site_id) %>%
        dplyr::summarise(minyear = min(season_starting),
                         maxyear = max(season_starting)) %>%
        dplyr::arrange(minyear)
first_last_season
```

```
## # A tibble: 71 x 3
##    site_id minyear maxyear
##    <chr>     <int>   <int>
##  1 ARDL       1980    2019
##  2 BART       1980    2019
##  3 BELL       1980    2016
##  4 CHAB       1980    2004
##  5 DEMA       1980    2015
##  6 LION       1980    2010
##  7 LLAN       1980    2015
##  8 PAPT       1980    2015
##  9 PENG       1980    2017
## 10 PTHO       1980    2006
## # i 61 more rows
```

```r
popy = merge(popy, first_last_season)

# subset so that you only predict for sites with counts at least 20 years from begin and end

#popy = subset(popy, popy$minyear < 1990)
#popy = subset(popy, popy$maxyear > 2010)

length(unique(popy$site_id))
```

```
## [1] 71
```

```r
popypred <- 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"))

head(popypred)
```

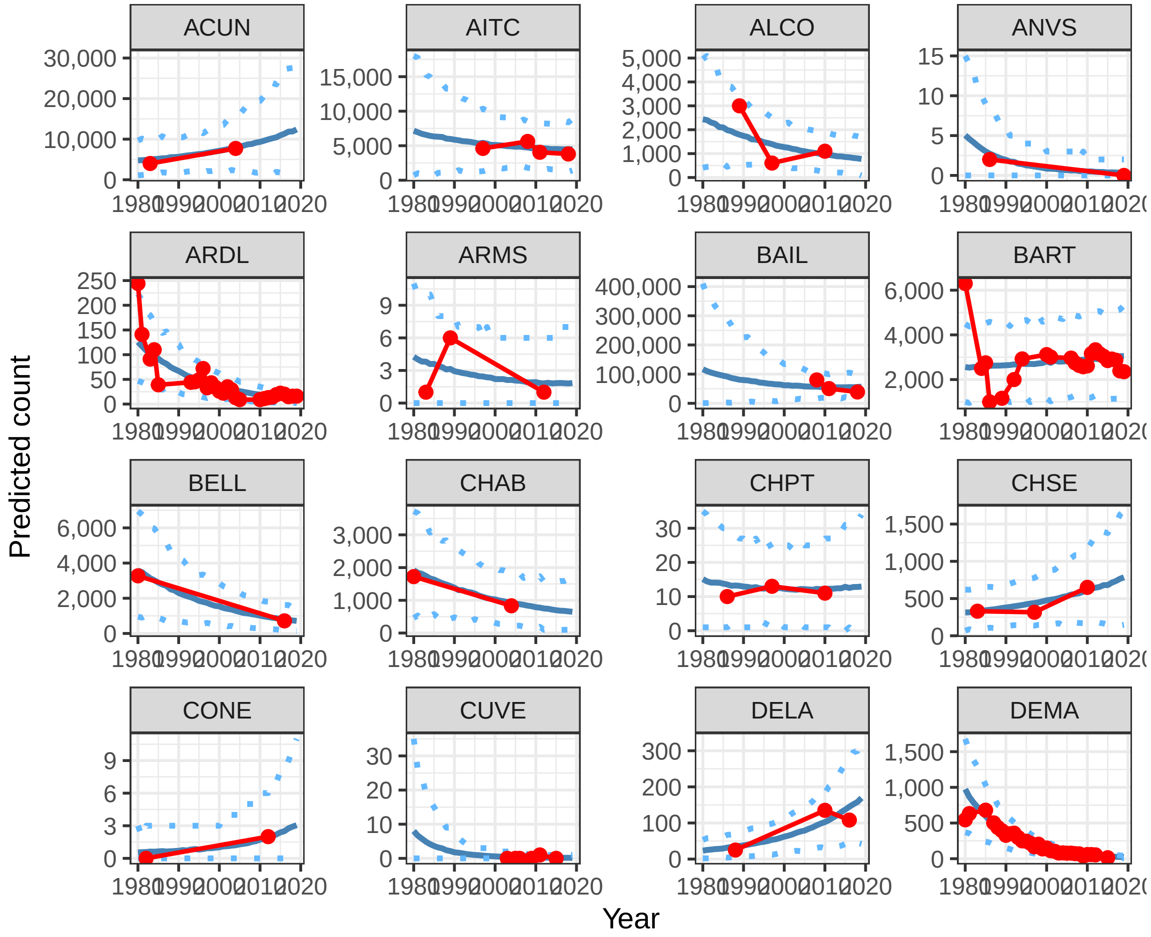
```
##          fit  lwr   upr
## 1 11904.431 1506 27566
## 2  4782.299 1089  9664
## 3 12407.094 1061 30525
## 4  6325.511 2015 11992
## 5  8181.861 2073 16091
## 6  5590.439 1588 10499
```

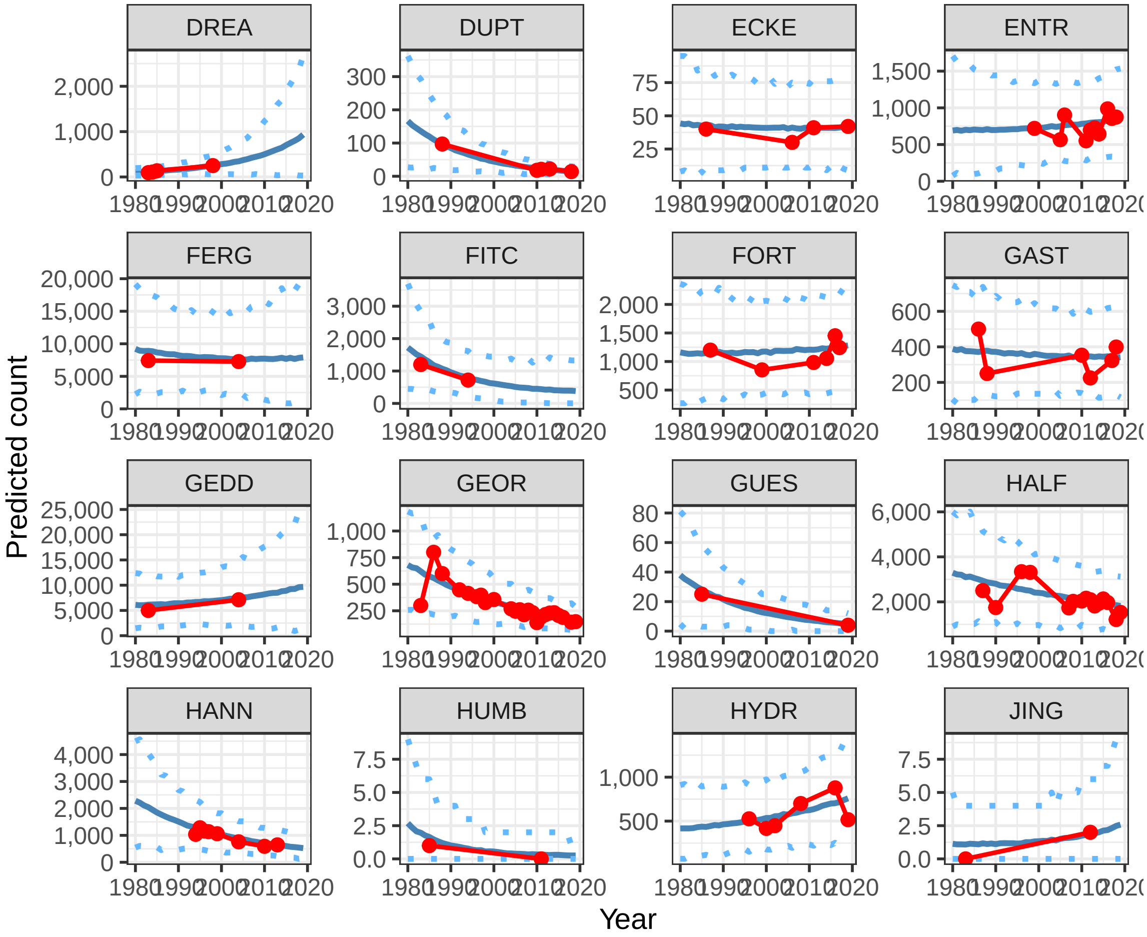
```r
popy$Zfit = popypred$fit
popy$Zlwr = popypred$lwr
popy$Zupr = popypred$upr

