The following supplement accompanies the article

Age and growth of olive ridley sea turtles in the main Brazilian nesting ground

Roberta Petitet*, Larisa Avens, Jaqueline C. Castilhos, Paul G. Kinas, Leandro Bugoni

*Corresponding author: rpetitet@hotmail.com

Marine Ecology Progress Series 541: 205–218 (2015)

SUPPLEMENT. MATERIALS AND METHODS

STATISTICAL ANALYSIS

All analyses were performed with R software (R Core Team 2014), and JAGS program (http://mcmc-jags.sourceforge.net [accessed 4 February 2015]) to specify models and perform the Bayesian analysis (Gilks et al. 1994). The R-code on all applications can be obtained as following:

1.1 Correction factor for lost LAGs (lines of arrested growth) in resorbed area in humerus bone of olive rildey sea turtles. The following R-codes are the two model applied to the relationship between LAG number:HSD (humerus section diameter from Zug et al. (2006):

```
# Power Function in JAGS #
rm(list=ls())

# Data entry #
data1 <- read.table("LAGNxLAGD2.txt", header=T) #HSD:LAG number data#
x <- data1$N
y <- data1$diam

# Carrying the data list for JAGs #
data1_lo <- list (L=length(y),x= log(x), y=log(y))

# The model #
sink("lostlags1.txt")
cat("model {
   for (i in 1:L){
   mu[i] <- b0 + b1*x[i]
   y[i] ~ dnorm(mu[i],tau)
   }
   b0 ~ dnorm(0.22,100)
   b1 ~ dnorm(0.0,1.0E-6)
   tau ~ dunif(0,10)
   sigma <- pow(tau,-0.5)
   } ")
sink()

# Set the parameter to get the posteriors #
params <- c("b0", "b1","sigma")

# Initialize the chains (3) #
outset1 <- list(list(b0=mean(y) ,b1=0, tau=1),
            list(b0=mean(y)+10 ,b1=0, tau=2),
            list(b0=mean(y)-10 ,b1=0, tau=2)
# Load the 'rjags' library #
library(rjags)
```

 # Let the model ready to generate the sample #

```
lostlags1 <- jags.model("lostlags1.txt",data = data1_lo,  outset1,n.chains=3,n.adapt=1000)
```

 # MCMC in JAGS #

```
lostlags.post <- coda.samples(lostlags1,params,n.iter=30000,thin=15)
```

 # Plot analysis #

```
plot(lostlags.post)
```

 # Plot analysis for each selected posterior #

```
plot(lostlags.post[,1],trace=F,density=T,main="Posterior",xlab="b0")
plot(lostlags.post[,2],trace=F,density=T,main="Posterior",xlab="b1")
plot(lostlags.post[,3],trace=F,density=T,main="Posterior",xlab="sigma")
```

 # Statistic summary of marginal posteriors #

```
summary(lostlags.post)
```

 # Generate headquarters with posterior sample of combined chains #

```
post.samp <- as.matrix(lostlags.post)
hist(post.samp[,1])
hist(post.samp[,2])
hist(post.samp[,3])
c(media=mean(post.samp[,1]), dp=sd(post.samp[,1]))
c(media=mean(post.samp[,2]), dp=sd(post.samp[,2]))
c(media=mean(post.samp[,3]), dp=sd(post.samp[,3]))
quantile(post.samp[,1],c(.025,.25,.50,.75,.975))
quantile(post.samp[,2],c(.025,.25,.50,.75,.975))
quantile(post.samp[,3],c(.025,.25,.50,.75,.975))
```

 # Diagnostic #

```
plot(lostlags.post[,1],trace=T,density=F)
plot(lostlags.post[,2],trace=T,density=F)
plot(lostlags.post[,3],trace=T,density=F)
cumuplot(lostlags.post,probs=c(0.025,.5,.975))
```

```
autocorr.plot(lostlags.post)
gelman.diag(lostlags.post)
gelman.plot(lostlags.post)

# DIC (Deviance Information Criteria) analysis #
DIC.1 <- dic.samples(lostlags1,type="pD",n.iter=10000,thin=5)
DIC.1

# Predicted x #
b0.post <- post.samp[,1]
b1.post <- post.samp[,2]
sigma.post <- post.samp[,3]

y.hat <- 22.23451 #this number is an example of a humerus diameter#
y.hat2 <- log(y.hat)
x.pred <- exp(y.hat2-b0.post)/b1.post

# Predictive odds #
table(x.pred)/length(x.pred)

# And in histogram #
barplot(table(round(x.pred))/length(x.pred))
round(mean(x.pred))
```

