Text S2. R Statistical Software Code for Analyzing Marine Mammal Population Abundance Trends (R 2.11.1 for Mac)

##### LOAD DATA

setwd #locates working directory

setwd("~/WORKING DIRECTORY NAME/ ") #sets working directory

##### RENAME DATA FOR USE IN FUNCTION

data0 <- read.csv("ABUNDANCE DAT FILE NAME.csv")

# load species .csv file with population abundance data; load separately for each species

##### SET UP DATAFRAME & REMOVE NA

data1 <- as.data.frame(data0) # create a data frame to manipulate columns

data1\_2 <- subset(data1, !is.na(data1$AbundMean))

# Or use PupCountMean instead of AbundMean if using pup count data

# remove values for which mean abundance value not available

data2 <- subset(data1\_2, !is.na(data1\_2$YearRead))

# remove values for which year value not available

# SPECIFY INFORMATION FOR EACH SPECIES

Species <- "gray\_whale\_scaled" # label -> character string name

# label data as pup count as needed to distinguish it from

# regular data

Pop\_int <- c(31000, 32000) # vector of populations of interest, indicated by Area\_ID

GenT <- 22.9 # generation time in years

Species\_com <- "Gray Whale" # species common name

Stype <- "Cetacean" # Cetacean, Pinniped or Other

##### SET UP AN EMPTY MATRIX TO STORE REGRESSION RESULTS

# AND THEN DELIVER RESULTS TO IT

# EACH TIME A SET OF REGRESSIONS IS PERFORMED FOR A POPULATION

# calls reg.rev1() command listed later

# run ten types of regression:

# (1) linear non-scaled (lm\_ns),

# (2) linear non-scaled weighted (lm\_ns\_w),

# (3) linear scaled (lm\_sds),

# (4) linear scaled weighted (lm\_sds\_w),

# (5) robust scaled (lmRob\_sds),

# (6) robust scaled weighted (lmRob\_sds\_w),

# (7) robust non-scaled (lmRob\_ns),

# (8) robust non-scaled weighted (lmRob\_ns\_w)

# (9) log linear (abundance) regression scaled non-weighted (log\_lm\_sds)

# (10) log linear (abundance) regression scaled weighted (log\_lm\_sds\_w)

# (11) robust log linear (abundance) regression scaled non-weighted (log\_lmRob\_sds)

# (12) robust log linear (abundance) regression scaled weighted (log\_lmRob\_sds\_w)

# 12 rows of results = 12 types of regression over one time period (tp3, i.e. 3 generations) for each population

library(MASS) # weighted regression command included in here

library(robust) # for lmRob() function

pick.pops <- function(data2, Species, Pop\_int, GenT)

{

results\_mat<-NULL

MinY <- c()

MaxY <- c()

Area\_Des <- c()

iter <- 0

for (i in Pop\_int){

iter <- iter + 1

data3 <- subset(data2, data2$AREAID==i) # organize by population Area ID

res1 <- reg.rev1(data3, Species, i, GenT, paste(Species,"\_",i,"\_",sep=""))

results\_mat <- rbind(results\_mat,res1)

# for each species, multiple population results are added in sequential order in output

# column labels for output data matrix

Spp\_Common\_Name <- rep(Species\_com, 12\*length(Pop\_int))

Species\_Type <- rep(Stype, 12\*length(Pop\_int))

Area\_ID <- rep(Pop\_int, each=12)

Area\_Des[iter] <- as.character(data3$AreaDes)

GenTime <- rep(GenT, 12\*length(Pop\_int))

print(as.numeric(tapply(data3$YearRead, data3$AREAID, min)))

MinY[iter] <- as.numeric(tapply(data3$YearRead, data3$AREAID, min))

MaxY[iter] <- as.numeric(tapply(data3$YearRead, data3$AREAID, max))

}

Area\_Des <- rep(Area\_Des, each=12)

MinYearPop <- rep(MinY, each=12)

MaxYearPop <- rep(MaxY, each=12)

TimePeriod\_Reg <- rep(rep(3, each=12), length(Pop\_int)) # repeat 3 (time period) 12 times (one for each regression kind) for each time period

RegType <- rep(c("lm\_ns", "lm\_ns\_w", "lm\_sds", "lm\_sds\_w", "lmRob\_sds", "lmRob\_sds\_w", "lmRob\_ns", "lmRob\_ns\_w", "log\_lm\_sds", "log\_lm\_sds\_w", "log\_lmRob\_sds", "log\_lmRob\_sds\_w"), (1\*length(Pop\_int)))

