Bruno Cid

Doutorando em Ecologia pela UFRJ. Interessado em conservação principalmente em reintrodução de espécies e restauração de interações ecológicas e serviços ecossistêmicos. O título da minha tese é “Uso do manejo adaptativo para otimizar a reintrodução de cutias (*Dasyprocta leporina*) na Mata Atlântica do Rio de Janeiro”, orientada pelo Dr. Fernando Fernandez no Laboratório de Ecologia e Conservação de Populações da UFRJ.

Meus exercícios

exec

Minha proposta

A reintrodução de indivíduos na natureza oferece uma ótima oportunidade para estudar o processo de formação de suas áreas de vida. Isso acontece porque podemos obter as localizações dos animais, desde o momento da soltura até sua morte. Espera-se que, logo após a soltura, os animais passem por uma fase de exploração (onde visitam muitos lugares diferentes a sua volta), para escolher onde viverão futuramente. Depois disso, espera-se que escolham o lugar onde vão realizar suas atividades diárias de forrageamento, acasalamento e cuidado com a prole, ficando mais “fiéis” a certa área. Como tenho interesse na dinâmica deste processo, escolhi como planos de função duas que me retornem indícios de como ele acontece.

PLANO A - Concluído

Título: quanto tempo os animais demoram para ficarem mais “fiéis” a certa área?

Para responder essa pergunta pensei em dividir temporalmente o total de localizações dos animais e calcular o desvio-padrão entre essas localizações em cada intervalo de tempo. O tamanho do intervalo (em dias) vai depender de um ajuste entre a quantidade de localizações em cada intervalo (lembrando que são necessárias pelo menos dois valores para o cálculo do desvio-padrão) e o intervalo de tempo que se considera adequado (baseado em seu conhecimento dos bichos) para a análise. Por esse motivo, o intervalo de tempo será um dos argumentos da função a ser definido pelo usuário. A entrada será uma planilha (classe data.frame) contendo nas colunas o id (“nome”) de cada indivíduo, as coordenadas da localização (uma coluna para a latitude e outra para a longitude, em UTM de preferência) e a data referente a cada localização existente. O resultado (saída) será uma tabela contendo a os valores dos desvios-padrão para cada indivíduo, em cada intervalo de tempo e um gráfico mostrando as mesmas informações. A hipótese de trabalho é a de que os desvio-padrão diminuirão como passar do tempo, mostrando que cada indivíduo se tornou mais “fiel” a certa área, formando uma área de vida.

PLANO B

Título: quanto tempo os animais demoram para parar de se afastar do ponto de soltura?

Essa função é mais simples do que a anterior e também traz uma informação interessante acerca do processo de formação das áreas de vida de indivíduos reintroduzidos. Para responder essa pergunta pensei em dividir temporalmente o total de localizações dos animais e escolher a localização mais distante do ponto de soltura dentre as existentes em cada intervalo. Nesse caso, o tamanho do intervalo de tempo pode ser menor do que na função anterior porque só se faz necessária uma localização em cada intervalo para realizar a análise. Nesta função, o intervalo de tempo será um dos argumentos a ser definido pelo usuário. A entrada será uma planilha (classe data.frame) contendo nas colunas o id (“nome”) de cada indivíduo, a coordenada do ponto de soltura (em UTM, de preferência), as coordenadas das localizações (uma coluna para a latitude e outra para a longitude, em UTM de preferência) e a data referente a cada localização existente. O resultado (saída) será uma planilha com as distâncias cumulativas de cada indivíduo para o ponto de soltura em cada intervalo de tempo e um gráfico mostrando as mesmas informações. A hipótese de trabalho é a de que essas distâncias crescerão em principio e depois estabilizarão com o passar do tempo.

Comentários

Seus dois planos são interessantes, e na verdade, muito parecidos. Por mais que o desvio padrão e a distância acumulada pareçam coisas diferentes, o verdadeiro desafio será separar os dados por intervalos de data, uma vez que você resolva esse problema, calcular qualquer coisa será tranquilo.