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

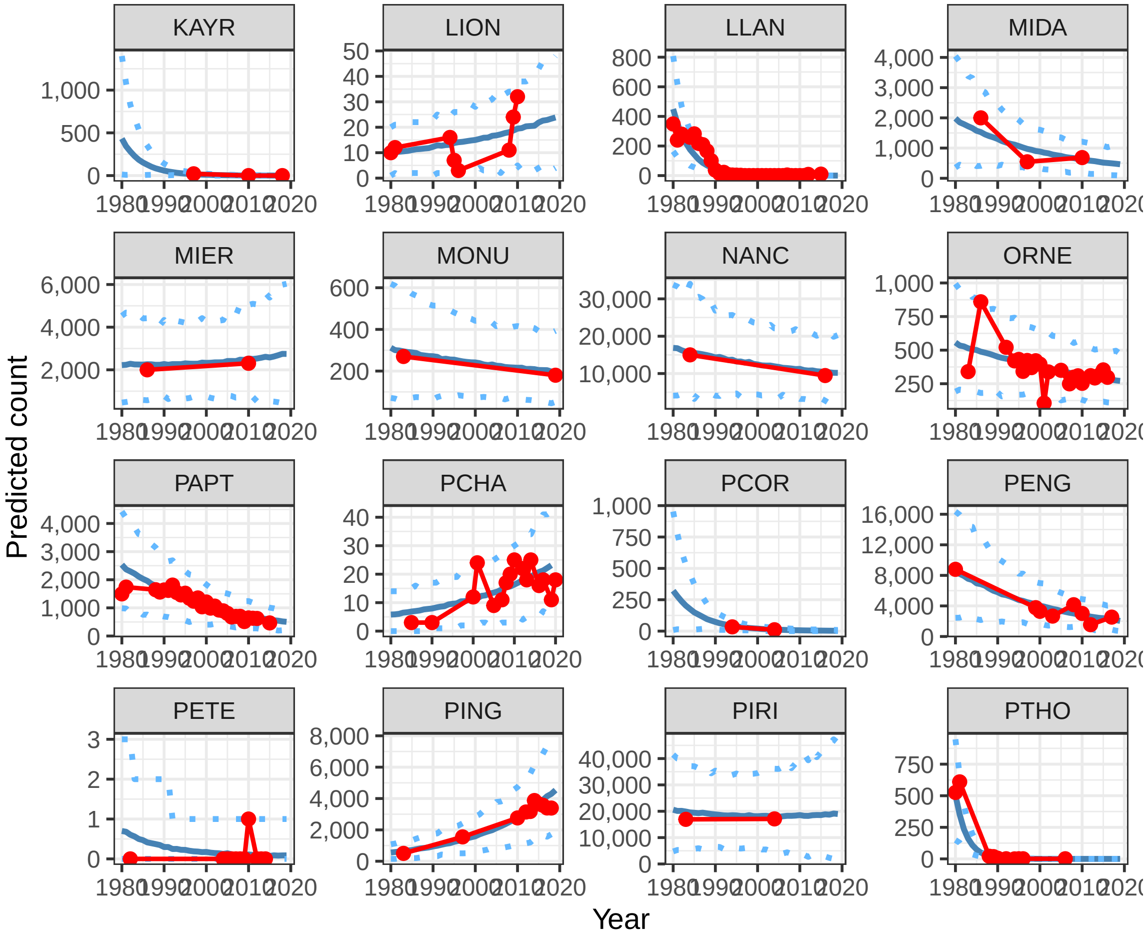
6.4 Conditional model predictions

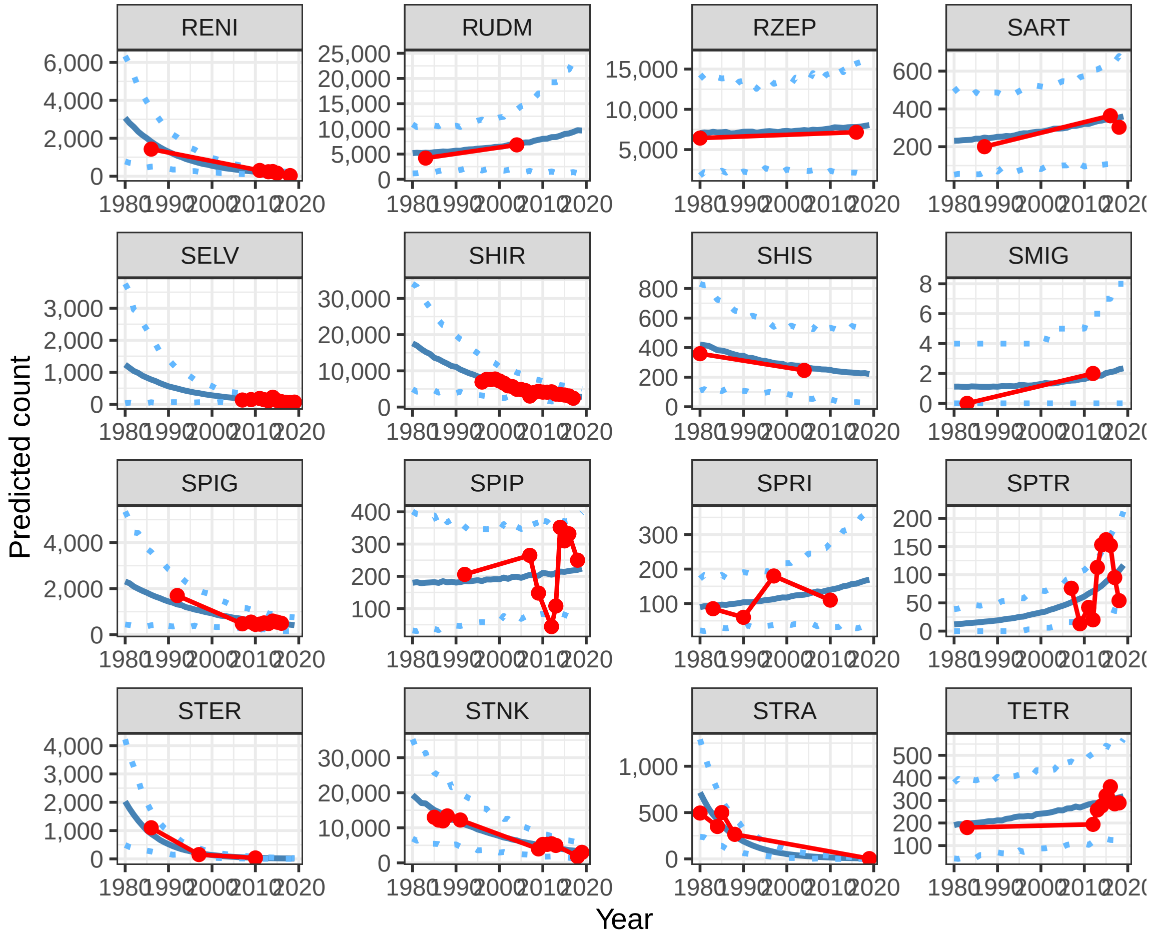
```r
#required_n_pages = round(101/16)+1
required_n_pages = round(70/16)+1
for(i in 1:required_n_pages){
```

