Habitat selection in transformed landscapes and the role of forest remnants and shade coffee in the conservation of resident birds.

Sánchez-Clavijo, L.M., Bayly, N.J., and Quintana-Ascencio P.F. (2019). *Journal of Animal Ecology* **V**(i): pp.

Supporting Information - Part 2: R code used for data analysis, following the order and structure of the manuscript.

1. Level 1: Separate indicators of habitat preference and performance

1.1. Habitat preference

1.1.1. Occurrence

```
### Occupancy model with effect of habitat on p and quadratic altitude & habitat on psi
### Based on code available at
### http://www.vogelwarte.ch/de/projekte/publikationen/bpa/code-for-running-bpa-using-jags.html
### Kéry & Schaub 2012 Bayesian population analysis using WinBUGS: a hierarchical perspective.

## Part 1. Get ready

setwd("")
data<-read.table("",header=TRUE)
y<-as.matrix(data[,5:20]) #Select columns with detected (1) / not detected (0) data
habitat<-data$hab
alt<-as.vector(scale(data$altitude))
library(jagsUI)

## Part 2. Set up and run Bayesian model in JAGS

cat("
  model {

  # Priors
  alpha.psi ~ dnorm(0,0.01)
  beta1.psi ~ dnorm(0,0.01)
  beta2.psi ~ dnorm(0,0.01)
  beta4.psi ~ dnorm(0,0.01)
  alpha.p ~ dnorm(0,0.01)
  beta1.p ~ dnorm(0,0.01)

  # Likelihood

  # Ecological model for true occurrence
  for (i in 1:R) {
  z[i] ~ dbern(psi[i])
  psi[i] <- 1 / (1 + exp(-lpsi.lim[i]))
  lpsi.lim[i] <- min(999, max(-999, lpsi[i]))
  lpsi[i] <- alpha.psi + beta1.psi*habitat[i] + beta2.psi*alt[i] + beta4.psi*pow(alt[i],2)

  # Observation model for replicated detection/non-detection observations
  for (j in 1:T){
  y[i,j] ~ dbern(mu.p[i,j])
  mu.p[i,j] <- z[i] * p[i,j]
  p[i,j] <- 1 / (1+exp(-lp.lim[i,j]))
  lp.lim[i,j] <- min(999, max(-999, lp[i,j]))
  lp[i,j] <- alpha.p + beta1.p*habitat[i]
  } #j
  } #i

  # Derived quantities
  occ.fs <- sum(z[]) # Number of occupied sites
```

```
  for (s in 1:9){
  odds1[s] <- alpha.psi + beta1.psi*habitat[s] + beta2.psi*alt[s] + beta4.psi*pow(alt[s],2)
  psi.site[s] <- exp(odds1[s]) / (1+exp(odds1[s]))
  } #site

  for (i in 1:R){
  for (j in 1:T){
  odds2[i,j] <- alpha.p + beta1.p*habitat[i]
  pij[i,j] <- exp(odds2[i,j]) / (1+exp(odds2[i,j]))
  } #occasion
  } #site

  }",file="name.txt")

data<-list(y=y,R=nrow(y),T=ncol(y),habitat=habitat,alt=alt)
zst<-apply(y,1,max,na.rm=TRUE)
inits<-function(){list(z=zst,alpha.p=runif(1,-3,3),alpha.psi=runif(1,-3,3))}
params<-c("alpha.psi","beta1.psi","beta2.psi","beta4.psi","psi.site","alpha.p","beta1.p","occ.fs")
ni<-120000
nb<-20000
nt<-100
nc<-3

m=jags(data,inits,params,model="name.txt",n.chains=nc,n.thin=nt,n.iter=ni,n.burnin=nb);m

## Note: output for psi for all species was manually put together into a single file

## Part 3. Build graphs

d<-read.table("occupancy.txt",header=T)
d<-data[data$SPP=="",] # Select species to make graph
attach(d)

plot(ALT,psi_m,type="n",ylab="Occupancy",xlab="Elevation",main="Turdus flavipes",ylim=c(0.95,1.0))
points(ALT[HABITAT=="COFFEE"],psi_m[HABITAT=="COFFEE"],col="red",pch=16)
points(ALT[HABITAT=="FOREST"],psi_m[HABITAT=="FOREST"],col="black",pch=16)
ALT2<-ALT^2
m<-lm(psi_m~ALT+ALT2+HABITAT)
x<-seq(min(ALT),max(ALT),1)
one<-rep("COFFEE",length(x))
two<-rep("FOREST",length(x))
y1<-predict(m,list(ALT=x,ALT2=x^2,HABITAT=one),interval=c("confidence"),level=0.95,type="response")
y2<-predict(m,list(ALT=x,ALT2=x^2,HABITAT=two),interval=c("confidence"),level=0.95,type="response")
lines(x,y1[,1],col="red")
lines(x,y1[,2],col="red",lty=2)
lines(x,y1[,3],col="red",lty=2)
lines(x,y2[,1],col="black")
lines(x,y2[,2],col="black",lty=2)
lines(x,y2[,3],col="black",lty=2)
```

