Mariella Butti

Analista Ambiental do Instituto Chico Mendes de Conservação da Biodiversidade atuando no Centro de Pesquisa e Conservação de Mamíferos Carnívoros. Possui graduação em Ciências Biológicas pela Universidade Federal de Minas Gerais (2008). Tem experiência na área de Zoologia, Geoprocessamento e Gestão de Unidades de Conservação. Atualmente é mestranda em Conservação da Biodiversidade e Desenvolvimento Sustentável pela Escola Superior de Conservação Ambiental e Sustentabilidade - IPÊ/ESCAS.

Meus exercícios

Página dos meus Exercícios

Proposta de trabalho final

Principal

ESTIMATIVAS POPULACIONAIS COM DADOS DE MARCAÇÃO E REAVISTAMENTO FEITO POR CAMERAS-TRAP

A marcação e recaptura é um método clássico de estimar populações animais. Esse baseia-se na proporção entre indivíduos marcados e não marcados nas recapturas para estimar a população total.

```
Y<- M*(t/m)
```

Onde Y= população total a ser estimada; M= número de indivíduos marcados e soltos na população; m= número de indivíduos marcados na recaptura; t = número de indivíduos total recapturado.

O método exige o cumprimento de três premissas básicas:

1. A população deve ser fechada, sem entrada ou perda de indivíduos: Normalmente essa premissa pode ser satisfeita com a redução do tempo de estudo, admitindo-se que o número de indivíduos total e marcados não variou, ou seja, mortes, nascimentos, imigração ou emigração não aconteceram no período do estudo.

2. A identificação da marcação deve ser inequívoca e as marcas não podem interferir na recaptura, ou seja, os animais não podem perder as marcas e nem ter a probabilidade de recaptura afetada pela marcação. Devido a esse aspecto a recaptura pode ser substituída, com vantagens, pelo reavistamento feito por armadilhamento fotográfico.

3. Por último, os indivíduos marcados devem se misturar aleatoriamente na população. Essa premissa gera problemas nos estudos com animais sociais, uma vez que esses animais se mantêm em bandos a probabilidade de captura de um animal marcado é afetada pela agregação entre eles. E por isso os modelos de estimativa por marcação e recaptura normalmente não aplicados a espécies que formam grupos. Entretanto, se essa premissa puder ser testada comparando os dados coletados com distribuições uniformes e agregadas, pode-se decidir pelo uso ou não das estimativas populacionais geradas com o método.

A função deverá estimar as populações de animais usando dados de uma tabela de contagens de indivíduos total e marcados em vídeos obtidos por armadilhamento fotográfico. A função permite gerar estimativas por simulações de Monte Carlo usando a soma de todos os registros.

```
p<- sum(t)/sum(m)
Y<- M*p
```

A função também retorna o resultado de um teste de aderência entre a tabela de dados e um modelo binomial a fim de verificar o cumprimento da terceira premissa do método de marcação-reavistamento. Podem ser escolhidos os testes: a) Exato binomial, b) Qui-quadrado ou c) Kolmogorov-Smirnov de acordo com a escolha do usuário.

Planejamento da função

Entrada:

Função mrsocial(marked, n.mark, n.total, n.sim, teste, nc)

- `marked` = número total de indivíduos marcados no estudo (classe: integer, `marked` > 0)
- `n.mark` = vetor de número indivíduos marcados no registro (classe: integer, `n.mark` > 0)
- `n.total` = vetor de número de número de indivíduos total no registro (classe: integer, 0 < `n.total` ≥ `n.mark`, mesmo comprimento que `n.mark`)
- `n.sim` = número de repetições que serão realizadas nas simulações de Monte Carlo (classe: integer, `n.sim` >0)
- `teste` = tipo de teste de aderência que será realizado (classe: character, `teste` = c(“binom.test”, “chisq.test”, “ks.test”)
- `nc` = nível de confiança do estudo (classe: numeric, `nc` = 0< `n.total` ≤1)