1.2 Schnute's growth model R-code applied to age-at-size data of olive ridley:

```
# Run the file containing all functions which will be used later #
source ("SGMfunctions.r")
# Entry the data #
lo_cres <- read.table("modelo_schnute_lo_crc1_power.txt",header=T) # Age:straight #
carapace length (SCL) data #
age <- lo_cres[,1]
size <- lo_cres[,2]
yg <- as.numeric(tapply(log(size), age, mean))
ng <- as.numeric(tapply (size,age, length))
xg <- as.numeric(names(table(age)))
plot(xg,exp(yg))

# Preliminary non-linear model fit #
# Choose Additive (35) or Multiplicative (36) error structure #
# mst = c("Additive") #
mst = c("Multiplic")

# Define the parameters in SGM #
x1 <- min(xg)             # tau1
x2 <- max(xg)             # tau2
y1 <- mean(size[age==x1])
y2 <- mean(size[age==x2])
c(x1=x1,y1=y1,x2=x2,y2=y2)

# initial parameters vector  ('a', 'b', 'y1', 'y2') #
p.in <- c(1/2,1,y1,y2)

# MLEs and Hessian approximation to the covariance matrix #
x <- as.numeric(age)
y <- size
if(mst=="Multiplic")
```

```
{  out1.schn <- nlm(dmq.schn.38,p.in,hessian=TRUE)
}
out1.teta <- out1.schn$estimate
out1.mtx.cov <- out1.schn$minimum/(length(y)-length(out1.teta))*solve(out1.schn$hessian)
S2e <- out1.schn$minimum/(length(y)-length(out1.teta))

# Graphic representation of data and fitted curve #
xfit <- seq(x1,x2,0.25)
yfit1.schn <- y.hat.schn.15(out1.teta,xfit,x1,x2)
plot(x,y,xlim=range(xfit))
lines(xfit,yfit1.schn)

# Bayesian Inference and Model fit #
# Bayesian fit of SGM using SIR algorithm #
# SAMPLING: Defining sample size for first stage of SIR #
is.ssize1 <- 50000
df.sir <- 8  # degrees of freedom for multivariate student importance function #

# Prior sample for sigma #
# Precision is sampled from a gamma density with expected value = 1/(S2e) #
# and coefficient of variation 'cv' #
cv <- 0.5  # coefficient of variation for precision parameter
alfa <- 1/(cv*cv)
beta <-  alfa*S2e
sigma2 <- 1/rgamma(is.ssize1,alfa,beta)

# prior sample for vector (a, b, y1, y2) #
prior.mtx.cov <- solve(out1.schn$hessian)
is.sample1 <- matrix(rep(0,4*is.ssize1),ncol=4)
for(i in 1: is.ssize1)
{   mtx.cov <- sigma2[i]*prior.mtx.cov
   is.sample1[i,] <- fsam(1,out1.teta,mtx.cov,df.sir)
}
```

```
# Calculating the weights of the sampled points #
# Calculating importance densities g(.) #
n <- length(y)
g.w <- rep(0,is.ssize1)
for(i in 1:is.ssize1)
{  mtx.cov <- sigma2[i]*prior.mtx.cov
   dv <- det(mtx.cov)
   mtx.inv <- solve(mtx.cov)
   g.w[i] <- log(fdens(is.sample1[i,],out1.teta,mtx.inv,dv,df.sir)) +
dgamma(1/sigma2[i],alfa,beta,log=T)
}

# Calculating posterior f(.) = L(data|teta)*p(teta) #
loglike <- rep(0,is.ssize1)
if(mst=="Multiplic")
{ for(i in 1:is.ssize1) loglike[i] <- (-1)*negloglike.38(c(is.sample1[i,],sigma2[i],n))
}

# Calculating weights w(.) #
is.w <- loglike-g.w
n.NA <- sum(is.na(is.w))
n.NA

# Eliminating points that result in NA #
is.sample1 <- is.sample1[!is.na(is.w),]
is.w <- is.w[!is.na(is.w)]
is.w <- exp(is.w - max(is.w))

# SIR diagnostics #
max(is.w)/sum(is.w)      # should be smaller than 0.05 #
# ERU
is.w[is.w == 0] <- 1E-16
k <- sum(is.w)
-sum(is.w*(log(is.w)-log(k)))/(k*log(is.ssize1))  # should exceed 0.90 #
```

```
# RESAMPLING  (2nd stage in SIR) #
# Set 2nd stage sample size (approximately 10% of first stage sample size #
is.ssize2 <- 4000
is.line <- sample(1:dim(is.sample1)[1], is.ssize2, replace=T, prob=is.w)
is.theta <- is.sample1[is.line,]    # posterior sample for (a, b, y1, y2)
is.sigma <- sqrt(sigma2[is.line])

# Posterior Deviance -2log(p(y | teta, sigma)) #
dev <- numeric()
if(mst=="Multiplic")
{  for(i in 1:is.ssize2)
   { s2 <- is.sigma[i]*is.sigma[i]
     dev[i] <- 2*negloglike.38(c(is.theta[i,],s2,n))
   }
}

# Posterior summary #
par.names = c("a","b","y1","y2","sigma","deviance")
post.summary <- bayes.post.summary(cbind(is.theta[,1:4],is.sigma,dev),par.names,r=3)
post.summary

# DIC #
post.mean <- as.numeric(as.vector(post.summary[,5]))
if(mst=="Multiplic")
{  dev.bar <- 2*negloglike.38(c(post.mean[1:4],post.mean[5]^2,n)) #dev(post.mean)
}
pD <- mean(dev) - dev.bar
pV <- var(dev)/2
DIC <- mean(dev) + pD
c(DIC=DIC, pD=pD, pV=pV )

# GRAPHICS #
# FIGURE 1: marginal posterior distributions #
```

```
par(mfrow=c(3,2))
for(i in 1:4) plot(density(is.theta[,i]),main=i,xlab=par.names[i])
plot(density(is.sigma),main=5,xlab=par.names[5])
plot(density(dev),main=6,xlab=par.names[6])
par(mfrow=c(1,1))
```