1.1.2. Abundance

```
### Closed population model with effect of quadratic effort on p
### Based on code available at
### http://www.vogelwarte.ch/de/projekte/publikationen/bpa/code-for-running-bpa-using-jags.html
### Kéry & Schaub 2012 Bayesian population analysis using WinBUGS: a hierarchical perspective.
### Forest version (code for coffee is exactly the same)
### Used for species with low recapture rates

## Part 1. Get ready

setwd("")
CH<-read.table("",header=F)
CH<-as.matrix(CH)
nz<-2000 # Set number of augmented individuals
CH.aug<-rbind(CH,matrix(0,ncol=dim(CH)[2],nrow=nz)) # Bind real data with augmented individuals
```

```
eff <- c(-0.6733507,-0.1116703,-0.8040866,1.0728217,-1.0522429,-0.3574055,1.7222646,1.1242687,-1.0855321,0.2690376,-1.0558744,0.9517698) #
Vector of effort per occasion (scaled mist net hours)
library(jagsUI)

## Part 2. Set up and run Bayesian model in JAGS

cat("
        model {

        # Priors and constraints
        psi ~ dunif(0,1)
        for (t in 1:n.occ){
                logit(p[t]) <- mp + beta1*x[t] + beta2*pow(x[t],2)
                p.est[t] <- 1 / (1+exp(-mp-beta1*x[t]-beta2*pow(x[t],2)))
        } #t
        mp ~ dnorm(0,0.001)
        mean.p <- 1 / (1+exp(-mp))
        beta1 ~ dnorm(0,0.001)I(-10,10)
        beta2 ~ dnorm(0,0.001)I(-10,10)

        # Likelihood
        for (i in 1:M){
        z[i] ~ dbern(psi)
        for (t in 1:n.occ){
        y[i,t] ~ dbern(mu[i,t])
        mu[i,t] <- z[i] * p[t]
        } #t
        } #i

        # Derived quantities
        N <- sum(z[])

        }",file="name.txt")

data <- list(y=CH.aug,M=dim(CH.aug)[1],n.occ=dim(CH.aug)[2],x=eff)
inits <- function(){list(z=rep(1,dim(CH.aug)[1]),mp=rnorm(1),beta1=runif(1,-5,5),beta2=runif(1,-5,5))}
params <- c("N","p.est","mean.p","beta1","beta2","psi")
ni <- 12000
nb <- 2000
nt <- 10
nc <- 3

m=jags(data=data,inits=inits,parameters=params,model="name.txt",n.chains=nc,n.iter=ni,n.burnin=nb,n.thin=nt,parallel=TRUE)

## Part 3. Collect output

write.table(m$summary,"C:/summ.txt",sep="\t") #Saves summary statistics for all iterations
write.table(m$sims.list,"C:/sims.txt",sep="\t") #Saves complete results for each iteration
pdf("C:/plots.pdf") #Next three lines save diagnostic plots in pdf files
plot(m)
dev.off()

## Part 4. Build graphs

library(ggplot2)
C_sim<-read.table("",header=T) #Simulation file for coffee
F_sim<-read.table("",header=T) #Simulation file for forest
Ncof<-C_sim$N
Nfor<-F_sim$N
Nsup<-c(Ncof,Nfor)
type<-c(rep(1,length(Ncof)),rep(2,length(Nfor)))
p.graph<-cbind(type,Nsup)
p.graph<-as.data.frame(p.graph)
p.graph$type<-factor(p.graph$type)
levels(p.graph$type)<-c("Coffee","Forest")
ggplot(p.graph,aes(p.graph$Nsup,fill=p.graph$type))+
  geom_histogram(position="dodge",binwidth=20)+
  scale_fill_manual(values=c("red","black"))+
  xlab("Total Abundance")+
```

```
 ylab("Frequency")+
 ggtitle("SPECIES")+
 xlim(min(Nsup),max(Nsup))+
 theme_classic()+
 geom_vline(xintercept=min(Nsup))+
 geom_hline(yintercept=0)+
 theme(axis.text=element_text(size=10),axis.title=element_text(size=12))+
 theme(legend.position="none")
```

```
### Jolly-Seber model with effect of quadratic effort on p and random variation of phi per occasion
### Parameterized as a multistate model based on code available at
### http://www.vogelwarte.ch/de/projekte/publikationen/bpa/code-for-running-bpa-using-jags.html
### Kéry & Schaub 2012 Bayesian population analysis using WinBUGS: a hierarchical perspective.
### Forest version (code for coffee is exactly the same)
### Used for species with high recapture rates

### Parameters
# phi: survival probability
# gamma: removal entry probability
# p1: capture probability in coffee
# p2: capture probability in forest

### States (S)
# 1 not yet entered
# 2 alive
# 3 dead

### Observations (O)
# 1 captured
# 2 not captured

## Part 1. Get ready

setwd("")
nz<-2000 # Set number of augmented individuals
CH<-read.table("",header=F)
V0<-rep(0,dim(CH)[1])
CH.du<-cbind(V0,CH)
mat.n<-matrix(0,ncol=dim(CH.du)[2],nrow=nz) # Bind real data with augmented individuals
colnames(mat.n)<-colnames(CH.du)
CH.ms<-rbind(CH.du,mat.n)
CH.ms[CH.ms==0]<-2
eff_forest <- c(1144,2072,928,4029,518,1666,5102,4114,463,2701,512,3829) # Vector of mist net hours
e_for <- as.vector(scale(eff_forest))
eff <- c(0,e_for)
library(jagsUI)

