Analysis of simulated population counts with GLMMs

Supporting documentation: Decreasing trends of chinstrap penguin breeding colonies in a region of major and ongoing environmental change suggest population level vulnerability: a reanalysis of Krüger (2023)

Chris Oosthuizen, Murray Christian, Mzabalazo Ngwenya

2023-10-05

Contents

1 Purpose

This script (1) simulates population counts for 26 sites over 60 years; (2) fit different glmer (lme4) and MCMCglmm (MCMCglmm) mixed model specifications to the simulated data; and (3) predict population counts from the fitted models, using various model choices for prediction.

Simulation studies, such as the one we conduct here, are useful to help understand the behavior and adequacy of different statistical models (i.e., to assess how well different models can recover the true relationships

between variables). The results show that Krüger (2023)'s MCMCglmm analysis fail to predict population trends (even when the data is strong) and that better modelling choices are required to reproduce the observed data. (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.)

2 Simulate data

The following line of code sigma2.lambda <- 0.01 controls the annual variation in growth rate. If the value is small (e.g., sigma2.lambda <- 0.0001) the counts at each site increase or decrease relatively smoothly. Annual variation (population increase or decrease) around the modelled trend increase as sigma2.lambda gets larger. Here, we use sigma2.lambda <- 0.0005. In the 'sparse data' analysis, we use # sigma2.lambda <- 0.005.

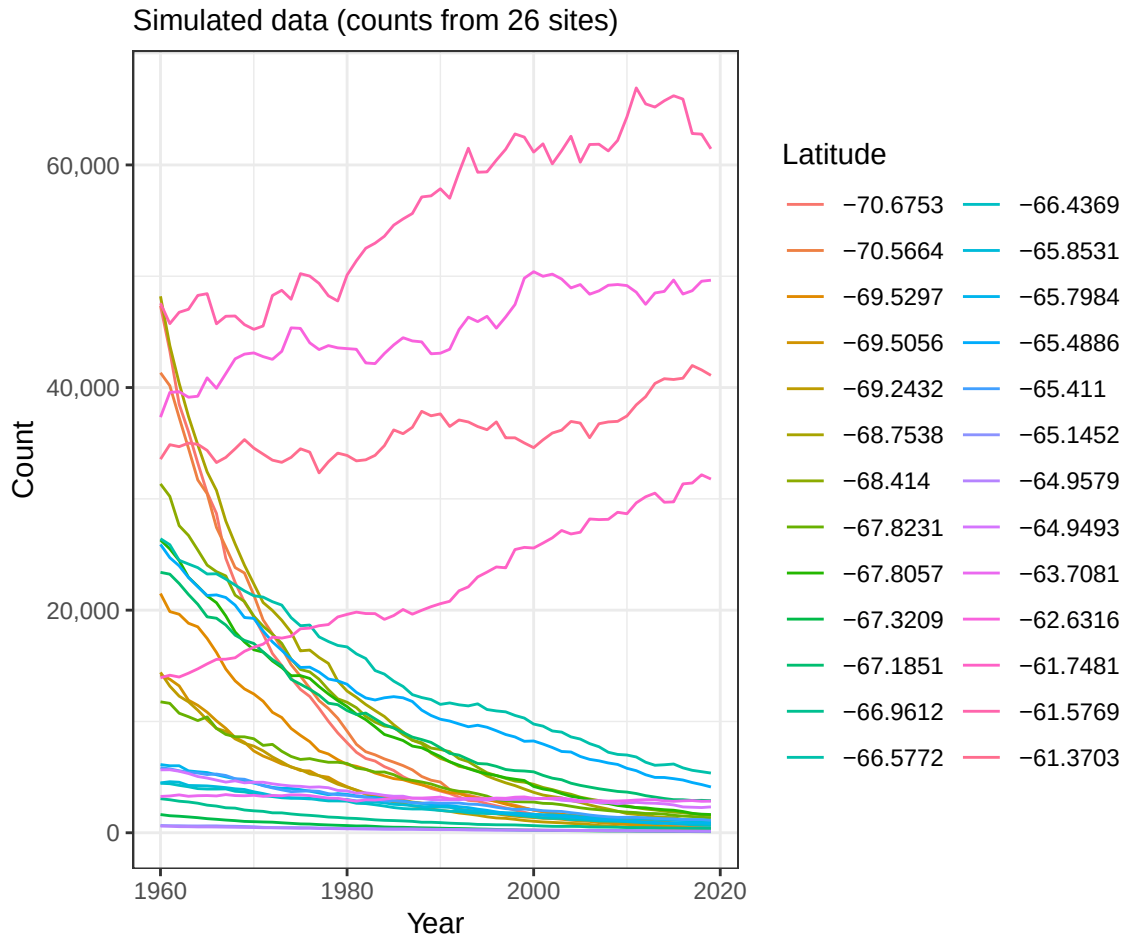
```
# load R packages
library(tidyverse)
library(lme4)
library(MCMCglmm)
library(scales)
library(colorspace)
library(gridExtra)
# ! make sure library plyr (used in Krüger (2023) code) is not loaded !

# 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())
}

#-----------------------------------
# Simulate data
#-----------------------------------

# Simulation based on Chapter 5 from State-Space Models for Population Counts from
# Bayesian Population Analysis using Winbugs by Marc Kery and Michael Schaub
# ISBN: 978-0-12-387020-9

# Make an empty list to save output in
list1 = list()   # for population counts
list2 = list()   # for lambda of each population

# Set seed for reproducibility
set.seed(1234)
```

```r
# Choose how many populations and how many years you want to simulate
n.populations = 26 # Number of populations (max = 26)
n.years <- 60 # Number of years
start.year = 1960  # Start year
years = start.year:(start.year+n.years-1)  # Year sequence

# simulate
for(i in 1:n.populations) {

  N1 <- runif(1, 500, 50000) # Initial population size
  mean.lambda <- runif(1, 0.92, 1.01) # Mean annual population growth rate
  sigma2.lambda <- 0.0005 # Process (temporal) variation of the growth rate
  sigma2.y <- 0 # Variance of observation error (0 assumes 100% accurate counts)

  y <- N <- numeric(n.years)
  N[1] <- N1

  lambda <- rnorm(n.years-1, mean.lambda, sqrt(sigma2.lambda))

  for (t in 1:(n.years-1)){
    N[t+1] <- N[t] * lambda[t]
  }

  for (t in 1:n.years){
    y[t] <- rnorm(1, N[t], sqrt(sigma2.y))
  }

  # Save output in list for each iteration
  list1[[i]] = as.data.frame(y)
  list2[[i]] = as.data.frame(mean.lambda)
}

# Build data frame from simulations of count
df = bind_rows(list1)
names(df) = "count"
# add year to simulated counts
df$year = as.integer(rep(years,n.populations))
# add site to simulated counts
df$site = rep(LETTERS[1:n.populations], each = n.years)
# make df a tibble
df = as_tibble(df)

# Use list 2 (lambda) to generate a latitude value for each site that
# correlate with the site's growth rate (lambda)
lambda = bind_rows(list2)
lambda$site = LETTERS[1:n.populations]
lambda$noise = runif(n.populations, -1, 1)

df = merge(df, lambda, by = "site")

df$r = df$mean.lambda-1   # convert lambda to growth rate r
df$r100 = df$r * 100   # rescale
```

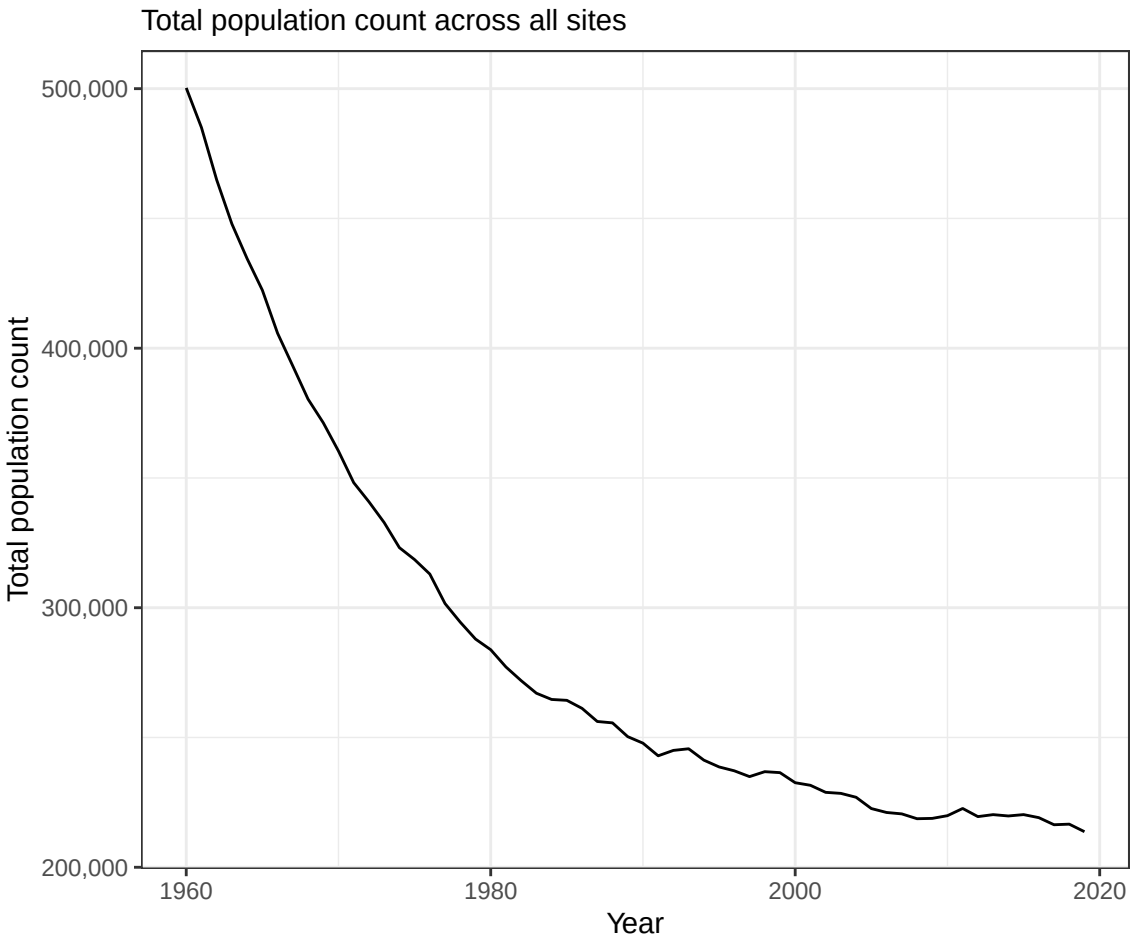
```r
# create one latitude value per site where the mean latitude (-63 degrees S)
# increase or decrease based on the growth rate of the population plus a small
# random component

df = df %>%
  group_by(site) %>%
  mutate(latitude = -63 + r100 + noise) %>%
  ungroup()

# Inspect df: every site should have 1 unique latitude value
df %>%
  group_by(site) %>%
  summarise(count = n_distinct(latitude)) %>%
  summarise(max_sites_per_lat = max(count))
```

```
## # A tibble: 1 x 1
##   max_sites_per_lat
##               <int>
## 1                 1
```

```r
# Counts must be positive and integers
df = df %>%
  dplyr::filter(count > 0) %>%
  dplyr::select(site, count, year, latitude) %>%
  mutate_at(vars(latitude), round, 4) %>%
  mutate_at(vars(count), round, 0)

# Plot data used for fitting models: every sites has a count every year.
ggplot(df, aes(x = year, y = count, color = as.factor(latitude))) +
  geom_line() +
  labs(x = "Year", y = "Count") +
  scale_color_discrete(name = "Latitude")+
  theme_bw()+
  scale_y_continuous(label = comma)+
  labs(subtitle = "Simulated data (counts from 26 sites)")
```




```r
# How many simulated individuals were there, per year, across all sites?
population = df %>%
             group_by(year) %>%
             summarise(totalN = sum(count))

# Plot total N (all sites) per year
ggplot(population, aes(x = year, y = totalN)) +
  geom_line() +
  labs(x = "Year", y = "Total population count") +
  theme_bw()+
  scale_y_continuous(label = comma)+
  labs(subtitle = "Total population count across all sites")
```



3 glmer (lme4) model fitting

Fit glmm models in a frequentist framework using lme4.

```
# random slope model with interaction between year and latitude that test
# whether latitude influence population trend
m1 = glmer(count ~ year * latitude + (year|site), family = "poisson", data = df)
```

```
## Warning: Some predictor variables are on very different scales: consider
## rescaling
```

```
## Error in (function (fr, X, reTrms, family, nAGQ = 1L, verbose = 0L, maxit = 100L, : Downdated VtV is
```

```
# model convergence problems occur. Can be avoided by scaling variables
# to mean = 0, sd = 1
df$zyear = scale(df$year)
df$zlatitude = scale(df$latitude)

# Refit with scaled predictors variables
m2 = glmer(count ~ zyear * zlatitude + (zyear|site), family = "poisson", data = df)
```

```r
# random slope model: assume latitude does not affect overall count (intercepts),
# only the slope of the year effect
m3 = glmer(count ~ zyear + zyear:zlatitude + (zyear|site), family = "poisson", data = df)

# Frequentist representation of the Kruger (2023) model
m_Kr = glmer(count ~ zyear + (zlatitude|site), family = "poisson", data = df)
```

```
## Warning in checkConv(attr(opt, "derivs"), opt$par, ctrl = control$checkConv, : Model is nearly unide
##  - Rescale variables?;Model is nearly unidentifiable: large eigenvalue ratio
##  - Rescale variables?
```

```r
# Note the warning message, even with z-standardized covariates

# Are populations nested in latitude? Nested random effects occur when a lower
# level factor appears only within a particular level of an upper level factor.

# How good is the relative fit of the models?
AIC(m2, m3, m_Kr)  # Note the AIC difference
```

```
##      df        AIC
## m2    7   38968.18
## m3    6   38970.19
## m_Kr  5 3510083.43
```

```r
summary(m2)
```

