Supplementary File S1: Detailed methods and R Scripts for for Bleisch, W.V.; Buzzard, P.; Souliya, D.; Li X..; Brooks, D.M. Ecology of Gamebirds in Namha National Protected Area, Lao People’s Demo-cratic Republic. Birds 2021, 2,

```
## Preparing Covariates and Camera trapping data from a study
## in the Controlled Use Zone of Namha National Protected Area, Lao PDR

## Prepare to add Class, Order, Family attributes using the IUCN database
iucn.full<-read.csv("IUCN.csv", sep=",",h=T,stringsAsFactors=T)
iucn<-iucn.full[,c("Class","Order","Family","Genus","Species")]

## load captures and add Class, Order and Family
rawcaptures<-read.csv(file="Capturesbysite4input.csv",sep=",",h=T,stringsAsFactors=F)

## Load site covariates
raw_data<-read.csv(file="LNT Camera Trap Covariates.csv", sep=",",h=T,stringsAsFactors=F)
any(is.na(raw_data)) # check for missing values
dat = subset(raw_data, Alt.meters != "N/A")  # manually remove missing values from OME
sites_all<-subset(dat,Alt.meters>0)
dim(sites_all)

sites=sites_all
## Remove one of each of pairs of trap stations that are too close to each other
# For dataset with 48 camera stations 700 meters apart, "700b"
loc_to_remove <-
c("HA04","HA11","HA22","HA25","HA27","NH01","NH14","NH15","NH21","NH22","NH28","NR09","NR12","NR19","NR
21")
exclude <- sites_all$Loc.ID %in% loc_to_remove
sites <- sites_all[!exclude,]
dim(sites)

J=dim(sites)[1] # number of camera trapping stations to be analyzed

## Set length of trapping and occasions for compiling captures
occ.length = 7  #set the length of occasions in days
max.days = 91  # set the days at which to truncate trapping

sites$Start.Number[40]=sites$Start.Number[41]  #Trim trap nights from the beginning of Hiker01

# Check difference in recorded elevation and that calculated from DEM
par(mfrow=c(1,1))
plot(sites$Alt.meters,sites$altitude.DEM)

#Take median of three NDVI measures
scaled_ndvi = scale(sites[,c('ndvi_2015','ndvi_2018','ndvi_2020')], center = TRUE, scale = TRUE)
sites$ndvi_med <- apply(scaled_ndvi,1,median)
plot(sites$ndvi_med,sites$ndvi_vals)
pairs(sites[,c(23:28)])
pairs(sites[,c(27:29)])

rawTN <- sites$Trap.nights
sites$Trap.nights <- max.days + as.numeric(rawTN<max.days)*(rawTN-max.days)
hist(rawTN, xlab="Total trap nights")
hist(sites$Trap.nights,xlab="Truncated trap nights")
k = trunc(sites$Trap.nights/occ.length)

# Adjust date & time from Excel to R format
```

```
sites$Start.Number = sites$Start.Number - 25569.0
sites$End.Number = sites$End.Number - 25569.0
par(mfrow=c(2,1))
hist(sites$Start.Number/365.25-30, xlab="Year")
hist(sites$End.Number/365.25-30, xlab="Year")
## First session ends in 2015

# Add season, roughly tracking annual rainfall and temperature in the region from Feb. 15 driest date (calendar day 45), phase shift to April 15 (calendar day 105)
sites$Mid.Number = sites$Start.Number + max.days/2.0
zero.date = 105/365.25   #April 15
sites$decimal.date = (sites$Mid.Number-zero.date)/365.25-trunc(((sites$Mid.Number-zero.date)/365.25))
sites$dryseason=sin(2*pi*(sites$decimal.date)-pi/2) #peak on April 15
sites$rainseason=sin(2*pi*(sites$decimal.date)) #peak on July 15
par(mfrow=c(1,1))
plot(sites$decimal.date,sites$dryseason,xlab='Date',ylab='Dry Season index')
plot(sites$dryseason,sites$rainseason)
par(mfrow=c(2,2))
plot((sites$Start.Number/365.25-30),sites$decimal.date,xlab='Year')
hist(sites$decimal.date,xlab="Decimal date")
plot((sites$Start.Number/365.25-30),sites$rainseason,xlab='Year',ylab='Rainy Season index')
hist(sites$rainseason,xlab="Rainy Season index")
par(mfrow=c(2,2))
plot((sites$Start.Number/365.25-30),sites$dryseason,xlab='Year',ylab='Dry Season index')
hist(sites$dryseason,xlab="Dry Season index")