# Or if pup count data, use the following

# RegType <- rep(c("pc\_lm\_ns", "pc\_lm\_ns\_w", "pc\_lm\_sds", "pc\_lm\_sds\_w", "pc\_lmRob\_sds", "pc\_lmRob\_sds\_w","pc\_lmRob\_ns", "pc\_lmRob\_ns\_w", "pc\_log\_lm\_sds", "pc\_log\_lm\_sds\_w", "pc\_log\_lmRob\_sds", "pc\_log\_lmRob\_sds\_w"), (1\*length(Pop\_int)))

results <- data.frame(Spp\_Common\_Name, Species\_Type, Area\_ID, Area\_Des, GenTime, MinYearPop, MaxYearPop, TimePeriod\_Reg, RegType, results\_mat)

return(results)

}

###### FUNCTION FOR RUNNING REGRESSIONS reg.rev1

# calls function regressions4, which is specified later

# runs regressions over tp3 (3 generations)

# generates columns of NAs in matrix if data are insufficient to perform robust regression

# 2 types of x and y data

 # (1) scaled data, simple linear regression,

 # (2) raw x & y abundance values

# build data matrix (results mat) with 13 columns and 12 rows for each time period, and specify inputs for each column

# will be added on to previously specified columns for each population for each species

# coef[2,] gives 1st 4 values, coef[1,1] gives intercept, no values get NA

# last two columns give bias test pvals for Mest and LSest

# Rows:

# (1) linear non-scaled (lm\_ns),

# (2) linear non-scaled weighted (lm\_ns\_w),

# (3) linear scaled (lm\_sds),

# (4) linear scaled weighted (lm\_sds\_w),

# (5) robust scaled (lmRob\_sds),

# (6) robust scaled weighted (lmRob\_sds\_w),

# (7) robust non-scaled (lmRob\_ns),

# (8) robust non-scaled weighted (lmRob\_ns\_w)

# (9) log linear (abundance) regression scaled non-weighted (log\_lm\_sds)

# (10) log linear (abundance) regression scaled weighted (log\_lm\_sds\_w)

# (11) robust log linear (abundance) regression scaled non-weighted (log\_lmRob\_sds)

# (12) robust log linear (abundance) regression scaled weighted (log\_lmRob\_sds\_w)

reg.rev1 <- function(data3, Species, Pop\_int, GenT, pdf\_prefix)

{

 years <- data3$YearRead

 maxyear <- max(years)

 three\_gen <- maxyear - (3\*GenT)

 time\_per3 <- years >= three\_gen & years <= maxyear

 if ((sum(time\_per3) > 2) & (length(unique(years[time\_per3])) > 2))

# use PupCountMean instead of AbundMean, ACID\_w\_pc instead of ACID\_w, and PCAbundCID instead of AbundCID in this section if using pup count data

# produce scaled x and y, regular x and y, and ACID weight

{ y\_lm\_scale <- (data3$AbundMean[time\_per3] - mean(data3$AbundMean[time\_per3]))/(sqrt(var(data3$AbundMean[time\_per3])))

 y\_log <- log(data3$AbundMean[time\_per3])

 y\_log\_scale <- (y\_log - mean(y\_log))/(sqrt(var(y\_log)))

 x\_lm\_scale <- (data3$YearRead[time\_per3] - mean(data3$YearRead[time\_per3]))/(sqrt(var(data3$YearRead[time\_per3])))

 y\_ns <- data3$AbundMean[time\_per3]

 x\_ns <- data3$YearRead[time\_per3]

 ACID\_w <- 1/(data3$AbundLCID[time\_per3]^2)

 print(y\_lm\_scale)

 print(y\_log)

 print(y\_log\_scale)

 print(x\_lm\_scale)

 print(y\_ns)

 print(x\_ns)

 print(ACID\_w)

# RUN regressions

 reg\_ns <- lm(y\_ns ~ x\_ns)

 reg\_ns\_w <- lm(y\_ns ~ x\_ns, weights = ACID\_w)

 reg\_sds <- lm(y\_lm\_scale ~ x\_lm\_scale)

 reg\_sds\_w <- lm(y\_lm\_scale ~ x\_lm\_scale, weights = ACID\_w)

 rreg\_sds <- lmRob(y\_lm\_scale ~ x\_lm\_scale)

 rreg\_sds\_w <- lmRob(y\_lm\_scale ~ x\_lm\_scale, weights = ACID\_w)

 rreg\_ns <- lmRob(y\_ns ~ x\_ns)

 rreg\_ns\_w <- lmRob(y\_ns ~ x\_ns, weights = ACID\_w)

 log\_reg\_sds <- lm(y\_log\_scale ~ x\_lm\_scale)

 log\_reg\_sds\_w <- lm(y\_log\_scale ~ x\_lm\_scale, weights = ACID\_w)

 log\_rreg\_sds <- lmRob(y\_log\_scale ~ x\_lm\_scale)

 log\_rreg\_sds\_w <- lmRob(y\_log\_scale ~ x\_lm\_scale, weights = ACID\_w)

 results\_mat <- matrix(0,12,13)