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: poisson  ( log )
## Formula: count ~ zyear * zlatitude + (zyear | site)
##    Data: df
##
##      AIC      BIC   logLik deviance df.resid
##  38968.2  39005.6 -19477.1  38954.2     1553
##
## Scaled residuals:
##      Min       1Q   Median       3Q      Max
## -24.8180  -1.7541  -0.0539   1.9760  19.6085
##
## Random effects:
##  Groups Name        Variance Std.Dev. Corr
##  site   (Intercept) 1.54458  1.24281
##         zyear       0.00902  0.09498  0.08
## Number of obs: 1560, groups:  site, 26
##
## Fixed effects:
##                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)      8.32304    0.24370  34.153   <2e-16 ***
## zyear           -0.58475    0.01864 -31.376   <2e-16 ***
## zlatitude        0.50747    0.24392   2.081   0.0375 *
## zyear:zlatitude  0.42604    0.01864  22.861   <2e-16 ***
```

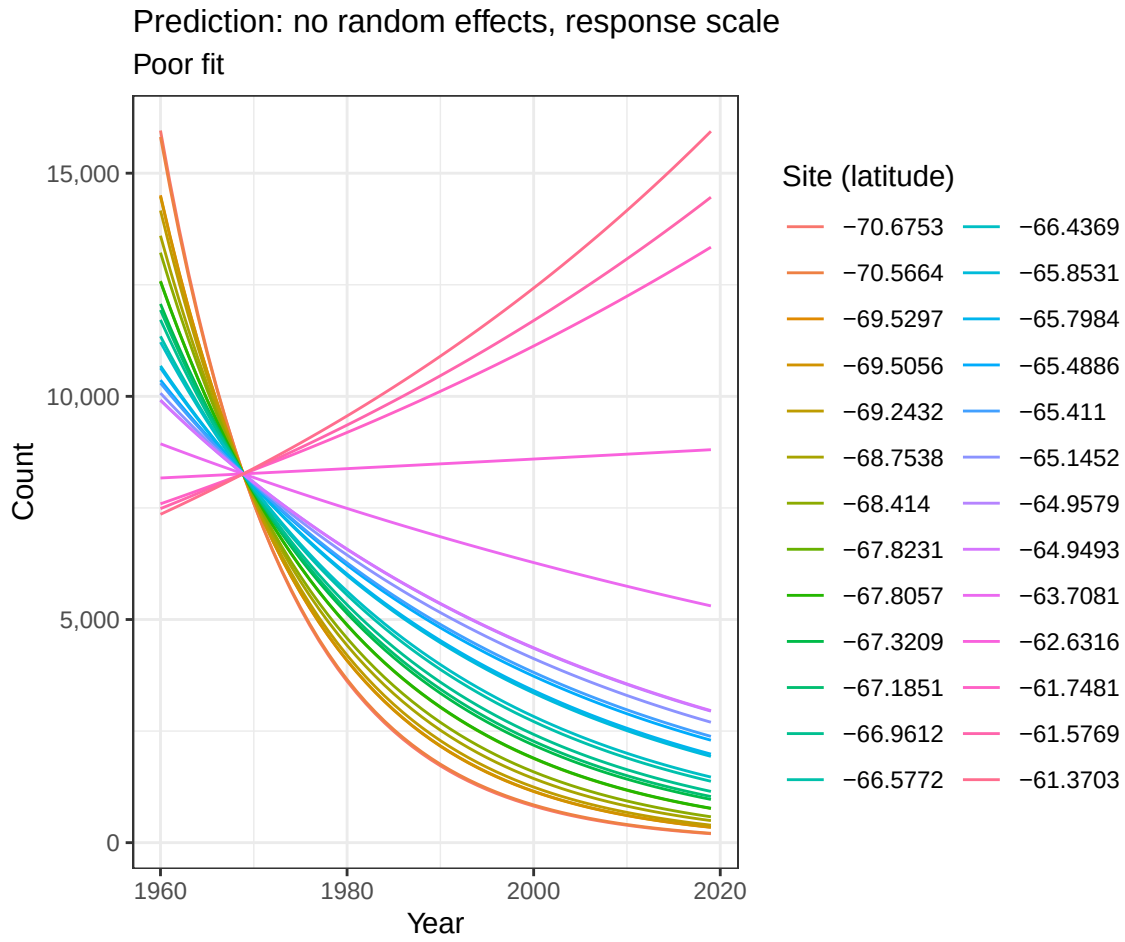
```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr) zyear zlattd
## zyear       0.080
## zlatitude   0.000  0.000
## zyear:zlttd 0.000  0.000 0.080
```

4 glmer (lme4) prediction

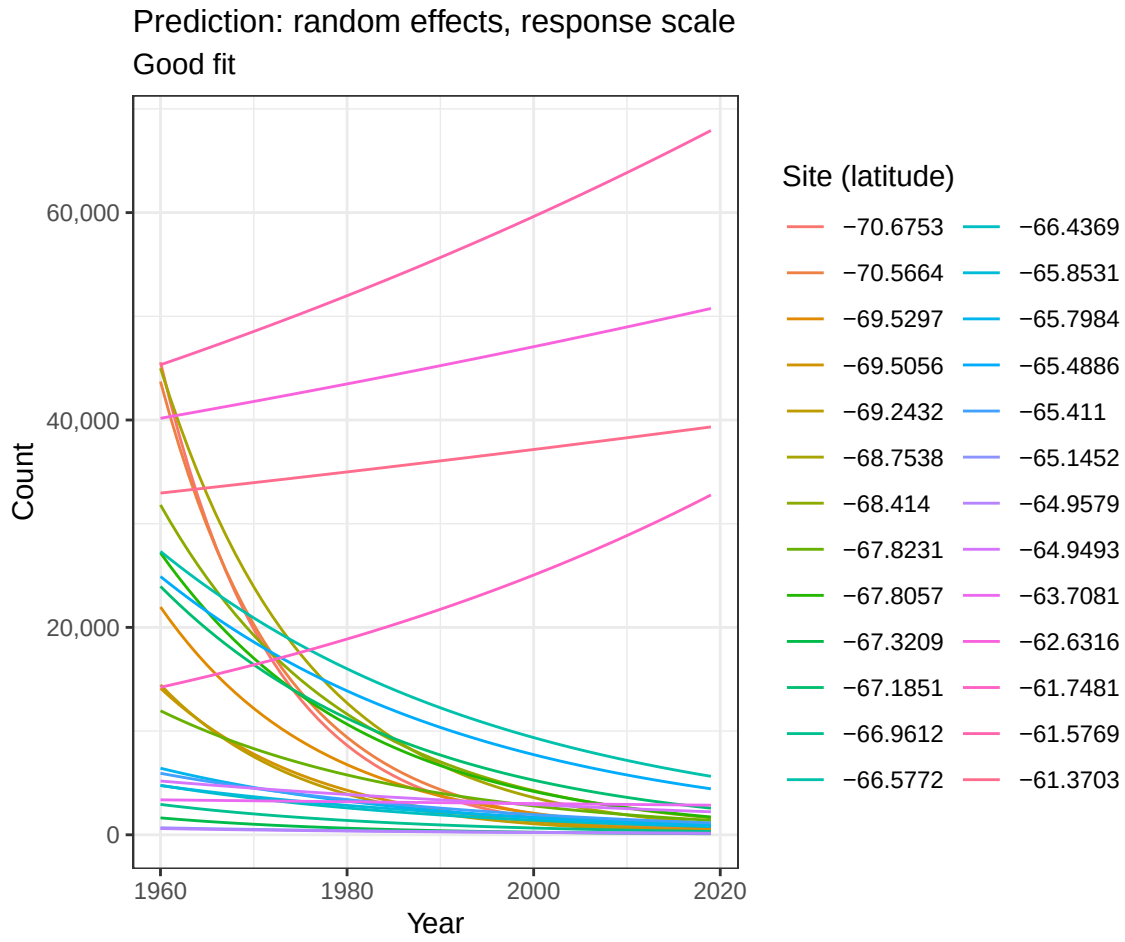
4.1 Prediction from m2

Predictions can be made with, or without, the contribution of random effects. The following plots shows the influence of this choice.

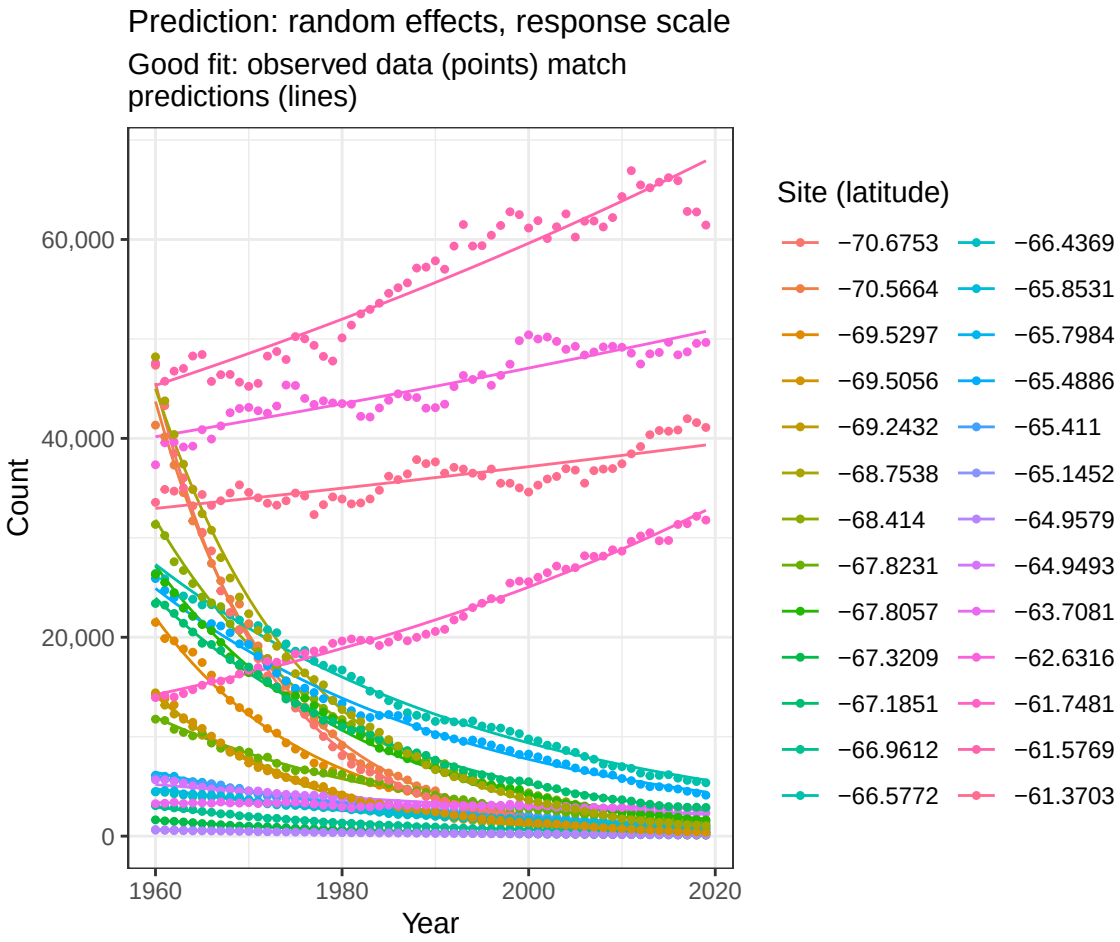
```
# glmer predict function: https://rdrr.io/cran/lme4/man/predict.merMod.html
# predict(object, newdata = NULL, newparams = NULL,
#         re.form = NULL, ReForm, REForm, REform,
#         random.only=FALSE, terms = NULL,
#         type = c("link", "response"), allow.new.levels = FALSE,
#         na.action = na.pass, ...)

# no random effects, response scale prediction
df$fit.m2_norand <- predict(m2, df, re.form=NA, type = "response")

ggplot(df, aes(x = year, y = fit.m2_norand, color = as.factor(latitude))) +
  geom_line() + theme_bw() +
  scale_y_continuous(label = comma)+
  labs(x = "Year", y = "Count") +
  scale_color_discrete(name = "Site (latitude)")+
  labs(title = "Prediction: no random effects, response scale",
       subtitle = "Poor fit")+
   theme(plot.title = element_text(size = 12))
```




```r
# with random effects, response scale
df$fit.m2 <- predict(m2, df, re.form=NULL, type = "response")

ggplot(df, aes(x = year, y = fit.m2, color = as.factor(latitude))) +
  geom_line() +  theme_bw() +
  scale_y_continuous(label = comma)+
  labs(x = "Year", y = "Count") +
  scale_color_discrete(name = "Site (latitude)") +
  labs(title = "Prediction: random effects, response scale",
       subtitle = "Good fit")+
   theme(plot.title = element_text(size = 12))
```

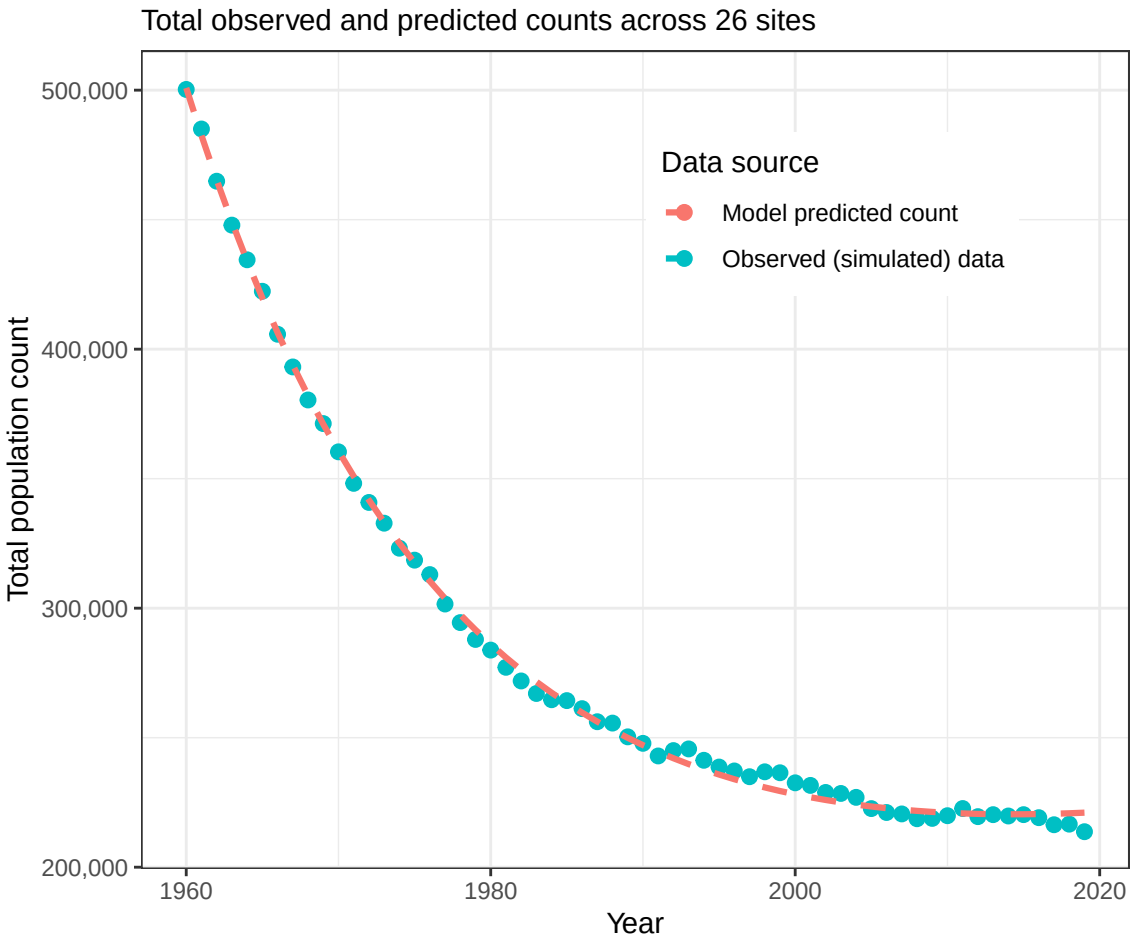



```
# now add observed data
ggplot(df, aes(x = year, y = fit.m2, color = as.factor(latitude))) +
  geom_line() +  theme_bw() +
  labs(x = "Year", y = "Count") +
  scale_y_continuous(label = comma)+
  scale_color_discrete(name = "Site (latitude)") +
  # add observed data
  geom_point(aes(x = year, y = count, color = as.factor(latitude)), size = 0.9) +
  labs(title = "Prediction: random effects, response scale",
       subtitle = "Good fit: observed data (points) match\npredictions (lines)")+
   theme(plot.title = element_text(size = 12))
```




```
# calculate total population size predicted (total N across all sites)
pop_predict = df %>%
              group_by(year) %>%
              summarise(total_pred = sum(fit.m2))