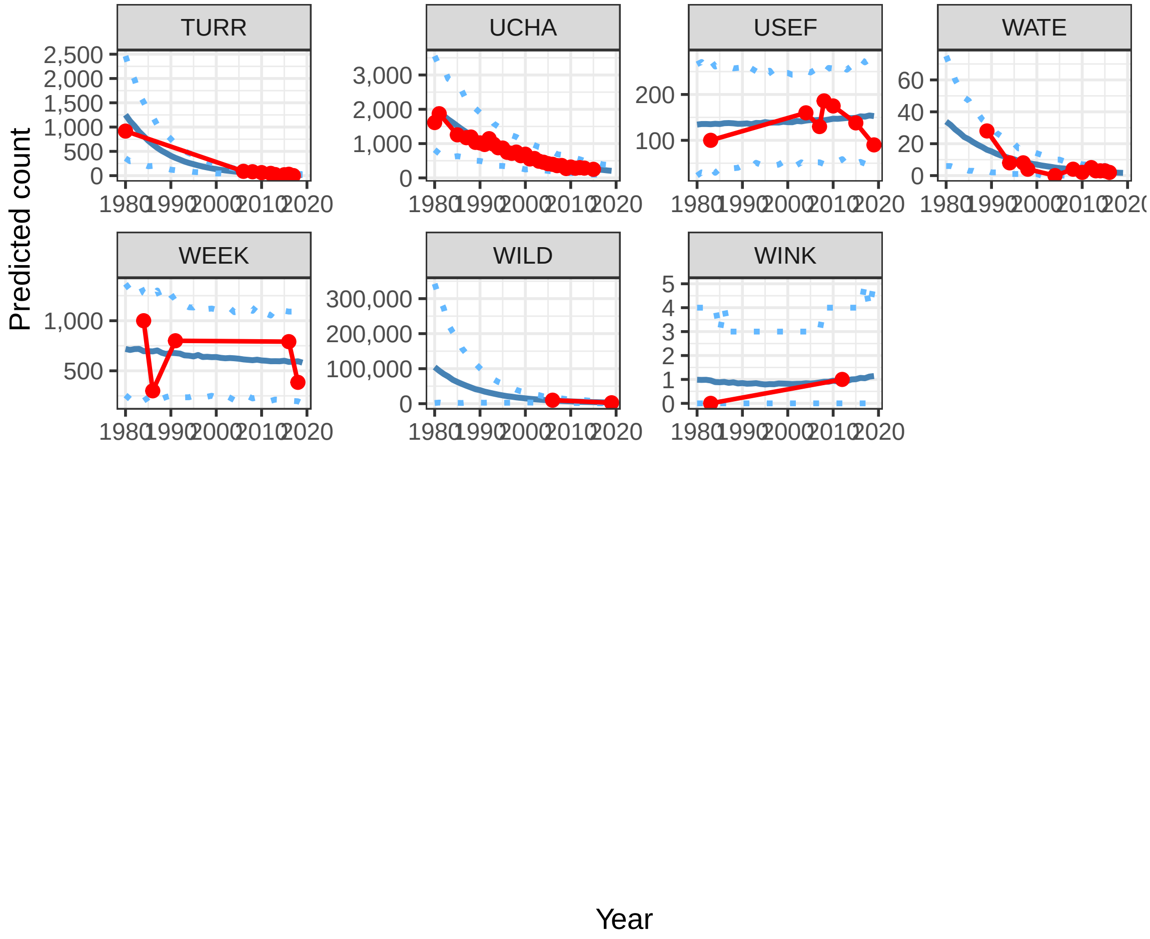
```
print(ggplot(data = popy) +
    geom_line(aes(x = season_starting, y = Zfit),
              col = "steelblue", linewidth=1.04) +
    geom_line(aes(x = season_starting, y = Zlwr),
              col = "steelblue1", linetype="dotted", linewidth = 1.02) +
    geom_line(aes(x = season_starting, y = Zupr),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




DREA
DUPT
ECKE
ENTR
FERG
FITC
FORT
GAST
GEDD
GEOR
GUES
HALF
HANN
HUMB
HYDR
JING
Predicted count
Year
0
1,000
2,000
0
100
200
300
25
50
75
0
500
1,000
1,500
0
5,000
10,000
15,000
20,000
0
1,000
2,000
3,000
500
1,000
1,500
2,000
200
400
600
0
5,000
10,000
15,000
20,000
25,000
250
500
750
1,000
0
20
40
60
80
2,000
4,000
6,000
0
1,000
2,000
3,000
4,000
0.0
2.5
5.0
7.5
500
1,000
1980
1990
2000
2010
2020


KAYR
LION
LLAN
MIDA
MIER
MONU
NANC
ORNE
PAPT
