Supplement 1

```
# *******Supplementary information: R code for Resource Selection
Function modelling for A. nigriceps activities by S. Dutta***********
## Loading data file and preliminary processing
data = read.csv("habdata.csv")
head(data)

## R script for descriptive statistics of microhabtiat variables

# z-standaradize seasonal variables by dry and wet season mean & sd
datDry = data[data$season=="Dry",c(1,8:16)]
datWet = data[data$season=="Wet",c(1,8:16)]
zdatDry = as.data.frame(scale(datDry[,-1], center=TRUE, scale=TRUE))
zdatWet = as.data.frame(scale(datWet[,-1], center=TRUE, scale=TRUE))
zdatDry$sn = datDry$sn
zdatWet$sn = datWet$sn
zdat = rbind(zdatDry,zdatWet)
zdatsort = zdat[order(zdat$sn),]
datcom = merge(data, zdatsort, by.x="sn", by.y="sn")           # binding
to main data
head(datcom)

# backtransform z-standardized variables by global mean & sd
datcom$br.gr.back = with(datcom, (br.gr.y * sd(br.gr.x) + mean(br.gr.x)))
datcom$sh.sw.back = with(datcom, (sh.sw.y * sd(sh.sw.x) + mean(sh.sw.x)))
datcom$md.sw.back = with(datcom, (md.sw.y * sd(md.sw.x) + mean(md.sw.x)))
datcom$tl.sw.back = with(datcom, (tl.sw.y * sd(tl.sw.x) + mean(tl.sw.x)))
datcom$vt.sw.back = with(datcom, (vt.sw.y * sd(vt.sw.x) + mean(vt.sw.x)))
datcom$ins = with(datcom, (inso.y * sd(inso.x) + mean(inso.x)))
datcom$frt = with(datcom, (frto.y * sd(frto.x) + mean(frto.x)))
datcom$grs.ht.back = with(datcom, (grs.ht.y * sd(grs.ht.x) +
mean(grs.ht.x)))
datcom$grs.cov.back = with(datcom, (grs.cov.y * sd(grs.cov.x) +
mean(grs.cov.x)))

# Fixing negative values as an artefact of backtransformation to 0 (min)
datcom$br.gr.back[datcom$br.gr.back < 0] = 0
datcom$sh.sw.back[datcom$sh.sw.back < 0] = 0
datcom$md.sw.back[datcom$md.sw.back < 0] = 0
datcom$tl.sw.back[datcom$tl.sw.back < 0] = 0
datcom$vt.sw.back[datcom$vt.sw.back < 0] = 0
datcom$ins[datcom$ins < 0] = 0
datcom$frt[datcom$frt < 0] = 0
datcom$grs.ht.back[datcom$grs.ht.back < 0] = 0
datcom$grs.cov.back[datcom$grs.cov.back < 0] = 0

# Estimate mean & 95% ci of variables across microsite types
names(datcom)
desc.dat = datcom[,c(17:23,33:41)]
dim = colnames(desc.dat)
loc = levels(datcom$loc_type)

# create dataframes to store output
```

```
mean = as.data.frame(matrix(0, nrow=length(dim), ncol=length(loc)))
sd = as.data.frame(matrix(0, nrow=length(dim), ncol=length(loc)))
n = as.data.frame(matrix(0, nrow=length(dim), ncol=length(loc)))
ci = as.data.frame(matrix(0, nrow=length(dim), ncol=length(loc)))

# add row and col names to dataframes
rowname = c("Shrub height", "Shrub cover", "Prosopis", "Zizyphus",
"Grazing intensity",
"Distance to road", "Distance to settlement","Bare ground", "Short
sward", "Moderate sward",
"Tall sward", "Very tall sward", "Insect count", "Fruit index", "Grass
height", "Grass cover")
colname = c("Breeding", "Display", "Foraging", "Others", "Random",
"Resting", "Roosting")
rownames(mean) = rowname
colnames(mean) = colname
rownames(sd) = rowname
colnames(sd) = colname
rownames(n) = rowname
colnames(n) = colname
rownames(ci) = rowname
colnames(ci) = colname

# start loop to estimate parameters for behavior-specific microhabitats
for(i in 1: length(dim)){
mean[i,] = tapply(desc.dat[,i],datcom$loc_type,FUN=mean)
sd[i,] = tapply(desc.dat[,i],datcom$loc_type,FUN=sd)
n[i,] = tapply(desc.dat[,i],datcom$loc_type,FUN=length)
}
ci = 1.96*(sd/sqrt(n))

# estimate parameters for general microhabitat
desc.dat.pr = datcom[datcom$pr==1,c(17:23,33:41)]
# create dataframes to store output
mean.o = vector("numeric", length=length(dim))
sd.o = vector("numeric", length=length(dim))
n.o = vector("numeric", length=length(dim))
ci.o = vector("numeric", length=length(dim))

# start loop to estimate parameters for general micro-habitat
for(o in 1: length(dim)){
mean.o[o] = mean(desc.dat.pr[,o])
sd.o[o] = sd(desc.dat.pr[,o])
}