# Plot total N observed per year and the predicted total abundance per year
ggplot() +
  geom_point(data = population, aes(x = year, y = totalN,
                color = "Observed (simulated) data"), size = 2.3)+
  geom_line(data = pop_predict, aes(x = year, y = total_pred,
                color = "Model predicted count"), lty = 2, size = 1.1)+
  labs(x = "Year", y = "Total population count") +
  theme_bw()+
  scale_y_continuous(label = comma)+
  labs(subtitle = "Total observed and predicted counts across 26 sites")+
  guides(color=guide_legend(title="Data source"))+
  theme(legend.position = c(0.7, 0.8))
```

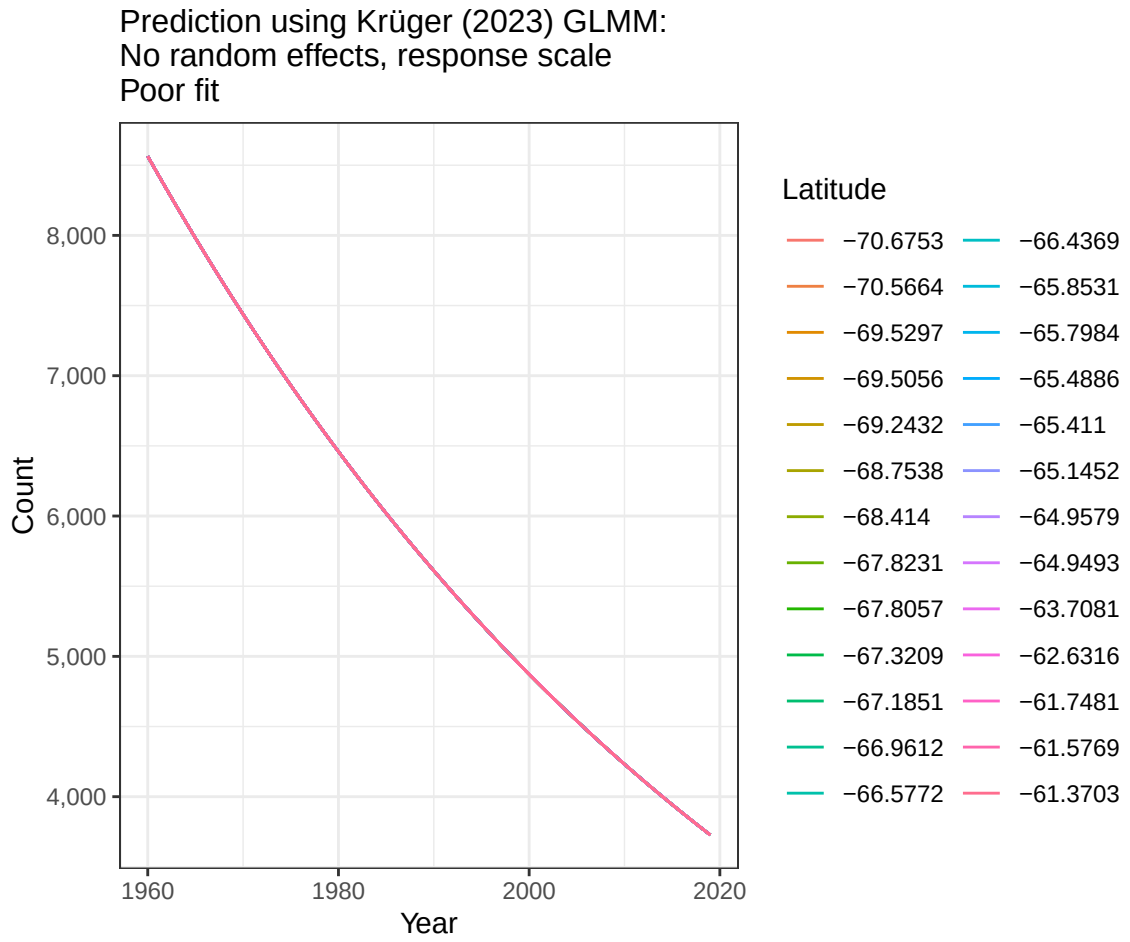
```
## Warning: Using ‘size‘ aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use ‘linewidth‘ instead.
## This warning is displayed once every 8 hours.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.
```



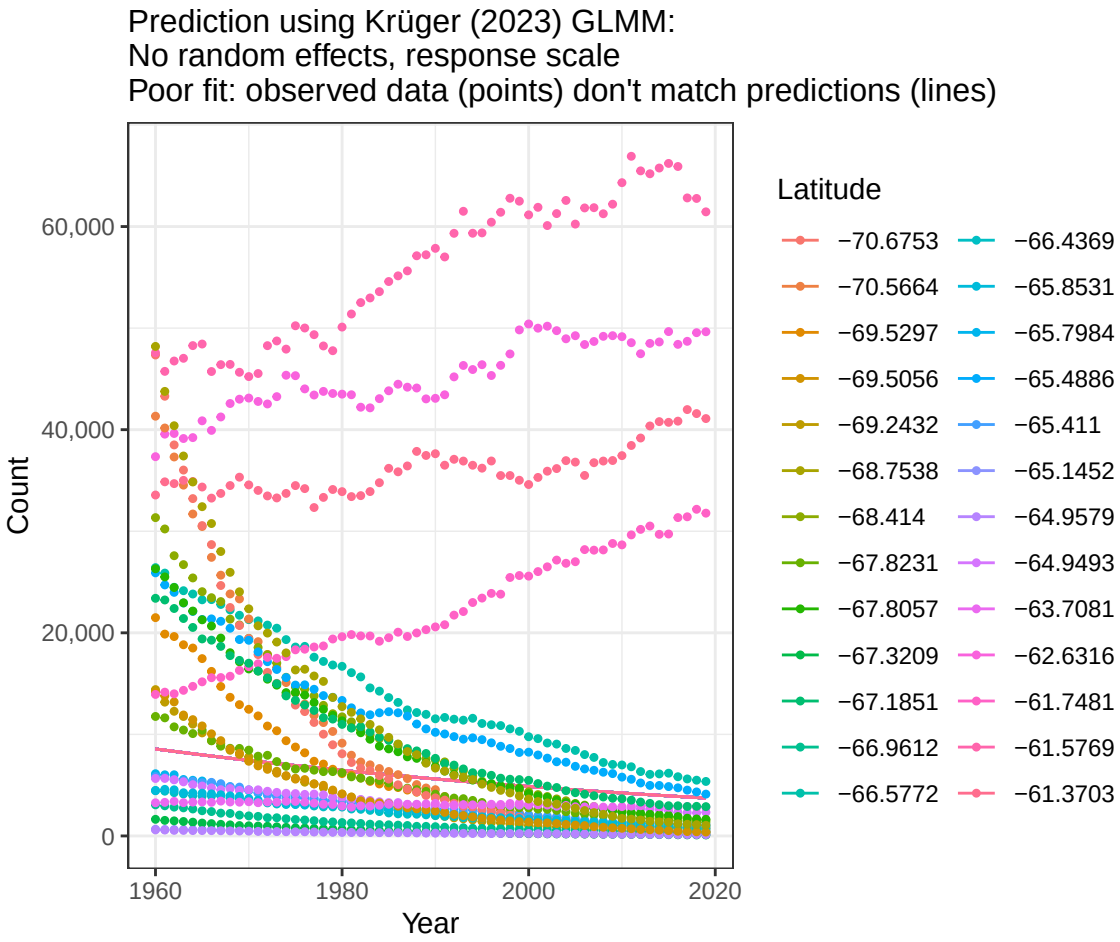
4.2 Prediction from m_Kr (frequentist GLMM syntax similar to Krüger 2023)

This prediction uses GLMM syntax similar to Krüger 2023, and is predicted without random effects (similar to Krüger 2023).

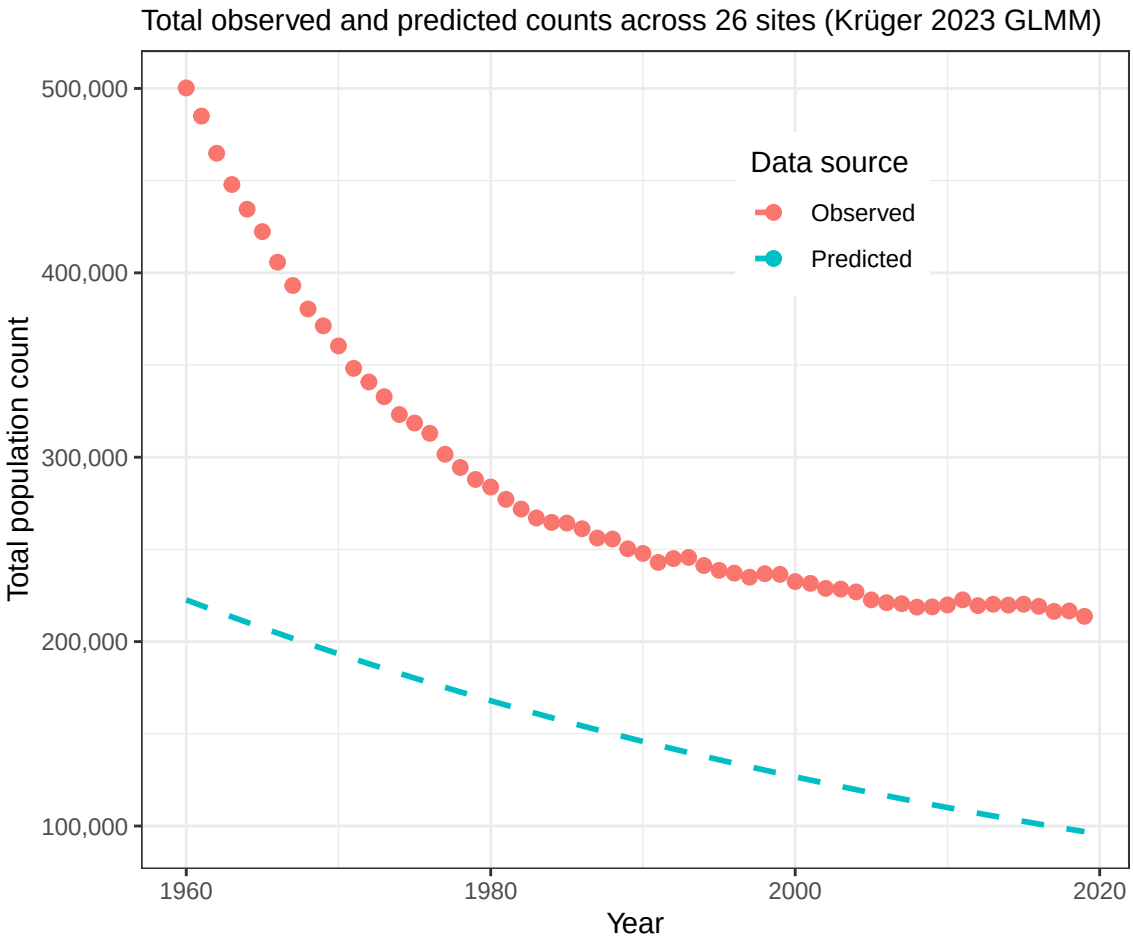
```
# With random effects, response scale prediction
df$fit.m_Kr <- predict(m_Kr, df, re.form=NA, type = "response")

ggplot(df, aes(x = year, y = fit.m_Kr, color = as.factor(latitude))) +
  geom_line() +
  labs(x = "Year", y = "Count") +
  scale_color_discrete(name = "Latitude") +
  theme_bw() +
  scale_y_continuous(label = comma)+
  labs(title = paste0("Prediction using Kr",ds4psy::Umlaut["u"],
        "ger (2023) GLMM: \nNo random effects, response scale",
        subtitle = "\nPoor fit"))+
    theme(plot.title = element_text(size = 12))
```




```r
# now add observed data
ggplot(df, aes(x = year, y = fit.m_Kr, color = as.factor(latitude))) +
  geom_line() +
  labs(x = "Year", y = "Count") +
  scale_color_discrete(name = "Latitude") +
  # add observed data
  geom_point(aes(x = year, y = count, color = as.factor(latitude)), size = 0.9)+
  theme_bw() +
  scale_y_continuous(label = comma)+
  labs(title =  paste0("Prediction using Kr",
       ds4psy::Umlaut["u"], "ger (2023) GLMM: \nNo random effects, response scale",
  subtitle = "\nPoor fit: observed data (points) don't match predictions (lines)"))+
  theme(plot.title = element_text(size = 12))
```




```r
# population prediction from Kruger
pop_predict_Kruger = df %>%
             group_by(year) %>%
             summarise(total_pred = sum(fit.m_Kr))

# Plot total N per year and predicted total count per year
ggplot() +
  geom_point(data = population, aes(x = year, y = totalN, col = "Observed"), size = 2.3)+
  geom_line(data = pop_predict_Kruger, aes(x = year, y = total_pred,
            col = "Predicted"), lty = 2, size = 1)+
  labs(x = "Year", y = "Total population count") +
  theme_bw()+
  scale_y_continuous(label = comma)+
  labs(subtitle =
        paste0("Total observed and predicted counts across 26 sites",
               " (Kr",ds4psy::Umlaut["u"], "ger 2023 GLMM)"))+
  guides(color=guide_legend(title="Data source"))+
  theme(legend.position = c(0.7, 0.8))
```



5 MCMCglmm model fitting

Repeat the analysis with MCMCglmm, comparing the Krüger 2023 model to a new model.

```
df = as.data.frame(df) # a data frame is expected by MCMCglmm

# Use prior from Kruger (2023) (change 133 to 26)
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(26, 2, 2))))


# Fit model m2 from above
# Works with default prior or with prior specified above
mc2 <- MCMCglmm(count ~ zyear * zlatitude,
                random = ~us(1 + zyear):site,
                rcov=~units,
                family="poisson",
                data = df,
                mev=NULL,start=NULL,
```

```
                # prior=NULL,
                prior=prior,
                nodes="ALL", scale=TRUE,
                nitt=13000, thin=10, burnin=3000, pr=T,
                pl=FALSE, verbose=TRUE, DIC=TRUE, singular.ok=FALSE, saveX=TRUE,
                saveZ=TRUE, saveXL=TRUE, slice=FALSE, ginverse=NULL, trunc=FALSE)
```

```
##
##                         MCMC iteration = 0
##
##  Acceptance ratio for liability set 1 = 0.000356
##
##                         MCMC iteration = 1000
##
##  Acceptance ratio for liability set 1 = 0.123172
##
##                         MCMC iteration = 2000
##
##  Acceptance ratio for liability set 1 = 0.182665
##
##                         MCMC iteration = 3000
##
##  Acceptance ratio for liability set 1 = 0.213670
##
##                         MCMC iteration = 4000
##
##  Acceptance ratio for liability set 1 = 0.204317
##
##                         MCMC iteration = 5000
##
##  Acceptance ratio for liability set 1 = 0.205298
##
##                         MCMC iteration = 6000
##
##  Acceptance ratio for liability set 1 = 0.204836
##
##                         MCMC iteration = 7000
##
##  Acceptance ratio for liability set 1 = 0.205613
##
##                         MCMC iteration = 8000
##
##  Acceptance ratio for liability set 1 = 0.205839
##
##                         MCMC iteration = 9000
##
##  Acceptance ratio for liability set 1 = 0.205028
##
##                         MCMC iteration = 10000
##
##  Acceptance ratio for liability set 1 = 0.204169
##
##                         MCMC iteration = 11000
```

```
##
##  Acceptance ratio for liability set 1 = 0.204524
##
##                         MCMC iteration = 12000
##
##  Acceptance ratio for liability set 1 = 0.205407
##
##                         MCMC iteration = 13000
##
##  Acceptance ratio for liability set 1 = 0.205367
```

```
# Now run MCMCglmm using Kruger (2023) model formulation
mc_Kr <- MCMCglmm(count ~ zyear,
              random = ~us(1+zlatitude):site,
              rcov=~units,
              family="poisson",
              data = df,
              mev=NULL,start=NULL,
              # prior=NULL,
              prior=prior,
              nodes="ALL", scale=TRUE,
              nitt=13000,
              thin=10,
              burnin=3000,
              pr=T,
              pl=FALSE, verbose=FALSE, DIC=TRUE, singular.ok=FALSE, saveX=TRUE,
              saveZ=TRUE, saveXL=TRUE, slice=FALSE, ginverse=NULL, trunc=FALSE)

summary(mc2)
```

```
##
##  Iterations = 3001:12991
##  Thinning interval  = 10
##  Sample size  = 1000
##
##  DIC: 18576.25
##
##  G-structure:  ~us(1 + zyear):site
##
##                               post.mean  l-95% CI u-95% CI eff.samp
## (Intercept):(Intercept).site   1.89904  0.958886  3.07351   1000.0
## zyear:(Intercept).site         0.01337 -0.056470  0.07859    811.4
## (Intercept):zyear.site         0.01337 -0.056470  0.07859    811.4
## zyear:zyear.site               0.01174  0.006123  0.01945   1000.0
##
##  R-structure:  ~units
##
##       post.mean l-95% CI u-95% CI eff.samp
## units   0.00143 0.001302 0.001555     1000
##
##  Location effects: count ~ zyear * zlatitude
##
##                 post.mean  l-95% CI  u-95% CI eff.samp  pMCMC
## (Intercept)      8.323904  7.796944  8.825263     1000 <0.001 ***
```

```
## zyear           -0.586120 -0.627885 -0.546343     1000 <0.001 ***
## zlatitude        0.504145  0.002876  1.076979     1000  0.062 .
## zyear:zlatitude  0.424806  0.385956  0.466796     1118 <0.001 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(mc_Kr)
```