Acho que você pode seguir com o plano A, mas se sobrar tempo, vale a pena incluir o plano B na mesma função.

Mas fiquei com uma dúvida prática:

A localização é um dado de coordenadas X e Y (latitude e longitude), como você vai calcular o desvio padrão disso? É um desvio para cada coordenada? Ou tem um modo específico de calcular o desvio padrão para coordenadas? __*Danilo*

Ale

Concordo com o Danilo! Faça A e se tiver tempo inclua B! — *Alexandre Adalardo de Oliveira 2013/03/24 11:38*

Bruno

Oi gente, no fim das contas achei que ia ficar meio “frankenstein” colocar os dois usos na mesma função. Para complexificar um pouco então a função proposta no plano A, inclui duas formas de calcular a janela de tempo, uma mais suavizada (argumento win.type=“p”) e uma mais categórica (argumento win.type=“c”). Espero que gostem!

FUNÇÃO DESVIOS

Página de ajuda

```
desvios                                  package:unknow
R Documentation


Representação gráfica dos desvios-padrão de coordenadas geográficas no tempo


Descrição

Calcula os desvios-padrão de coordenadas geográficas em UTM separadas em
conjuntos determinados por janelas temporais


Uso

desvios(long, lat, id, date, win.type="p", win)


Argumentos

long        vetor numérico correspondente aos valores de longitude em UTM

lat     vetor numérico correspondente aos valores de latitude em UTM

id      vetor de caracteres correspondentes a identificação de cada
indivíduo

date        vetor de datas correspondentes a cada localização no formato
(dd/mm/aaaa)

win.type    Determina a forma como a função calcula as janelas de tempo. Se
"p" as janelas são calculadas progressivamente, se "c" são calculadas de

        forma categórica

win     tamanho da janela de tempo em dias


Detalhes

Essa função é útil para ajudar a indicar em qual momento determinado
indivíduo se torna mais fiel a uma certa área. Especialmente útil para
entender o processo de formação de áreas de vida por animais reintroduzidos.
Quanto mais uniformemente espalhadas no tempo forem as localizações, mais
confiáveis serão os padrões revelados, sendo assim, a função é mais adequada
pra aplicação em dados provenientes de GPS telemetria.

Se as localizações estiverem espalhadas por mais de uma zona UTM, os
cálculos de distâncias entre médias e pontos ficará deturpada o que se
refletirá no cálculo dos desvios-padrão.
```

```
As janelas de tempo progressivas (win.type="p") são calculadas a partir de
cada data contida em date. A cada uma delas se acrescenta sequencialmente o
número de dias determinado no argumento win para a definição das janelas de
tempo.

As janelas de tempo categóricas (win.type="c") são calculadas separando-se o
intervalo de dias contido em date em n janelas de tempo no tamanho do número
de dias determinado no argumento win para a definição das janelas de tempo.

Algumas vezes o número de localizações em cada janela será insuficiente para
o cálculo de um desvio-padrão confiável, para saber quantas localizações
existem em cada janela para cada indivíduo consulte o dataframe n.loc do
output da função desvios.


Valores

Retorna uma lista com quatro tabelas. A primeira ("desvios") mostra os
valores dos desvios-padrão para cada janela de tempo, para cada indivíduo. A
segunda ("n.win") retorna o número de janelas de tempo calculadas para cada
indivíduo. A terceira ("time") retorna o intervalo de tempo (de dias) que
foi considerado na análise, para cada indivíduo. A quarta retorna o número
de localizações em cada janela de tempo, para cada indivíduo. Retorna também
um gráfico contendo as linhas de variação dos desvios-padrão no tempo para
cada indivíduo com as medias e erros-padrão.


Autor

Bruno Cid Crespo Guimarães


Exemplos

x<-rnorm(120, 5, 10)
y<-rnorm(120, 5, 20)
ident<-rep(c("A", "B", "C"), each=40)
datas<-seq(as.Date(Sys.Date()), as.Date(Sys.Date())+100*5, length=120)
desvios(long=x, lat=y, id=ident, date=datas, win.type="p", win=15)
```