 # FIGURE 2: posterior growth curve #

 # 1: average SGM with posterior probability intervals (without sigma) #

```
id <- seq(x1,x2,by=0.5)
c.id <- matrix(rep(0,5*length(id)),ncol=5)
pred <- as.numeric()
for(i in 1:length(id))
{ for(j in 1:is.ssize2)
  { pred[j] <- y.hat.schn.15(is.theta[j,],id[i],x1,x2)
  }
  c.id[i,] <- as.numeric(quantile(pred,c(.025,0.10,.5,0.90,.975),na.rm=T))
}
```

 # FIGURE 3: average size per age #

```
plot(x,y,xlim=c(0,28),ylim=c(0,80),xlab="Estimated Age (yr)",
    ylab="Mean size (cm SCL)")
lines(id,c.id[,3],lwd=2,lty="solid") # mediana
lines(id,c.id[,1],lty="dashed")      #linf 95%
lines(id,c.id[,5],lty="dashed")      #lsup 95%
lines(id,c.id[,2],lty="dotted")    #linf 80%
lines(id,c.id[,4],lty="dotted")    #lsup 80%
dados_BC <- read.table("BC.txt",header=T)
points(dados_BC$id,dados_BC$L, pch=3)
points(dados_zug$age,dados_zug$size, pch=3)
```