Verificando os parâmetros:

- `marked` é um número inteiro e maior que 0? Se não escreve: “marked precisa ser um número inteiro e maior que 0”.
- `n.mark` é um número inteiro e maior que 0? Se não escreve: “n.mark precisa ser um número inteiro e maior que 0”.
- `n.total` é um número inteiro e maior que 0? Se não escreve: “n.total precisa ser um número inteiro e maior que 0”.
- `n.total` tem o mesmo tamanho que `n.mark`? é um número inteiro e maior que 0? Se não escreve: “n.total precisa ter o mesmo número de elementos que n.mark”.
- `n.sim` é um número inteiro e maior que 0? Se não escreve: “n.total precisa ser um número inteiro e maior que 0. O resultado será apresentado usando o padrão 2000 simulações”
- `teste` é igual a “binom.test”, “chisq.test” ou “ks.test”? Se não escreve “o teste indicado não foi encontrado, o resultado será apresentado usando o padrão binom.test”.
- `nc` é maior que zero e menor ou igual a 1? Se não escreve: “o nível de confiança não foi indicado corretamente, o resultado será apresentado usando 0.95.”

Pseudo-código:

1. Cria um objeto `marked` para guardar o número de indivíduos marcados na população
2. Cria um objeto `pop.sim` para receber valores de população da simulação
3. Cria um objeto `mark.sim` com mesmo comprimento de `n.mark`
4. Cria um objeto `total.sim` com mesmo comprimento de `n.total`
5. Salva na primeira posição de `pop.sim` o resultado da estimativa populacional para os dados reais, dado por: `marked` vezes a somatória de `n.total`, dividido pela somatória de `n.mark`
6. Entra num ciclo for com contador i de 2:`n.sim` repetições
7. Sorteia posições aleatórias de 1:length(`n.mark`) com reposição e salva em um objeto `boot.indice`
8. Reamostra `n.mark` nas posições sorteadas em `boot.indice` e salva em `mark.sim`
9. Reamostra `n.total` nas posições sorteadas em `boot.indice` e salva em `total.sim`
10. Salva `pop.sim` na posição `i` o resultado da estimativa populacional para a simulação, dado por: `marked`vezes a somatória de `total.sim`, dividido pela somatória de `mark.sim`
11. Cria o objeto media.pop que recebe a média das estimativas geradas em
12. Cria o objeto `ic` que recebe os quantis de `pop.sim` referentes ao nível de confiança `nc`
13. Cria o objeto `sd` que recebe o valor de desvio padrão de `pop.sim`
14. Mostrar `pop.sim` em um histograma
15. Indicar o valor da estimativa de população gerada pelos dados (`pop.sim`[1]) em uma reta vertical no gráfico e uma curva dos valores acumulados sobre o histograma.
16. Um controle de fluxo encaminha ao teste de aderência indicado pelo usuário:

1. Se teste= “binom.test”,
2. Cria o objeto `aderencia` que recebe o resultado da função `binom.test()` usando como argumentos:
3. x = vetor de sucessos, ou seja, o vetor de individuos marcados nas observações (`n.mark`)
4. y = vetor de tentativas, ou seja, o vetor de individuos total nas observações (`n.total`)
5. p= hipótese de probabilidade, ou seja, a soma de individuos marcados dividido pela soma do total de individuos nas observações
6. alternative=”`two-sided`”, uma vez que o teste é para confirmar a aderência dos dados ao modelo binomial e confirmar a proporção encontrada pelas observações,
7. conf.level= o mesmo indicado pelo usuário em `ic`

1. Se teste= “chisq.test”
2. Item de lista ordenada
3. Cria um objeto `binom`, com um vetor de sucesso aleatorizado pelo uso da função `rbinom()` com os seguintes parâmetros:
4. n= comprimento do vetor `n.mark`, ou seja o número de registros da espécie estudada,
5. size= n.total, ou seja o vetor de tentativas,
6. prob= probabilidade de sucesso, ou seja, a soma de indivíduos marcados dividido pela soma do total de individuos nas observações
7. b. Cria um objeto `aderencia` que recebe o resultado da função `chisq.test()`, comparando as observações reais (n.mark) com as observações geradas pelo modelo nulo (`binom`).