código

```
desvios<-function(long, lat, id, date, win.type="p", win)
{
  if(length(long)!=length(lat))
  {
    stop("vetores de coordenadas geograficas possuem tamanhos
incompatíveis")
  }
```

```
  if(length(long)!=length(id))
  {
    stop("vetores de coordenadas geograficas e de identificação individual
possuem tamanhos incompatíveis")
  }
  if(length(id)!=length(date))
  {
    stop("vetor de identificação individual e de datas possuem tamanhos
incompatíveis")
  }
  date<-as.Date(date, format="%d/%m/%Y")
  if(win.type=="p")
  {
    sd.ind<-list(rep(NA, times=length(unique(id))))
    len.ind<-rep(NA, times=length(unique(id)))
    n.loc.ind<-list(rep(NA, times=length(unique(id))))
    time.ind<-rep(NA, times=length(unique(id)))
    for(i in 1:length(unique(id)))
    {
      long.ind<-long[id==unique(id)[i]]
      lat.ind<-lat[id==unique(id)[i]]
      data.ind<-date[id==unique(id)[i]]
      time<-abs(diff(range(data.ind)))
      time.ind[i]<-time
      ind.adap<-data.ind[data.ind<data.ind[length(data.ind)]-win]
      sds.progress.ind<-rep(NA, times=length(ind.adap))
      n.loc.win<-rep(NA, times=length(ind.adap))
      for(j in 1:length(data.ind[data.ind<data.ind[length(data.ind)]-win]))
      {
        long.ind.win<-
long.ind[data.ind>=data.ind[j]&data.ind<=data.ind[j]+win]
        lat.ind.win<-
lat.ind[data.ind>=data.ind[j]&data.ind<=data.ind[j]+win]
        dists.long.ind.win<-abs(outer(long.ind.win, mean(long.ind.win), "-
"))
        dists.lat.ind.win<-abs(outer(lat.ind.win, mean(lat.ind.win), "-"))
        dists.ind.win<-sqrt(dists.long.ind.win^2 + dists.lat.ind.win^2)
        sd.ind.win<-sd(as.vector(dists.ind.win))
        sds.progress.ind[j]<-sd.ind.win
        n.loc.win[j]<-length(dists.ind.win)
      }
      sd.ind[[i]]<-sds.progress.ind
      len.ind[i]<-length(sds.progress.ind)
      n.loc.ind[[i]]<-n.loc.win
    }
    x11()
    par(mfrow=c(1,1))
    plot(x=max(len.ind), y=max(unlist(sd.ind), na.rm=T), xlim=c(0,
max(len.ind, na.rm=T)), ylim=c(0, max(unlist(sd.ind), na.rm=T)), type="n",
xlab="Número de janelas", ylab="Desvio-padrão")
    for(p in 1:length(unique(id)))
```

```
    {
      lines(1:length(sd.ind[[p]]), sd.ind[[p]], type="l", col="grey")
    }
    sd.matrix<-matrix(NA, max(len.ind),length(unique(id)))
    for(n in 1:length(unique(id)))
    {
      sd.matrix[1:length(sd.ind[[n]]),n]<-as.vector(sd.ind[[n]])
    }
    medias.inds<-apply(sd.matrix, MARGIN=1, mean, na.rm=T)
    points(1:max(len.ind), medias.inds, pch=19)
    desvios.inds<-apply(sd.matrix, MARGIN=1, sd, na.rm=T)
segments(x0=(1:length(medias.inds)),x1=(1:length(medias.inds)),y0=medias.ind
s-desvios.inds,y1=medias.inds+desvios.inds)
    resu<-list(sd.ind,len.ind, time.ind, n.loc.ind)