PCHA
PCOR
PENG
PETE
PING
PIRI
PTHO
Predicted count
Year
0
500
1,000
10
20
30
40
50
200
400
600
800
2,000
3,000
4,000
6,000
20,000
30,000
250
750
12,000
16,000
40,000
1
2
3
8,000
1980
1990
2000
2010
2020

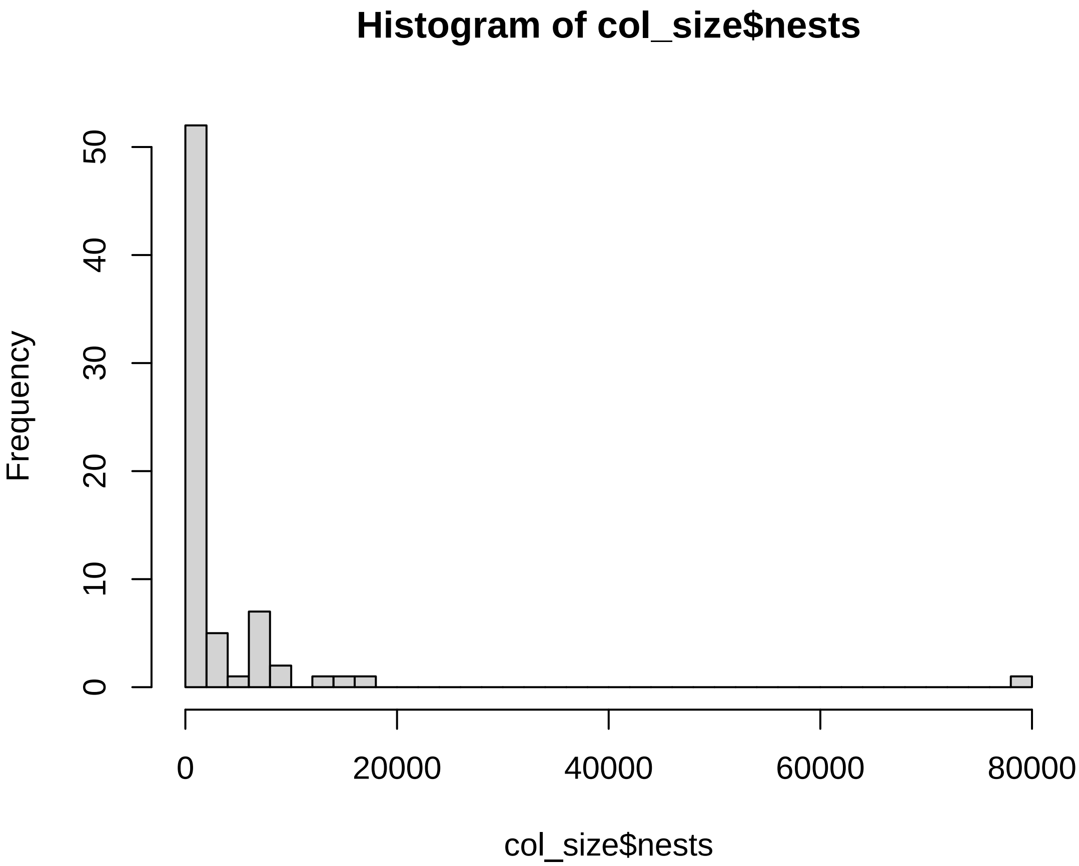

RENI
RUDM
RZEP
SART
SELV
SHIR
SHIS
SMIG
SPIG
SPIP
SPRI
SPTR
STER
STNK
STRA
TETR
6,000
4,000
2,000
0
25,000
20,000
15,000
10,000
5,000
0
15,000
10,000
5,000
600
400
200
3,000
2,000
1,000
0
30,000
20,000
10,000
0
800
600
400
200
0
8
6
4
2
0
4,000
2,000
0
400
300
200
100
300
200
100
200
150
100
50
0
4,000
3,000
2,000
1,000
0
30,000
20,000
10,000
0
1,000
500
0
500
400
300
200
100
1980
1990
2000
2010
2020
Predicted count
Year



```
# Predictions are generally good, but back-predicting to 1960 is extrapolation
# (there are only 1 or 2 counts prior to 1970) so uncertainty (prediction intervals)
# is high.