```
##
##  Iterations = 3001:12991
##  Thinning interval  = 10
##  Sample size  = 1000
##
##  DIC: 18962.15
##
##  G-structure:  ~us(1 + zlatitude):site
##
##                              post.mean l-95% CI u-95% CI eff.samp
## (Intercept):(Intercept).site    1.7152 0.785217    2.895    900.7
## zlatitude:(Intercept).site      0.9461 0.147499    1.811    437.1
## (Intercept):zlatitude.site      0.9461 0.147499    1.811    437.1
## zlatitude:zlatitude.site        0.6077 0.003168    1.332    312.3
##
##  R-structure:  ~units
##
##       post.mean l-95% CI u-95% CI eff.samp
## units    0.1949   0.1831   0.2088     1103
##
##  Location effects: count ~ zyear
##
##             post.mean l-95% CI u-95% CI eff.samp  pMCMC
## (Intercept)    8.2140   7.9128   8.5509     1033 <0.001 ***
## zyear         -0.5859  -0.6062  -0.5627     1000 <0.001 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# Compare the effective sample sizes between mc2 and mc_Kr
# The rvariance components of the Kruger model has MCMC sampling problems
coda::effectiveSize(mc2$VCV)
```

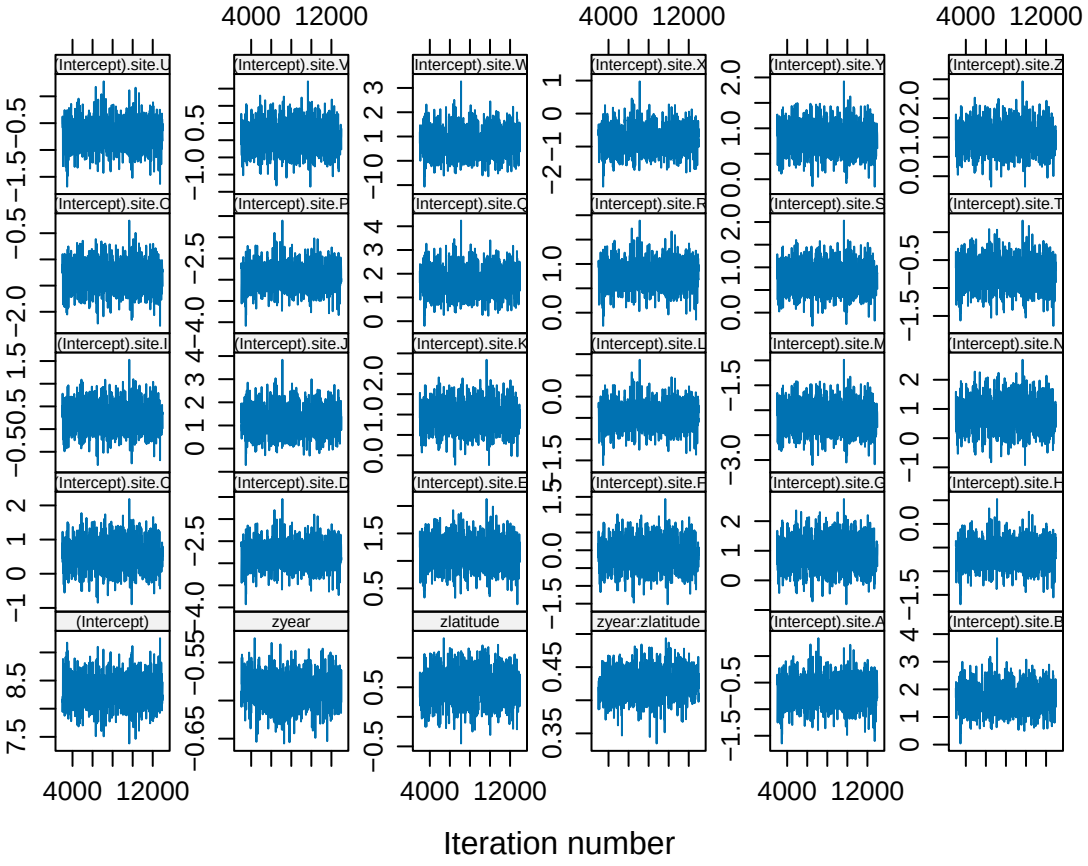
```
## (Intercept):(Intercept).site       zyear:(Intercept).site
##                    1000.0000                     811.4209
##       (Intercept):zyear.site             zyear:zyear.site
##                     811.4209                    1000.0000
##                        units
##                    1000.0000
```

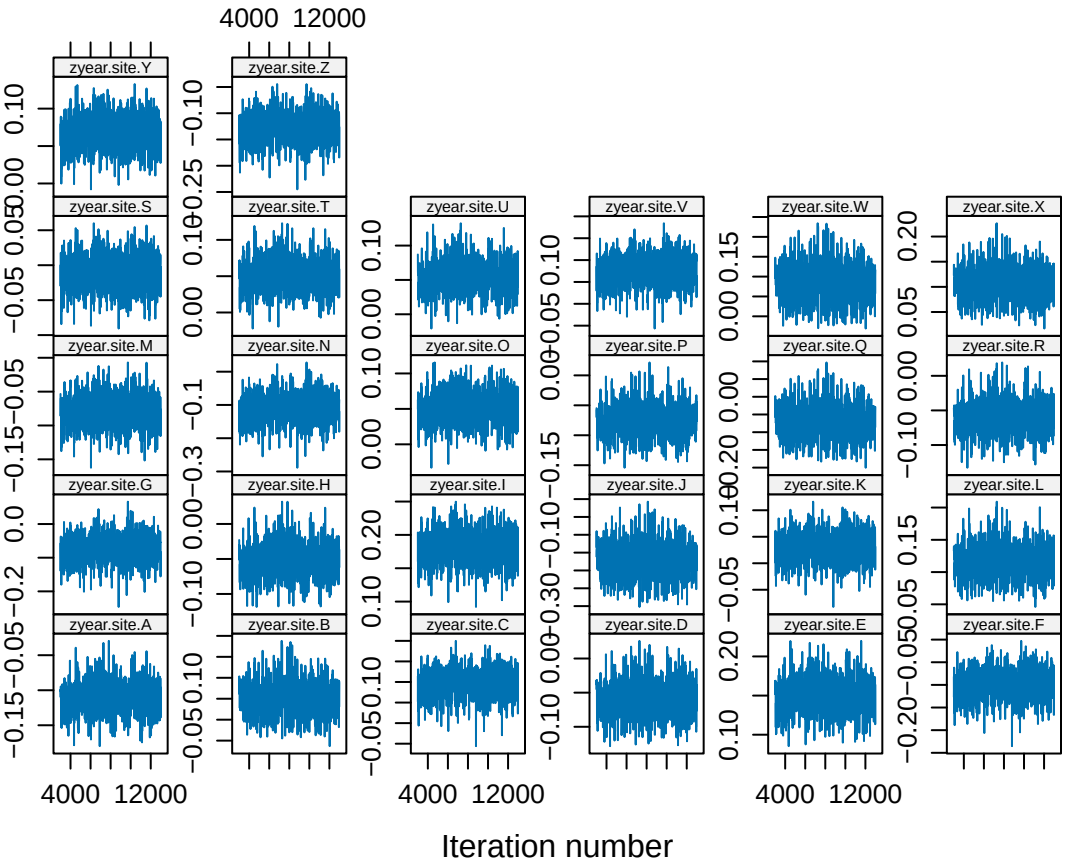
```
coda::effectiveSize(mc_Kr$VCV)
```

```
## (Intercept):(Intercept).site   zlatitude:(Intercept).site
##                     900.7301                     437.0647
##   (Intercept):zlatitude.site     zlatitude:zlatitude.site
```

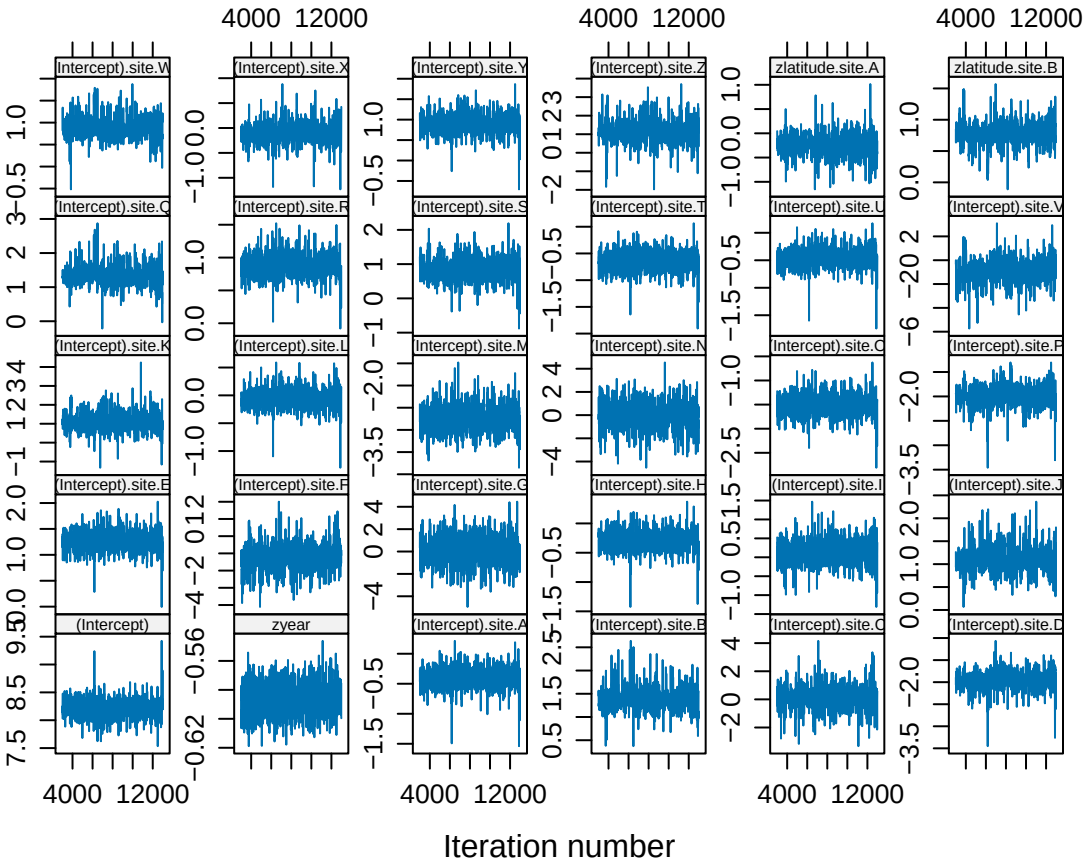
```
##                     437.0647                     312.2533
##                        units
##                    1102.9256
```

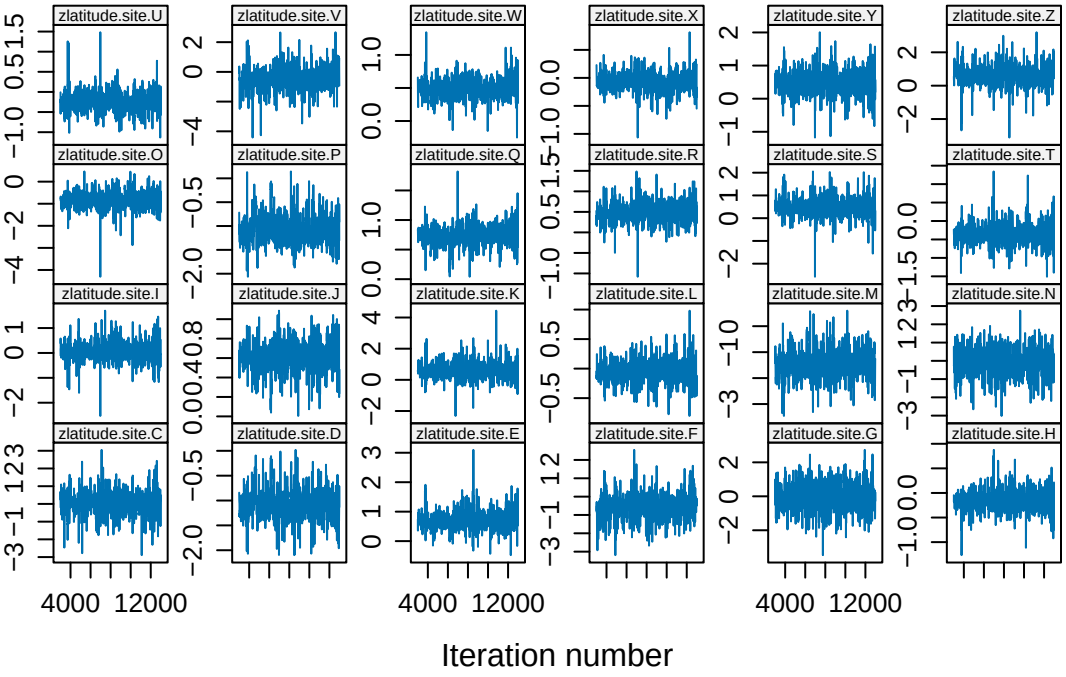
```
# check that the mcmc chain is mixing well - should be "white noise"
lattice::xyplot(as.mcmc(mc2$Sol), layout=c(6,5), par.strip.text=list(cex=0.5))
```



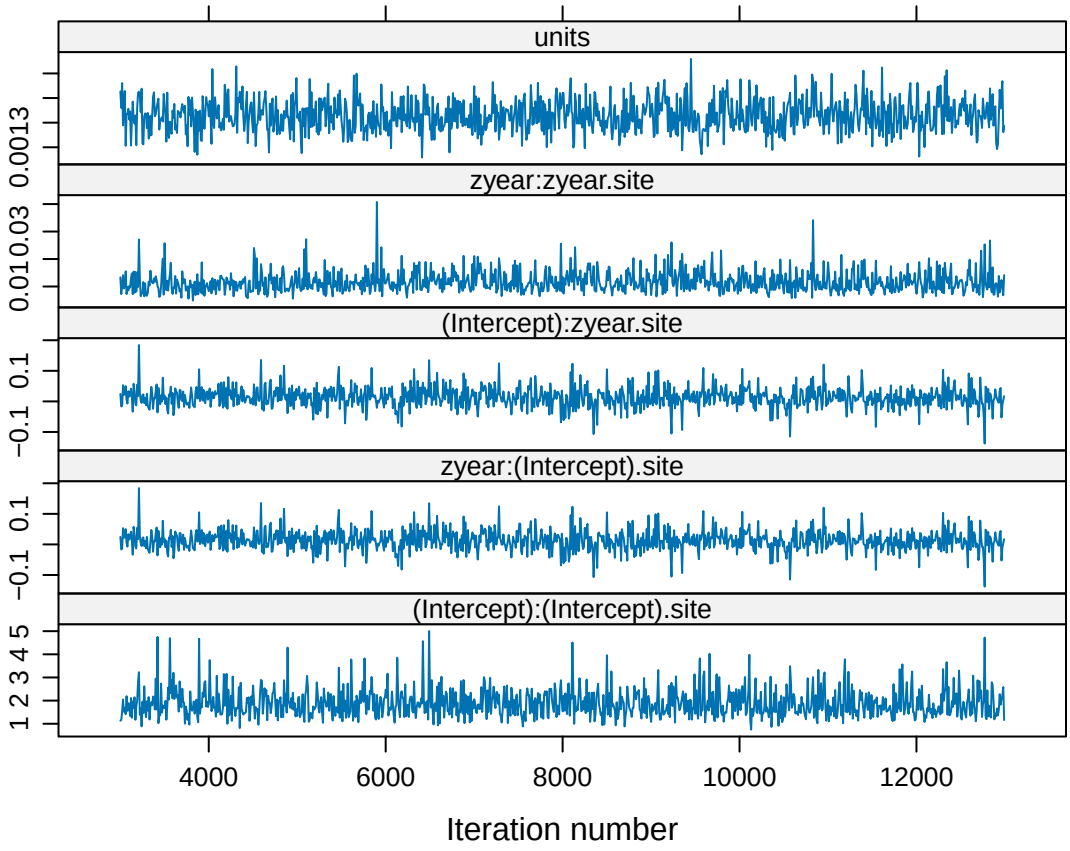



```
lattice::xyplot(as.mcmc(mc_Kr$Sol), layout=c(6,5), par.strip.text=list(cex=0.5))
```

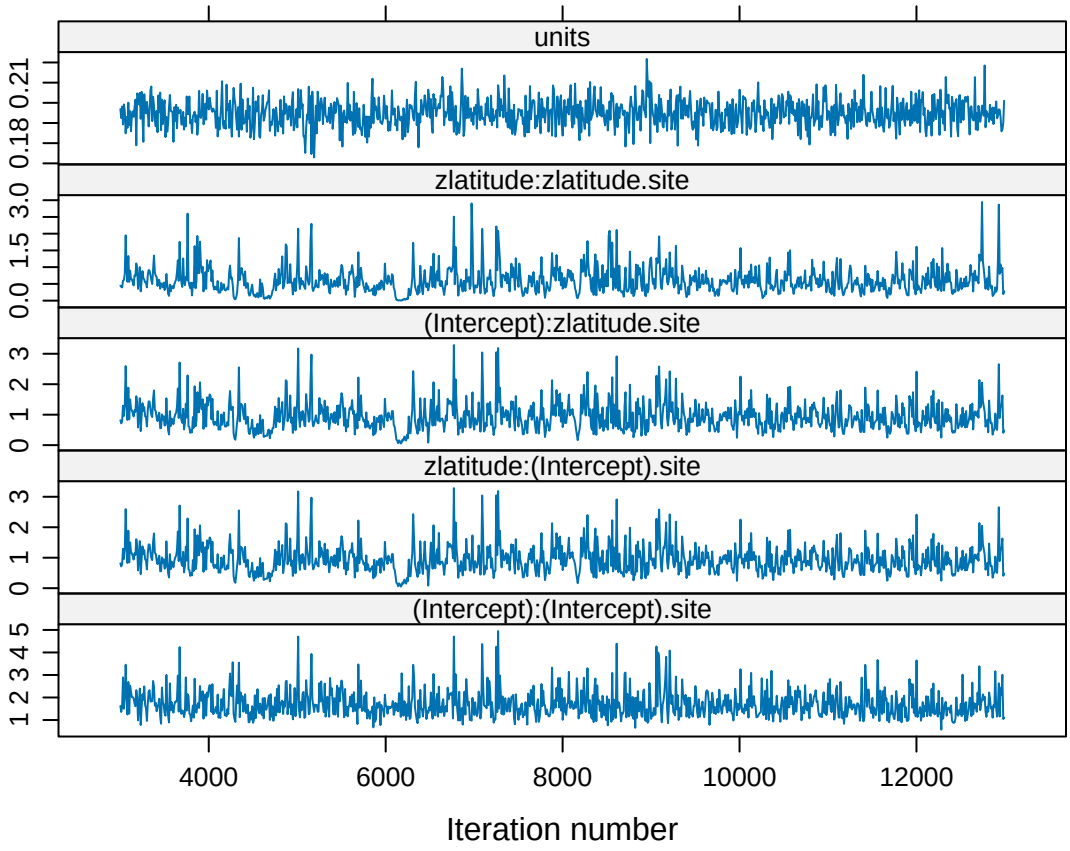

(Intercept).site.W
(Intercept).site.X
(Intercept).site.Y
(Intercept).site.Z
zlatitude.site.A
zlatitude.site.B
(Intercept).site.Q
(Intercept).site.R
(Intercept).site.S
(Intercept).site.T
(Intercept).site.U
(Intercept).site.V
(Intercept).site.K
(Intercept).site.L
(Intercept).site.M
(Intercept).site.N
(Intercept).site.O
(Intercept).site.P
(Intercept).site.E
(Intercept).site.F
(Intercept).site.G
(Intercept).site.H
(Intercept).site.I
(Intercept).site.J
(Intercept)
zyear
(Intercept).site.A
(Intercept).site.B
(Intercept).site.C
(Intercept).site.D
4000 12000
Iteration number




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