 # FIGURE 4: predictions of individual size per age (including sigma) #

```
c.ind <- matrix(rep(0,5*length(id)),ncol=5)
n.post <- dim(is.theta)[1]
for(i in 1:length(id))
```

```
{  for(j in 1:is.ssize2)
   { pred[j] <- y.hat.schn.15(is.theta[j,],id[i],x1,x2)
     yy <- exp(rnorm(n.post,log(pred[j]),is.sigma[i]))
   }
  c.ind[i,] <- as.numeric(quantile(yy,c(.10,0.25,.5,0.75,.90),na.rm=T))
}

plot(x,y,xlim=range(id),xlab="Age (yr)",ylab="Individual size (cm SCL)", main="Schnute
Growth Model",
ylim= c(min(c.ind[,1][!is.na(c.ind[,1])])-1, max(c.ind[,5][!is.na(c.ind[,5])])+1))
lines(id,c.ind[,3],lwd=2,lty="solid") # mediana
lines(id,c.ind[,1],lty="dashed")      #linf 80%
lines(id,c.ind[,5],lty="dashed")      #lsup 80%
lines(id,c.ind[,2],lty="dotted")    #linf 50%
lines(id,c.ind[,4],lty="dotted")    #lsup 50%

# Probability for model to fit some cases #
# case 1: 0 < a and 0 < b < 1 #
case1 <- sum(is.w[is.sample1[,1] > 0 & is.sample1[,2] > 0 & is.sample1[,2] < 1])/k

# case 2: 0 < a and 1 <= b #
case2 <- sum(is.w[is.sample1[,1] > 0 & is.sample1[,2] >= 1])/k

# case 3: -b*log(y2/y1)/(x2-x1) < a <= 0  and 1 < b #
lim.a <- -is.sample1[,2]*log(is.sample1[,4]/is.sample1[,3])/(x2-x1)
case3 <- sum(is.w[is.sample1[,1] > lim.a & is.sample1[,1] <= 0 & is.sample1[,2] > 1])/k

# case 4: -b*log(y2/y1)/(x2-x1) < a <= 0  and 0 <= b <= 1 #
case4 <- sum(is.w[is.sample1[,1] > lim.a & is.sample1[,1] <= 0 & is.sample1[,2] <= 1 &
is.sample1[,2] >= 0])/k

# case 5: a <= -b*log(y2/y1)/(x2-x1)  and 0 <= b #
case5 <- sum(is.w[is.sample1[,1] <= lim.a &  is.sample1[,2] >= 0])/k
```

```
# case 6: a < 0  and b < 0 #
case6 <- sum(is.w[is.sample1[,1] < 0 &  is.sample1[,2] < 0])/k

# case 7: 0 <= a <= -b*log(y2/y1)/(x2-x1)  and b < 0 #
case7 <- sum(is.w[is.sample1[,1] > 0 & is.sample1[,1] <= lim.a & is.sample1[,2] < 0])/k

# case 8: -b*log(y2/y1)/(x2-x1) < a  and  b <= 0 #
case8 <- sum(is.w[is.sample1[,1] > lim.a & is.sample1[,2] <= 0])/k
round(c(p.M1=case1,
p.M2=case2,p.M3=case3,p.M4=case4,p.M5=case5,p.M6=case6,p.M7=case7,p.M8=case8),4)

# GRAPHICS: Posterior distributed in Fig. 1 #
minimos <- c(min(is.theta[,1]),min(is.theta[,2]))
maximos <- c(max(is.theta[,1]),max(is.theta[,2]))
y1.post <- mean(is.theta[,3],na.rm=T)
y2.post <- mean(is.theta[,4],na.rm=T)

# Option: dispersion #
plot( is.theta[,1],is.theta[,2],xlim=c(minimos[1],maximos[1]),xlab="a", ylab="b",
            ylim= c(minimos[2],maximos[2]),main="SGM: regions (Fig.1)")

# Option: contours #
# sca.contour(is.theta[,1],is.theta[,2],n.classes=16) #
abline(h=0)
abline(v=0)
abline(0,-(x2-x1)/log(y2.post/y1.post))
segments(-log(y2.post/y1.post)/(x2-x1),1, maximos[1]+1,1)

# Calculating probabilities for RESAMPLED posterior #
# case 1: 0 < a and 0 < b < 1 #
case1 <- sum(is.theta[,1] > 0 & is.theta[,2] > 0 & is.theta[,2] < 1)/is.ssize2

# case 2: 0 < a and 1 <= b #
case2 <- sum(is.theta[,1] > 0 & is.theta[,2] >= 1)/is.ssize2
```

```
# case 3: -b*log(y2/y1)/(x2-x1) < a <= 0  and 1 < b #
lim.a <- -is.theta[,2]*log(is.theta[,4]/is.theta[,3])/(x2-x1)
case3 <- sum(is.theta[,1] > lim.a & is.theta[,1] <= 0 & is.theta[,2] > 1)/is.ssize2

# case 4: -b*log(y2/y1)/(x2-x1) < a <= 0  and 0 <= b <= 1 #
case4 <- sum(is.theta[,1] > lim.a & is.theta[,1] <= 0 & is.theta[,2] <= 1 & is.theta[,2] >=
0)/is.ssize2

# case 5: a <= -b*log(y2/y1)/(x2-x1)  and 0 <= b #
case5 <- sum(is.theta[,1] <= lim.a &  is.theta[,2] >= 0)/is.ssize2

# case 6: a < 0  and b < 0 #
case6 <- sum(is.theta[,1] < 0 &  is.theta[,2] < 0)/is.ssize2

# case 7: 0 <= a <= -b*log(y2/y1)/(x2-x1)  and b < 0 #
case7 <- sum(is.theta[,1] > 0 & is.theta[,1] <= lim.a & is.theta[,2] < 0)/is.ssize2

# case 8: -b*log(y2/y1)/(x2-x1) < a  and  b <= 0 #
case8 <- sum(is.theta[,1] > lim.a & is.theta[,2] <= 0)/is.ssize2
round(c(p.M1=case1,
p.M2=case2,p.M3=case3,p.M4=case4,p.M5=case5,p.M6=case6,p.M7=case7,p.M8=case8),4)

# Calculating tau.zero, tau.star, y.star  and y.inf #
# Expressions (24) (25) (26) and (27) [Schnute, 1981] #
linhas <- 1:is.ssize2

# number of excluded lines (a = 0 and b = 0) #
is.ssize2 - sum(is.theta[,1]!=0 & is.theta[,2]!=0)
linesel <- linhas[is.theta[,1]!=0 & is.theta[,2]!=0]
new.is <- is.theta[linesel,]
gg <- (exp(x2*new.is[,1])*(new.is[,4]^new.is[,2]) -
exp(x1*new.is[,1])*(new.is[,3]^new.is[,2]))/((new.is[,4]^new.is[,2]) -
(new.is[,3]^new.is[,2]))
```

```
y.gg<- (exp(x2*new.is[,1])*(new.is[,4]^new.is[,2]) -
exp(x1*new.is[,1])*(new.is[,3]^new.is[,2]))/(exp(x2*new.is[,1]) - exp(x1*new.is[,1]))
```