## Transform aspect to make it more biologically meaningful.
sites$northness = cos((2.0*pi)*(sites$aspect/360))  # cf. Wilson et al. 2007
sites$eastness = sin((2.0*pi)*(sites$aspect/360))
par(mfrow=c(2,1))
plot(sites$aspect,sites$northness)
plot(sites$aspect,sites$eastness)
par(mfrow=c(1,1))
plot(sites$eastness,sites$northness)
par(mfrow=c(3,1))
hist(sites$aspect,xlim=c(0,360))
hist(sites$northness)
hist(sites$eastness)

sites$is_2ndsession = as.numeric((sites$Start.Number/364.75)>46)
sites$is_tourism = as.numeric(sites$Habitat.type=="Trekking")
sites$is_north = as.numeric(sites$Northing>2320000)

pairs(sites[,4:6])
pairs(sites[,c(9:10,14)])
pairs(sites[,16:21])
## Conclude that slope, tri ("terrain roughness index"") and roughness
## are all highly correlated.

rawcovariates<-sites[,c("is_tourism","is_2ndsession","is_north","is_Aggressor","is_WideAngle","Northing","Easting.47Q","Start.Number","End.Number","Trap.nights","dryseason","rainseason","altitude.DEM","northness","eastness","tpi","slope","tri","roughness","forest.cover","ndvi_med","km.village","mtrs.to.tourism")]
row.names(rawcovariates)<-sites[,1]

Logit = function(p) {
log(p/(1-p))
```

```
}

## Choose log transformed data for Alt.meters, tri, roughness,  and km.village (despite outlier) and meters to tourism.
## Choose arcsine(sqrt) transformation of slope.
## Choose Logit transformation of percentage.of.forest.cover.  Still has a skewed distribution.
## two outliers in ndvi. One is directly across river from Nalan illage.
## rainseason has a uniform distribution
## northness, rain season and meters to tourism have a bimodal distribution.

Locations <- rownames(rawcovariates)

## Load captures of hunters and other humans
huntercaptures <- read.csv(file="hunters4input.csv",sep=",",h=T,stringsAsFactors=F)
hunters <- huntercaptures[,c(5,4,2,3,6)]
names(hunters)

humancaptures <- read.csv(file="humans4input.csv",sep=",",h=T,stringsAsFactors=F)
humans <- humancaptures[,c(5,4,2,3,6)]
names(humans)

## Prepare matrix of all captures and Index of relative abundance (RAI) of hunters ##
hunters_onehour<-matrix(nrow=J,ncol=1,dimnames = list(Locations,"hunters_captures"))
hunters_RAI <-matrix(nrow=J,ncol=1,dimnames = list(Locations,"hunters_RAI"))
for (i in 1:J) {
  station=Locations[i]
  huntersbysttn =  hunters[hunters$Loc.ID==station,]
  hunters_onehour[i] =   nrow(huntersbysttn)
  hunters_RAI[i] = 100*hunters_onehour[i]/rawcovariates$Trap.nights[i]
}

## Prepare matrix of all captures and Index of relative abundance (RAI) of other humans##
humans_onehour<-matrix(nrow=J,ncol=1,dimnames = list(Locations,"humans_captures"))
humans_nh_RAI <-matrix(nrow=J,ncol=1,dimnames = list(Locations,"humans_RAI"))
for (i in 1:J) {
  station=Locations[i]
  humansbysttn =  humans[humans$Loc.ID==station,]
  humans_onehour[i] =   nrow(humansbysttn)
  humans_nh_RAI[i] = 100*humans_onehour[i]/rawcovariates$Trap.nights[i]
}
humans_RAI <- humans_nh_RAI + hunters_RAI

## Add transformed covariates

covariates = cbind(rawcovariates[,1:10],
                   northness=rawcovariates$northness,
                   eastness=rawcovariates$eastness,
                   dryseason=rawcovariates$dryseason,
                   rainseason=rawcovariates$rainseason,
                   altitude=rawcovariates$altitude.DEM,
                   slope=rawcovariates$slope,
                   roughness=rawcovariates$roughness,
                   forest.cover=rawcovariates$forest.cover,
                   ndvi=rawcovariates$ndvi_med,
                   kmvillage = rawcovariates$km.village,
                   mtrtourism = rawcovariates$mtrs.to.tourism,
                   logalt=log(rawcovariates$altitude.DEM),
                   logroughness=log(rawcovariates$roughness),
                logitcover = Logit(rawcovariates$forest.cover/100),
                logvillage = log(rawcovariates$km.village),
                logtourism = log(rawcovariates$mtrs.to.tourism),