ci.o = 1.96*(sd.o/sqrt(nrow(desc.dat.pr)))

# write results of mean & ci to a table
para = as.data.frame(matrix(0, nrow=length(dim), ncol=length(loc)))
rownames(para) = rowname
colnames(para) = colname
for(i in 1:length(dim)){
 for(j in 1:length(loc)){
  para[i,j] = paste(round(mean[i,j],2)," (",round(ci[i,j],2),")")
 }
```

```
}
para$Occurrence = 0
for(p in 1:length(dim)){
 para[p,8] = paste(round(mean.o[p],2)," (",round(ci.o[p],2),")")
}

para

# Compute relative frequency of land-covers by microsite types
count = table(datcom$loc_type, datcom$hab)
pcount = count/rowSums(count)
tcount = t(pcount)
tcount = round(tcount,2)

# Compute relative frequency of land-covers by microsite types
count.o = table(datcom$pr, datcom$hab)
pcount.o = count.o/rowSums(count.o)
tcount.o = t(pcount.o)
tcount.o = round(tcount.o,2)

write.csv(para, "microhab_est.csv") # print results
write.csv(tcount, "freq_lndcov.csv") # print results
write.csv(tcount.o, "freq_lndcovo.csv") # print results


## Checking multicollinearity among microhabitat variables
# Subset habitat data
names(datcom)
cordat = datcom[,c(17:23,33:41)]
names(cordat)
cordat1 =
cbind(cordat[,8],cordat[,9],cordat[,10],cordat[,11],cordat[,12],cordat[,1
5],cordat[,1],
cordat[,16],cordat[,2],cordat[,3],cordat[,4],cordat[,13],cordat[,14],cord
at[,5],cordat[,6],cordat[,7])
cordat1 = as.data.frame(cordat1)
colnames(cordat1) = corcol = c("Bare ground","Short sward","Moderate
sward","Tall sward",
"Very tall sward", "Grass height", "Shrub height","Grass cover","Shrub
cover","Prosopis",
"Zizyphus","Insect","Fruit","Grazing","Dist. road", "Dist. settlement")

# store rho & p-values of Spearman's correlation test
len = length(cordat1)
cor_out = as.data.frame(matrix(0, nrow=len, ncol=len))
rownames(cor_out) = corcol
colnames(cor_out) = corcol

# start loop
for(r in 1:len){
 for(c in 1:len){
  corr = cor.test(cordat1[,r],cordat1[,c], method="spearman")
  cor_out[c,r] = corr$p.value
  cor_out[r,c] = corr$estimate
```

```
 }
}
cor_out1 = round(cor_out,2)

# print results
write.csv(t(cor_out1), "microhab_corr.csv")


## Factor analysis to extract synthetic uncorrelated vegetation
components
# Prepare data
names(datcom)
datcom$z.sh.ht = scale(datcom$shr.ht, center=TRUE, scale=TRUE)
datcom$z.sh.cov = scale(datcom$shr.cov, center=TRUE, scale=TRUE)
facdat = datcom[,c(24:28,31,32,42,43)]

# Run factor analysis & store scores in data (veg1-3)
fact = factanal(facdat, factors=4, scores="regression", rotation="none")
fact$loadings
datcom$veg1.raw = fact$scores[,1]
datcom$veg2.raw = fact$scores[,2]
datcom$veg3.raw = fact$scores[,3]
datcom$veg4.raw = fact$scores[,4]
datcom$veg1 = datcom$veg1.raw - min(datcom$veg1.raw)
datcom$veg2 = datcom$veg2.raw - min(datcom$veg2.raw)
datcom$veg3 = datcom$veg3.raw - min(datcom$veg3.raw)
datcom$veg4 = datcom$veg4.raw - min(datcom$veg4.raw)

factload = ifelse(abs(fact$loadings) <= 0.3, 0, round(fact$loadings,2))
factload
write.csv(factload, "factor.csv")


## Micro-habitat distributions across used and available sites using
boxplots
# Reading files
names(datcom)
dat = datcom[,c(1:7,48:51,38:39,19,20,21:23)]
head(dat)

# Code location types
dat$loc = 0
dat$loc[dat$loc_type=="Random"] = "A"
dat$loc[dat$loc_type=="Breeding"] = "B"
dat$loc[dat$loc_type=="Display"] = "D"
dat$loc[dat$loc_type=="Foraging"] = "F"
dat$loc[dat$loc_type=="Resting"] = "Rd"
dat$loc[dat$loc_type=="Roosting"] = "Rn"
dat1 = dat[dat$loc != 0,]

# plot box-plots
cairo_pdf("Fig2.pdf", width=28/2.54, height=21/2.54, pointsize=8)

par(mfrow=c(3,4))
```

```
par(oma=c(2.5,2.5,0,0))
par(mar=c(1.2,2,1.2,3))#MARGINS
par(mgp=c(5, 0.6, 0))
par(las=0,tck=0,ps=12)

boxplot(veg1~loc, dat1,
col=c(0,"chartreuse","chartreuse1","chartreuse2","chartreuse3","chartreus
e4"), varwidth=TRUE, notch=TRUE, outline=FALSE,