## Part 2. Set up and run Bayesian model in JAGS

cat("model {

 # Priors and constraints
  for (t in 1:(n.occasions-1)){
  gamma[t] ~ dunif(0,1)
  phi[t] ~  dunif(0,1)
  } #t

 alpha.p ~ dnorm(0,0.001)
 beta1.p ~ dnorm(0,0.001)
 beta2.p ~ dnorm(0,0.001)

 # Define state-transition and observation matrices
  for (j in 1:M){

  # Define probabilities of state S(t+1) given S(t)
   for (t in 1:(n.occasions-1)){
   ps[1,j,t,1] <- 1-gamma[t]
   ps[1,j,t,2] <- gamma[t]
   ps[1,j,t,3] <- 0
```

```
    ps[2,j,t,1] <- 0
    ps[2,j,t,2] <- phi[t]
    ps[2,j,t,3] <- 1-phi[t]
    ps[3,j,t,1] <- 0
    ps[3,j,t,2] <- 0
    ps[3,j,t,3] <- 1

  # Define probabilities of O(t) given S(t)
    po[1,j,t,1] <- 0
    po[1,j,t,2] <- 1
    po[2,j,t,1] <- p[t]
    po[2,j,t,2] <- 1-p[t]
    po[3,j,t,1] <- 0
    po[3,j,t,2] <- 1

   } #t
  } #i

# Likelihood

for (t in 1:n.occasions){
lin.p[t] <- alpha.p + beta1.p * eff[t] + beta2.p * pow(eff[t],2)
p[t] <- exp(lin.p[t]) / (1+exp(lin.p[t]))
} #t

for (i in 1:M){

  # Define latent state at first occasion
  z[i,1] <- 1

  for (t in 2:n.occasions){

    # State process: draw S(t) given S(t-1)
    z[i,t] ~ dcat(ps[z[i,t-1], i, t-1,])

    # Observation process: draw O(t) given S(t)
    y[i,t] ~ dcat(po[z[i,t], i, t-1,])

  } #t
} #i

# Calculate derived population parameters

for (t in 1:(n.occasions-1)){
qgamma[t] <- 1-gamma[t]
} #t

cprob[1] <- gamma[1]

for (t in 2:(n.occasions-1)){
cprob[t] <- gamma[t] * prod(qgamma[1:(t-1)])
} #t

psi <- sum(cprob[])

for (t in 1:(n.occasions-1)){
b[t] <- cprob[t] / psi
} #t

for (i in 1:M){
  for (t in 2:n.occasions){
  al[i,t-1] <- equals(z[i,t], 2)
  } #t
  for (t in 1:(n.occasions-1)){
  d[i,t] <- equals(z[i,t]-al[i,t],0)
  } #t
alive[i] <- sum(al[i,])
} #i
```

```
  for (t in 1:(n.occasions-1)){
  N[t] <- sum(al[,t])
  B[t] <- sum(d[,t])
  } #t

  for (i in 1:M){
  w[i] <- 1-equals(alive[i],0)
  } #i

  Nforest <- sum(w[])

} #model
", file="name.txt")

data <- list(y=CH.ms,n.occasions=dim(CH.ms)[2],M=dim(CH.ms)[1],eff=eff)

js.multistate.init <- function(ch,nz){
  ch[ch==2] <- NA
  state <- ch
  for (i in 1:nrow(ch)){
    n1 <- min(which(ch[i,]==1))
    n2 <- max(which(ch[i,]==1))
    state[i,n1:n2] <- 2
  }
  state[state==0] <- NA
  get.first <- function(x) min(which(!is.na(x)))
  get.last <- function(x) max(which(!is.na(x)))
  f <- apply(state, 1, get.first)
  l <- apply(state, 1, get.last)
  for (i in 1:nrow(ch)){
    state[i,1:f[i]] <- 1
    if(l[i]!=ncol(ch)) state[i, (l[i]+1):ncol(ch)] <- 3
    state[i, f[i]] <- 2
  }
  m.z <- matrix(1, ncol = ncol(ch), nrow = nz)
  colnames(m.z) <- colnames(state)
  state <- rbind(state,m.z)
  state <-as.matrix(state)
  state[,1] <- NA
  return(state)
}

inits <- function(){list(z = js.multistate.init(CH.du,nz),
                         alpha.p=runif(1,-1,1),beta1.p=runif(1,-1,1),beta2.p=runif(1,-1,1),
                         phi=runif(12,0,1))}

params <- c("Nforest","p","alpha.p","beta1.p","beta2.p","phi","psi","b","B","N","gamma")

ni <- 12000
nb <- 2000
nt <- 10
nc <- 3

Jm=jags(data=data,inits=inits,parameters=params,model="name.txt",n.chains=nc,n.iter=ni,n.burnin=nb,n.thin=nt,parallel=TRUE)

## Part 3. Collect output

write.table(m$summary," C:/summ.txt",sep="\t") #Saves summary statistics for all iterations
write.table(m$sims.list," C:/summ.txt",sep="\t") #Saves complete results for each iteration
pdf("C:/plots.pdf") #Next three lines save diagnostic plots in pdf files
plot(m)
dev.off()

## Part 4. Build graphs

library(ggplot2)
C_sim<-read.table("",header=T) #Simulation file for coffee
F_sim<-read.table("",header=T) #Simulation file for forest
Ncof<-C_sim$Ncoffee
```

```
Nfor<-F_sim$Nforest
Nsup<-c(Ncof,Nfor)
type<-c(rep(1,length(Ncof)),rep(2,length(Nfor)))
p.graph<-cbind(type,Nsup)
p.graph<-as.data.frame(p.graph)
p.graph$type<-factor(p.graph$type)
levels(p.graph$type)<-c("Coffee","Forest")
ggplot(p.graph,aes(p.graph$Nsup,fill=p.graph$type))+
 geom_histogram(position="dodge",binwidth=5)+
 scale_fill_manual(values=c("red","black"))+
 xlab("Total Abundance")+
 ylab("Frequency")+
 ggtitle("MYCO")+
 xlim(min(Nsup),max(Nsup))+
 theme_classic()+
 geom_vline(xintercept=min(Nsup))+
 geom_hline(yintercept=0)+
 theme(axis.text=element_text(size=10),axis.title=element_text(size=12))+
 theme(legend.position="none")
```