                hunters_RAI,
                humans_RAI)

names(covariates)
Locations <- rownames(covariates)

### EXAMINATION OF COVARIATES

Cor = cor(covariates)
write.csv(Cor, file = "Covariate.correlations.csv", quote=F)
pairs(covariates[,c(19,22:26)])
Corshort = cor(covariates[,c(19,22:26)])
par(mfrow=c(1,1))
corrplot(Corshort,type = "upper", method = "ellipse", tl.pos = "tl")
corrplot(Corshort,type = "lower", method = "number", col = "black",
   add = TRUE, diag = FALSE, tl.pos = "n", cl.pos = "n")

## Scale transformed numeric covariates
names(covariates)
n_loc = dim(covariates)[1]
scaled = scale(covariates[,c(11:28)], center = TRUE, scale = TRUE)
round(colMeans(scaled)*10000)
apply(scaled,2,sd)
Xcov=cbind(intercept=rep(1,n_loc),
         is_tourism=covariates$is_tourism,
         is_2ndsession=covariates$is_2ndsession,
         is_Aggressor=covariates$is_Aggressor,
         is_WideAngle,
         is_north=covariates$is_north,
         Trap.nights=covariates$Trap.nights,
         Start.Number=covariates$Start.Number,
         as.data.frame(scaled))
names(Xcov)

### PREPARE CAMERA TRAP CAPTURE DATA ###

## Looking only at identified Pheasants and kin
pheasant_captures<-iucncaptures[iucncaptures$Class=="AVES",]
write.csv(pheasant_captures,file="Pheasant_Capturesbysite.csv",quote=F,row.names=F)
AllPheasants<-unique(pheasant_captures$Binomial)
(N_Pheasants=length(AllPheasants))
captures<-pheasant_captures
(AllSpp=AllPheasants)
N=N_Pheasants

## Calculate Naive Species Richness for each site and add to dataframe covariates
Locations<-unique(sites$Loc.ID)
(n_loc=length(Locations))
richness<-data.frame(stringsAsFactors=FALSE)
sppbysite<-vector()
Sppdump<-vector()
Listbysite<-list()

for (j in 1:n_loc) {
```

```
trapsite=Locations[j]
allbysite<-captures[captures$Loc.ID==trapsite,]
sppbysite<-unique(allbysite$Binomial)
rich <- length(sppbysite)
richness <- rbind(richness,data.frame("Loc.ID"=trapsite,"Spp.richness"=rich))
Sppdump<-c(c(Sppdump),c(sppbysite))

## Create a table of species at each trap site
Listbysite[[trapsite]]=sppbysite
}
Listbysite
#write.txt(Listbysite,file="Listbysite.txt")

## Prepare matrix of all captures and Index of relative abundance (RAI) ##
Y_onehour<-matrix(nrow=n_loc,ncol=N,dimnames = list(Locations,AllSpp))
RAI <-matrix(nrow=n_loc,ncol=N,dimnames = list(Locations,AllSpp))
for (i in 1:n_loc) {
        trapsite = Locations[i]
        allbysite =  captures[captures$Loc.ID==trapsite,]
                for (j in 1:N) {
                species = AllSpp[j]
                sitesppcaptures = allbysite[allbysite$Binomial==species,]
                Y_onehour[i,j] = nrow(sitesppcaptures)
                RAI[i,j] = 100*Y_onehour [i,j]/Xcov$Trap.nights[i]
                }
}
write.csv(Y_onehour,file="Y_onehour.csv")
write.csv(RAI,file="RAI.csv")
Spp_Rec <- cbind(Y_onehour,"is_2ndsession"=covariates$is_2ndsession,"is_tourism"=covariates$is_tourism)
write.csv(Spp_Rec,file="SppRecords.csv")

Loc_Rec=matrix(nrow=n_loc,ncol=N,rep(x=0.0,times=n_loc*N),dimnames = list(Locations,AllSpp))
for (i in 1:n_loc) {
for(j in 1:N) {
Loc_Rec[i,j] = as.numeric(Y_onehour[i,j]>0)
}
}
Loc_Rec <- cbind(Loc_Rec,covariates$is_2ndsession,covariates$is_tourism)
write.csv(Loc_Rec,file="LocRecords.csv")

## Prepare matrix of captures for Hierarchical Occupancy Model with k occasions ##
captures$Capture.Time=paste(captures$Photo.Date,captures$Photo.Time)
captures$Photo.Number = as.numeric(as.POSIXct(captures$Capture.Time))/(24*60*60)
par(mfrow=c(3,1))
hist((covariates$Start.Number/365.25-30),xlab="Year", xlim=c(13,20))
hist((covariates$End.Number/365.25-30),xlab="Year", xlim=c(13,20))
hist((captures$Photo.Number/365.25-30),xlab="Year", xlim=c(13,20))

Y_allspp=matrix(nrow=n_loc,ncol=N,rep(x=0.0,times=n_loc*N),dimnames = list(Locations,AllSpp))
for (i in 1:N){
species=AllSpp[i]
tmp.spp=captures[captures$Binomial==species,]
for (j in 1:n_loc){
trapsite=Locations[j]
start=covariates$Start.Number[j]