names=c("","","","","","",""), pars=list(boxwex = 1, staplewex = 0.3, outwex
= 0.5), ylim=c(0,5))
mtext(side=2, line=2.2, "Sward biomass", cex=0.8, font=2)

boxplot(veg2~loc, dat1,
col=c(0,"darkolivegreen1","darkolivegreen2","darkolivegreen3","darkoliveg
reen4", "darkolivegreen"), varwidth=TRUE, notch=TRUE, outline=FALSE,
names=c("","","","","","",""), pars=list(boxwex = 1, staplewex = 0.3, outwex
= 0.5), ylim=c(0,5))
mtext(side=2, line=2.2, "Tall vs. short sward", cex=0.8, font=2)

boxplot(veg3~loc, dat1,
col=c(0,"palegreen","palegreen1","palegreen2","palegreen3","palegreen4"),
varwidth=TRUE, notch=TRUE, outline=FALSE, names=c("","","","","","",""),
pars=list(boxwex = 1, staplewex = 0.3, outwex = 0.5), ylim=c(0,4))
mtext(side=2, line=2.2, "Shrubiness", cex=0.8, font=2)

boxplot(veg4~loc, dat1,
col=c(0,"azure","azure1","azure2","azure3","azure4"), varwidth=TRUE,
notch=TRUE, outline=FALSE, names=c("","","","","","",""), pars=list(boxwex =
1, staplewex = 0.3, outwex = 0.5), ylim=c(2,6))
mtext(side=2, line=2.2, "Moderate vs. tall sward", cex=0.8, font=2)

boxplot(grz~loc, dat1,
col=c(0,"bisque","bisque1","bisque2","bisque3","bisque4"), varwidth=TRUE,
notch=TRUE, outline=FALSE, names=c("","","","","","",""), pars=list(boxwex =
1, staplewex = 0.3, outwex = 0.5), ylim=c(0,4))
mtext(side=2, line=2.2, "Grazing intensity", cex=0.8, font=2)

boxplot(dvil~loc, dat1,
col=c(0,"coral","coral1","coral2","coral3","coral4"), varwidth=TRUE,
notch=TRUE, outline=FALSE, names=c("","","","","","",""), pars=list(boxwex =
1, staplewex = 0.3, outwex = 0.5), ylim=c(0,4))
mtext(side=2, line=2.2, "Distance to settlement", cex=0.8, font=2)

boxplot(drd~loc, dat1,
col=c(0,"brown1","brown2","brown3","brown","brown4"), varwidth=TRUE,
notch=TRUE, outline=FALSE, names=c("","","","","","",""), pars=list(boxwex =
1, staplewex = 0.3, outwex = 0.5), ylim=c(0,4))
mtext(side=2, line=2.2, "Distance to road", cex=0.8, font=2)

boxplot(pro~loc, dat1, col=c(0,"pink","pink1","pink2","pink3","pink4"),
varwidth=TRUE, notch=TRUE, outline=FALSE,
names=c("A","B","D","F","Rd","Rn"), pars=list(boxwex = 1, staplewex =
0.3, outwex = 0.5), ylim=c(0,4))
mtext(side=2, line=2.2, "Prosopis dominance", cex=0.8, font=2)
```

```
boxplot(ins~loc, dat1, col=c(0,"cyan","cyan1","cyan2","cyan3","cyan4"),
varwidth=TRUE, notch=TRUE, outline=FALSE,
names=c("A","B","D","F","Rd","Rn"), pars=list(boxwex = 1, staplewex =
0.3, outwex = 0.5), ylim=c(0,40))
mtext(side=2, line=2.2, "Insect availability", cex=0.8, font=2)

boxplot(frt~loc, dat1,
col=c(0,"deepskyblue","deepskyblue1","deepskyblue2","deepskyblue3","deeps
kyblue4"), varwidth=TRUE, notch=TRUE, outline=FALSE,
names=c("A","B","D","F","Rd","Rn"), pars=list(boxwex = 1, staplewex =
0.3, outwex = 0.5), ylim=c(0,4))
mtext(side=2, line=2.2, "Fruit availability", cex=0.8, font=2)

boxplot(ziz~loc, dat1,
col=c(0,"slateblue1","slateblue2","slateblue3","slateblue","slateblue4"),
varwidth=TRUE, notch=TRUE, outline=FALSE,
names=c("A","B","D","F","Rd","Rn"), pars=list(boxwex = 1, staplewex =
0.3, outwex = 0.5), ylim=c(0,4))
mtext(side=2, line=2.2, "Zizyphus dominance", cex=0.8, font=2)

count = table(dat1$loc, dat1$hab)
pcount = count/rowSums(count)
tcount = t(pcount)

par(mar=c(1.2,2,3.5,3))#MARGINS
barplot(tcount, col=c("royalblue","yellowgreen","gold","sienna"),
legend.text=rownames(tcount), args.legend=list(x=9,y=1.09, bty="n",
horiz=T), border="black", beside=FALSE)
# ("topright", inset=c(-0.2,0), legend=rownames(tcount))
mtext(side=2, line=2.2, "Frequency occurrence", cex=0.8, font=2)

dev.off() # to print as pdf



# ********* Start of Resource Selection Function modelling for specific
uses ************ #

## Checking multicollinearity in habitat data prior to modeling
library(fmsb)
names(dat)
datcor = dat[,8:17]
cor = round(cor(datcor,method="spearman"),2)
cor ##no variable pair is strongly correlated