# Look at some sites with temporally sparse data and strong extrapolation
nestM3 %>% dplyr::filter(site_id == "MUCK")
```

```
##  [1] site_id          Lat              Lon              season_starting
##  [5] nests            ncounts          minseason        maxseason
##  [9] interval         Zseason_starting ZLat
## <0 rows> (or 0-length row.names)
```

```
col_size = nestM3 %>%
          group_by(site_id) %>%
          slice(which.max(nests))

hist(col_size$nests, breaks = 30)
```



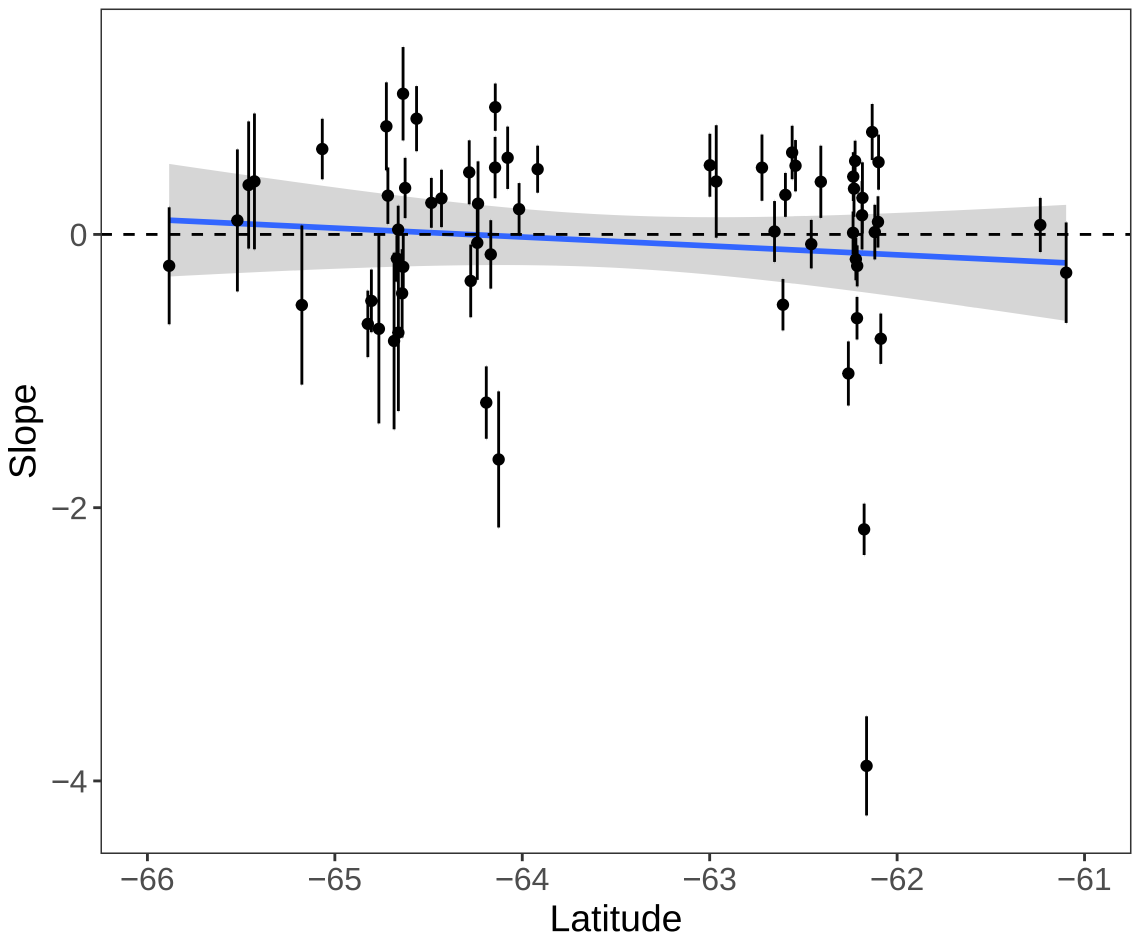
```r
# extract random effects from MCMCglmm
# https://stackoverflow.com/questions/64562052/extract-random-effects-from-mcmcglmm
library(broom.mixed)
re = tidy(mc2, effects="ran_vals")
unique(re$group)
```

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

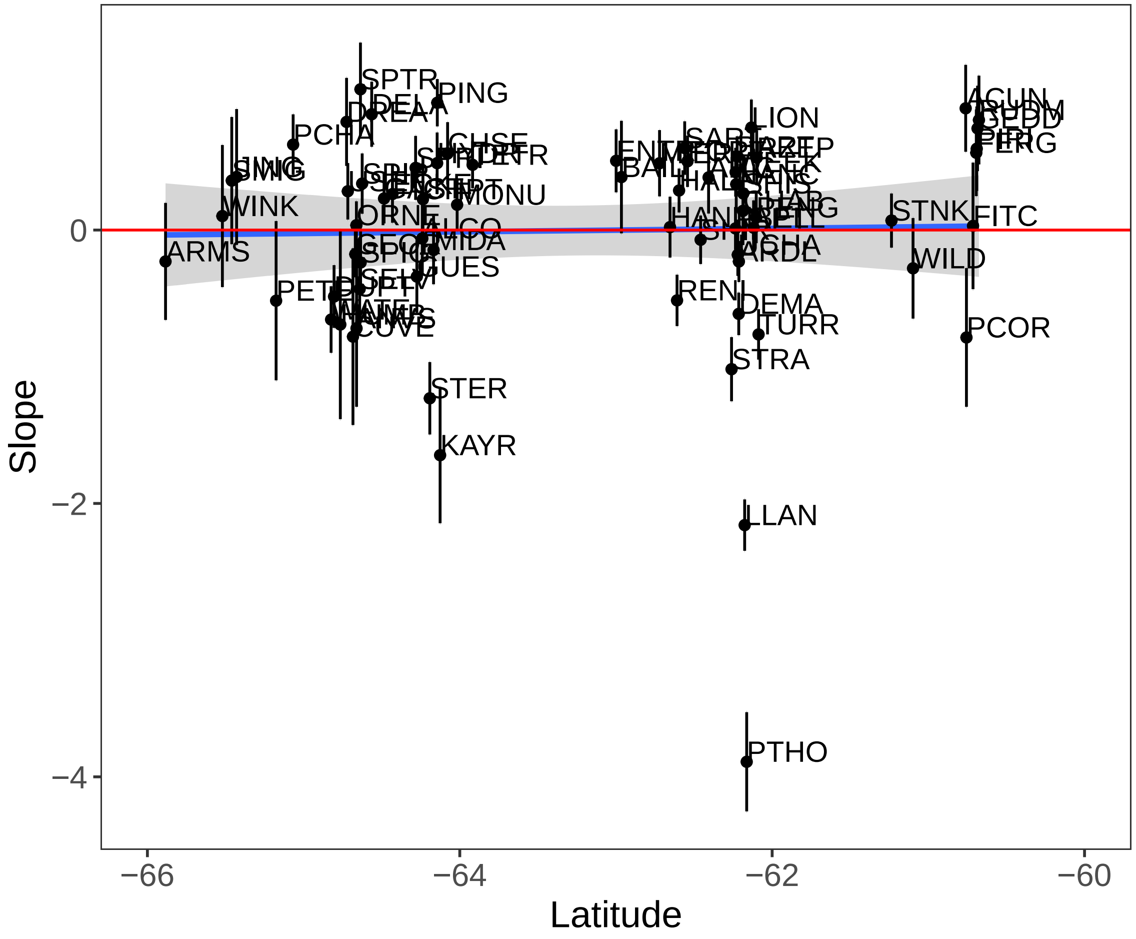
```r
re = re %>%
    dplyr::select(-group, -effect) %>%
    pivot_wider(names_from = term, values_from = c(estimate, std.error))