tmpdat=tmp.spp[tmp.spp$Loc.ID==trapsite,]
for (h in 1:k[j]){
```

```
Y_allspp[j,i] = Y_allspp[j,i] + (sum((tmpdat$Photo.Number>=(start+(h-1)*occ.length))&
        (tmpdat$Photo.Number<(start+(h*occ.length))))>0)
}}}

## Reduce number of species to those with greater than cutoff=x captures ####
cutoff=3
dim(Y_allspp)
records<-colSums(Y_allspp)>cutoff
Y<-Y_allspp[,records]
N=dim(Y)[2]

meangroup <- vector(length=N)
for (j in 1:N) {
  species = AllSpp[j]
  allsites =  captures[captures$Binomial==species,]
  meangroup[j] <- mean(allsites$Number.of.Animals)
}

covariates$Naive.Spp.richness <- richness$Spp.richness

## SUMMARY AND SAVE ##
by(data=covariates,covariates$is_tourism,summary)
(J=nrow(Y))  #Number of trap stations included
table(Xcov$is_tourism,Xcov$is_2ndsession)
dim(Xcov) #Scaled covariates for modeling
(N=ncol(Y))  #Number of species considered
dim(Y) #Capture occasions trap site by species
max.days  #maximum length of trapping sessions
occ.length #length of trap occasions in days
k #occasions per trap site

write.csv(covariates,file="Namha_Covariates.csv",quote=F,row.names=T)
write.csv(Xcov,file="Xcov.csv")  # Scaled and normalized covariates
write.csv(k,file="k.csv") # truncated trap days for each camera trapping station
write.csv(Y,file="Y.csv") # records of events (7day) for each species x each camera trapping station

############################################################
## Hierarchical community RN model JAGS code
## The model code below is modified from Appendix S1 of Li et al. 2019, adapted from Tobler et al. 2015 and Rich et al. 2016.
# The m-day pooled observation data are 3-dimensional, i.e. Station, Species, Count.
# The camera trap station data are nx-dimensional, e.g. Station, Habitat, Session, Trap Occasions,  Elevation, Cover (NDVI), km from nearest village, Topographic indices (altitude, roughness, northness, eastness)
# N= number of observed species
# k= number of occasions each camera station was operating (trunc(Station$Days/occ.length), where occ.lngth is the days of each occassion)
# J= number of camera stations
# Y= a 2-D Station X Species matrix of detection counts with dimension J by N

attach(Xcov)
# Choose covariates for availability/use (mu.a and lambda)

ENV = data.frame(
  #is_tourism,  #factor with 2 states, 1 or 0
is_2ndsession, #factor with 2 states, 1 or 0
  #ndvi,
  #logndvi,
logitcover,
kmvillage,
  #logvillage,
altitude,
(Xcov$altitude)^2,
  #logalt,
  logroughness,
dryseason,
rainseason,
  #northness,
  #eastness
  #Xcov$altitude*Xcov$logvillage
logtourism
)

# Choose covariates for individual detection, p
DET = data.frame(
  #is_tourism,
  #is_2ndsession,
is_Aggressor, #factor with 2 states, 1 or 0
is_WideAngle #factor with 2 states, 1 or 0
ndvi,
  #logndvi,
  #logitcover,
  #kmvillage,
  #logvillage,
  #altitude,
  #(Xcov$altitude)^2,
  #logalt,
  #logroughness,
  #dryseason,
  #rainseason,
northness,
eastness
  #Xcov$altitude*Xcov$logvillage
  # logtourism
)

(nx=dim(ENV))
(nz=dim(DET))
(N=ncol(Y))
(J=nrow(Y))

#Specify the data
occ.data = list(N=as.numeric(N), J=as.numeric(J), k=as.numeric(k),
             ENV=ENV,DET=DET,nx=dim(ENV),nz=dim(DET), y=as.matrix(Y))
             #,logScaledGrpMass=logScaledGrpMass
#Specify the parameters to be monitored
occ.params = c('fit', 'fit.new', 'u','v','mbeta','mpbeta','sbeta','spbeta','p','lambda','mu.p','z') #Test of model’s fit, species
specific intercepts, community coefficients of association and species specific coefficients of association, means of
availability/use, cumulative detection probability, individual detection probability, and occupancy

#Specify the initial values
occ.inits <- function() { list(
      mu.a=matrix(rbinom(N*J,size=1,prob=1),nrow=J,ncol=N))
                          # psi.mean=runif(1), p.mean=runif(1),
#If using Occupancy modeling, don't forget to initialize the Z matrix
```

```
                        }
#Specify the number of chains, number of iterations...