# Only run to subsample available micro-habitat data!!
s = dat[dat$loc=="A",]
s$sel = 1
for(sr in 1:nrow(s)){
s$sel[sr] = ifelse(runif(1)>=0.5,0,1)}
head(s)
write.csv(s, "modeldata.csv") # only run to save the subsampling for
future calls
```

```
s = read.csv("modeldata.csv") # subsampled data used for modeling;
results may vary between data subsamples
# Subsetting data
foraging = rbind(s[s$sel==1,-c(1,21)],dat[dat$loc_type=="Foraging",])
resting = rbind(s[s$sel==1,-c(1,21)],dat[dat$loc_type=="Resting",])
roosting = rbind(s[s$sel==1,-c(1,21)],dat[dat$loc_type=="Roosting",])
displ = rbind(s[s$sel==1,-c(1,21)],dat[dat$loc_type=="Display",])
breed = rbind(s[s$sel==1,-c(1,21)],dat[dat$loc_type=="Breeding",])

library(fmsb)
library(car)
library(MuMIn)

## Modelling RSF for foraging use
for_full =
glm(pr~hab+veg1+I(veg1^2)+veg2+veg3+pro+frt+ziz+ins+grz+I(grz^2)+dvil+drd
, binomial, foraging)
for_null = glm(pr~1, binomial, foraging)
summary(for_full)
vif(for_full)
NagelkerkeR2(for_full)
anova(for_null, for_full, test="Chisq")

# Candidate models
for_hab = glm(pr~hab, binomial, foraging)
for_veg1l = glm(pr~veg1, binomial, foraging)
for_veg1q = glm(pr~veg1+I(veg1^2), binomial, foraging)
for_veg2l = glm(pr~veg2, binomial, foraging)
for_veg3l = glm(pr~veg3, binomial, foraging)
for_veg1 = glm(pr~veg1+I(veg1^2)+veg2+veg3, binomial, foraging)
for_veg2 = glm(pr~veg1+I(veg1^2)+veg3, binomial, foraging)
for_pro = glm(pr~pro, binomial, foraging)
for_grz1 = glm(pr~grz, binomial, foraging)
for_grz2 = glm(pr~grz+I(grz^2), binomial, foraging)
for_res1 = glm(pr~frt, binomial, foraging)
for_res2 = glm(pr~frt+ziz, binomial, foraging)
for_res3 = glm(pr~ins, binomial, foraging)
for_res4 = glm(pr~frt+ziz+ins, binomial, foraging)
for_dstb1 = glm(pr~dvil, binomial, foraging)
for_dstb2 = glm(pr~drd, binomial, foraging)
for_dstb3 = glm(pr~drd+dvil, binomial, foraging)
for_hab_res = glm(pr~hab+frt+ziz, binomial, foraging)
for_veg_res1 = glm(pr~veg1+I(veg1^2)+veg3+frt+ziz, binomial, foraging)
for_veg_res2 = glm(pr~veg1+I(veg1^2)+pro+frt+ziz, binomial, foraging)
for_hab_veg1 = glm(pr~hab+veg1+I(veg1^2)+veg3, binomial, foraging)
for_hab_veg2 = glm(pr~hab+veg1+I(veg1^2)+pro, binomial, foraging)
for_hab_veg_res1 = glm(pr~hab+veg1+I(veg1^2)+veg3+frt+ziz, binomial,
foraging)
for_hab_veg_res2 = glm(pr~hab+veg1+I(veg1^2)+pro+frt+ziz, binomial,
foraging)
for_hab_res_grz = glm(pr~hab+frt+ziz+grz+I(grz^2), binomial, foraging)
for_hab_veg_res_grz1 = glm(pr~hab+veg3+frt+ziz+grz+I(grz^2), binomial,
foraging)
```

```
for_hab_veg_res_grz2 = glm(pr~hab+pro+frt+ziz+grz+I(grz^2), binomial,
foraging)
for_hab_veg_res_grz3 =
glm(pr~hab+veg1+I(veg1^2)+veg3+frt+ziz+grz+I(grz^2), binomial, foraging)
for_hab_veg_res_grz4 =
glm(pr~hab+veg1+I(veg1^2)+pro+frt+ziz+grz+I(grz^2), binomial, foraging)
for_hab_veg_res_grz_dstb1 = glm(pr~hab+veg3+frt+ziz+grz+I(grz^2)+dvil,
binomial, foraging)
for_hab_veg_res_grz_dstb2 = glm(pr~hab+pro+frt+ziz+grz+I(grz^2)+dvil,
binomial, foraging)
for_hab_veg_res_grz_dstb3 =
glm(pr~hab+veg1+I(veg1^2)+veg3+frt+ziz+grz+I(grz^2)+dvil, binomial,
foraging)
for_hab_veg_res_grz_dstb4 =
glm(pr~hab+veg1+I(veg1^2)+pro+frt+ziz+grz+I(grz^2)+dvil, binomial,
foraging)
for_hab_veg_res_dstb1 = glm(pr~hab+veg1+I(veg1^2)+veg3+frt+ziz+dvil,
binomial, foraging)
for_hab_veg_res_dstb2 = glm(pr~hab+veg1+I(veg1^2)+pro+frt+ziz+dvil,
binomial, foraging)

# Model selection & parameter estimates
rank_for =