head(re)
```

```
## # A tibble: 6 x 5
##   level ‘estimate_(Intercept)‘ estimate_Zseason_starting ‘std.error_(Intercept)‘
##   <chr>                  <dbl>                     <dbl>                  <dbl>
## 1 ACUN                   0.737                     0.891                  0.695
## 2 AITC                   2.33                      0.385                  0.433
## 3 ALCO                   3.11                     -0.0613                 0.440
## 4 ANVS                  -3.98                     -0.719                  1.04
## 5 ARDL                  -3.00                     -0.230                  0.412
## 6 ARMS                  -1.31                   -0.230                    0.841
## # i 1 more variable: std.error_Zseason_starting <dbl>
```

```
# 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

lat_plot = 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")+
  scale_x_continuous(lim = c(-66,-61), breaks = seq(-66, -61, by = 1))+
  xlab("Latitude")+
  geom_hline(yintercept=0,
             color = "black", lty = 2)

lat_plot
```



```
## Save Plot
# pdf("./figures/MS_Latitude.pdf",
#       useDingbats = FALSE, width = 5, height = 4)
# plot(lat_plot)
# dev.off()

ggplot(data = re, aes(x = Lat, y = estZss, label = site_id))+
  stat_smooth(method="gam",formula=y~s(x,k=2))+
  # geom_smooth(method='lm', formula= y~x)+
  geom_point()+
  geom_text(hjust=0, vjust=0) +
  geom_errorbar(aes(ymin=estZss-seZss,
                    ymax=estZss+seZss))+
  theme_bw()+th+
  ylab("Slope")+xlim(-66,-60)+
  xlab("Latitude")+
  geom_hline(yintercept=0,
             color = "red")
```



7 Oosthuizen et al Population change:

```
# How many penguins were there, per year, across all sites?
# We don't know, as we only have intermittent counts.

# calculate total population size predicted (how many penguins were there in all populations?)
head(popy)
```

```
##   site_id     Lat season_starting nests Zseason_starting     ZLat minyear
## 1    ACUN -60.761            2018     0        1.3259096 1.879658    1983
## 2    ACUN -60.761            1980     0       -1.9853633 1.879658    1983
## 3    ACUN -60.761            2019     0        1.4130484 1.879658    1983
## 4    ACUN -60.761            1995     0       -0.6782819 1.879658    1983
## 5    ACUN -60.761            2005     0        0.1931057 1.879658    1983
## 6    ACUN -60.761            1989     0       -1.2011144 1.879658    1983
##   maxyear      Zfit Zlwr  Zupr
## 1    2004 11904.431 1506 27566
## 2    2004  4782.299 1089  9664
## 3    2004 12407.094 1061 30525
## 4    2004  6325.511 2015 11992
## 5    2004  8181.861 2073 16091
## 6    2004  5590.439 1588 10499
```

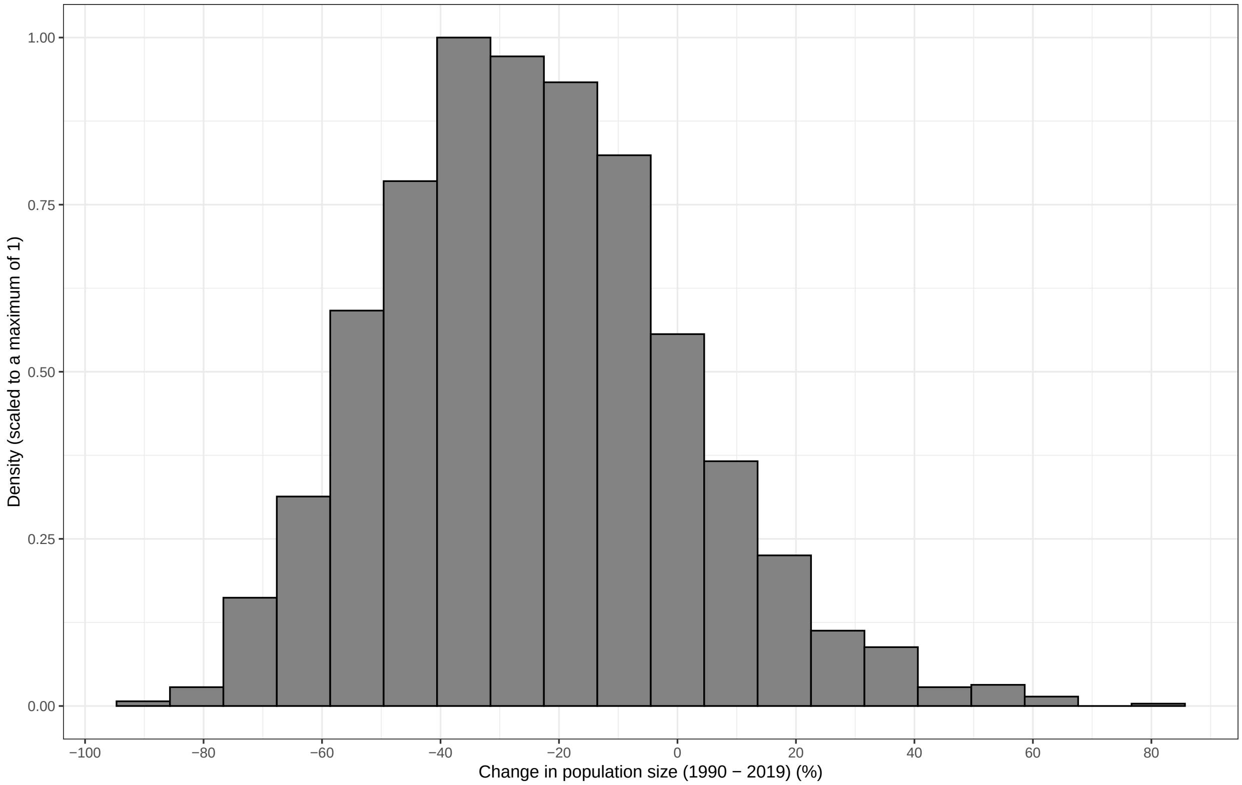
```
pop_predict = popy %>%
#           dplyr::filter(site_id != "HARM") %>%
              dplyr::group_by(season_starting) %>%
              dplyr::summarise(total_pred = sum(Zfit),
                       min_pred = sum(Zlwr),
                       max_pred = sum(Zupr))