1.1.3. Fidelity

```
### Cormack-Jolly-Seber model with effect of quadratic effort on p and fixed phi
### Based on code available at
### http://www.vogelwarte.ch/de/projekte/publikationen/bpa/code-for-running-bpa-using-jags.html
### Kéry & Schaub 2012 Bayesian population analysis using WinBUGS: a hierarchical perspective.
### Forest version (code for coffee is exactly the same)
### Used for species with low recapture rates

## Part 1. Get ready

setwd("")
CH<-read.table("",header=F)
CH<-as.matrix(CH)
get.first<-function(x)min(which(x!=0))
f<-apply(CH,1,get.first)
eff <- c(-0.6733507,-0.1116703,-0.8040866,1.0728217,-1.0522429,-0.3574055,1.7222646,1.1242687,-1.0855321,0.2690376,-1.0558744) # Vector of
effort per occasion (scaled mist net hours)
library(jagsUI)

## Part 2. Set up and run Bayesian model in JAGS

cat("model {

        # Priors and constraints
        for (i in 1:nind){
                for (t in f[i]:(n.occasions-1)){
                phi[i,t] <- phi.occ[t]
                logit(p[i,t]) <- mu + beta1*x[t] + beta2*pow(x[t],2)
                } #t
        } #i
        for (t in 1:(n.occasions-1)){
        phi.occ[t] ~ dunif(0,1)
        p.est[t] <- 1 / (1+exp(-mu-beta1*x[t]-beta2*pow(x[t],2)))
        } #t
        mu ~ dnorm(0,0.001)
        mean.p <- 1 / (1+exp(-mu))
        beta1 ~ dnorm(0,0.001)I(-10,10)
        beta2 ~ dnorm(0,0.001)I(-10,10)

        # Likelihood
        for (i in 1:nind){

        # Define latent state at first capture
        z[i,f[i]] <-1
                for (t in (f[i]+1):n.occasions){

        # State process
```

```
            z[i,t] ~dbern(mu1[i,t])
            mu1[i,t]<-phi[i,t-1] *z[i,t-1]

            # Observation process
            y[i,t] ~dbern(mu2[i,t])
            mu2[i,t]<-p[i,t-1]*z[i,t]
                        } #t
            } #i

            }",file="name.txt")

            known.state.cjs <- function(ch){
                        state <- ch
                        for (i in 1:dim(ch)[1]){
                        n1 <- min(which(ch[i,]==1))
                        n2 <- max(which(ch[i,]==1))
                        state[i,n1:n2] <- 1
                        state[i,n1] <- NA
                        }
            state[state==0] <- NA
            return(state)
            }

data<-list(y=CH,f=f,nind=dim(CH)[1],n.occasions=dim(CH)[2],z=known.state.cjs(CH),x=eff)

            cjs.init.z <- function(ch,f){
            for (i in 1:dim(ch)[1]){
                        if (sum(ch[i,])==1) next
                        n2 <- max(which(ch[i,]==1))
                        ch[i,f[i]:n2] <- NA
                        }
            for (i in 1:dim(ch)[1]){
                        ch[i,1:f[i]] <- NA
                        }
            return(ch)
            }

inits <- function(){list(z=cjs.init.z(CH,f),mu=rnorm(1),beta1=runif(1,-5,5),beta2=runif(1,-5,5),phi.occ=runif(11,0,1))}
params<-c("phi.occ","p.est","mean.p","beta1","beta2")
ni <- 12
nb <- 2
nt <- 1
nc <- 3

m=jags(data=data,inits=inits,parameters=params,model="name.txt",n.chains=nc,n.iter=ni,n.burnin=nb,n.thin=nt,parallel=TRUE)

## Part 3. Collect output

write.table(m$summary,"C:/summ.txt",sep="\t") #Saves summary statistics for all iterations
write.table(m$sims.list,"C:/sims.txt",sep="\t") #Saves complete results for each iteration
pdf("C:/plots.pdf") #Next three lines save diagnostic plots in pdf files
plot(m)
dev.off()

## Part 4. Calculate summary statistics for phi and build graphs

sim<-read.table("",header=T) #Simulation file for species: habitat of interest
n.iter<-dim(sim)[1]
phi_m<-as.vector(rep(0,n.iter))
phi_v<-as.vector(rep(0,n.iter))
phi_sd<-as.vector(rep(0,n.iter))
phi_cv<-as.vector(rep(0,n.iter))
for(i in 1:n.iter){
  phi_m[i]<-mean(as.numeric(sim[i,20:30])) #20:30 are the columns in which phi [1-11] were recorded
  phi_v[i]<-var(as.numeric(sim[i,20:30]))
  phi_sd[i]<-sd(as.numeric(sim[i,20:30]))
  phi_cv[i]<-phi_sd[i]/phi_m[i]
}
phi.F<-matrix(0,2,4)
```

```
phi.F[1,1]<-round(mean(phi_m),4)
phi.F[1,2]<-round(mean(phi_v),4)
phi.F[1,3]<-round(mean(phi_sd),4)
phi.F[1,4]<-round(mean(phi_cv),4)
phi.F[2,1]<-round(var(phi_m),4)
phi.F[2,2]<-round(var(phi_v),4)
phi.F[2,3]<-round(var(phi_sd),4)
phi.F[2,4]<-round(var(phi_cv),4)

library(ggplot2)
C_sim<-read.table("",header=T) #Simulation file for coffee
F_sim<-read.table("",header=T) #Simulation file for forest

Cphi<-as.vector(rep(0,n.iter))
for(i in 1:n.iter){
 Cphi[i]<-mean(as.numeric(C_sim[i,20:30]))
}

Fphi<-as.vector(rep(0,n.iter))
for(i in 1:n.iter){
 Fphi[i]<-mean(as.numeric(F_sim[i,20:30]))
}

phi2<-c(Cphi,Fphi)
type<-c(rep(1,length(Cphi)),rep(2,length(Fphi)))
p.graph<-cbind(type,phi2)
p.graph<-as.data.frame(p.graph)
p.graph$type<-factor(p.graph$type)
levels(p.graph$type)<-c("Coffee","Forest")
ggplot(p.graph,aes(p.graph$phi2,color=p.graph$type))+
 geom_density()+
 scale_color_manual(values=c("red","black"))+
 xlab("Survival")+
 ylab("Density")+
 ggtitle("MYCO")+
 xlim(0,1)+
 theme_classic()+
 geom_vline(xintercept=0)+
 theme(axis.text=element_text(size=10),axis.title=element_text(size=12))+
 theme(legend.position="none")
```