model.sel(for_full,for_null,for_hab,for_veg1l,for_veg1q,for_veg2l,for_veg
3l,for_veg1,for_veg2,for_pro,

for_grz1,for_grz2,for_res1,for_res2,for_res3,for_res4,for_dstb1,for_dstb2
,for_dstb3,for_hab_res,for_veg_res1,for_veg_res2,

for_hab_veg1,for_hab_veg2,for_hab_veg_res1,for_hab_veg_res2,for_hab_res_g
rz,for_hab_veg_res_grz1,for_hab_veg_res_grz2,

for_hab_veg_res_grz3,for_hab_veg_res_grz4,for_hab_veg_res_grz_dstb1,for_h
ab_veg_res_grz_dstb2,for_hab_veg_res_grz_dstb3,
 for_hab_veg_res_grz_dstb4,for_hab_veg_res_dstb1,for_hab_veg_res_dstb2)
modrank_for = as.data.frame(rank_for)

write.csv(modrank_for,"mod_for.csv")

list(rownames(modrank_for[modrank_for$delta < 2,]))
topfor = for_hab_veg_res_grz_dstb2    # most common top model(s) based on
bootstrap - change model name if this model(s) is not returned as top
candidate
topfor

# Storing rsf values following Manly et al (2002)
coef(topfor)
habcoef1 = ifelse(foraging$hab=="Agro-
veg",coef(topfor)[2],ifelse(foraging$hab=="Grassland",coef(topfor)[3],
           ifelse(foraging$hab=="Scrub",coef(topfor)[4],0)))
foraging$rsf_for = exp(habcoef1 + coef(topfor)[5]*foraging$pro +
coef(topfor)[6]*foraging$frt + coef(topfor)[7]*foraging$ziz
 + coef(topfor)[8]*foraging$grz + coef(topfor)[9]*foraging$grz^2 +
coef(topfor)[10]*foraging$dvil)
```

```
habcoef2 = ifelse(s$hab=="Agro-
veg",coef(topfor)[2],ifelse(s$hab=="Grassland",coef(topfor)[3],
           ifelse(s$hab=="Scrub",coef(topfor)[4],0)))
s$rsf_for = exp(habcoef2 + coef(topfor)[5]*s$pro + coef(topfor)[6]*s$frt
+ coef(topfor)[7]*s$ziz
 + coef(topfor)[8]*s$grz + coef(topfor)[9]*s$grz^2 +
coef(topfor)[10]*s$dvil)

head(foraging)
head(s)

## End of RSF modelling for foraging use ##

## Modelling RSF for resting use
rest_full = glm(pr~veg1+veg3+dvil, binomial, resting)
rest_null = glm(pr~1, binomial, resting)
summary(rest_full)
vif(rest_full)
NagelkerkeR2(rest_full) # poor fit of global model
anova(rest_null, rest_full, test="Chisq") # poor fit of global model

# Candidate models
rest_veg1 = glm(pr~veg1, binomial, resting)
rest_veg2 = glm(pr~veg3, binomial, resting)
rest_veg3 = glm(pr~veg1+veg3, binomial, resting)
rest_dstb = glm(pr~dvil, binomial, resting)
rest_veg_dstb1 = glm(pr~veg1+dvil, binomial, resting)

# Model selection & parameter estimates
modrank_rest = model.sel(rest_full, rest_null, rest_veg1, rest_veg2,
rest_veg3,
 rest_dstb, rest_veg_dstb1)

modrank_rest = as.data.frame(rank_rest)
write.csv(modrank_rest, "mod_rest.csv")

candrest = list(rownames(modrank_rest[modrank_rest$delta < 2,]))
candrest
toprest = model.avg(rest_veg1,rest_veg_dstb1,rest_veg3)
summary(toprest)
# since resting RSF lacked good fit to data, ignoring resting rsf
predictions

## End of RSF modelling for resting use ##

## Modelling RSF for roosting use
roost_full = glm(pr~hab+veg1+veg2+pro+dvil, binomial, roosting)
roost_null = glm(pr~1, binomial, roosting)
summary(roost_full)
vif(roost_full)
NagelkerkeR2(roost_full)
anova(roost_null, roost_full, test="Chisq")
```

```
# Candidate models
roost_hab = glm(pr~hab, binomial, roosting)
roost_veg1 = glm(pr~veg1, binomial, roosting)
roost_veg2 = glm(pr~veg2, binomial, roosting)
roost_veg3 = glm(pr~pro, binomial, roosting)
roost_veg4 = glm(pr~veg1+veg2+pro, binomial, roosting)
roost_veg5 = glm(pr~veg2+pro, binomial, roosting)
roost_dstb = glm(pr~dvil, binomial, roosting)
roost_hab_veg = glm(pr~hab+veg1+veg2+pro, binomial, roosting)