nc=3
ni=150e3
nb=3e3
nthin=50

#################################################
## The community model ##
modRNH_string = " model {
#Priors for occupancy random species effect
mu.u ~ dnorm(0,0.001)
#Choose between alternative priors for precision tau
sigma.u ~ dunif(0,10)
tau.u <- pow(sigma.u,-2)
#tau.u~dgamma(0.1,0.1)

#Priors for detection random species effect
mu.v ~ dnorm(0,0.001)
sigma.v ~ dunif(0,10)
tau.v <- pow(sigma.v,-2)
#tau.v~dgamma(0.1,0.1)

#Define priors for availability/occupancy covariates (percent forest cover, altitude, disturbance, etc.)
#mbeta is the community-level hyperparameter for each of the covariates,
#tbeta is the amount of variability in each of the species specific betas,
#sbeta is the species-specific covariate effects
for (a in 1:nx[2]){
#Choose priors for community coefficients of association
#mbeta[a]~dnorm(0,0.001)  #noninformative prior
# or, alternatively, more skeptical double exponential prior
mbeta[a] ~ ddexp(0.0,sqrt(2.0))  # variance of 1.0
#Choose precision for species specific coefficients of association
sigma.beta[a] ~ dunif(0,10)
tbeta[a] <- pow(sigma.beta[a],-2)
#tbeta[a]~dgamma(0.1,0.1) #alternative gamma prior
for (i in 1:N){
sbeta[i,a]~dnorm(mbeta[a],tbeta[a])}
}

#For detection
for (b in 1:nz[2]){
# mpbeta[b]~dnorm(0,0.001)
# or, alternatively, more skeptical double exponential prior
mpbeta[b] ~ ddexp(0.0,sqrt(2.0))  # variance of 1.0
sigma.pbeta[b] ~ dunif(0,10)
tpbeta[b] <- pow(sigma.pbeta[b],-2)
#tpbeta[b]~dgamma(0.1,0.1)
for (i in 1:N){
spbeta[i,b]~dnorm(mpbeta[b],tpbeta[b])}
}

#mspmbeta~dnorm(0.0,0.001)  ## coefficient for species body mass effect on detection

#Specify priors for species i from the community level prior distributions
for (i in 1:N)
{
```

```
#Mean for all species from the community level prior distributions
u[i] ~ dnorm(mu.u, tau.u)
v[i] ~ dnorm(mu.v, tau.v)
#loop over all camera stations
for (j in 1:J)
{
##Occupancy Model##
#Logistic model for occupancy
#logit(psi[j,i]) <- u[i] +inprod(sbeta[i,],ENV[j,])
#z[j,i] ~ dbern(psi[j,i]) # Occupancy with Bernoulli distribution
#Logistic model for detectability
#logit(p[j,i])<- v[i] + inprod(spbeta[i,],DET[j,])
#mu.p[j,i]<- p[j,i]*z[j,i]

##Alternative Royle-Nichols model##
#Poisson model for abundance
log(lambda[j,i]) <-  u[i] + inprod(sbeta[i,],ENV[j,])
mu.a[j,i]~dpois(lambda[j,i]) #Abundance if species is present
z[j,i]<-step(mu.a[j,i]-1)      #Occupancy
#Logistic model for detection
logit(p[j,i]) <- v[i] + inprod(spbeta[i,],DET[j,])
mu.p[j,i] <- 1-pow(1-p[j,i],mu.a[j,i])

y[j,i] ~ dbin(mu.p[j,i],k[j]) # from actual data, Y

#Create simulated dataset to calculate the Bayesian p-value
ynew[j,i] ~ dbin(mu.p[j,i],k[j])  # from simulation
#Pearson residuals
d[j,i]<-  (y[j,i] - mu.p[j,i]*k[j])/sqrt((mu.p[j,i]+0.0001)*k[j]*(1-mu.p[j,i]-0.0001))
dnew[j,i]<- (ynew[j,i]-mu.p[j,i]*k[j])/sqrt((mu.p[j,i]+0.0001)*k[j]*(1-mu.p[j,i]-0.0001))
sq[j,i]<- pow(d[j,i],2)
sq.new[j,i]<- pow(dnew[j,i],2)
}}

fit<-sum(sq[1:J,1:N])
fit.new<-sum(sq.new[1:J,1:N])
} "

modRNH = jags.model(textConnection(modRNH_string), data=occ.data, inits=occ.inits, n.chains=nc)
update(modRNH, nb)
modRNH_sim = coda.samples(model=modRNH,
variable.names=occ.params,
thin=nthin,
n.iter=ni)
modRNH_csim = as.mcmc(do.call(rbind, modRNH_sim))