1.1.4. Inter-seasonal variance

```
### Calculation for inter-seasonal variance per iteration uses output for Jolly-Seber model as input

F_sim<-read.table("",header=T) #Simulation file for forest
n.iter<-dim(F_sim)[1]
F_m<-as.vector(rep(0,n.iter))
F_sd<-as.vector(rep(0,n.iter))
F_CV<-as.vector(rep(0,n.iter))
for(i in 1:n.iter){
 F_m[i]<-mean(as.numeric(F_sim[i,55:66]))
 F_sd[i]<-sd(as.numeric(F_sim[i,55:66]))
 F_CV[i]<-F_sd[i]/F_m[i]
}

mean(F_CV)
sd(F_CV)

C_sim<-read.table("",header=T) #Simulation file for coffee
n.iter<-dim(C_sim)[1]
C_m<-as.vector(rep(0,n.iter))
C_sd<-as.vector(rep(0,n.iter))
C_CV<-as.vector(rep(0,n.iter))
for(i in 1:n.iter){
 C_m[i]<-mean(as.numeric(C_sim[i,55:66]))
 C_sd[i]<-sd(as.numeric(C_sim[i,55:66]))
 C_CV[i]<-C_sd[i]/C_m[i]
```

```
}

mean(C_CV)
sd(C_CV)
```