pop_predict.p = ggplot(data = pop_predict) +
 geom_line(aes(x = season_starting, y = total_pred),
          lty = 1, linewidth = 1.1)+
  geom_line(aes(x = season_starting, y = min_pred), lty = 2, linewidth = 0.8)+
  geom_line(aes(x = season_starting, y = max_pred), lty = 2, linewidth = 0.8)+
  labs(x = "Year", y = "Total population count") +
  theme_bw()+
  scale_y_continuous(label = comma)
 # labs(subtitle = "Total predicted counts across all sites")+
 # guides(color=guide_legend(title="Data source"))+
 # theme(legend.position = c(0.7, 0.9))

# pop_predict.p
#
# delta.y = 100 * (pop_predict[40,2] - pop_predict[1,2]) / pop_predict[1,2]
# delta.y
```

8 Predicting population change with entire posterior distribution

```
# extract posterior draws of fixed effects and random effects
posterior <- as.matrix(mc2$Sol)

# collect site-level information
site_and_lat <- nestM3 %>%
  as_tibble() %>%
  select(site_id, ZLat) %>%
  distinct()

site_and_lat
```

```
## # A tibble: 71 x 2
##    site_id ZLat[,1]
##    <chr>      <dbl>
##  1 ACUN       1.88
##  2 AITC       0.604
##  3 ALCO      -0.817
##  4 ANVS      -1.14
##  5 ARDL       0.754
##  6 ARMS      -2.09
##  7 BAIL       0.172
##  8 BART       0.746
##  9 BELL       0.827
## 10 CHAB       0.774
## # i 61 more rows
```

```
# map years which to predict to (standardised scale)
# Here, the first year is 1990 and the last year is 2020 (30 year change)

year1 = 1990
year2 = 2019

first_year <- (year1 - mean(nestM3$season_starting)) / sd(nestM3$season_starting)
last_year <- (year2 - mean(nestM3$season_starting)) / sd(nestM3$season_starting)


# define function to predict with or without random effects
get_predictions <- function(posterior,
                            site_and_lat,
                            first_year,
                            last_year,
                            use_random_effects = FALSE) {
  # matrices for predictions at each site in year 1 (1990) and year 2 (2020)
  # each row is a prediction from a different posterior sample, each column is a site
  pred_pop_per_site.first <- matrix(NA, nrow = nrow(posterior), ncol = nrow(site_and_lat))
  pred_pop_per_site.last <- matrix(NA, nrow = nrow(posterior), ncol = nrow(site_and_lat))

  for (s in 1:nrow(posterior)) {
    theta <- posterior[s,]
    for (j in 1:nrow(site_and_lat)) {
      site_id <- site_and_lat$site_id[j]
      ZLat <- site_and_lat$ZLat[j]

      # predict pop at site j in first year
      lin_pred <- theta["(Intercept)"] +
        theta["Zseason_starting"] * first_year +
        theta["ZLat"] * ZLat +
        theta["Zseason_starting:ZLat"] * first_year * ZLat
      if (use_random_effects) {
        lin_pred <- lin_pred +
          theta[ str_c("(Intercept).site_id.",site_id) ] +
          theta[ str_c("Zseason_starting.site_id.",site_id) ] * first_year
      }
      pred_pop_per_site.first[s,j] <- exp( lin_pred )

      # predict pop at site j in last year
      lin_pred <- theta["(Intercept)"] +
        theta["Zseason_starting"] * last_year +
        theta["ZLat"] * ZLat +
        theta["Zseason_starting:ZLat"] * last_year * ZLat
      if (use_random_effects) {
        lin_pred <- lin_pred +
          theta[ str_c("(Intercept).site_id.",site_id) ] +
          theta[ str_c("Zseason_starting.site_id.",site_id) ] * last_year
      }
      pred_pop_per_site.last[s,j] <- exp( lin_pred )
    }
  }

  # sum over sites for population level predictions
  pred_pop.first <- rowSums(pred_pop_per_site.first)
  pred_pop.last <- rowSums(pred_pop_per_site.last)

  # percent change from year1 to year2
  pred_pop_change <- 100 * ( pred_pop.last - pred_pop.first ) / pred_pop.first

  # outputs
  predictions <- list(pop_per_site.first = pred_pop_per_site.first,
                      pop_per_site.last = pred_pop_per_site.last,
                      pop.first = pred_pop.first,
                      pop.last = pred_pop.last,
                      pop_change = pred_pop_change)
  predictions
}

#------ Set Plotting theme----------------
gg_theme <- function () {
  theme_bw() %+replace%
    theme(
      axis.text = element_text(colour = "black", size = 11),
      axis.title = element_text(size=13),
      axis.ticks = element_line(colour = "black"),
    #  panel.grid = element_blank(),
    #  strip.background = element_blank(),
      panel.border = element_rect(colour = "black", fill = NA),
      axis.line = element_line(colour = "black"),
    legend.background = element_blank())
}
# Make predictions with and without random effects
pred_re <- get_predictions(posterior, site_and_lat, first_year, last_year,
                           use_random_effects = TRUE)