# create results matrix with various different columns headings

dimnames(results\_mat) <- list(NULL, c("coeff\_est", "SE", "tval", "pval", "CI95\_Lower", "CI95\_Upper", "RSq", "RSq\_Adj", "sqrtRSq", "sqrtRSq\_Adj", "Intercept", "Mest\_biaspval", "LSest\_biaspval"))

results\_mat[1,] <- c(summary(reg\_ns)$coef[2,], ((summary(reg\_ns)$coef[2,1])-(1.96\*(summary(reg\_ns)$coef[2,2]))), ((summary(reg\_ns)$coef[2,1])+(1.96\*(summary(reg\_ns)$coef[2,2]))), summary(reg\_ns)$r.squared, summary(reg\_ns)$adj.r.squared, sqrt(abs(summary(reg\_ns)$r.squared)), sqrt(abs(summary(reg\_ns)$adj.r.squared)), summary(reg\_ns)$coef[1,1], NA, NA)

results\_mat[2,] <- c(summary(reg\_ns\_w)$coef[2,], ((summary(reg\_ns\_w)$coef[2,1])-(1.96\*(summary(reg\_ns\_w)$coef[2,2]))), ((summary(reg\_ns\_w)$coef[2,1])+(1.96\*(summary(reg\_ns\_w)$coef[2,2]))), summary(reg\_ns\_w)$r.squared, summary(reg\_ns\_w)$adj.r.squared, sqrt(abs(summary(reg\_ns\_w)$r.squared)), sqrt(abs(summary(reg\_ns\_w)$adj.r.squared)), summary(reg\_ns\_w)$coef[1,1], NA, NA)

results\_mat[3,] <- c(summary(reg\_sds)$coef[2,], ((summary(reg\_sds)$coef[2,1])-(1.96\*(summary(reg\_sds)$coef[2,2]))), ((summary(reg\_sds)$coef[2,1])+(1.96\*(summary(reg\_sds)$coef[2,2]))), summary(reg\_sds)$r.squared, summary(reg\_sds)$adj.r.squared, sqrt(abs(summary(reg\_sds)$r.squared)), sqrt(abs(summary(reg\_sds)$adj.r.squared)), summary(reg\_sds)$coef[1,1], NA, NA)

results\_mat[4,] <- c(summary(reg\_sds\_w)$coef[2,], ((summary(reg\_sds\_w)$coef[2,1])-(1.96\*(summary(reg\_sds\_w)$coef[2,2]))), ((summary(reg\_sds\_w)$coef[2,1])+(1.96\*(summary(reg\_sds\_w)$coef[2,2]))), summary(reg\_sds\_w)$r.squared, summary(reg\_sds\_w)$adj.r.squared, sqrt(abs(summary(reg\_sds\_w)$r.squared)), sqrt(abs(summary(reg\_sds\_w)$adj.r.squared)), summary(reg\_sds\_w)$coef[1,1], NA, NA)

results\_mat[5,] <- c(summary(rreg\_sds)$coef[2,], ((summary(rreg\_sds)$coef[2,1])-(1.96\*(summary(rreg\_sds)$coef[2,2]))), ((summary(rreg\_sds)$coef[2,1])+(1.96\*(summary(rreg\_sds)$coef[2,2]))), summary(rreg\_sds)$r.squared, NA, sqrt(abs(summary(rreg\_sds)$r.squared)), NA, summary(rreg\_sds)$coef[1,1], summary(rreg\_sds)$biasTest[1,2], summary(rreg\_sds)$biasTest[2,2])

results\_mat[6,] <- c(summary(rreg\_sds\_w)$coef[2,], ((summary(rreg\_sds\_w)$coef[2,1])-(1.96\*(summary(rreg\_sds\_w)$coef[2,2]))), ((summary(rreg\_sds\_w)$coef[2,1])+(1.96\*(summary(rreg\_sds\_w)$coef[2,2]))), summary(rreg\_sds\_w)$r.squared, NA, sqrt(abs(summary(rreg\_sds\_w)$r.squared)), NA, summary(rreg\_sds\_w)$coef[1,1], summary(rreg\_sds\_w)$biasTest[1,2], summary(rreg\_sds\_w)$biasTest[2,2])