 # Removing points outside possible range #

```
tau.zero <- x1 + x2 - log(gg)/new.is[,1]
tau.star <- x1 + x2 - log(new.is[,2]*gg)/new.is[,1]
y.star <- ((1-new.is[,2])*y.gg)^(1/new.is[,2])
y.inf <- (y.gg)^(1/new.is[,2])
```

 # Posterior summary #

```
par.trans = c("tau.zero","tau.star","y.star","y.inf")
bayes.post.summary(cbind(tau.zero,tau.star,y.star,y.inf),par.trans,r=3)
```

 # Effective sample size #

```
c(n.tau.zero=is.ssize2-sum(is.na(tau.zero)),n.tau.star=is.ssize2-sum(is.na(tau.star)),
  n.y.star=is.ssize2-sum(is.na(y.star)),n.y.inf=is.ssize2-sum(is.na(y.inf)))
```

```
# Function to find MLE of the parameters for Schnute's model by "nlm()" #
# This function will assume that ages ='x', sizes='y', tau1='x1, tau2='x2' #
# source ("SGMfunctions.r") #
dmq.schn.37 <- function(p)
 sum((y -(p[3]^p[2] + (p[4]^p[2]-p[3]^p[2])*(1-exp(-p[1]*(x-x1)))/(1-exp(-p[1]*(x2-
x1))))^(1/p[2]))^2)
dmq.schn.38 <- function(p)
 sum((log(y) - log((p[3]^p[2] + (p[4]^p[2]-p[3]^p[2])*(1-exp(-p[1]*(x-x1)))/(1-exp(-
p[1]*(x2-x1))))^(1/p[2])))^2)
dmq.schn.38.pos <- function(p)
 sum((log(y) - log((p[3]^(exp(p[2])) + (p[4]^exp(p[2])-p[3]^exp(p[2]))*(1-exp(-exp(p[1])*(x-
x1)))/(1-exp(-exp(p[1])*(x2-x1))))^(1/exp(p[2]))))^2)
dmq.vbgm.38 <- function(p)
 sum((log(y) - log(p[2] + (p[3]- p[2])*(1-exp(-exp(p[1])*(x-x1)))/(1-exp(-exp(p[1])*(x2-
x1)))))^2)

# Function to calculate the predictive values y.hat by equation 15 from Schnute (1981) #
y.hat.schn.15 <- function(p,x,x1,x2)
(p[3]^p[2] + (p[4]^p[2]-p[3]^p[2])*(1-exp(-p[1]*(x-x1)))/(1-exp(-p[1]*(x2-x1))))^(1/p[2])
y.hat.vbgm.15 <- function(p,x,x1,x2)
(p[2] + (p[3]- p[2])*(1-exp(-exp(p[1])*(x-x1)))/(1-exp(-exp(p[1])*(x2-x1))))

# Functions by HUMBER AGRELLI ANDRADE (HAA) #
# Function for a multidimensional student sample #
# This function is used to generate the first sample, it is the first important #
# function #
# A sample of size "n" would be obtained from a student with an average "m" #
# covariance matrix "E" and degrees of freedom "gl" #
# The function output is a vector or matrix (depending on the number of parameters) #
# of length or number of lines (depending on the case) equal to n #
fsam <-function(n,m,E,gl){
  teta<-rep(NA,length(m))
  cv<-t(chol(E))
  for(i in 1:n) teta<-rbind(teta,c(m+cv%*%rt(length(m),gl)))
```

```
  teta<-teta[-c(1),]
  teta}
fdens<-function(x,m=m0,iv=iv0,dv=dv0,gll=gl0)
  {res<-t(x-m)%*%iv%*%(x-m)
  res1<-gamma(gll/2)*gll^(length(x)/2)*pi^(length(x)/2)
  res1<-gamma((gll+length(x))/2)/res1
  res<-res1*1/sqrt(dv)*(1+1/gll*res)^(-(gll+length(x))/2)
  res}
```

 # End of the HAA's functions #

 # Function to calculate -log(L(data|teta)) #

```
negloglike.38 <- function(param){
dmq.schn.38(param[1:4])/(2*param[5]) + param[6]*log(param[5])/2
}
```