# Plot histogram of population change using random effects in prediction:

hist_growth =
  ggplot(data = data.frame(pred_re$pop_change), aes(x = pred_re.pop_change,
                                                     after_stat(ndensity)))+
  geom_histogram(bins = 20, colour = "black", fill = "grey51")+
  theme_bw()+
  scale_x_continuous(n.breaks = 10)+
  labs(y= "Density (scaled to a maximum of 1)",
       x = "Change in population size (1990 - 2019) (%)")

hist_growth
```



```
# can calculate the probability that the population has decreased by
# at least thirty percent with
sum(pred_re$pop_change < -30)/2000  # McElreath Chapter 3
```

```
## [1] 0.434
```

```
mean(pred_re$pop_change < -30)
```

```
## [1] 0.434
```

```
mean(pred_re$pop_change < 0)
```

```
## [1] 0.8405
```

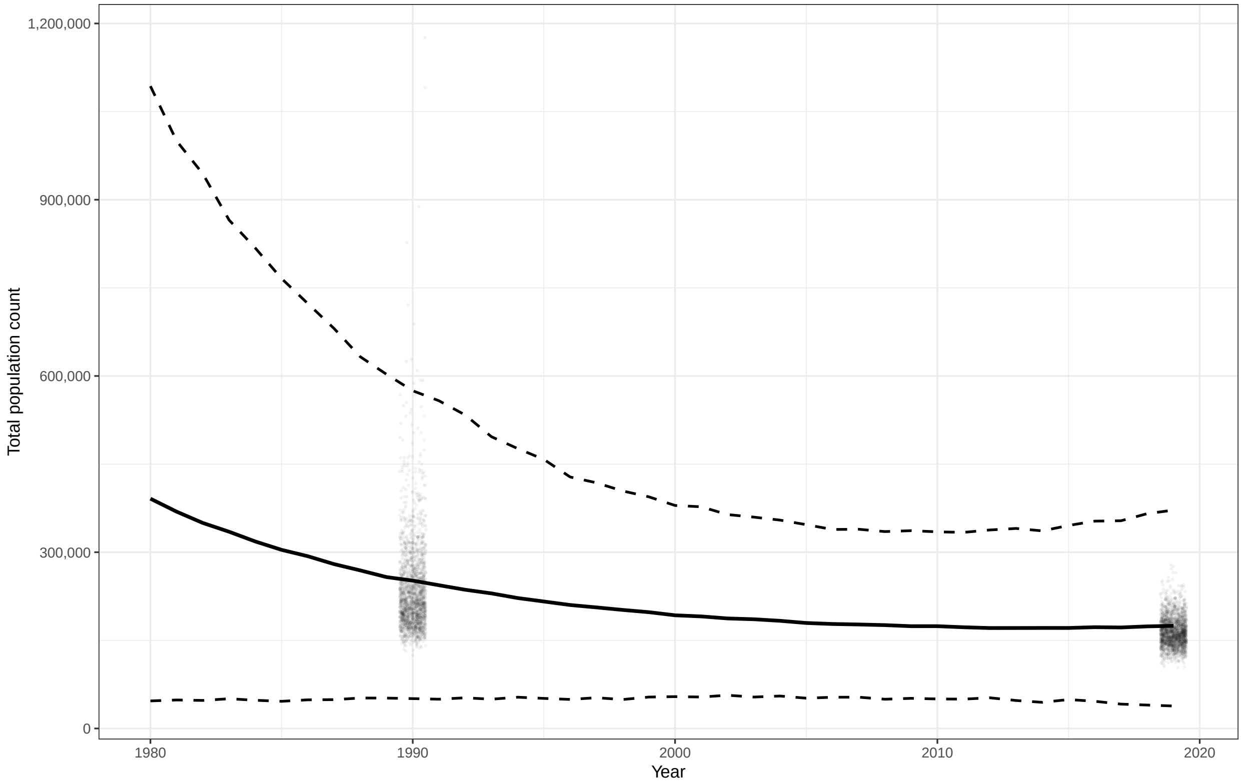
```
quantile(pred_re$pop_change,c(0.05,0.95))
```

```
##         5%       95%
## -61.98294  19.95688
```

```
length(unique(nestm3$site_id))
```

```
## [1] 71
```

```r
# estimated population size in year1 (with random effects)
pred_first = as.data.frame(pred_re$pop.first)
names(pred_first) = "pred_first"
#hist(pred_first$pred_first, breaks = 20)

# estimated population size in year2 (with random effects)
pred_last = as.data.frame(pred_re$pop.last)
names(pred_last) = "pred_last"
#hist(pred_last$pred_last, breaks = 20)

#-----------------------------------------------------------------------------
# pred_no_re <- get_predictions(posterior, site_and_lat, first_year, last_year,
#                                 use_random_effects = FALSE)
#
# # Plot histogram of population change without random effects in prediction:
# hist(pred_no_re$pop_change, breaks = 20)
#
# # estimated population size in year1 (no random effects)
# pred_first_noRE = as.data.frame(pred_no_re$pop.first)
# names(pred_first_noRE) = "pred_first"
#
# # estimated population size in year2 (no random effects)
# pred_last_noRE = as.data.frame(pred_no_re$pop.last)
# names(pred_last_noRE) = "pred_last"
#------------------------------------------------------------------------------
pop_predict.p = pop_predict.p +
  geom_jitter(data = pred_first, aes(x = year1, y = pred_first),
             col = "black",size = 1, height = 0, width = 0.5,
              alpha = 0.05, stroke = NA) +
  geom_jitter(data = pred_last, aes(x = year2, y = pred_last),
             col = "black",size = 1, height = 0, width = 0.5,
              alpha = 0.05, stroke = NA)

pop_predict.p
```



```
## Save Plot
# pdf("./figures/MS_Population_change.pdf",
#       useDingbats = FALSE, width = 10, height = 4)
# cowplot::plot_grid(pop_predict.p, hist_growth, labels = c('A', 'B'), ncol = 2, label_size = 12)
# dev.off()
```