roost_veg_dstb1 = glm(pr~veg1+veg2+dvil, binomial, roosting)
roost_veg_dstb2 = glm(pr~veg1+veg2+pro+dvil, binomial, roosting)

# Model selection & parameter estimates
rank_roost =
model.sel(roost_full,roost_null,roost_hab,roost_veg1,roost_veg2,

roost_veg3,roost_veg4,roost_veg5,roost_dstb,roost_hab_veg,roost_veg_dstb1
,roost_veg_dstb2)
modrank_roost = as.data.frame(rank_roost)
write.csv(modrank_roost,"mod_roost.csv")
list(rownames(modrank_roost[modrank_roost$delta < 2,]))
toproost = model.avg(roost_veg4,roost_veg5,roost_veg_dstb2)  # most
common top model(s) based on bootstrap - change model name if this
model(s) is not returned as top candidate

# Storing rsf values following Manly et al (2002)
coef(toproost)
roosting$rsf_roost = exp(coef(toproost)[2]*roosting$veg1 +
coef(toproost)[3]*roosting$veg2
 + coef(toproost)[4]*roosting$pro + coef(toproost)[5]*roosting$dvil)
s$rsf_roost = exp(coef(toproost)[2]*s$veg1 + coef(toproost)[3]*s$veg2
 + coef(toproost)[4]*s$pro + + coef(toproost)[5]*s$dvil)

head(roosting)
head(s)

## End of RSF modelling for roosting use ##

## RSF modeling for display use ##
displ$hab1 = ifelse(displ$hab == "Grassland", 1, 0)
disp_full = glm(pr~hab1+veg2+dvil, binomial, displ)
disp_null = glm(pr~1, binomial, displ)
summary(disp_full)
vif(disp_full)
NagelkerkeR2(disp_full)
anova(disp_full, disp_null, test="Chisq")

# Candidate models
disp_hab = glm(pr~hab1, binomial, displ)
disp_veg = glm(pr~veg2, binomial, displ)
disp_hab_veg = glm(pr~hab1+veg2, binomial, displ)
disp_hab_dstb = glm(pr~hab1+dvil, binomial, displ)
disp_veg_dstb = glm(pr~veg2+dvil, binomial, displ)
```

```
# Model selection & parameter estimates
rank_disp = model.sel(disp_full,disp_null,disp_hab,disp_veg,
 disp_hab_veg,disp_hab_dstb,disp_veg_dstb)
modrank_disp = as.data.frame(rank_disp)
write.csv(modrank_disp,"mod_disp.csv")
list(rownames(modrank_disp[modrank_disp$delta < 2,]))
topdisp = disp_full   # most common top model(s) based on bootstrap -
change model name if this model(s) is not returned as top candidate

# Storing rsf values following Manly et al (2002)
coef(topdisp)
disp_coef1 = ifelse(displ$hab=="Grassland",coef(topdisp)[2],0)
displ$rsf_disp = exp(disp_coef1 + coef(topdisp)[3]*displ$veg2 +
coef(topdisp)[4]*displ$dvil)

disp_coef2 = ifelse(s$hab=="Grassland",coef(topdisp)[2],0)
s$rsf_disp = exp(disp_coef2 + coef(topdisp)[3]*s$veg2 +
coef(topdisp)[4]*s$dvil)

head(displ)
head(s)
## End of RSF modeling for display use ##


## RSF modeling for nesting use ##
breed$hab1 = ifelse(breed$hab == "Grassland", 1, 0)
breed_full = glm(pr~hab1+veg2+ins, binomial, breed)
breed_null = glm(pr~1, binomial, breed)
summary(breed_full)
vif(breed_full)
NagelkerkeR2(breed_full)
anova(breed_full, breed_null, test="Chisq")

# Candidate models
breed_hab = glm(pr~hab1, binomial, breed)
breed_veg = glm(pr~veg2, binomial, breed)
breed_res = glm(pr~ins, binomial, breed)
breed_hab_veg = glm(pr~hab1+veg2, binomial, breed)
breed_hab_res = glm(pr~hab1+ins, binomial, breed)
breed_veg_res = glm(pr~veg2+ins, binomial, breed)

# Model selection & parameter estimates
rank_breed =
model.sel(breed_full,breed_null,breed_hab,breed_veg,breed_res,
 breed_hab_veg, breed_hab_res, breed_veg_res)
modrank_breed = as.data.frame(rank_breed)
write.csv(modrank_breed,"mod_breed.csv")
list(rownames(modrank_breed[modrank_breed$delta < 2,]))
topbreed = breed_full  # most common top model based on bootstrap -