# quickly store summary of results before the computer crashes again
save.image("~/R Camera Trapping 2016/Working Directory/.RData_LiRNHmod6.RData")
HPD=HPDinterval(modRNH_csim)
SummaryOut=summary(modRNH_sim)
Estimates=as.data.frame(SummaryOut[1])
Quantiles=as.data.frame(SummaryOut[2])

SummaryTable=cbind(as.data.frame(Estimates),as.data.frame(Quantiles),as.data.frame(HPD))
ColHead=c('Mean','SD','Naive SE','Time-series SE','2.5%','25%','50%','75%','97.5%','HPDI lower','HPDI upper')
colnames(SummaryTable)=ColHead
write.csv(SummaryTable,file="SummaryTable.csv",row.names=T)
```

```r
# Data summary
nsample=nc*ni/nthin
ncoeffs=nx[2] #Environmental covariates
npcoeffs=nz[2] #Detection covariates
nbetas=ncoeffs+npcoeffs
dim(Y) #species x trap site capture occasions
nlocs=J # camera trap locations
N # species considered
k #occasions per trap site
# Goodness of fit
mod=as.data.frame(modRNH_csim)
bpvalue<-mean(mod$fit.new>mod$fit)  # Bayesian p value
fit4plot<-cbind(mod$fit,mod$fit.new)
write.csv(fit4plot,file="fit4plot.csv",row.names=F)
par(mfrow=c(1,1))
plot(x=fit4plot[,1], y = fit4plot[,2], xlab="Data fit",ylab="Simulated data fit", xlim=c(0,2000),ylim=c(0,1500))
Details=c(SummaryOut[3:6],nsample=nsample,ncoeffs=ncoeffs,npcoeffs=npcoeffs,max_days=max.days,occ_days=occ.length,locs=J,spp=N,bpvalue=bpvalue)
write.csv(Details,file="Details.csv")

#Prepare to export values as matrices
LocNames <- rownames(Y)
SpeciesNames <- colnames(Y)
CoeffNames <- c(colnames(ENV),colnames(DET))
write.csv(SpeciesNames, file="SpeciesNames.csv")
write.csv(LocNames,file="LocationNames.csv")
write.csv(CoeffNames,file="CoefficientNames.csv")

## DIAGNOSTICS ##
## convergence diagnostics
#gelman.plot(modRNH_sim)
gelman.diag(modRNH_sim)
autocorr.diag(modRNH_sim)
#autocorr.plot(modRNH_sim)
effectiveSize(modRNH_sim)
raftery.diag(as.mcmc(modRNH_csim), q=0.025, r=0.001, s=0.95)
raftery.diag(as.mcmc(modRNH_csim), q=0.05, r=0.01, s=0.95)

##########################################################################
##Summarize output ('fit', 'fit.new', 'lambda', 'mbeta','mpbeta', 'mu.p', 'p','sbeta','spbeta', 'u','v','z')
## Examine Community-wide meta-parameters ##
## Calculate Posterior P values for all mbeta coefficients ##
MbetaStart = 2 + N*J #index for first row of mbeta statistics
mbetasummary<-SummaryTable[(MbetaStart+1):(MbetaStart+nbetas),]

mbetas=as.data.frame(modRNH_csim[,(MbetaStart+1):(MbetaStart+nbetas)])
mod_log= as.data.frame(mbetas>0.0)
mbppp = colMeans(mod_log)
mbetas_BF <- cbind(mbppp,(mbppp/(1-mbppp)))
colnames(mbetas_BF) = c('PPP of > 0','Bayes Factor')

mbeta_pp_percent <- matrix(nrow=(nbetas),ncol=5)
for (i in 1:5){
  slope= -0.3+i*0.1
  mod_log = as.data.frame(mbetas>slope)
  mbeta_pp_percent[,i] = colMeans(mod_log)
}
colnames(mbeta_pp_percent)<-c("-.2","-.1","0.0","0.1","0.2")
mbetatable <- cbind(mbetasummary, mbetas_BF,mbeta_pp_percent)
rownames(mbetatable) <- CoeffNames
                  #,"mspgrpmass")
write.csv(mbetatable,file="mbeta_table.csv")

par(mfrow=c(3,2))
densplot(modRNH_csim[,(MbetaStart+1):(MbetaStart+ncoeffs)])
par(mfrow=c(3,2))
densplot(modRNH_csim[,(MbetaStart+ncoeffs+1):(MbetaStart+ncoeffs+npcoeffs)])

############################################################