1.1.5. Age

```
### Generalized linear models for mist-net data - AGE

## Part 1. Get ready (and modify and define response)

library(AICcmodavg)
setwd("")
data<-read.table("",header=T)
d<-data[data$SP=="SSPP",] #Select species
d1<-d[d$AGE=="3" | d$AGE=="4",] #Subset to only immature and adult individuals
d1$AGE<-as.character(d1$AGE)
d1$AGE[d1$AGE=="3"] <- 0 # FAIL defined as capturing an immature individual
d1$AGE[d1$AGE=="4"] <- 1 # SUCCESS defined as capturing an adult individual
d1$AGE<-as.numeric(d1$AGE)
d1$RESP<-d1$AGE
d1$HAB<-as.factor(d1$HAB)
d1$DAY2<-d1$DAY^2
d1$DAYs<-scale(d1$DAY)
d1$DAY2s<-scale(d1$DAY2)
attach(d1)

## Part 2. Define models and run model selection

M0<-glm(RESP~1,family=binomial)
Mt<-glm(RESP~DAYs+DAY2s,family=binomial)
Mh<-glm(RESP~HAB,family=binomial)
Mth<-glm(RESP~DAYs+DAY2s+HAB,family=binomial)
summary(M0)
summary(Mt)
summary(Mh)
summary(Mth)
Cand.mods <- list(M0,Mt,Mh,Mth)
modnames<-c("M0","Mt","Mh","Mth")
aictab(Cand.mods,modnames,sort=TRUE)
modavg(Cand.mods,parm="(Intercept)",modnames)
modavg(Cand.mods,parm="DAYs",modnames)
modavg(Cand.mods,parm="DAY2s",modnames)
modavg(Cand.mods,parm="HABFOREST",modnames)

## Part 3. Build graphs

plot(DAY,RESP,type="n",ylab="p(variable)",xlab="Day of year",main="SSPP-RES",ylim=c(0,1))
points(DAY[HAB=="COFFEE"],RESP[HAB=="COFFEE"],col="red",pch=16)
points(DAY[HAB=="FOREST"],RESP[HAB=="FOREST"],col="black",pch=16)
m<-glm(RESP~DAY+DAY2+HAB,family=binomial)
plotdatc<-data.frame(DAY=min(DAY):max(DAY))
plotdatc$DAY2<-plotdatc$DAY^2
plotdatc$HAB<-"COFFEE"
preddatc<-predict(m,newdata=plotdatc,se.fit=TRUE)
with(preddatc,lines(min(DAY):max(DAY),exp(fit)/(1+exp(fit)),col="red",lwd=2))
with(preddatc,lines(min(DAY):max(DAY),exp(fit+1.96*se.fit)/(1+exp(fit+1.96*se.fit)),lty=2,col="red"))
with(preddatc,lines(min(DAY):max(DAY),exp(fit-1.96*se.fit)/(1+exp(fit-1.96*se.fit)),lty=2,col="red"))
plotdatf<-data.frame(DAY=min(DAY):max(DAY))
plotdatf$DAY2<-plotdatf$DAY^2
plotdatf$HAB<-"FOREST"
preddatf<-predict(m,newdata=plotdatf,se.fit=TRUE)
with(preddatf,lines(min(DAY):max(DAY),exp(fit)/(1+exp(fit)),col="black",lwd=2))
with(preddatf,lines(min(DAY):max(DAY),exp(fit+1.96*se.fit)/(1+exp(fit+1.96*se.fit)),lty=2))
with(preddatf,lines(min(DAY):max(DAY),exp(fit-1.96*se.fit)/(1+exp(fit-1.96*se.fit)),lty=2))
```