```
lattice::xyplot(as.mcmc(mc_Kr$VCV), par.strip.text=list(cex=0.8))
```

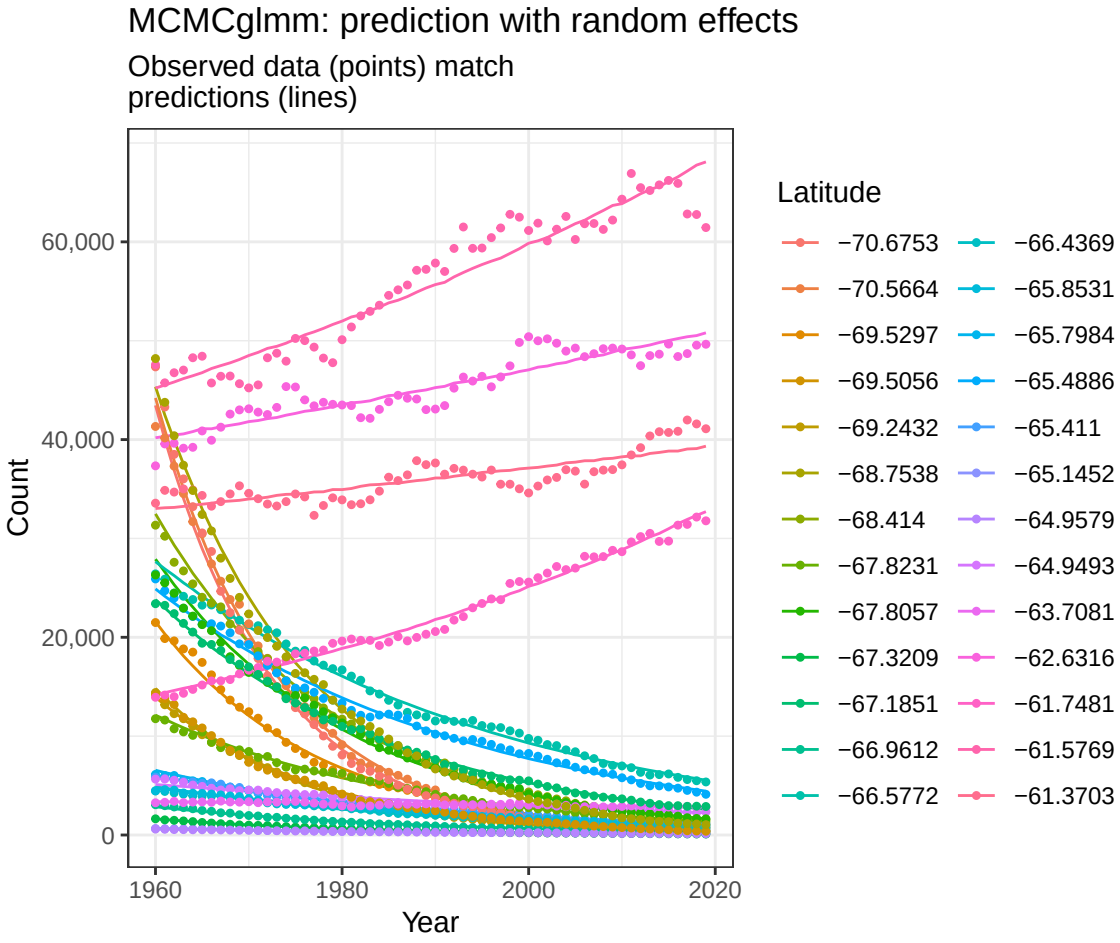


6 MCMCglmm predictions

6.1 MCMCglmm predictions with mc2

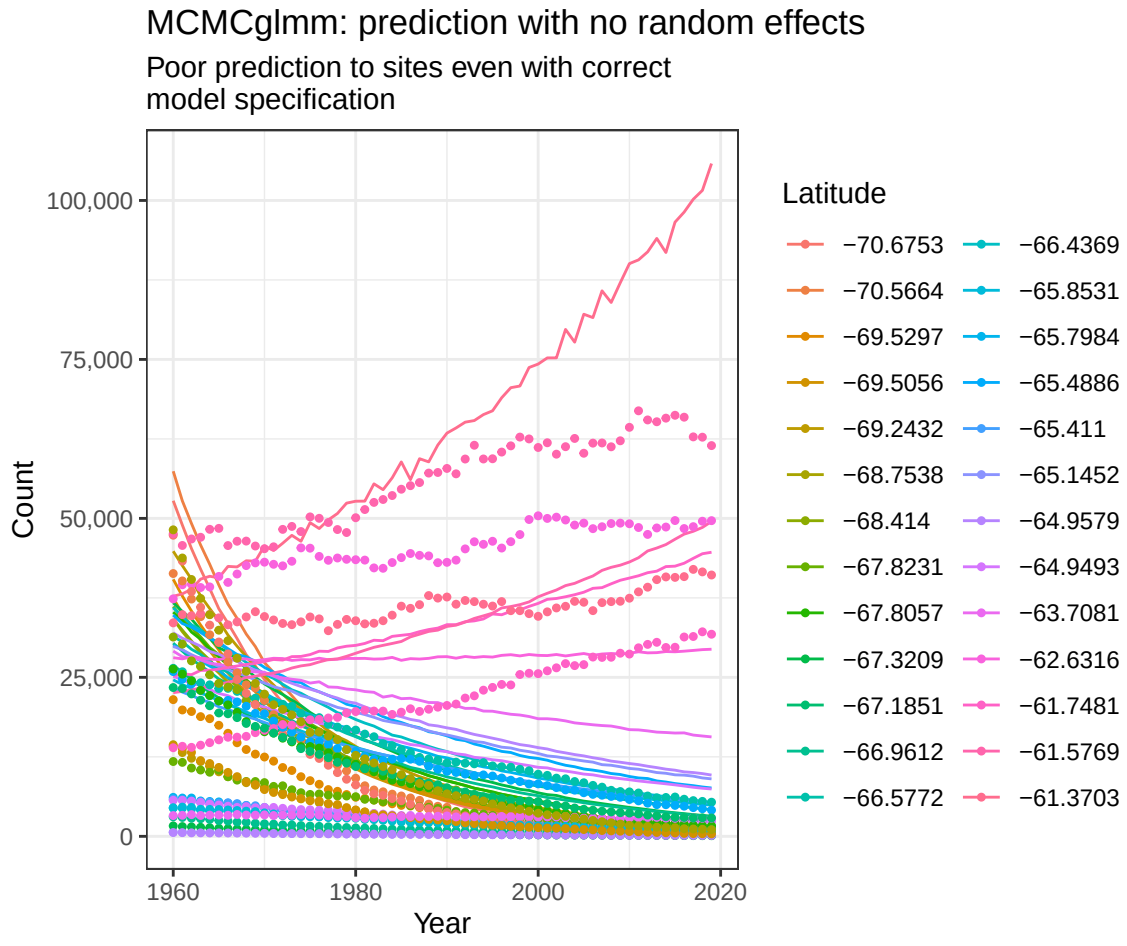
```
# https://www.rdocumentation.org/packages/MCMCglmm/versions/2.34/topics/predict.MCMCglmm
#predict(object, newdata=NULL, marginal=object$Random$formula,
#        type="response", interval="none", level=0.95, it=NULL,
#        posterior="all", verbose=FALSE, approx="numerical", ...)

# marginal = formula defining random effects to be maginalised.
# You don't want the random effects maginalised, so need to set this to NULL.

# First do a prediction with random effects:
pred <- data.frame(predict(mc2,
                      newdata=df,
                      type="response",
                      marginal=NULL,
                      interval="prediction",
                      posterior="all"))  # "all" is better (Kruger 2023 used 'mean')
```

```r
df$fit.mc2 <- pred$fit
df$lwr.mc2 <- pred$lwr
df$upr.mc2 <- pred$upr

# predict with observed data
fit.mc2p = ggplot(df, aes(x = year, y = fit.mc2, color = as.factor(latitude))) +
  geom_line() +  theme_bw() +
  labs(x = "Year", y = "Count") +
  scale_color_discrete(name = "Latitude") +
  scale_y_continuous(label = comma)+
  # add observed data
  geom_point(aes(x = year, y = count, color = as.factor(latitude)), size = 0.9)+
  labs(title = "MCMCglmm: prediction with random effects",
       subtitle = "Observed data (points) match\npredictions (lines)")

fit.mc2p
```



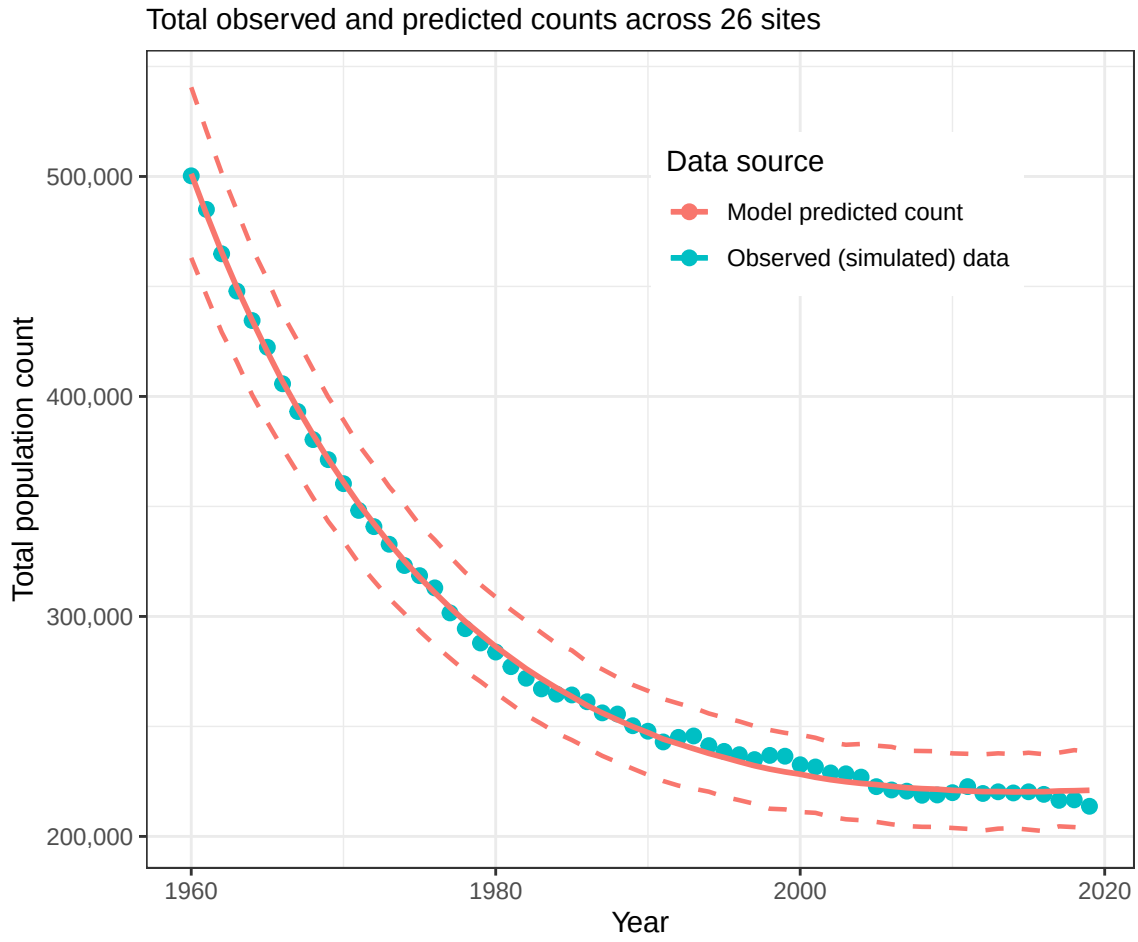

```r
# Kruger (2023) maginalised the random effects.
# Let's also do that to show what the impact is on this data set.
# Now predict with maginalised random effects (Kruger (2023)):
pred_margin <- data.frame(predict(mc2,
                          newdata=df,
```

```
                            type="response",
                            marginal=mc2$Random$formula,
                            interval="prediction",
                            posterior="all"))    # "all" is better (Kruger 2023 used 'mean')

df$fit.mc2_margin <- pred_margin$fit

ggplot(df, aes(x = year, y = fit.mc2_margin, color = as.factor(latitude))) +
  geom_line() +
  labs(x = "Year", y = "Count") +
  scale_color_discrete(name = "Latitude") +
  # add observed data
  geom_point(aes(x = year, y = count, color = as.factor(latitude)), size = 0.9)+
  theme_bw() +
   scale_y_continuous(label = comma)+
  labs(title = "MCMCglmm: prediction with no random effects",
       subtitle = "Poor prediction to sites even with correct\nmodel specification")
```



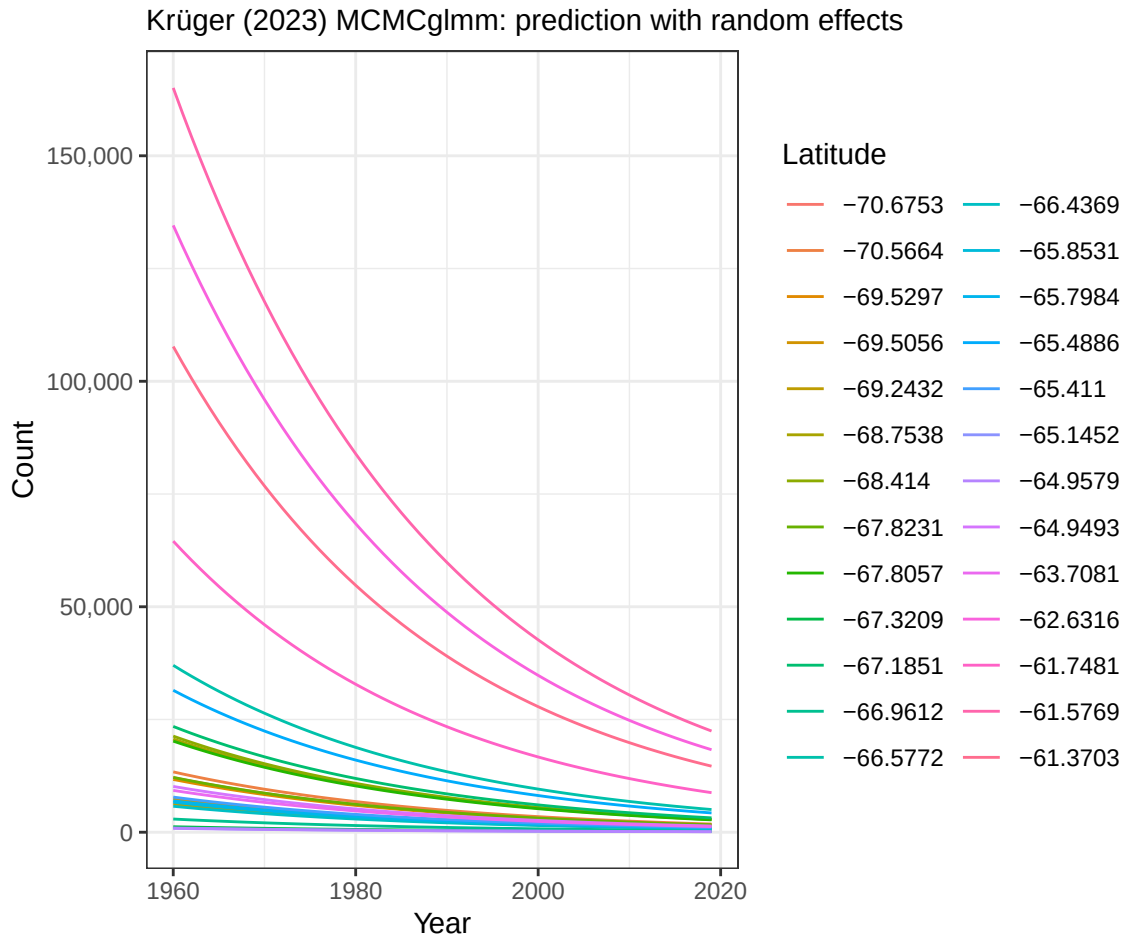