1. Se teste = “ks.test”
2. Cria um objeto `aderencia` que recebe o resultado da função `ks.test()` com os seguintes parâmetros
3. x = vetor de sucessos, ou seja, o vetor de individuos marcados nas observações (`n.mark`)
4. y = “`pbinom`”, ou seja, a função de distribuição cumulativa bionomial,
5. size= comprimento do vetor `n.mark`, ou seja o número de registros da espécie estudada,
6. prob= probabilidade de sucesso, ou seja, a soma de indivíduos marcados dividido pela soma do total de individuos nas observações,
7. alternative=”`two-sided`”, uma vez que o teste é para confirmar a aderência dos dados ao modelo binomial e confirmar a proporção encontrada pelas observações

Saida:

1. Apresenta os resultados em uma lista contendo valores médio da simulação (`media`), intervalo de confiança a 95% (`ic`), desvio padrão (`sd`), número de simulações de Monte Carlo realizado (`n.sim`) e estatísticas do teste realizado (`aderência`).

> — *Alexandre Adalardo de Oliveira 2019/06/04 15:21*
>
> Parece uma boa proposta, a primeira parte da função faz o cálculo do intervalo de confiança bootstrap para a estimativa do tamanho da população, se entendi corretamente. A segunda parte, parece ok, não gosto muito de testes de aderência, mas parece que conceitualmente está tudo certo, apesar de alguma redundância. Ambas as propostas são boas e factíveis, pode escolher uma delas [1)]. Cuidados apenas que o objeto `marked` é um criado duas vezes, como argumento e depois como um objeto no pseudo-código, não entendi se foi só uma redundância ou se irá criar o objeto novamente.

Plano B

Criar uma função geo.t.apply(), que permite aplicar uma outra função a um conjunto de dados matriciais (raster), a partir de um shapefile de polígono e retorna uma tabela de valores por feição atributo(linhas) e raster (coluna).

Os rasters e o shapefile devem estar na mesma pasta, sendo que o shapefile deve estar na raiz e os rasters podem estar em subpastas.

Planejamento da função

Entrada

função: geo.t.apply(dir, padrao, subpastas, shapefile, ID.pol, codificacao.shp, funcao, salvar.como)
adaptado de
https://rstudio-pubs-static.s3.amazonaws.com/254726_caf5ac8c774645d890f97a674f6afa33.html

- `dir`= diretorio dos dados (classe: character)
- `padrao` = texto que identifica os rasters, podendo ser apenas a extensão por exemplo (classe: character)
- `subpastas` = lógico, identifica se os rasters serão buscados apenas na raiz ou também nas subpastas (classe: logico)
- `shapefile` = nome do arquivo shapefile, com a extensão (classe: character)
- `ID.pol` = nome do campo que identifica os polígonos (Classe: character)
- `codificacao.shp` = codificação do arquivo .sph (em geral UTF8)
- `funcao` = função que será aplicada aos dados por exemplo `mean`, `sum`,`min`, `max`, etc
- `salvar.como`= nome e extensão do arquivo que será salvo na saída.