1.2. Habitat performance

1.2.1. Body condition

```
### Generalized linear models for mist-net data - BCI

## Part 1. Calculate Body Condition Index for each capture event following guidelines from
## Peig, J. & Green, A.J. (2009) New perspectives for estimating body condition from mass/length data: ## the scaled mass index as an alternative
method. Oikos 118:1883-1891.

library(smatr)
setwd("")
data<-read.table("",header=T) #data table with all measurements
d1<-data[!is.na(data$WC) & !is.na(data$BM),] #WC stands for wing chord, BM for body mass
attach(d1)

d2<-d1[d1$SP=="SSPP",] #select species
k<-3 #recommended value to exclude outliers that are probably mistakes but not extreme cases
QLS_WC<-quantile(d2$WC)
LQ_WC<-QLS_WC[[2]]
UQ_WC<-QLS_WC[[4]]
IQR_WC<-UQ_WC - LQ_WC
d3<-d2[d2$WC<(UQ_WC+(k*IQR_WC)) & d2$WC>(LQ_WC-(k*IQR_WC)),]
QLS_BM<-quantile(d3$BM)
LQ_BM<-QLS_BM[[2]]
UQ_BM<-QLS_BM[[4]]
IQR_BM<-UQ_BM - LQ_BM
d4<-d3[d3$BM<(UQ_BM+(k*IQR_BM)) & d3$BM>(LQ_BM-(k*IQR_BM)),] #These lines remove outliers from data

Mi.2<-log(d4$BM) #log of BM
Li.2<-log(d4$WC) #log of WC
Lo.2<-mean(Li.2)
sma_model.2<-sma(Mi.2~Li.2,method="SMA") #Standard major axis regression

M1.2<-sma_model.2$coef[[1]]
M2.2<-as.matrix(M1.2)
bSMA.2<-M2.2[2,1]
BCI.2<-Mi.2*((Lo.2/Li.2)^bSMA.2) #Apply formula suggested by Peig & Green 2009
t2<-cbind(d4,BCI.2) #Create a data vector with BCI values for each capture that had both WC and BM
write.table(t2,"",sep="\t")

## Part 2. Define generalized linear models and run model selection

library("bbmle")
library(AICcmodavg)
data<-read.table("t2.txt",header=T) #Output from previous step
attach(data)
DAY2<-DAY^2
DAYs<-scale(DAY)
DAY2s<-scale(DAY2)
BCIs<-scale(BCI.2)

M0<-lm(BCIs~1)
Mt<-lm(BCIs~DAYs+DAY2s)
Mh<-lm(BCIs~HAB)
Mth<-lm(BCIs~DAYs+DAY2s+HAB)
summary(M0)
summary(Mt)
summary(Mh)
summary(Mth)
Cand.mods <- list(M0,Mt,Mh,Mth)
modnames<-c("M0","Mt","Mh","Mth")
aictab(Cand.mods,modnames,sort=TRUE)
modavg(Cand.mods,parm="(Intercept)",modnames)
modavg(Cand.mods,parm="DAYs",modnames)
modavg(Cand.mods,parm="DAY2s",modnames)
modavg(Cand.mods,parm="HABFOREST",modnames)
```

```
## Part 3. Build graphs

DAY2<-DAY^2
m<-lm(BCI.2~DAY+DAY2+HAB)
plot(DAY,BCI.2,type="n",ylab="Body Condition Index",xlab="Day of year",main="TUFL-BCI",ylim=c(3.9,4.1))
x<-seq(min(DAY),max(DAY),1)
one<-rep("COFFEE",length(x))
two<-rep("FOREST",length(x))
y1<-predict(m,list(DAY=x,DAY2=x^2,HAB=one),interval=c("confidence"),level=0.95,type="response")
y2<-predict(m,list(DAY=x,DAY2=x^2,HAB=two),interval=c("confidence"),level=0.95,type="response")
lines(x,y1[,1],col="red",lwd=2)
lines(x,y1[,2],col="red",lty=2)
lines(x,y1[,3],col="red",lty=2)
lines(x,y2[,1],col="black",lwd=2)
lines(x,y2[,2],col="black",lty=2)
lines(x,y2[,3],col="black",lty=2)
```

1.2.2. Muscle

```
### Generalized linear models for mist-net data - MUSCLE

## Part 1. Get ready (and modify and define response)

library(AICcmodavg)
setwd("")
data<-read.table("",header=T)
d<-data[data$SP=="SSPP",] #Select species
d6<-d[d$MUS=="2" | d$MUS=="3",] #Subset to individuals with muscle score 2 and 3
d6<-d6[!is.na(d6$MUS),]
d6$MUS<-as.character(d6$MUS)
d6$MUS[d6$MUS=="2"] <- 0 # FAIL defined as individuals with medium muscle
d6$MUS[d6$MUS=="3"] <- 1 # SUCCESS defined as individuals with high muscle
d6$MUS<-as.numeric(d6$MUS)
d6$RESP<-d6$MUS
d6$HAB<-as.factor(d6$HAB)
d6$DAY2<-d6$DAY^2
d6$DAYs<-scale(d6$DAY)
d6$DAY2s<-scale(d6$DAY2)
attach(d6)

## Part 2. Define models and run model selection
############ Same code as for AGE

## Part 3. Build graphs
############ Same code as for AGE
```