## Matrix for coefficient x species parameters  ##
SbetaStart = 2+nbetas+3*(N*J)
mean.sbetas <-
matrix(data=SummaryTable[(SbetaStart+1):(SbetaStart+N*(nbetas+2)),1],nrow=(nbetas+2),ncol=(N),byrow=TRUE)
rownames(mean.sbetas) <- c(CoeffNames,'u','v')
colnames(mean.sbetas) <- SpeciesNames
write.csv(mean.sbetas,file="SBetaMatrix.csv",row.names=T)
se.betas <-
matrix(data=SummaryTable[(SbetaStart+1):(SbetaStart+N*(nbetas+2)),4],nrow=(nbetas+2),ncol=(N),byrow=TRUE)
rownames(se.betas) <- c(CoeffNames,'u','v')
colnames(se.betas) <- SpeciesNames
write.csv(se.betas,file="SBeta_SE_Matrix.csv",row.names=T)

###############################################
## Summarize species-specific beta values ##
sbetas <- as.data.frame(modRNH_csim[,(SbetaStart+1):(SbetaStart+N*nbetas)])
Summary_sbetas <- summary(as.mcmc(sbetas))
Estimates=as.data.frame(Summary_sbetas[1])
Quantiles=as.data.frame(Summary_sbetas[2])
HPD_sbetas=HPDinterval(as.mcmc(sbetas))
sbeta_zero <- as.data.frame(sbetas>0.0)
sbeta_PPP <- colMeans(sbeta_zero)
sBetas_Summary=cbind(as.data.frame(Estimates),as.data.frame(Quantiles),as.data.frame(HPD_sbetas),
as.data.frame(sbeta_PPP))
sBetas_Summary=cbind(c(rep(SpeciesNames,times=nbetas)),c(rep(CoeffNames,each=N)),sBetas_Summary)
ColHead=c('Species','Covariate','Mean','SD','Naive SE','Time-series SE','2.5%','25%','50%','75%','97.5%','HPDI
lower','HPDI upper', 'PPP of > 0')
colnames(sBetas_Summary)=ColHead
## Calculate Posterior P values for all sbeta coefficients ##
sbeta_pp_percent <- matrix(nrow=nbetas*N, ncol=5)
for (i in 1:5){
  slope= -0.3+i*0.1
  mod2_log = as.data.frame(sbetas>slope)
  sbeta_pp_percent[,i] = colMeans(mod2_log)
}
colnames(sbeta_pp_percent)<-c("-.2","-.1","0.0","0.1","0.2")
rownames(sbeta_pp_percent)<-names(sbeta_PPP)
Betas_Summary <- cbind(sBetas_Summary,sbeta_pp_percent)
write.csv(sBetas_Summary,file="sBetas_Summary.csv",row.names=T)

##############################################
## Matrix for location x species parameters ##
Param1Start = 2
m.params1 <- matrix(data=SummaryTable[(Param1Start+1):(Param1Start+N*J),1],nrow=J,ncol=N)
```

```
rownames(m.params1) <- LocNames
colnames(m.params1) <- SpeciesNames
write.csv(m.params1,file="Mean_Lambda.csv",row.names=T)
SE.params1 <- matrix(data=SummaryTable[(Param1Start+1):(Param1Start+N*J),4],nrow=J,ncol=N)
rownames(SE.params1) <- LocNames
colnames(SE.params1) <- SpeciesNames
write.csv(SE.params1,file="SE_Lambda.csv",row.names=T)

ParamFrame <- as.data.frame(modRNH_csim[,(Param1Start+1):(Param1Start+N*J)])
meanParam <- matrix(nrow=nsample,ncol=N)
for (l in 1:N){
  spspParam <-ParamFrame[,((l-1)*nlocs):(l*nlocs)]
  meanParam[,l] <- rowMeans(spspParam)
}
Summary_meanParam <- summary(as.mcmc(meanParam))
HPD=HPDinterval(as.mcmc(meanParam))
SummaryOut=summary(as.mcmc(meanParam))
Estimates=as.data.frame(SummaryOut[1])
Quantiles=as.data.frame(SummaryOut[2])
SummaryParam=cbind(as.data.frame(Estimates),as.data.frame(Quantiles),as.data.frame(HPD))
ColHead=c('Mean','SD','Naive SE','Time-series SE','2.5%','25%','50%','75%','97.5%','HPDI lower','HPDI upper')
colnames(SummaryParam)=ColHead
rownames(SummaryParam)=SpeciesNames
write.csv(SummaryParam,file="simmean_lambda.csv",row.names=T)

Param2Start = MbetasStart + nbetas
m.params2 <- matrix(data=SummaryTable[(Param2Start+1):(Param2Start+N*J),1],nrow=J,ncol=N)
