

File S1. R codes for reproduction of analysis in the paper

Supplementary material for the paper " **Decreasing trends of Chinstrap Penguin breeding colonies on a region of major and ongoing rapid environmental changes suggest population level vulnerability**"

R code

```
# 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"))

# data from MAPPPD 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("AllCounts_V_4_0.csv")

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

```

summary(as.factor(chins$common_name))

## chinstrap penguin
##           1342

summary(as.factor(chins$count_type))

## adults chicks  nests
##      91    147   1104

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

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)) # some populations had
                                # multiple counts over the
                                # same season
                                # this summarises the count
                                #with the maximum nests

##### summarizing number of populations and number of counts

countsN<-ddply(nestM, c("site_id","Lat","Lon"), summarise,
  ncouns=length(nests),
  interval=(max(season_starting)-min(season_starting)))

countsN2<-ddply(subset(nestM,nests>0), c("site_id","Lat","Lon"), summarise,
  ncouns=length(nests),
  interval=(max(season_starting)-min(season_starting)))

summary(as.factor(countsN2$ncouns))
##   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

npops=length(countsN2$ncouns[countsN2$ncouns>1])
npops # number of populations

## [1] 133

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

poisson.mtest(nestM2$nests[nestM2$ncouns>1 & nestM2$nests>0],R=199)

##test for poisson distribution

##
## Poisson M-test

```

```

##
## data: nestM2$nested[nestM2$ncounts > 1 & nestM2$nested > 0] replicates: 199
## M-CvM = 158.43, p-value = 0.2211
## sample estimates:
## [1] 3006.691

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

### calculating population level slope

slopeN<-na.omit(data.frame((nestm3%>%
  group_by(site_id,Lat,Lon) %>%
  do({
    mod=glm(nested~season_starting,family="poisson",
            data=.)
    data.frame(Intercept=coef(mod)[1],
               Slope=coef(mod)[2])))))

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

length(sloN$Slope) # number of populations

## [1] 133

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

## [1] 83

```

```

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

## [1] 0.6240602

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

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

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

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

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

decr<-subset(Slope.Counts,Slope<0) # subset only decreasing populations

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%"
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

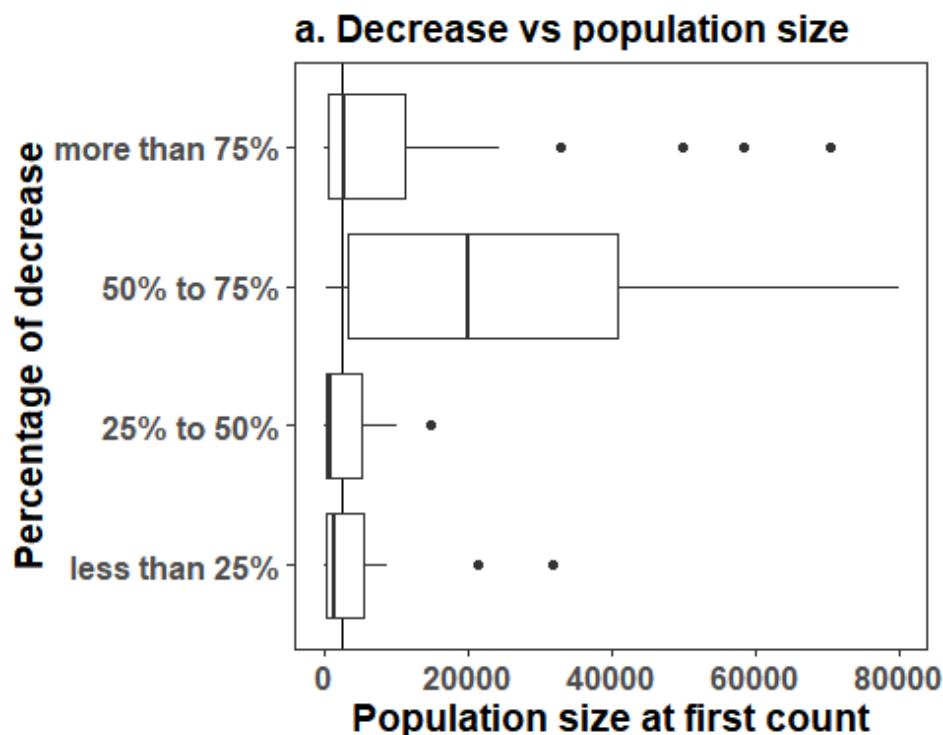
##          decrCat  N
## 1 less than 25% 19
## 2   25% to 50% 21
## 3   50% to 75%  5
## 4 more than 75% 38

sum(n$N) # check number of pops

## [1] 83

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

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") #figure 2
```



```
write.csv(decr,"decreasing_populations.csv") # for figure 1 b

nestM3<-nestm3

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))))