1.2.3. Primary molt

```
### Generalized linear models for mist-net data – PP MOLT

## Part 1. Get ready (and modify and define response)

library(AICcmodavg)
setwd("")
data<-read.table("",header=T)
d<-data[data$SP=="SSPP",] #Select species
d4<-d[d$AGE=="3" | d$AGE=="4",] #Subset to only immature and adult individuals
d4<-d4[!is.na(d4$PLU),]
d4$PLU<-as.character(d4$PLU)
d4$PLU[d4$PLU=="2"] <- 0 # FAIL defined as individual not IN FULL PPMolting
d4$PLU[d4$PLU=="3"] <- 0 # FAIL defined as individual not IN FULL PPMolting
d4$PLU[d4$PLU=="5"] <- 0 # FAIL defined as individual not IN FULL PPMolting
d4$PLU[d4$PLU=="4"] <- 1 # SUCCESS defined as individual is IN FULL PPMolting
d4$PLU[d4$PLU=="6"] <- 1 # SUCCESS defined as individual is IN FULL PPMolting
d4$PLU<-as.numeric(d4$PLU)
```

```
d4$RESP<-d4$PLU
d4$HAB<-as.factor(d4$HAB)
d4$DAY2<-d4$DAY^2
d4$DAYs<-scale(d4$DAY)
d4$DAY2s<-scale(d4$DAY2)
attach(d4)

## Part 2. Define models and run model selection
############ Same code as for AGE

## Part 3. Build graphs
############ Same code as for AGE
```

1.2.4. Breeding

```
### Generalized linear models for mist-net data – BREEDING

## Part 1. Get ready (and modify and define response)

library(AICcmodavg)
setwd("")
data<-read.table("",header=T)
d<-data[data$SP=="SSPP",] #Select species
d3<-d[!is.na(d$BRE),]
d3$BRE<-as.character(d3$BRE)
d3$BRE[d3$BRE=="NO"] <- 0 # FAIL defined as individual not breeding
d3$BRE[d3$BRE=="YES"] <- 1 # SUCCESS defined as individual is breeding
d3$BRE<-as.numeric(d3$BRE)
d3$RESP<-d3$BRE
d3$HAB<-as.factor(d3$HAB)
d3$DAY2<-d3$DAY^2
d3$DAYs<-scale(d3$DAY)
d3$DAY2s<-scale(d3$DAY2)
attach(d3)

## Part 2. Define models and run model selection
############ Same code as for AGE

## Part 3. Build graphs
############ Same code as for AGE
```

1.2.5. Juveniles

```
### Generalized linear models for mist-net data – JUVENILES

## Part 1. Get ready (and modify and define response)

library(AICcmodavg)
setwd("")
data<-read.table("",header=T)
d<-data[data$SP=="SSPP",] #Select species
d5<-d[d$AGE!="0",] #Remove individuals with age unknown
d5$AGE<-as.character(d5$AGE)
d5$AGE[d5$AGE=="4"] <- 0 # FAIL defined as AHY individual
d5$AGE[d5$AGE=="3"] <- 0 # FAIL defined as AHY individual
d5$AGE[d5$AGE=="2"] <- 1 # SUCCESS defined as JUVENILE individual
d5$AGE<-as.numeric(d5$AGE)
d5$RESP<-d5$AGE
d5$HAB<-as.factor(d5$HAB)
d5$DAY2<-d5$DAY^2
d5$DAYs<-scale(d5$DAY)
d5$DAY2s<-scale(d5$DAY2)
attach(d5)

## Part 2. Define models and run model selection
############ Same code as for AGE
```

```
## Part 3. Build graphs
############ Same code as for AGE
```

2. Level 2: Composite indexes of habitat preference and performance

2.1. Habitat preference

No R code was used for this part of the analysis.

2.2. Habitat performance

```
### Meta-analysis of habitat performance

setwd("")
library("metafor")
data<-read.table("",header=T) #Read file with the mean and standard error for each species: variable
d<-data[data$SPP=="SSPP",] #Select species
d<-d[!is.na(d$MEAN),] #Remove NAs (analysis not carried out for a species)
x<-rma(yi=MEAN,sei=STE,data=d,method="FE",level=90);x
forest(x,slab=c("BCI","BRE","JUV","MUS","PLU"))
```

3. Level 3: Is habitat selection acting adaptively?

No R code was used for this part of the analysis.