change model name if this model(s) is not returned as top candidate

# Storing rsf values following Manly et al (2002)
coef(topbreed)
breed_coef1 = ifelse(breed$hab=="Grassland",coef(topbreed)[2],0)
```

```
breed$rsf_breed = exp(breed_coef1 + coef(topbreed)[3]*breed$veg2 +
coef(topbreed)[4]*breed$ins)

breed_coef2 = ifelse(s$hab=="Grassland",coef(topbreed)[2],0)
s$rsf_breed = exp(breed_coef2 + coef(topbreed)[3]*s$veg2 +
coef(topbreed)[4]*s$ins)

head(breed)
head(s)
## End of RSF modeling for breeding use ##


## Collatig RSF values for behaviors at available sites
for.rsf.q1 = quantile(foraging$rsf_for[foraging$pr==1],0.25)
roost.rsf.q1 = quantile(roosting$rsf_roost[roosting$pr==1],0.25)
disp.rsf.q1 = quantile(displ$rsf_disp[displ$pr==1],0.25)
breed.rsf.q1 = quantile(breed$rsf_breed[breed$pr==1],0.25)

s$res_forl = ifelse(s$rsf_for >= for.rsf.q1,1,0)
s$res_roostl = ifelse(s$rsf_roost >= roost.rsf.q1,1,0)
s$res_displ = ifelse(s$rsf_disp >= disp.rsf.q1,1,0)
s$res_breedl = ifelse(s$rsf_breed >= breed.rsf.q1,1,0)
head(s)

# ********* End of Resource Selection Function modelling ************ #

# ********* Start of Resource Selection Function modelling for general
use ************ #
library(glmulti)
names(s)
names(dat)

u = dat[dat$pr==1,]
u1 = u[u$loc_type!="Others",]           # remove 'others' use-type from
data
summary(u1)
used = rbind(s[s$sel==1,2:20],u1[,1:19])

# global model
use_full =
glm(pr~hab+veg1+I(veg1^2)+veg2+veg3+pro+ziz+frt+grz+I(grz^2)+dvil,
binomial, used)
use_null = glm(pr~1, binomial, used)
anova(use_null, use_full, test="Chisq")
summary(use_full)
NagelkerkeR2(use_full)
vif(use_full)
par(mfrow=c(2,2))
plot(use_full)

# Automated model selection using "glmulti" to choose top candidate
models explaining general occurrence
moduseulti =
glmulti(pr~hab+veg1+I(veg1^2)+veg2+veg3+pro+ziz+frt+grz+I(grz^2)+dvil,
```

```
 data = used, level=1, crit=aicc, method="g")
plot(moduseulti, type = "p")
importances = summary(moduseulti)$modelweights
modusecoef = coef(moduseulti, select=1.99, varweighting="Buckland")
write.csv(modusecoef, "modusecoef.csv")

# Storing & printing rsf values following Manly et al (2002)
modusecoef
habcoef_use1 = ifelse(used$hab=="Agro-
veg",modusecoef[10],ifelse(used$hab=="Grassland",modusecoef[11],
ifelse(used$hab=="Scrub",modusecoef[12],0)))
used$rsf_use = exp(habcoef_use1 + modusecoef[1]*used$veg1^2 +
modusecoef[2]*used$veg1 + modusecoef[3]*used$grz +
modusecoef[4]*used$dvil
 + modusecoef[5]*used$grz^2 + modusecoef[6]*used$veg2 +
modusecoef[7]*used$frt + modusecoef[8]*used$ziz)
head(used)

habcoef_use2 = ifelse(s$hab=="Agro-
veg",modusecoef[10],ifelse(s$hab=="Grassland",modusecoef[11],
ifelse(s$hab=="Scrub",modusecoef[12],0)))
s$rsf_use = exp(habcoef_use2 + modusecoef[1]*s$veg1^2 +
modusecoef[2]*s$veg1 + modusecoef[3]*s$grz + modusecoef[4]*s$dvil
 + modusecoef[5]*s$grz^2 + modusecoef[6]*s$veg2 + modusecoef[7]*s$frt +
modusecoef[8]*s$ziz)
head(s)

use.rsf.q1 = quantile(used$rsf_use[used$pr==1],0.25)
s$res_usel = ifelse(s$rsf_use >= use.rsf.q1,1,0)

## Correlation between behaviorally explicit resource selection values at
micro-sites
head(s)
pair = s[,c("rsf_for","rsf_roost","rsf_disp","rsf_breed", "rsf_use")]
cor.rsf = cor(pair, method="pearson")
round(cor.rsf,2)

# plot behavioral RSFs against general use RSF
colnames(pair) = c("Forage","Roost","Display","Nest","Use")

