JAE R Code

Adam Kane & Corinne J. Kendall

6 February 2017

Cue from above: Understanding how mammalian scavengers use information from avian scavengers

Summarise Data

Clear everything and load in data

```
rm(list=ls())
setwd("")
mydata<-read.csv("arrivalDataJAE.csv", header = T, sep = ",")```
```

Count the number of times each species arrived to a carcass over the whole dataset

```
speciesSuccess<-colSums(mydata>0); speciesSuccess
```

Find out which species arrived first at each carcass

```
sapply(mydata, function(x) which(x==1))
```

Who follows who?

```
mydata[mydata==0]<-100
sapply(colnames(mydata),function(v1){
  return(sapply(colnames(mydata),
                function(v2){
                  return(sum(mydata[,v1]-mydata[,v2] == 1))
                }))
})
```

Probability of discovering a carcass (Jackson et al 2008)

```
probLocate <- function(M,r,v,t,pi,A){
  (M*((2*r*v*t) + (pi*r^2)))/A
}

probLocate(1,12,45,6,pi,1530) # vulture
probLocate(1,1,10,6,pi,1530) # mammal
```

Kendall coefficient of concordance for vague data

```
# Create the function
vagueKendall<-function(u,v,n){
  u<-as.matrix (u)
  v<-as.matrix (v)
  # divide by the number of columns - 1
```

```
  uColNo<-u/(n-1)
  vColNo<-v/(n-1)
  # take the averages
  colMeansU<-colMeans(uColNo)
  colMeansV<-colMeans(vColNo)
  # measure the distances from the averages
  au = (colMeansU - 1/2)^2
  av = (colMeansV - 1/2)^2
  # calculate component before sum
  outside<-6*(n-1)/(n*(n+1))
  # sum of squared distances from averages
  sumSqdDiff<-sum(au+av)
  # The product of these gives the modified Kendall's W
  W<-outside*sum(au+av)
  return(W)
}
```

Load in the arrival order data and apply the function

```
kendalldata<-read.csv("orderedDataJAE.csv", header = T, sep = ",")
# transform into a matrix
M<-as.matrix(kendalldata)
# check the dimensions of the matrix with:
dim(M)
n<-6 # number of columns
k<-73 # number of rows
# Clean up the matrix to create the worse element matrix u and better element matrix v
M[M==0]<-NA;apply(M,1,rank,na.last="keep")
rowSums(M>0, na.rm = TRUE)
sumsOfRows<-rowSums(M>0, na.rm = TRUE)
u<-abs(sweep(M,MARGIN=1,sumsOfRows,`-`))
v<-abs(sweep(M,MARGIN=1,1,`-`))
u[is.na(u)] <- 0
v[is.na(v)] <- 0
# apply our functions to get a value W for concordance
vagueKendall(u,v,n)
```

Visual acuity of scavengers

```
library(aspace)

# Gyps fulvus measured at 104 cycles per degree (Fischer 1968)
target<-2
distance<-12000
target/distance
atan_d(target/distance) # degrees per cycle
# To get cycles per degree, divide one by that number.
1/atan_d(target/distance) # cycles per degree
```

Code to analyse NetLogo arrival times

Set your working directory to the folder where you have the csv files from the NetLogo model. We group all of these output files into one new file called arrivals.

```
# select the variables of interest that we want to keep hold of
myvars <- c("who", "breed", "got.here", "mycarcass")
# loop over the output files, clean them up and stitch them together in a new csv = "arrivals.csv"
out.file<-""
filenames <- dir(pattern =".csv")
lst <- vector("list", length(filenames ))

for (i in 1:length(filenames)) {
  tmp.file <- read.csv(filenames[i],header=TRUE, sep=",", skip=12, stringsAsFactors=FALSE) ##
  # tmp.file<-head(tmp.file, -451) ## trim off the useless data from the end
  # alternative way to do so by only keeping the rows with useful data
  tmp.file<-tmp.file[tmp.file$hidden. == "false", ]
  tmp.file<- tmp.file[myvars]
  tmp.file<-tmp.file[with(tmp.file, order(mycarcass, got.here)), ] ## order the arrival times
  out.file <- rbind(out.file, tmp.file)
}

write.table(out.file, file = "arrivals.csv", row.names=F, sep=",")
# have to clean the resulting table by removing the first row, manually changing the {} and filling
# the spaces with _
```

We now use the arrivals.csv in subsequent analysis

```
# df<-read.csv("arrivals.csv", header = T, sep = ",")
head(df)
names(df)
df<-df[complete.cases(df),]
head(df)
# get rid of curly braces
df$breed<-gsub("\\{|\\}", "", df$breed)
df$mycarcass<-gsub("\\{|\\}", "", df$mycarcass)
head(df)
# replace spaces with underscores
df$breed<-gsub("[ ]", "_", df$breed)
df$mycarcass<-gsub("[ ]", "_", df$mycarcass)
head(df)
# change breed and mycarcass columns to factors
df$breed<-as.factor(df$breed)
df$mycarcass<-as.factor(df$mycarcass)
# drop the carcass breed because we're not interested in it as it doesn't arrive to anything
df <- subset(df, breed != "breed_carcasses")
df <- subset(df, mycarcass != "0")
length(df$breed)
levels(df$breed)

# can extract the arrival times of specific species as follows
df$got.here[df$breed=="breed_hyenas"]
mean(df$got.here[df$breed=="breed_hyenas"])

# Average Arrival Time of Species to Carcasses
```

```r
meanArrivalData<-with(df,tapply(df$got.here, df$breed, mean));meanArrivalData
sdArrivalData<-with(df,tapply(df$got.here, df$breed, sd));sdArrivalData

# First Species to Arrive at Every Carcass
firstArrivalData<-df[df$got.here == ave(df$got.here, df$mycarcass, FUN=min), ]
summary(firstArrivalData)
length(firstArrivalData$mycarcass)
length(unique(firstArrivalData$mycarcass))

lastArrivalData<-df[df$got.here == ave(df$got.here, df$mycarcass, FUN=max), ]
summary(lastArrivalData)
length(lastArrivalData$mycarcass)
length(unique(lastArrivalData$mycarcass))

# Arrival Times - Code for Jackals, same approach for hyenas
# for each carcass, calculate the first jackal arrival
first_jackals <- aggregate(got.here~mycarcass,
                           data=df[df$breed=="breed_jackals",], FUN=min)

# tabulate the number of other animals arriving before the jackal
beat_jackals <- sapply(unique(df$mycarcass), function(i) {
  table(df$breed[df$mycarcass==i &
                       df$got.here < first_jackals$got.here[first_jackals$mycarcass==i]])})

# drop unwanted breeds
beat_jackals <- beat_jackals[row.names(beat_jackals) != "breed_jackals",]
# add carcass names to the columns
colnames(beat_jackals) <- unique(df$mycarcass)

arrival_order_jackal <- sapply(unique(df$mycarcass), function(i) {
  unique(df[df$mycarcass==i, "breed"])})

sapply(arrival_order_jackal, function(i) i[(which(i=="breed_jackals"))-1])
precedeJackals<-sapply(arrival_order_jackal, function(i) i[(which(i=="breed_jackals"))-1])
tableBeatJackals<-sapply(precedeJackals, table)
rowSums(tableBeatJackals)
```