 # Calculating of -log(L(param)) for the data #

```
negloglike.37 <- function(param){
dmq.schn.37(param[1:4])/(2*param[5]) + param[6]*log(param[5])/2
# calculo da -log(L(param)) para os dados
}
negloglike.38vbgm <- function(param){
dmq.vbgm.38(param[1:3])/(2*param[4]) + param[5]*log(param[4])/2
# calculo da -log(L(param)) para os dados
}
```

 # Bayes-posterior summary #

```
bayes.post.summary <- function(mc.matrix,parnames,r=4)
{   nr.col <- dim(mc.matrix)[2]
    stats.sum <- matrix(rep(0,nr.col*5),ncol=5)
    for(i in 1:nr.col)
    {  stats.sum[i,1:3] <- quantile(mc.matrix[,i],c(.025,.5,.975),na.rm=T)
       stats.sum[i,4] <- mean(mc.matrix[,i],na.rm=T)
       stats.sum[i,5] <- sd(mc.matrix[,i],na.rm=T)
    }
```

```
   stats <- as.data.frame(cbind(parnames,round(stats.sum,r)))
   names(stats) <- c("param","2.5%","median","97.5%","mean","sdev")
   stats}
sca.contour <- function(var1,var2, n.classes = 30,g.pc = 0.1,
                    q.line=c(0.01,0.25,0.5,0.80))

# This function creates a contour plot from scatterplots #
# x = var1 and  y = var2 #
# n.classes and g.pc control smoothness and window width #
# q.line gives percentage of maximum high for contour lines #
{ x.axis <- var1
  y.axis <- var2

# Creating classes #
  x.lim <- c(min(x.axis),max(x.axis))
  y.lim <- c(min(y.axis),max(y.axis))
  dif.x <- x.lim[2]-x.lim[1]
  dif.y <- y.lim[2]-y.lim[1]
  c.x <- seq(x.lim[1]-g.pc*dif.x,x.lim[2]+g.pc*dif.x, length.out = n.classes)
  c.y <- seq(y.lim[1]-g.pc*dif.y,y.lim[2]+g.pc*dif.y, length.out = n.classes)

# Calculating matrix of frequencies #
  mp.x <- rep(0,n.classes-1)
  mp.y <- mp.x
  for(i in 2:n.classes)
  { mp.x[i-1] <- mean(c.x[c(i-1,i)])
    mp.y[i-1] <- mean(c.y[c(i-1,i)])
  }
  p1.axis <- cut(x.axis,breaks<-c.x,right=F)
  p2.axis <- cut(y.axis,breaks<-c.y,right=F)
  f.axis <- matrix(as.numeric(table(p1.axis,p2.axis)),ncol= n.classes-1 )
  f.axis <- f.axis/max(f.axis)

# Making the contourplot #
```

```
  plot(mp.x,mp.y,xlab="a", ylab="b",main="SGM (Regions)",type="n")
  contour(mp.x,mp.y,f.axis,levels=q.line,lwd=2.0,add=T)
}
```

1.3 Von Bertalanffy growth model (VBGM) R-code applied to age-at-size data of olive ridley:

```
# Entry the data #
data2 <- read.table("crc_lo_von3.txt", header=T) # Age:SCL data #
age <- data2$id
size <- data2$L
maxage <- max(age)
ma<-mean(size[age==maxage])
mo <-mean(size[age==0])

# Carrying the data list for JAGs #
data2_lo <- list(M=length(age),age=age,size=size)

# Set the parameter to get the posteriors #
# VBGM parameterized for L0 (in place of t0)
parameters <- c("linf","L0","k","sigma")

# Initialize the chains (3) #
outset2 <- list(list(loglinf=log(max(comp)),logL0=1.4,logk=log(0.04),sigma=1),
          list(loglinf=log(max(comp)+0.2),logL0=1.3,logk=log(0.02),sigma=1.5),
          list(loglinf=log(max(comp))-0.2,logL0=1.2,logk=log(0.06),sigma=0.8))

# The model #
sink("vb_L0_PGK2.txt")
cat("model {
   logL0 ~ dnorm(1.3,0.25)I(0.5,2.5)
   loglinf ~ dnorm(4.5,0.5)I(4,5)
   logk ~ dnorm(0,0.005)I(-10,10)
   sigma ~ dunif(0,5)
   k <- exp(logk)
   linf <- exp(loglinf)
   L0 <- exp(logL0)
   tau <- 1/(sigma*sigma)
   for(i in 1:M){
```

```
    expkt[i] <- exp(-k*idade[i])
    expmu[i] <- linf - (linf-L0)*expkt[i]
    mu[i] <- log(expmu[i])
    comp[i] ~ dlnorm(mu[i],tau)
    }
    } ")
 sink()
library(R2jags)
n.cad <- 3
salto <- 3
comp.cad <- 12000
n.aq <- 3000
vbgm <- jags(data2_lo,outset2,parameters,"vb_L0_PGK2.txt",n.chains=n.cad,
            n.thin=salto,n.iter=comp.cad,n.burnin=n.aq)
print(vbgm,dig=3)
```