```
# calculate total population size predicted (total N across all sites)
pop_predict_mc2 = df %>%
             group_by(year) %>%
             summarise(total_pred = sum(fit.mc2),
```

```
                        min_pred = sum(lwr.mc2),
                        max_pred = sum(upr.mc2))

# Plot total N observed per year and the predicted total abundance per year
pop_predict_mc2p = ggplot() +
  geom_point(data = population, aes(x = year, y = totalN,
                color = "Observed (simulated) data"),   size = 2.3)+
  geom_line(data = pop_predict_mc2, aes(x = year, y = total_pred,
                color = "Model predicted count"), lty = 1, size = 1)+
    geom_line(data = pop_predict_mc2, aes(x = year, y = min_pred,
                color = "Model predicted count"), lty = 2, size = 0.8)+
    geom_line(data = pop_predict_mc2, aes(x = year, y = max_pred,
                color = "Model predicted count"), lty = 2, size = 0.8)+
  labs(x = "Year", y = "Total population count") +
  theme_bw()+
  scale_y_continuous(label = comma)+
  labs(subtitle = "Total observed and predicted counts across 26 sites")+
  guides(color=guide_legend(title="Data source"))+
  theme(legend.position = c(0.7, 0.8))

pop_predict_mc2p
```

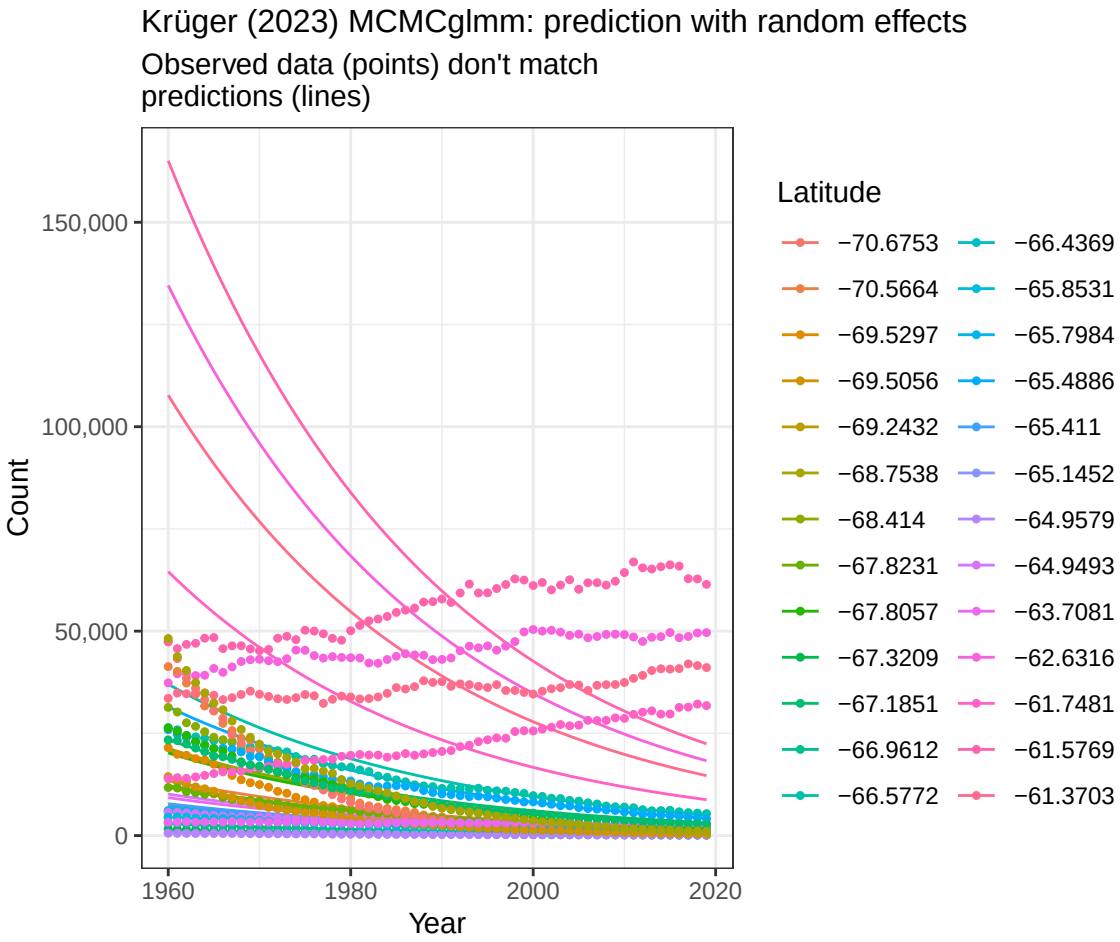


6.2 MCMCglmm predictions with Krüger (2023) model

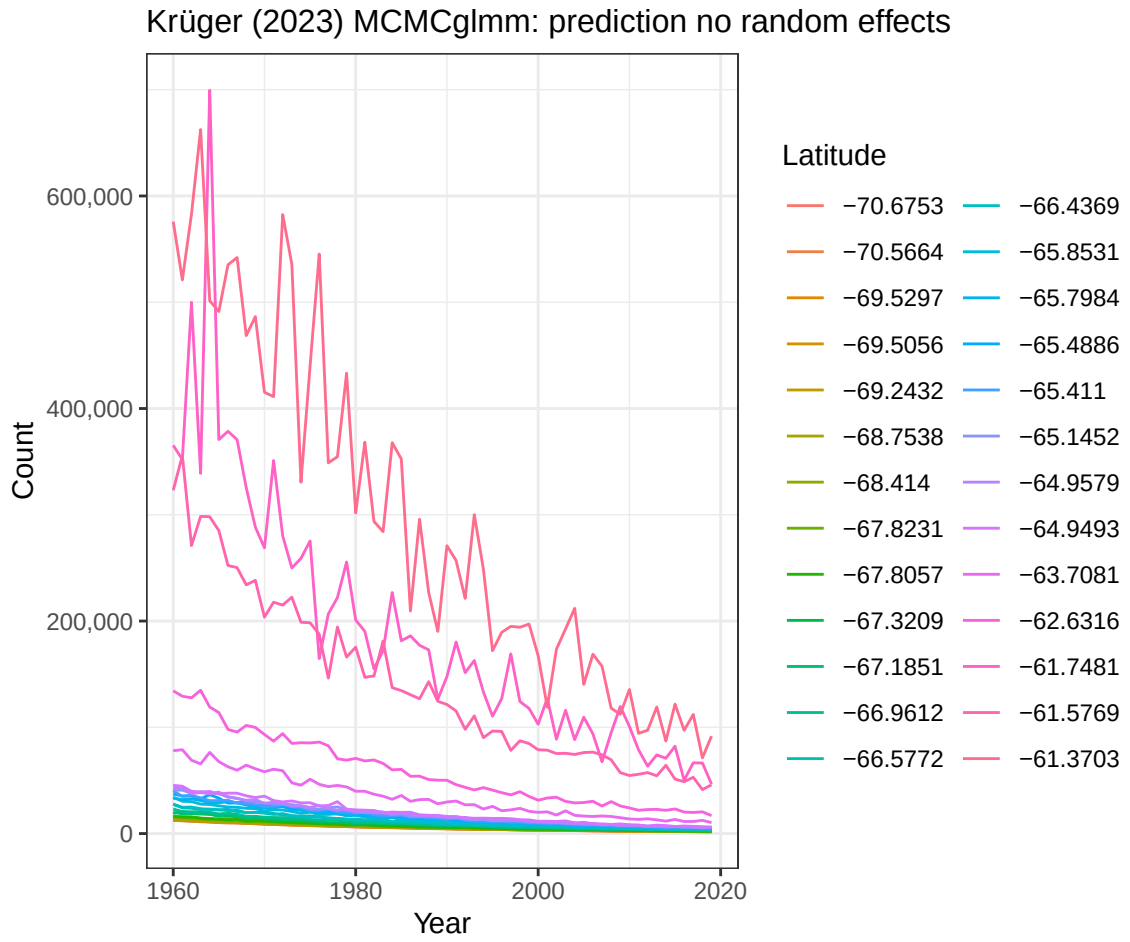
```
# Predict from Kruger (2023) MCMCglmm, using random effects
# and confidence (not prediction) intervals (for comparison with random intercept model)
pred_Kr <- data.frame(predict(mc_Kr,
                  newdata=df,
                  type="response",
                  marginal=NULL,
                  interval="confidence",
                  posterior="all"))  # "all" is better, but Kruger used 'mean'

df$fit.mc_Kr <- pred_Kr$fit

# Without observed data
ggplot(df, aes(x = year, y = fit.mc_Kr, color = as.factor(latitude))) +
  geom_line() +
  labs(x = "Year", y = "Count") +
  scale_color_discrete(name = "Latitude") +
  theme_bw() +
  scale_y_continuous(label = comma)+
  labs(subtitle = paste0("Kr",
       ds4psy::Umlaut["u"], "ger (2023) MCMCglmm: prediction with random effects"))+
  theme(plot.title = element_text(size = 12))
```



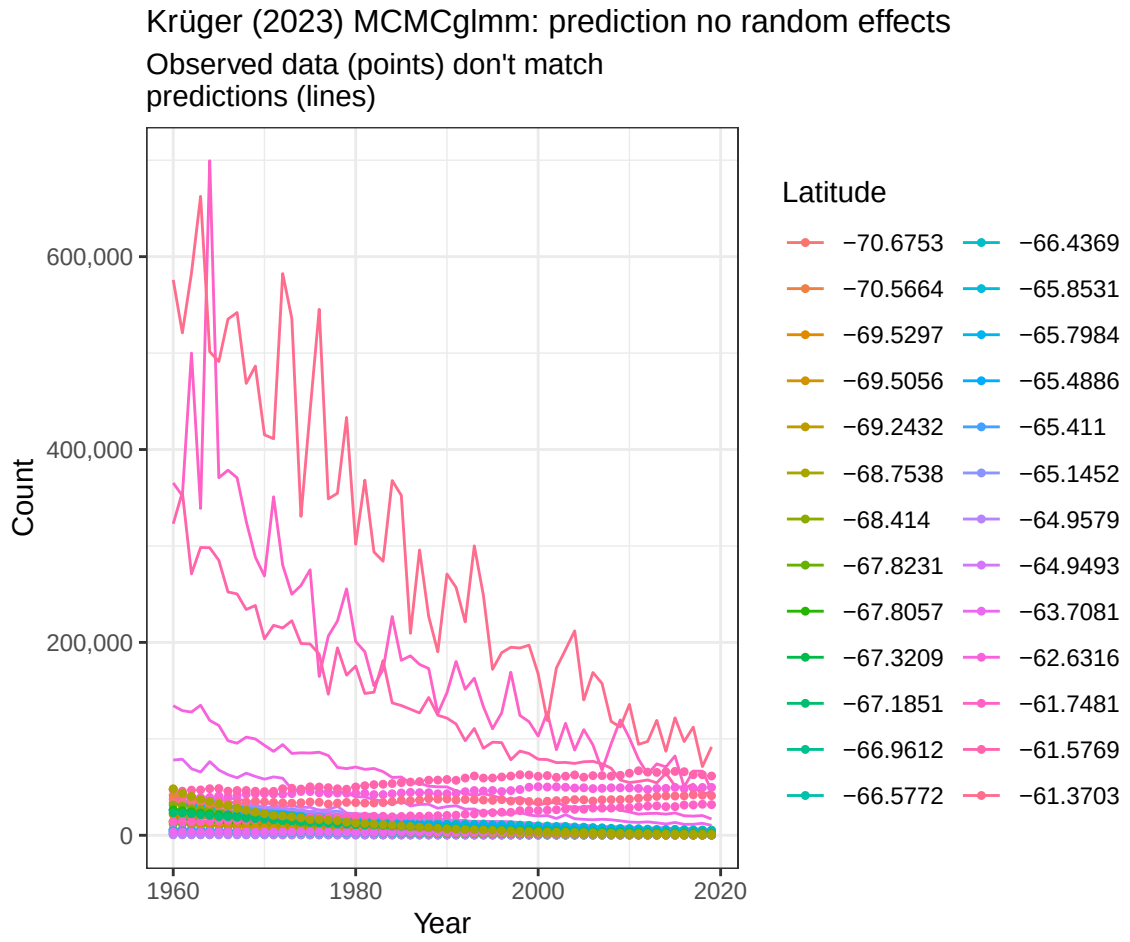

```
# Add observed data
ggplot(df, aes(x = year, y = fit.mc_Kr, color = as.factor(latitude))) +
  geom_line() +
  labs(x = "Year", y = "Count") +
  scale_color_discrete(name = "Latitude") +
  # add observed data
  geom_point(aes(x = year, y = count, color = as.factor(latitude)), size = 0.9)+
  theme_bw() +
  scale_y_continuous(label = comma)+
  labs(title = paste0("Kr",
       ds4psy::Umlaut["u"], "ger (2023) MCMCglmm: prediction with random effects"),
       subtitle = "Observed data (points) don't match\npredictions (lines)")+
  theme(plot.title = element_text(size = 12))
```




```r
# Now predict with maginalised random effects:
# this is the GLMM and prediction used by Kruger (2023)
pred_margin_Kr <- data.frame(predict(mc_Kr,
                             newdata=df,
                             type="response",
                             marginal=mc_Kr$Random$formula,
                             interval="prediction",
                             posterior="mean"))  # Kruger used mean, but "all" is better

df$fit.mc_Kr_margin <- pred_margin_Kr$fit
df$lwr.mc_Kr_margin <- pred_margin_Kr$lwr
df$upr.mc_Kr_margin <- pred_margin_Kr$upr

ggplot(df, aes(x = year, y = fit.mc_Kr_margin, color = as.factor(latitude))) +
  geom_line() +
  labs(x = "Year", y = "Count") +
  scale_color_discrete(name = "Latitude") +
  theme_bw() +
  scale_y_continuous(label = comma)+
   labs(title = paste0("Kr",
        ds4psy::Umlaut["u"], "ger (2023) MCMCglmm: prediction no random effects"))+
  theme(plot.title = element_text(size = 12))
```

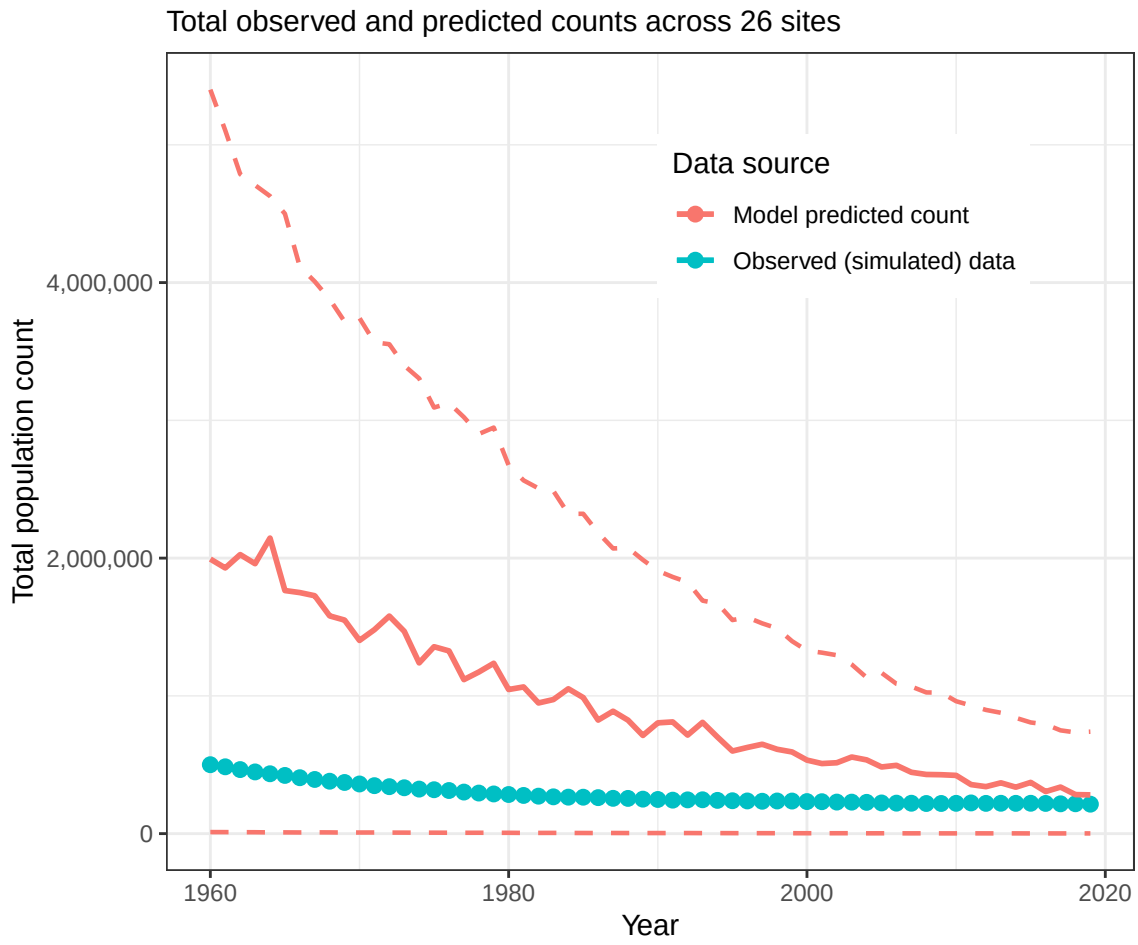