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, burn
in=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 = 0.000405
##
##           MCMC iteration = 1000
##
## Acceptance ratio for liability set 1 = 0.222841
##
##           MCMC iteration = 2000
```

```
##
## Acceptance ratio for liability set 1 = 0.288098
##
##           MCMC iteration = 3000
##
## Acceptance ratio for liability set 1 = 0.308042
##
##           MCMC iteration = 4000
##
## Acceptance ratio for liability set 1 = 0.318167
##
##           MCMC iteration = 5000
##
## Acceptance ratio for liability set 1 = 0.317816
##
##           MCMC iteration = 6000
##
## Acceptance ratio for liability set 1 = 0.318200
##
##           MCMC iteration = 7000
##
## Acceptance ratio for liability set 1 = 0.318353
##
##           MCMC iteration = 8000
##
## Acceptance ratio for liability set 1 = 0.317591
##
##           MCMC iteration = 9000
##
## Acceptance ratio for liability set 1 = 0.318545
##
##           MCMC iteration = 10000
##
## Acceptance ratio for liability set 1 = 0.318230
##
##           MCMC iteration = 11000
##
## Acceptance ratio for liability set 1 = 0.317054
##
##           MCMC iteration = 12000
##
## Acceptance ratio for liability set 1 = 0.318073
##
##           MCMC iteration = 13000
##
## Acceptance ratio for liability set 1 = 0.317737
summary(mc1)
```

```

##
## Iterations = 3001:12991
## Thinning interval = 10
## Sample size = 1000
##
## DIC: 4792.617
##
## G-structure: ~us(1 + Lat):site_id
##
##               post.mean    1-95% CI u-95% CI eff.samp
## (Intercept):(Intercept).site_id 1277.2062  0.0058523 4014.798    27.06
## Lat:(Intercept).site_id         20.9097 -0.0201395   64.956    26.47
## (Intercept):Lat.site_id         20.9097 -0.0201395   64.956    26.47
## Lat:Lat.site_id                 0.3434  0.0006516    1.052    25.91
##
## R-structure: ~units
##
##               post.mean 1-95% CI u-95% CI eff.samp
## units      0.2947  0.2522  0.344    887.4
##
## Location effects: nests ~ season_starting
##
##               post.mean 1-95% CI u-95% CI eff.samp pMCMC
## (Intercept)    27.335619 20.463759 34.547765    1000 <0.001 ***
## season_starting -0.010281 -0.013781 -0.006845    1000 <0.001 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

sol<-data.frame(mc1$Sol) # random effects

solm<-melt(sol,id.vars=c("X.Intercept.", "season_starting"))

solm$site_id<-substring(solm$variable,first=22,last=26)

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")

years<-data.frame(season_starting=c(1960:2020)) # construct an hypothetical dataframe
to generate the populations estimaters
pops<-data.frame(site_id=countsN2$site_id[countsN2$ncounts>1],Lat=countsN2$Lat[counts
N2$ncounts>1])

popy<-merge(pops,years)

```



```

poppy$nested<-c(0)    ### MCMCglmm needs a column with the response variable

popypred<-data.frame(predict(mc1,newdata=poppy,type="response",marginal=mc1$Random$for
mula,interval="prediction",
                        posterior="mean"))

poppy$fit<-popypred$fit

#figure 3

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

p1v2<-ggplot(poppy,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 predic
ted fit

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")

### 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

library(lubridate)
library(tidyr)
library(tidyquant)

library(dplyr)
library(broom)
library(purrr)
library(stringr)
library(knitr)
library(timetk)

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

```

```

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"))) ) # create a time stamp
for year

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

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

### 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.
## -0.3609 -0.2895 -0.2571 -0.2605 -0.2310 -0.1471

quantile(chm$value,probs=0.95)

##          95%
## -0.1932163

quantile(chm$value,probs=0.05)

##          5%
## -0.3381993

mean(chm$value)

## [1] -0.2605174

sd(chm$value)

## [1] 0.04364916

```

```

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")

```

p1v2/p2/p3