 # Marginal posterior densities for each parameter #

```
output <- vbgm$BUGSoutput
Linf.post <- output$sims.matrix[,"linf"]
k.post <- output$sims.matrix[,"k"]
L0.post <- output$sims.matrix[,"L0"]
sigma.post <- output$sims.matrix[,"sigma"]

hist(Linf.post,nclass=150,main="Linf")
hist(k.post,nclass=150,main="k")
hist(L0.post,nclass=150,main="L0")
hist(sigma.post,nclass=150,main="sigma")
```

 # Joint posterior distribution #

```
plot(k.post, Linf.post)
```

 # Plots #

 # 1: Size predictive for VBGM with credibility interval #

```
id <- seq(0,max(age),by=1)
```

```
c.ind <- matrix(rep(0,5*length(id)),ncol=5)
n.post <- length(Linf.post)
for(i in 1:length(id))
{ pred <- log(Linf.post - (Linf.post-L0.post)*exp(-k.post*id[i]))
  yy <- exp(rnorm(n.post,pred,sigma.post))
  c.ind[i,] <- as.numeric(quantile(yy,c(0.10,0.25,.5,0.75,0.90)))
}
plot(data2$id,data2$L,xlim=c(0,28),ylim=c(0,80),
     xlab="Estimated Age (yr)", ylab="Mean size (cm SCL)")
lines(id,c.ind[,3],lwd=2,lty="solid") # mediana
lines(id,c.ind[,1],lty="dashed")       #linf 95%
lines(id,c.ind[,5],lty="dashed")       #lsup 95%
lines(id,c.ind[,2],lty="dotted")     #linf 80%
lines(id,c.ind[,4],lty="dotted")     #lsup 80%
```

 # Insert values for the species you are working on #

```
abline(h=60.0) # size data of a mature olive ridley sea turtle #
y0 <- 60.0
x.pred <- log((y0-Linf.post)/-(Linf.post-L0.post))/-k.post
table(round(x.pred))/length(x.pred)
barplot(table(round(x.pred))/length(x.pred),xlim=c(0,10),ylim=c(0,0.6),
       xlab="Estimated age (yr)",ylab="Frequency")
round(mean(x.pred,na.rm=T))
quantile(x.pred,c(0.025,0.5,0.975),na.rm=T)
```

1.4 Back-calculate model r-code applied to humerus section diamenter (HSD) and size data

```
# Entry the data #
data3 <- read.table("bph_lo_crc.txt", header=T) # HSD:SCL data #
lop <- 4.03 # Mean value of SCL of olive ridley hatchlings #
dop <- 2.20 # Mean value of humerus diameter from olive rildey hatchlings #

# Carrying the data list for JAGs #
data3_lo <- list(n=length(data3$l), x=log((data3$d)-dop),
                 y=log((data3$l)-lop))

# The model #
sink("bph1.txt")
cat("model {
    for (i in 1:n){
    mu[i] <- B + c*x[i]
    y[i] ~ dnorm(mu[i],tau)
    }
     B ~ dnorm(4.0,1.0E-6)I(1,8)
     c ~ dnorm(0.0,1.0E-6)
     tau <- 1/(sigma*sigma)
     #tau ~ dgamma(0.01,0.01)
    sigma ~ dunif(0,10)
     b <- exp(B)
    } ")
sink()

# Defining the estimated parameters #
parameters2 <- c("B", "b", "c", "sigma")

y <- log((data3$l)-lop)
x <- log((data3$d)-dop)
plot(x,y)
outset3 <- list(list(B=mean(y) ,c=0, sigma=1),
                list(B=mean(y)+1 ,c=0, sigma=2),
```

```
        list(B=mean(y)-1,c=0, sigma=3))
library(rjags)
bph_fit <- jags.model("bph1.txt", data = data3,
                      outset3, n.chains=3, n.adapt=100000)
bph_fit.post <- coda.samples(bph_fit,parameters2,n.iter=400000,thin=200)

# Output analysis #
plot(bph_fit.post)

# Or from selected posterior #
plot(bph_fit.post[,1],trace=F,density=T,main="Posterior",xlab="B")
plot(bph_fit.post[,2],trace=F,density=T,main="Posterior",xlab="b")
plot(bph_fit.post[,3],trace=F,density=T,main="Posterior",xlab="c")
plot(bph_fit.post[,4],trace=F,density=T,main="Posterior",xlab="sigma")
B <- numeric(bph_fit.post[,2])

# Statistical summary of marginal posteriors #
summary(bph_fit.post)

# Creating matrixes with posterior sample of combined chains #
post.samp2 <- as.matrix(bph_fit.post)
hist(post.samp2[,1])
hist(post.samp2[post.samp2[,2]<40,2])
hist(post.samp2[,3])
c(media=mean(post.samp2[,1]), dp=sd(post.samp2[,1]))
c(media=mean(post.samp2[,2]), dp=sd(post.samp2[,2]))
c(media=mean(post.samp2[,3]), dp=sd(post.samp2[,3]))
quantile(post.samp2[,1],c(.025,.25,.50,.75,.975))
quantile(post.samp2[,2],c(.025,.25,.50,.75,.975))
quantile(post.samp2[,3],c(.025,.25,.50,.75,.975))

# Diagnostics #
plot(bph_fit.post[,1],trace=T,density=F)
plot(bph_fit.post[,2],trace=T,density=F)
```

```
plot(bph_fit.post[,3],trace=T,density=F)
plot(bph_fit.post[,4],trace=T,density=F)
cumuplot(bph_fit.post,probs=c(0.025,.5,.975))
autocorr.plot(bph_fit.post)
gelman.diag(bph_fit.post)
gelman.plot(bph_fit.post)
```