```
# add observed data
fit.mc_Kr_marginp =
ggplot(df, aes(x = year, y = fit.mc_Kr_margin, color = as.factor(latitude))) +
  geom_line() +
  labs(x = "Year", y = "Count") +
  scale_color_discrete(name = "Latitude") +
  # add observed data
  geom_point(aes(x = year, y = count, color = as.factor(latitude)), size = 0.9)+
  theme_bw() +
  scale_y_continuous(label = comma)+
  labs(title = paste0("Kr",
       ds4psy::Umlaut["u"], "ger (2023) MCMCglmm: prediction no random effects"),
       subtitle = "Observed data (points) don't match\npredictions (lines)")+
  theme(plot.title = element_text(size = 12))

fit.mc_Kr_marginp
```




```
# calculate total population size predicted (total N across all sites)
pop_predict_Kr = df %>%
             group_by(year) %>%
             summarise(total_pred = sum(fit.mc_Kr_margin),
                       min_pred = sum(lwr.mc_Kr_margin),
                       max_pred = sum(upr.mc_Kr_margin))

# Plot total N observed per year and the predicted total abundance per year
pop_predict_Krp = ggplot() +
  geom_point(data = population, aes(x = year, y = totalN,
             color = "Observed (simulated) data"), size = 2.3)+
  geom_line(data = pop_predict_Kr, aes(x = year, y = total_pred,
             color = "Model predicted count"), lty = 1, size = 1)+
     geom_line(data = pop_predict_Kr, aes(x = year, y = min_pred,
             color = "Model predicted count"), lty = 2, size = 0.8)+
    geom_line(data = pop_predict_Kr, aes(x = year, y = max_pred,
             color = "Model predicted count"), lty = 2, size = 0.8)+
  labs(x = "Year", y = "Total population count") +
  theme_bw()+
  scale_y_continuous(label = comma)+
  labs(subtitle = "Total observed and predicted counts across 26 sites")+
  guides(color=guide_legend(title="Data source"))+
  theme(legend.position = c(0.7, 0.8))
```

```
pop_predict_Krp
```



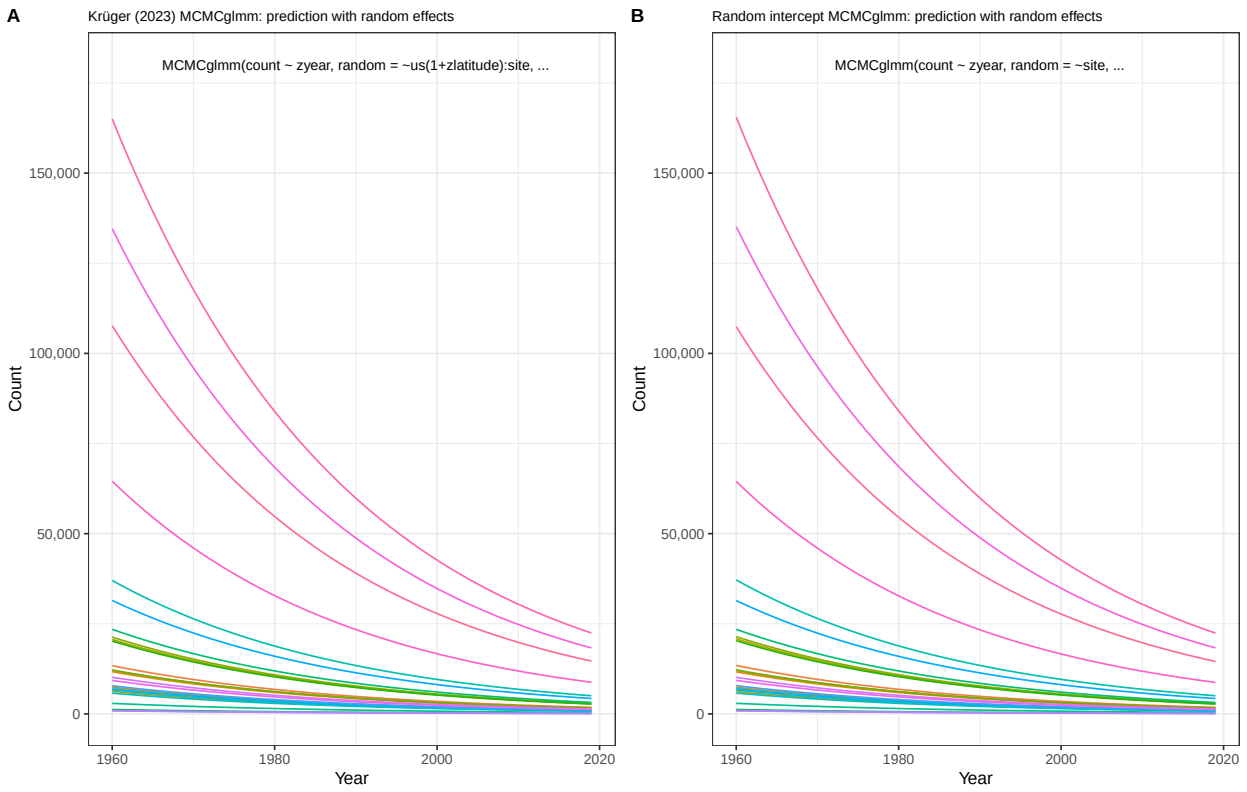
7 Intercept-only MCMCglmm model

Run intercept-only MCMCglmm model to show equivalency to Krüger (2023) model

```
# Now run MCMCglmm intercept only model, using the default prior
mc_I <- MCMCglmm(count ~ zyear,
                 random = ~site,
                 rcov=~units,
                 family="poisson",
                 data = df,
                 mev=NULL,start=NULL,
                 prior=NULL,
                 #prior=prior,
                 nodes="ALL", scale=TRUE,
                 nitt=13000,
                 thin=10,
                 burnin=3000,
                 pr=T,
```

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

```
# Predict from intercept model (include random site)
pred_I <- data.frame(predict(mc_I,
                             newdata=df,
                             type="response",
                             marginal=NULL,
                             interval="confidence",
                             posterior="all"))

df$fit.mc_I <- pred_I$fit
df$lwr.mc_I <- pred_I$lwr
df$upr.mc_I <- pred_I$upr

# figure (from above) with Lat as a random effect
lat_id = ggplot(df, aes(x = year, y = fit.mc_Kr, color = as.factor(latitude))) +
  geom_line() +
  labs(x = "Year", y = "Count") +
  scale_color_discrete(name = "Latitude") +
  # add observed data
  # geom_point(aes(x = year, y = count, color = as.factor(latitude)), size = 0.9)+
  theme_bw() +  theme(legend.position="none") +
  scale_y_continuous(label = comma)+
  labs(subtitle = paste0("Kr",
       ds4psy::Umlaut["u"], "ger (2023) MCMCglmm: prediction with random effects")) +
  annotate('text', x = 1990, y = 180000,
        label = "MCMCglmm(count ~ zyear, random = ~us(1+zlatitude):site, ...",
        size=3) +
  theme(plot.subtitle = element_text(size = 9))


# Intercept only model
interc.m = ggplot(df, aes(x = year, y = fit.mc_I, color = as.factor(latitude))) +
  geom_line() +
  labs(x = "Year", y = "Count") +
  scale_color_discrete(name = "Latitude") +
  scale_y_continuous(label = comma)+
  # add observed data
  # geom_point(aes(x = year, y = count, color = as.factor(latitude)), size = 0.9)+
  theme_bw() +  theme(legend.position="none") +
  labs(subtitle = "Random intercept MCMCglmm: prediction with random effects")+
  annotate('text', x = 1990, y = 180000,
        label = "MCMCglmm(count ~ zyear, random = ~site, ...",
        size=3) +
  theme(plot.subtitle = element_text(size = 9))

cowplot::plot_grid(lat_id, interc.m, labels = c('A', 'B'), ncol = 2, label_size = 12)
```




```
## Save Plot
 # pdf("./figures/Supp_Kruger_intercept model.pdf",
 #      useDingbats = FALSE, width = 9, height = 5)
 # cowplot::plot_grid(lat_id, interc.m, labels = c('A', 'B'), ncol = 2, label_size = 12)
 # dev.off()
```

The two models produce the same estimates for 'fit'.

8 Estimating population change from year x to year y

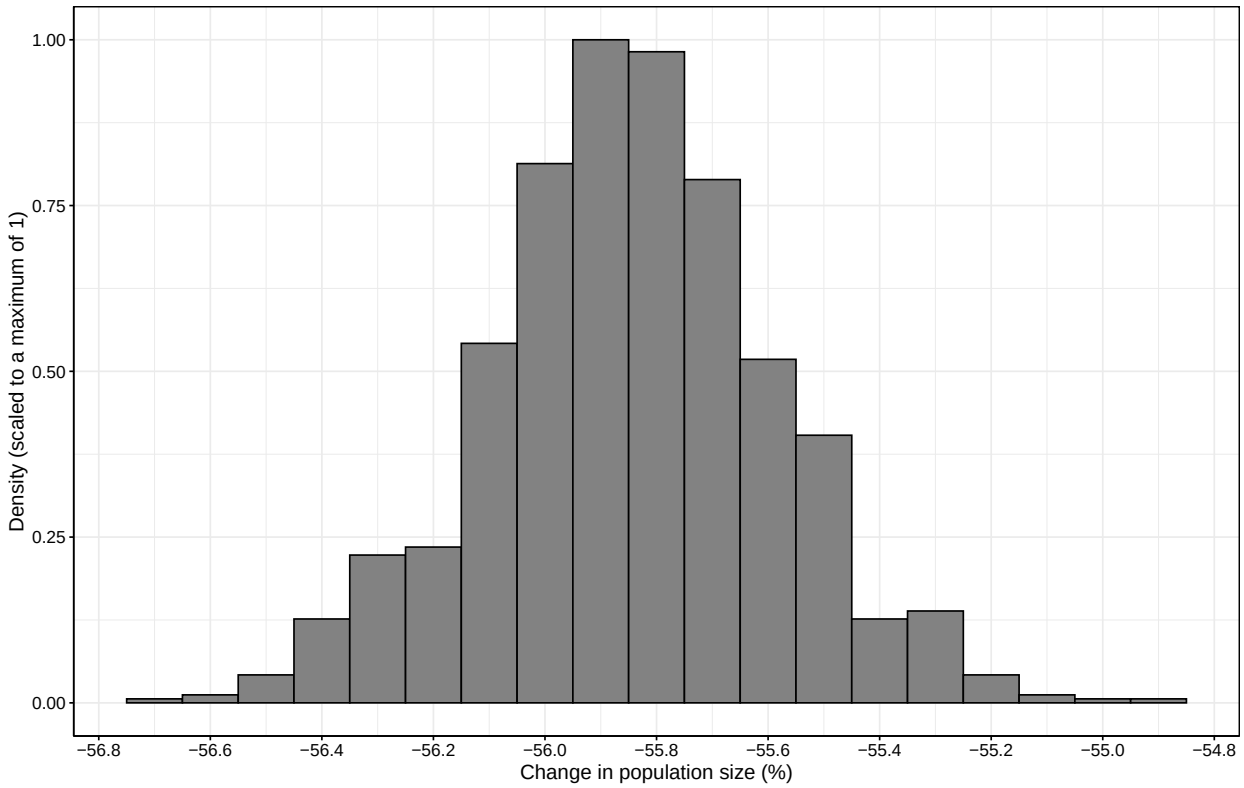
```
# extract posterior draws of fixed effects and random effects
posterior <- as.matrix(mc2$Sol)

# collect site-level information
site_and_lat <- df %>%
  as_tibble() %>%
  select(site, zlatitude) %>%
  distinct()

site_and_lat
```

```
## # A tibble: 26 x 2
##    site  zlatitude[,1]
##    <chr>         <dbl>
##  1 A             0.215
```

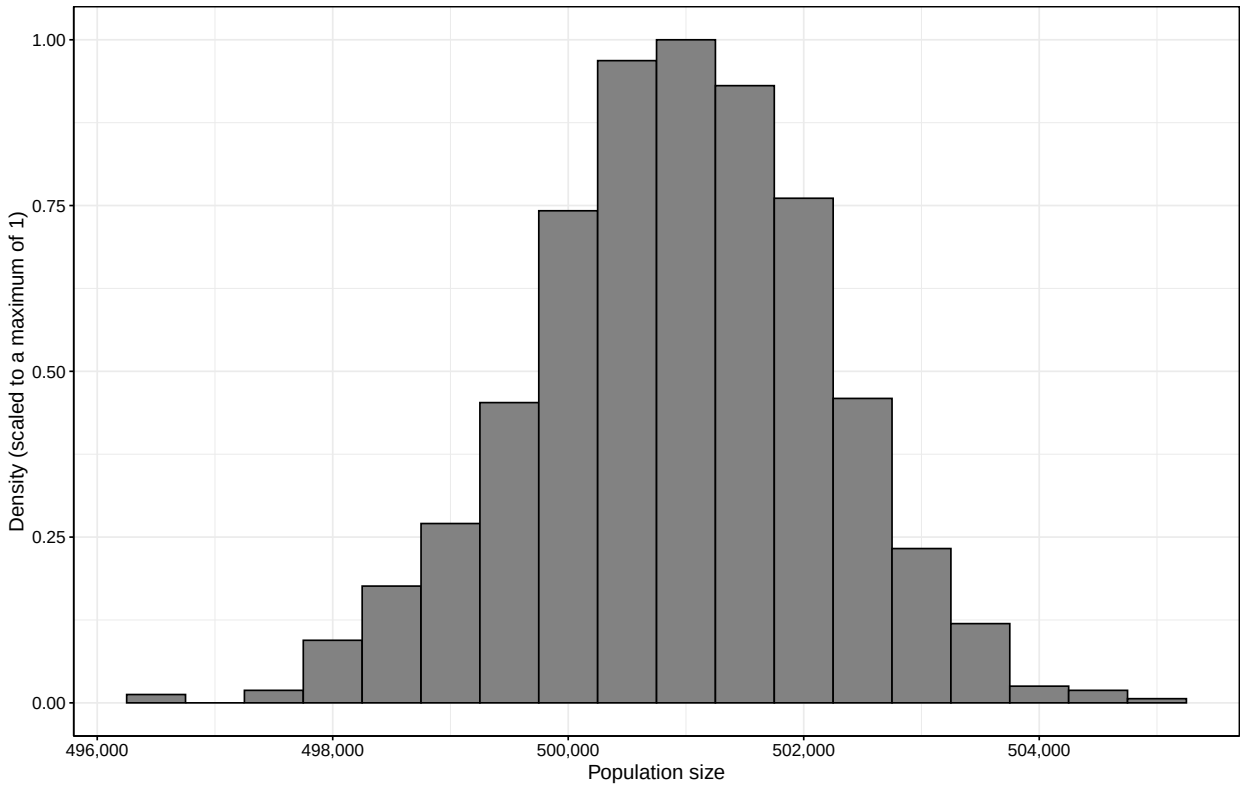
```
##  2 B            1.42
##  3 C           -1.21
##  4 D            0.465
##  5 E           -0.0818
##  6 F           -1.10
##  7 G           -1.60
##  8 H            0.363
##  9 I           -0.557
## 10 J            1.91
## # i 16 more rows
```