results\_mat[7,] <- c(summary(rreg\_ns)$coef[2,], ((summary(rreg\_ns)$coef[2,1])-(1.96\*(summary(rreg\_ns)$coef[2,2]))), ((summary(rreg\_ns)$coef[2,1])+(1.96\*(summary(rreg\_ns)$coef[2,2]))), summary(rreg\_ns)$r.squared, NA, sqrt(abs(summary(rreg\_ns)$r.squared)), NA, summary(rreg\_ns)$coef[1,1], summary(rreg\_ns)$biasTest[1,2], summary(rreg\_ns)$biasTest[2,2])

results\_mat[8,] <- c(summary(rreg\_ns\_w)$coef[2,], ((summary(rreg\_ns\_w)$coef[2,1])-(1.96\*(summary(rreg\_ns\_w)$coef[2,2]))), ((summary(rreg\_ns\_w)$coef[2,1])+(1.96\*(summary(rreg\_ns\_w)$coef[2,2]))), summary(rreg\_ns\_w)$r.squared, NA, sqrt(abs(summary(rreg\_ns\_w)$r.squared)), NA, summary(rreg\_ns\_w)$coef[1,1], summary(rreg\_ns\_w)$biasTest[1,2], summary(rreg\_ns\_w)$biasTest[2,2])

results\_mat[9,] <- c(summary(log\_reg\_sds)$coef[2,], ((summary(log\_reg\_sds)$coef[2,1])-(1.96\*(summary(log\_reg\_sds)$coef[2,2]))), ((summary(log\_reg\_sds)$coef[2,1])+(1.96\*(summary(log\_reg\_sds)$coef[2,2]))), summary(log\_reg\_sds)$r.squared, summary(log\_reg\_sds)$adj.r.squared, sqrt(abs(summary(log\_reg\_sds)$r.squared)), sqrt(abs(summary(log\_reg\_sds)$adj.r.squared)), summary(log\_reg\_sds)$coef[1,1], NA, NA)

results\_mat[10,] <- c(summary(log\_reg\_sds\_w)$coef[2,], ((summary(log\_reg\_sds\_w)$coef[2,1])-(1.96\*(summary(log\_reg\_sds\_w)$coef[2,2]))), ((summary(log\_reg\_sds\_w)$coef[2,1])+(1.96\*(summary(log\_reg\_sds\_w)$coef[2,2]))), summary(log\_reg\_sds\_w)$r.squared, summary(log\_reg\_sds\_w)$adj.r.squared, sqrt(abs(summary(log\_reg\_sds\_w)$r.squared)), sqrt(abs(summary(log\_reg\_sds\_w)$adj.r.squared)), summary(log\_reg\_sds\_w)$coef[1,1], NA, NA)

results\_mat[11,] <- c(summary(log\_rreg\_sds)$coef[2,], ((summary(log\_rreg\_sds)$coef[2,1])-(1.96\*(summary(log\_rreg\_sds)$coef[2,2]))), ((summary(log\_rreg\_sds)$coef[2,1])+(1.96\*(summary(log\_rreg\_sds)$coef[2,2]))), summary(log\_rreg\_sds)$r.squared, NA, sqrt(abs(summary(log\_rreg\_sds)$r.squared)), NA, summary(log\_rreg\_sds)$coef[1,1], summary(log\_rreg\_sds)$biasTest[1,2], summary(log\_rreg\_sds)$biasTest[2,2])

results\_mat[12,] <- c(summary(log\_rreg\_sds\_w)$coef[2,], ((summary(log\_rreg\_sds\_w)$coef[2,1])-(1.96\*(summary(log\_rreg\_sds\_w)$coef[2,2]))), ((summary(log\_rreg\_sds\_w)$coef[2,1])+(1.96\*(summary(log\_rreg\_sds\_w)$coef[2,2]))), summary(log\_rreg\_sds\_w)$r.squared, NA, sqrt(abs(summary(log\_rreg\_sds\_w)$r.squared)), NA, summary(log\_rreg\_sds\_w)$coef[1,1], summary(log\_rreg\_sds\_w)$biasTest[1,2], summary(log\_rreg\_sds\_w)$biasTest[2,2])

return(results\_mat)

}

 else {tp3\_res <- matrix(NA,12,13)}

 }

##### RUN ALL FUNCTIONS

spp\_regdata <- pick.pops(data2, Species, Pop\_int, GenT)

write.csv(spp\_regdata, file=paste(Species,"\_","table12regtp3",".csv", sep="")) # save matrix to a .csv

##### END