 # DIC analysis #

```
DIC.1 <- dic.samples(bph_fit,type="pD",n.iter=10000,thin=5)
DIC.1
```

 # Predicted x #

```
B.post_lo <- post.samp2[,1]
b.post_lo <- post.samp2[,2]
c.post_lo <- post.samp2[,3]
sigma.post_lo <- post.samp2[4]
```

 # Estimated SCL from humerus diameter #

```
data4 <- read.table("tc_lo_crc.txt", header=T) # HSD:SCL data #
attach(data4)
outset4 <- list (L=length(l1),l1=l1, d1=d1, d.hat=d.hat)
crc.est <- function(l1,d1,d.hat)
{ x <- log(d1-dop)
  y.pred <- B.post_lo + c.post_lo*x
  x.hat <- log(d.hat-dop)
  y.hat <- B.post_lo + c.post_lo*x.hat
  l.hat <- exp(y.hat)+lop
  list(L=length(l1), l.hat=l.hat)
}
object <- numeric()
other.obj <- numeric()
for(i in 1:68) {
    exit <- crc.est(l1[i],d1[i],d.hat[i])
    other.obj[i] <- mean(exit$l.hat)
```

```
   }
other.obj
as.data.frame(other.obj)

# Estimated SCL from each LAG #
data5 <- read.table("tc_lo_1.txt", header=T) # LAG diameter:SCL data #
attach(data5)
outset5 <- list (L=length(l1),l1=l1, d1=d1, d.hat.1=d.hat.1)
scl.est_1 <- function(l1,d1,d.hat.1)
{ x <- d1-dop
  y.pred <- b0.post_lo + b1.post_lo*x
  taxa <- l1/(y.pred+lop)
  x.hat <- d.hat.1-dop
  y.hat <- b0.post_lo + b1.post_lo*x.hat
  l.hat2 <- y.hat+lop
  l.hat.1 <- l.hat2*taxa
  list(L=length(l1), l.hat.1=l.hat.1, taxa=taxa)
}
other.obj2 <- numeric()
object2 <- numeric()
for(i in 1:68) {
    exit2 <- scl.est(l1[i],d1[i],d.hat.1[i])
    other.obj2[i] <- mean(exit2$l.hat.1)
    object2[i] <- mean(exit2$taxa)
   }
other.obj2
as.data.frame(other.obj2)

# Growth rate calculated from the difference between estimated SCL from each LAG #
# Plot of growth rate from each LAG #
data_gr <- read.table("idxtc.txt", header=T) # Age:growth rate data #
nro.linhas <- as.numeric(tapply(data_gr$tc,data_gr$id,length))
x <- as.factor(data_gr$id)
plot(data_gr$tc~x, ylim=c(0,6)
```

,xlab="Estimated age (yr)", ylab="Growth rate (cm/yr)", outline=FALSE)

LITERATURE CITED

Gilks WR, Thomas A, Spiegelhalter DJ (1994) A language and program for complex Bayesian modeling. Statistician 43:169−177

R Core Team (2014) R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, www.R-project.org

Schnute J (1981) A versatile growth model with statistically stable parameters. Can J Fish Aquat Sci 38:1128–1140

Zug GR, Chaloupka M, Balazs GH (2006) Age and growth in olive ridley sea turtles (*Lepidochelys olivacea*) from the north-central Pacific: A skeletochronological analysis. Mar Ecol 27:263–270