```r
# map years which to predict to (standardised scale)
# Here, the first year is 1960 and the last year is 2020 (30 year change)

year1 = 1960
year2 = 2020

first_year <- (year1 - mean(df$year)) / sd(df$year)
last_year <- (year2 - mean(df$year)) / sd(df$year)

# 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 (1960) 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 <- site_and_lat$site[j]
      zlatitude <- site_and_lat$zlatitude[j]

      # predict pop at site j in first year
      lin_pred <- theta["(Intercept)"] +
        theta["zyear"] * first_year +
        theta["zlatitude"] * zlatitude +
        theta["zyear:zlatitude"] * first_year * zlatitude
      if (use_random_effects) {
        lin_pred <- lin_pred +
          theta[ str_c("(Intercept).site.",site) ] +
          theta[ str_c("zyear.site.",site) ] * 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["zyear"] * last_year +
        theta["zlatitude"] * zlatitude +
        theta["zyear:zlatitude"] * last_year * zlatitude
```

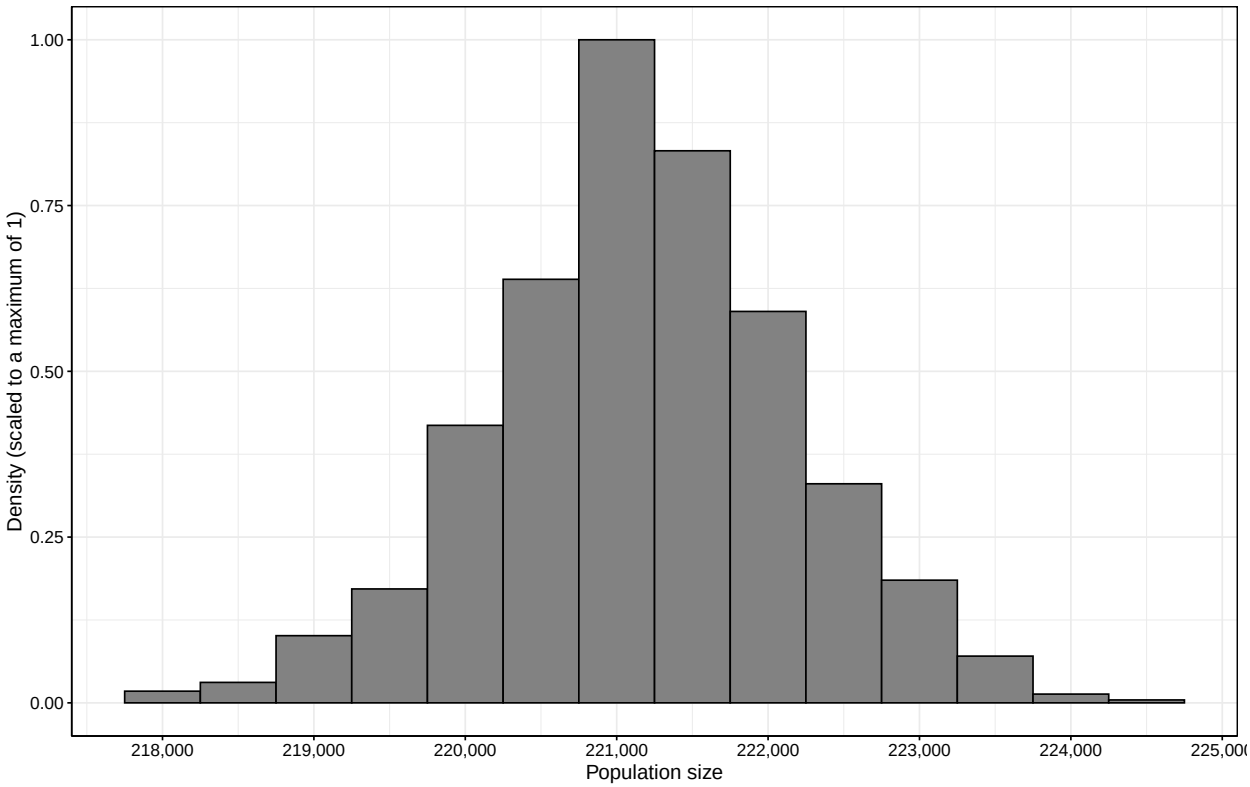
```
      if (use_random_effects) {
        lin_pred <- lin_pred +
          theta[ str_c("(Intercept).site.",site) ] +
          theta[ str_c("zyear.site.",site) ] * last_year
      }
      pred_pop_per_site.last[s,j] <- exp( lin_pred )
    }
  }

  # 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
}


# Make predictions with and without random effects
pred_re <- get_predictions(posterior, site_and_lat, first_year, last_year,
                           use_random_effects = TRUE)

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:
# ggplot(data = data.frame(pred_no_re$pop_change), aes(x = pred_no_re.pop_change,
#                                                      after_stat(ndensity)))+
#   geom_histogram(binwidth = 5, colour = "black", fill = "grey51")+
#   gg_theme()+
#   scale_x_continuous(n.breaks = 10)+
#   labs(y= "Density (scaled to a maximum of 1)",
#        x = "Change in population size (%)")


# Plot histogram of population change using random effects in prediction:
ggplot(data = data.frame(pred_re$pop_change), aes(x = pred_re.pop_change,
                                                  after_stat(ndensity)))+
  geom_histogram(binwidth = 0.1, colour = "black", fill = "grey51")+
  gg_theme()+
  scale_x_continuous(n.breaks = 10)+
  labs(y= "Density (scaled to a maximum of 1)",
       x = "Change in population size (%)")
```




```
# estimated population size in year1 (with random effects)
pred_first = as.data.frame(pred_re$pop.first)
names(pred_first) = "pred_first"
ggplot(data = data.frame(pred_re$pop.first), aes(x = pred_re.pop.first,
                                                 after_stat(ndensity)))+
  geom_histogram(binwidth = 500, colour = "black", fill = "grey51")+
  gg_theme()+
  scale_x_continuous(n.breaks = 7, label = comma)+
  labs(y= "Density (scaled to a maximum of 1)",
       x = "Population size")
```




```
# estimated population size in year2 (with random effects)
pred_last = as.data.frame(pred_re$pop.last)
names(pred_last) = "pred_last"
ggplot(data = data.frame(pred_re$pop.last), aes(x = pred_re.pop.last,
                                                after_stat(ndensity)))+
  geom_histogram(binwidth = 500, colour = "black", fill = "grey51")+
  gg_theme()+
  scale_x_continuous(n.breaks = 7, label = comma)+
  labs(y= "Density (scaled to a maximum of 1)",
       x = "Population size")
```

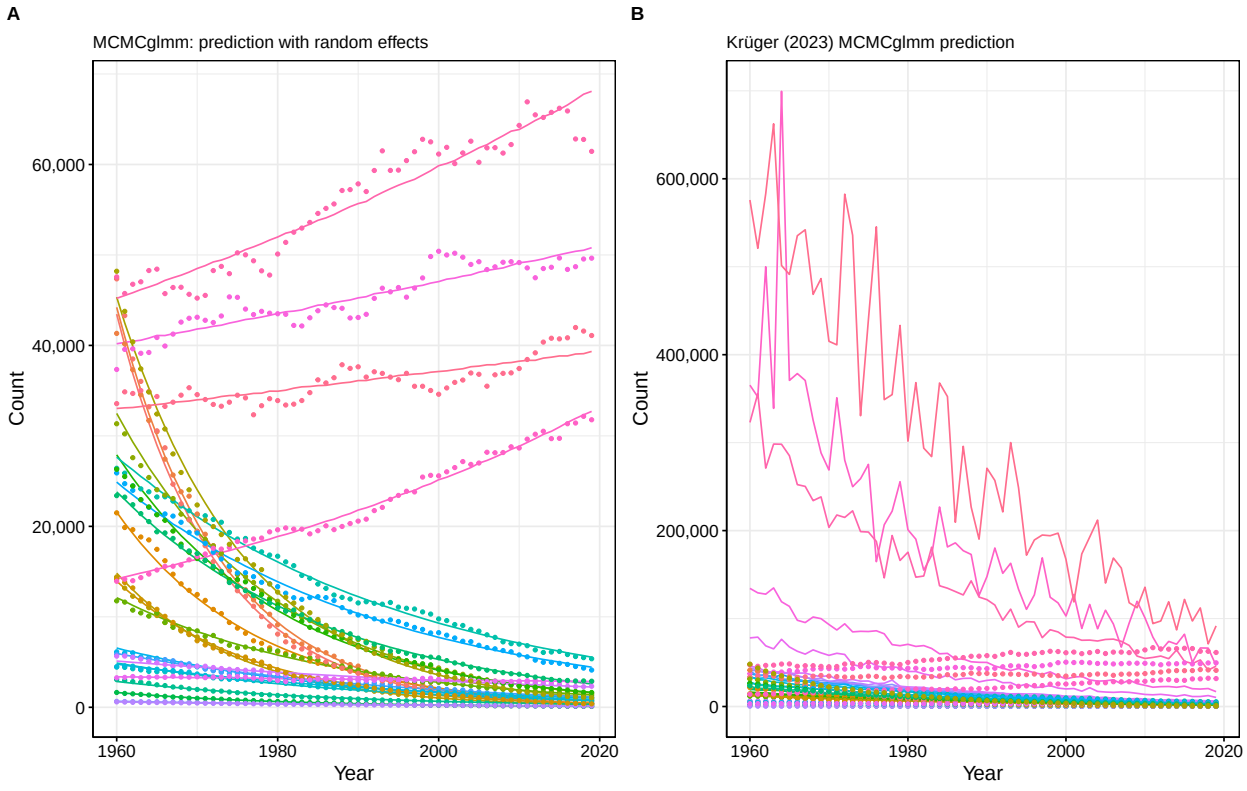



```
# # estimated population size in year1 (no random effects)
# pred_first_noRE = as.data.frame(pred_no_re$pop.first)
# names(pred_first_noRE) = "pred_first"
#
# # estimated population size in year2 (no random effects)
# pred_last_noRE = as.data.frame(pred_no_re$pop.last)
# names(pred_last_noRE) = "pred_last"
```

9 Manuscript figures

```
fit.mc2p =
fit.mc2p +
  gg_theme()+
  labs(title = "", subtitle = paste0("MCMCglmm: prediction with random effects"))+
  theme(legend.position="none")

fit.mc_Kr_marginp =
fit.mc_Kr_marginp +
  gg_theme()+
  labs(title = "", subtitle = paste0("Kr",
       ds4psy::Umlaut["u"], "ger (2023) MCMCglmm prediction"))+
  theme(legend.position="none")
```

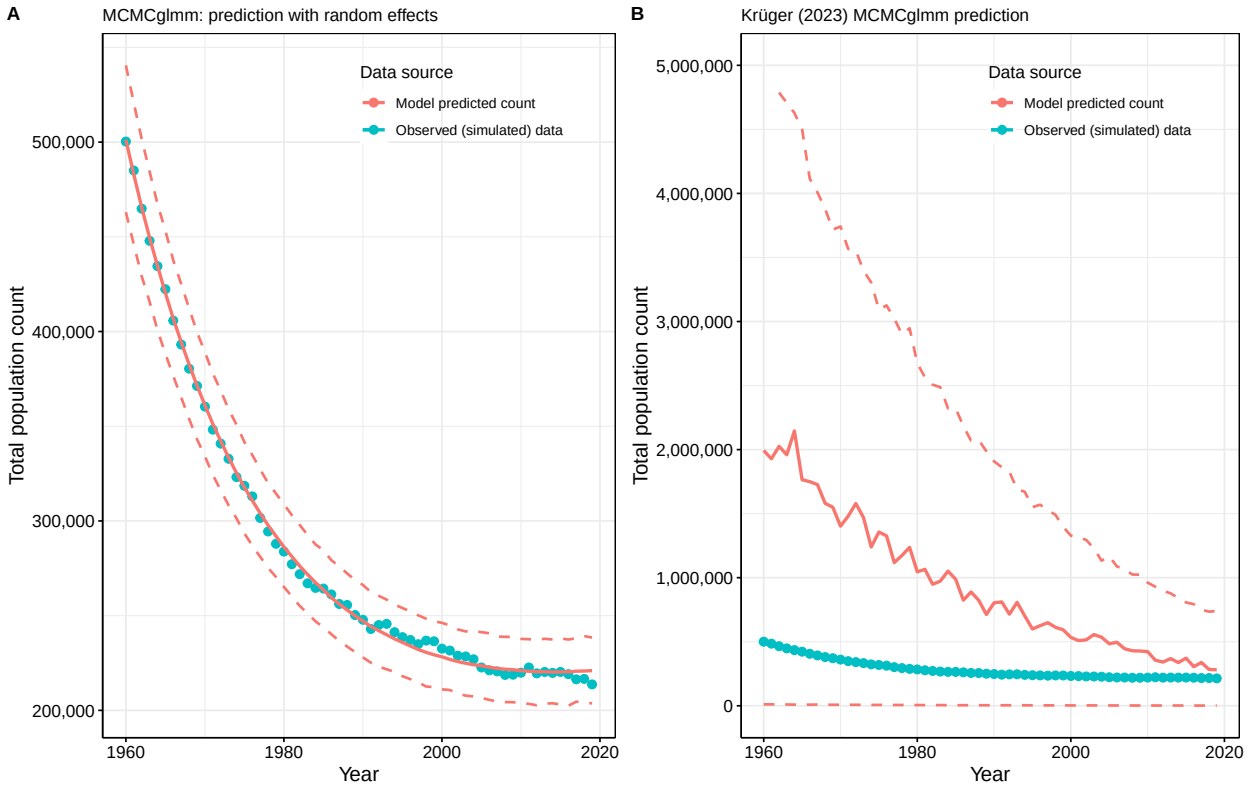
```
cowplot::plot_grid(fit.mc2p, fit.mc_Kr_marginp, labels = c('A', 'B'),
                   ncol = 2, label_size = 12)
```




```
pop_predict_mc2p =
pop_predict_mc2p  +
  gg_theme()+
  labs(subtitle = "MCMCglmm: prediction with random effects")+
  theme(legend.position = c(0.7, 0.9))
 # geom_point(data = pred_first_noRE, aes(x = year1, y = pred_first), col = "red") +
 # geom_point(data = pred_last_noRE, aes(x = year2, y = pred_last), col = "red")+
# geom_jitter(data = pred_first, aes(x = year1, y = pred_first),
#               col = "black",size = 1, height = 0, width = 0.5,
#                alpha = 0.1, stroke = NA) +
#   geom_jitter(data = pred_last, aes(x = year2, y = pred_last),
#               col = "black",size = 1, height = 0, width = 0.5,
#                alpha = 0.1, stroke = NA)


pop_predict_Krp =
pop_predict_Krp  +
  gg_theme()+
  labs(subtitle = paste0("Kr",ds4psy::Umlaut["u"], "ger (2023) MCMCglmm prediction "))+
  theme(legend.position = c(0.7, 0.9)) +
  scale_y_continuous(label = comma, limits = c(0, 5000000),
                     breaks = seq(0, 5000000, by = 1000000))
```

```
cowplot::plot_grid(pop_predict_mc2p, pop_predict_Krp, labels = c('A', 'B'),
                   ncol = 2, label_size = 12)
```

```
## Warning: Removed 2 rows containing missing values (`geom_line()`).
```




```
# cowplot::plot_grid(fit.mc2p, fit.mc_Kr_marginp,
#                    pop_predict_mc2p, pop_predict_Krp,
#                    labels = c('A', 'B', 'C', 'D'),
#                    ncol = 2, label_size = 12)


## Save Plot
# pdf("./figures/MS_Kruger_Oosthuizen models 4x4.pdf",
#     useDingbats = FALSE, width = 10, height = 9)
# cowplot::plot_grid(fit.mc_Kr_marginp, fit.mc2p,
#                    pop_predict_Krp, pop_predict_mc2p,
#                    labels = c('A', 'B', 'C', 'D'),
#                    ncol = 2, label_size = 10,
#                    vjust = 2, hjust = -1.5)
# dev.off()
```