rownames(m.params2) <- LocNames
colnames(m.params2) <- SpeciesNames
write.csv(m.params2,file="Mean_mu_P.csv",row.names=T)
SE.params2 <- matrix(data=SummaryTable[(Param2Start+1):(Param2Start+N*J),4],nrow=J,ncol=N)
rownames(SE.params2) <- LocNames
colnames(SE.params2) <- SpeciesNames
write.csv(SE.params2,file="SE_mu_P.csv",row.names=T)

ParamFrame <- as.data.frame(modRNH_csim[,(Param2Start+1):(Param2Start+N*J)])
meanParam <- matrix(nrow=nsample,ncol=N)
for (l in 1:N){
  spspParam <-ParamFrame[,((l-1)*nlocs):(l*nlocs)]
  meanParam[,l] <- rowMeans(spspParam)
}
Summary_meanParam <- summary(as.mcmc(meanParam))
HPD=HPDinterval(as.mcmc(meanParam))
SummaryOut=summary(as.mcmc(meanParam))
Estimates=as.data.frame(SummaryOut[1])
Quantiles=as.data.frame(SummaryOut[2])
SummaryParam=cbind(as.data.frame(Estimates),as.data.frame(Quantiles),as.data.frame(HPD))
colnames(SummaryParam)=ColHead
rownames(SummaryParam)=SpeciesNames
write.csv(SummaryParam,file="simmean_mu_P.csv",row.names=T)

Param3Start = Param2Start + N*J
m.params3 <- matrix(data=SummaryTable[(Param3Start+1):(Param3Start+N*J),1],nrow=J,ncol=N)
rownames(m.params3) <- LocNames
colnames(m.params3) <- SpeciesNames
write.csv(m.params3,file="Mean_Ind_P.csv",row.names=T)
SE.params3 <- matrix(data=SummaryTable[(Param3Start+1):(Param3Start+N*J),4],nrow=J,ncol=N)
```

```
rownames(SE.params3) <- LocNames
colnames(SE.params3) <- SpeciesNames
write.csv(SE.params3,file="SE_Ind_P.csv",row.names=T)

ParamFrame <- as.data.frame(modRNH_csim[,(Param3Start+1):(Param3Start+N*J)])
meanParam <- matrix(nrow=nsample,ncol=N)
for (l in 1:N){
  spspParam <-ParamFrame[,((l-1)*nlocs):(l*nlocs)]
  meanParam[,l] <- rowMeans(spspParam)
}
Summary_meanParam <- summary(as.mcmc(meanParam))
HPD=HPDinterval(as.mcmc(meanParam))
SummaryOut=summary(as.mcmc(meanParam))
Estimates=as.data.frame(SummaryOut[1])
Quantiles=as.data.frame(SummaryOut[2])
SummaryParam=cbind(as.data.frame(Estimates),as.data.frame(Quantiles),as.data.frame(HPD))
colnames(SummaryParam)=ColHead
rownames(SummaryParam)=SpeciesNames
write.csv(SummaryParam,file="simmean_Ind_P.csv",row.names=T)

Param4Start = SBetasStart + N*(nbetas + 2)
m.params4 <- matrix(data=SummaryTable[(Param4Start+1):(Param4Start+N*J),1],nrow=J,ncol=N)
rownames(m.params4) <- LocNames
colnames(m.params4) <- SpeciesNames
write.csv(m.params4,file="mean_Psi.csv",row.names=T)
SE.params4 <- matrix(data=SummaryTable[(Param4Start+1):(Param4Start+N*J),4],nrow=J,ncol=N)
rownames(SE.params4) <- LocNames
colnames(SE.params4) <- SpeciesNames
write.csv(SE.params4,file="SE_Psi.csv",row.names=T)

ParamFrame <- as.data.frame(modRNH_csim[,(Param4Start+1):(Param4Start+N*J)])
meanParam <- matrix(nrow=nsample,ncol=N)
for (l in 1:N){
  spspParam <-ParamFrame[,((l-1)*nlocs):(l*nlocs)]
  meanParam[,l] <- rowMeans(spspParam)
}
Summary_meanParam <- summary(as.mcmc(meanParam))
HPD=HPDinterval(as.mcmc(meanParam))
SummaryOut=summary(as.mcmc(meanParam))
Estimates=as.data.frame(SummaryOut[1])
Quantiles=as.data.frame(SummaryOut[2])
SummaryParam=cbind(as.data.frame(Estimates),as.data.frame(Quantiles),as.data.frame(HPD))
colnames(SummaryParam)=ColHead
rownames(SummaryParam)=SpeciesNames
write.csv(SummaryParam,file="simmean_Psi.csv",row.names=T)
```