    names(resu)<-c("desvios","n.win", "time", "n.loc")
    names(resu$desvios)<-unique(id)
    names(resu$n.win)<-unique(id)
    names(resu$time)<-unique(id)
    names(resu$n.loc)<-unique(id)
    return(resu)
  }
  if(win.type=="c")
  {
    sd.ind<-list(rep(NA, times=length(unique(id))))
    len.ind<-rep(NA, times=length(unique(id)))
    n.loc.ind<-list(rep(NA, times=length(unique(id))))
    time.ind<-rep(NA, times=length(unique(id)))
    for(i in 1:length(unique(id)))
    {
      long.ind<-long[id==unique(id)[i]]
      lat.ind<-lat[id==unique(id)[i]]
      data.ind<-date[id==unique(id)[i]]
      time<-abs(diff(range(data.ind)))
      time.ind[i]<-time
      sds.progress.ind<-rep(NA,
times=ceiling(abs(diff(range(data.ind)))/win))
      n.loc.win<-rep(NA, times=ceiling(abs(diff(range(data.ind)))/win))
      for(j in 1:ceiling(abs(diff(range(data.ind)))/win))
      {
        long.ind.win<-long.ind[data.ind>=data.ind[1]+(win*j)-
win&data.ind<data.ind[1]+(win*j)]
        lat.ind.win<-lat.ind[data.ind>=data.ind[1]+(win*j)-
win&data.ind<data.ind[1]+(win*j)]
        dists.long.ind.win<-abs(outer(long.ind.win, mean(long.ind.win), "-
"))
        dists.lat.ind.win<-abs(outer(lat.ind.win, mean(lat.ind.win), "-"))
        dists.ind.win<-sqrt(dists.long.ind.win^2 + dists.lat.ind.win^2)
        sd.ind.win<-sd(as.vector(dists.ind.win))
        sds.progress.ind[j]<-sd.ind.win
        n.loc.win[j]<-length(dists.ind.win)
```

```
      }
      sd.ind[[i]]<-sds.progress.ind
      len.ind[i]<-length(sds.progress.ind)
      n.loc.ind[[i]]<-n.loc.win
    }
    x11()
    par(mfrow=c(1,1))
    plot(x=max(len.ind), y=max(unlist(sd.ind), na.rm=T), xlim=c(0,
max(len.ind, na.rm=T)), ylim=c(0, max(unlist(sd.ind), na.rm=T)), type="n",
xlab="Tempo (dias)", ylab="Desvio-padrão", xaxt="n")
    axis(1, at=(1:max(time.ind, na.rm=T)), label=(1:max(time.ind,
na.rm=T))*win)
    for(p in 1:length(unique(id)))
    {
      lines(1:length(sd.ind[[p]]), sd.ind[[p]], type="l", col="grey")
    }
    sd.matrix<-matrix(NA, max(len.ind),length(unique(id)))
    for(n in 1:length(unique(id)))
    {
      sd.matrix[1:length(sd.ind[[n]]),n]<-as.vector(sd.ind[[n]])
    }
    medias.inds<-apply(sd.matrix, MARGIN=1, mean, na.rm=T)
    points(1:max(len.ind), medias.inds, pch=19)
    desvios.inds<-apply(sd.matrix, MARGIN=1, sd, na.rm=T)
segments(x0=(1:length(medias.inds)),x1=(1:length(medias.inds)),y0=medias.ind
s-desvios.inds,y1=medias.inds+desvios.inds)
    resu<-list(sd.ind,len.ind, time.ind, n.loc.ind)
    names(resu)<-c("desvios","n.win", "time", "n.loc")
    names(resu$desvios)<-unique(id)
    names(resu$n.win)<-unique(id)
    names(resu$time)<-unique(id)
    names(resu$n.loc)<-unique(id)
    return(resu)
  }
}
```

arquivos da função

help.txt

codigo.txt