cairo_pdf("Fig4.pdf", width=14/2.54, height=11/2.54, pointsize=8)
par(mfrow=c(1,1))
par(oma=c(2.5,2.5,0,0))
par(mar=c(1.2,2,1.2,3))#MARGINS
par(mgp=c(5, 0.6, 0))
par(las=0,tck=0,ps=12)
pairs(pair)

#Plot titles
title(ylab="Fitted RSF values",outer=T,line=0.9,cex.lab=1.1)
title(xlab="Fitted RSF values",outer=T,line=1.3,cex.lab=1.1)

dev.off()
```

```
s$max = pmax(s$res_forl,s$res_roostl,s$res_displ,s$res_breedl)

# Storing RSF values from behavior and general-use models
write.csv(s, "RSFvalues.csv")

# Estimate availability of behaviorally explicit resource units based on
binomial GLMs
#Function to back transform LOGIT values from binomial GLMs
logitback = function (x) {
  1/(1+exp(-x))
}#End logitback

# Function to directly calculate SE from General Linear Model Object,
library(msm)
delta.2<- function (x) {
  deltamethod(~1/(1+exp(-x1)),
              coef(summary(x))[1],
              vcov(x)[1,1])
}#End delta.2

# Run binomial GLMs
prb.for = glm(res_forl ~ 1, binomial, s)
mu.for = round(logitback(coef(summary(prb.for))[1]),digits=2)
ci.for = round(delta.2(prb.for)*1.96,digits=2)

prb.roost = glm(res_roostl ~ 1, binomial, s)
mu.roost = round(logitback(coef(summary(prb.roost))[1]),digits=2)
ci.roost = round(delta.2(prb.roost)*1.96,digits=2)

prb.disp = glm(res_displ ~ 1, binomial, s)
mu.disp = round(logitback(coef(summary(prb.disp))[1]),digits=2)
ci.disp = round(delta.2(prb.disp)*1.96,digits=2)

prb.breed = glm(res_breedl ~ 1, binomial, s)
mu.breed = round(logitback(coef(summary(prb.breed))[1]),digits=2)
ci.breed = round(delta.2(prb.breed)*1.96,digits=2)

prb.use = glm(res_usel ~ 1, binomial, s)
mu.use = round(logitback(coef(summary(prb.use))[1]),digits=2)
ci.use = round(delta.2(prb.use)*1.96,digits=2)

# Collate mean & 95% ci of resource availabilities
res_avl = matrix(0,nrow=5, ncol=2)
rownames(res_avl) = c("Foraging","Roosting","Display","Nesting","Use")
colnames(res_avl) = c("Mean", "CI")
res_avl[1,1] = mu.for
res_avl[1,2] = ci.for
res_avl[2,1] = mu.roost
res_avl[2,2] = ci.roost
res_avl[3,1] = mu.disp
res_avl[3,2] = ci.disp
res_avl[4,1] = mu.breed
res_avl[4,2] = ci.breed
res_avl[5,1] = mu.use
```

```
res_avl[5,2] = ci.use
res_avl     # report availability of behaviorally explicit resource units

# ********* End of Resource Selection Function modelling for general use
************ #

## ---- Examining the relationship between GIB usage and resource
availability ---- ##

# Transect data
trn = read.csv("tr_rsf_gib.csv")
head(trn)
trn$sch = trn$h/max(trn$h)

trnrsf0 = lm(gib_den ~ 1, trn)
trnrsf1 = lm(gib_den ~ max, trn)
trnrsf2 = lm(gib_den ~ I(max*sch), trn)
trnrsf3 = lm(gib_den ~ res_usel, trn)

model.sel(trnrsf0,trnrsf1,trnrsf2,trnrsf3)
summary(trnrsf3)
par(mfrow=c(2,2))
plot(trnrsf3)

# Plotting space-use vs. resource availability graphs
cairo_pdf("Fig3.pdf", width=14/2.54, height=8/2.54, pointsize=8)

par(mfrow=c(1,1))
par(oma=c(2.5,2.5,0,0))
par(mar=c(1.2,2,1.2,3))#MARGINS
par(mgp=c(5, 0.6, 0))
par(las=0,tck=0,ps=12)

plot(trn$gib_den ~ I(trn$max*trn$sch), pch=16, cex=1.1)
abline(a=coef(trnrsf3)[1], b=coef(trnrsf3)[2], col="gray80", lwd=1,
lty=1)
axis(1,labels=T)
axis(2,labels=F)

#Plot titles
title(ylab="Bird density",outer=T,line=0.9,cex.lab=1.1)
title(xlab="Resource availability x
heterogeneity",outer=T,line=1.3,cex.lab=1.1)

dev.off()

## ---- End of examining the relationship between GIB usage and resource
availability ---- ##
```