Pseudocódigo:

1. cria objeto `rasters` com nomes dos rasters no diretório `dir` localizados de acordo com `padrao`indicado pelo usuário, podendo ser verificado todo o diretório caso `subpastas`=TRUE, ou apenas na raiz.
2. cria objeto `n.rasters` com o número de rasteres localizados
3. Cria uma lista `raster.list` de rasters para processamento
4. cria objeto `feicoese` le o arquivo shapefile, podendo corrigir a codificação de acordo com o informado em codificação.shp
5. cria objeto `n.feicoes` com numero de feições no shapefile
6. cria objeto `ID.pol` com identificação dos poligonos ou atributo usado para identificar o grupo
7. Cria dataframe vazio (NA) com `n.feicoes` linhas e `n.rasters+1` colunas
8. insere `ID.pol` como primeira coluna de valores no dataframe
9. insere “ID.Pol” e `rasters` como nome das colunas do dataframe
10. entra num ciclo for de 1:`n.rasters`
11. aplica a funcao indicada ao raster[i] usando funçao extract in(pacote raster)
12. salva valores relativos a todas `ID.pol` na coluna i+1 do dataframe
13. entra em um controle de fluxo que permite que o dataframe gerado seja salvo em formato csv ou txt de acordo com o indicado em `salvar.como`, essa função é interessante pois a análise dos rasters leva muito tempo e por isso é interessante ter um arquivo de saída independente do código.

Saída

Dataframe, ou arquivo .csv contendo o resultado das funções aplicadas aos rasters, por polígono.

Plus: se for seguir essa proposta gostaria de poder escolher os polígonos por atributo e aplicar as funções também por zona, acredito que um`raster::aggregate()` mas não sei ainda onde isso ficaria no pseudocodigo.

> Parece um bom plano também. Tenho pouco conhecimento de shapefile, mas consegui entender a proposta e o que a função faz. Parece bem útil para síntese de dados de um raster agrupado pelos polígonos do shapefile. Sugiro o nome `geo.tapply` (sem o segundo ponto) para ficar claro a analogia com a função `tapply` Pode decidir por qualquer uma das propostas, caso seja essa, deixamos de fora o `aggregate`... nesse momento.

> — *Mariella Butti 2019/06/09 16:30*
>
> Seguem abaixo os códigos da função e respectiva ajuda. Tive que fazer uma alteração quanto aos testes de aderência: como o Alê já tinha alertado eles não funcionaram como eu imaginei e por isso incluí uma `anova` em substituição ao `ks.test` e `binom.test` e incluí a opção de o usuário realizar um ou ambos os testes.

Função ''socialmr()''

```
socialmr<-function(marked,  # numero total de individuos marcados no estudo (classe: integer, marked > 0)
                   n.mark,  #vetor de numero individuos marcados no registro (classe: integer, n.mark > 0)
                   n.total, #vetor de numero de numero de individuos total no registro (classe: integer, 0 < n.total e" n.mark, mesmo comprimento que n.mark)
                   site,    #identificacao da estacao fotografica (cameratrap) (classe: factor)
                   n.sim = 2000, #numero de simulaces que o teste ir? fazer
                   teste = "ambos", #tipo de teste de aderencia que sera realizado (classe: character, teste = c( binom.test ,  chisq.test ,  ks.test )
                   hist.curv = TRUE,#lógico para insercao da curva de no histograma
                   nc = 0.95)   #nivel de confianca do estudo (classe: numeric, nc = 0< n.total d"1 )

{
  #
  ###.......................... Inicio da
```

```
função.............................####
 ##############################
 #  Verificando os parametros  #
 ##############################

if(marked<0|as.integer(marked)!=(marked)) #marked e um numero inteiro e
maior que 0?
  stop(" marked precisa ser um numero inteiro e maior que 0 ") #Se nao para
e indica o erro

if(ifelse(sum( n.mark<0|as.integer(n.mark)!=(n.mark))!=0,TRUE,FALSE))#n.mark
e um numero inteiro e maior que 0?
  #O ifelse() foi necessário para transformar o vetor lógico em um único
valor
  stop("n.mark precisa ser um numero inteiro e maior que 0") #Se nao para e
indica o erro

if(ifelse(sum( n.total<0|as.integer(n.total)!=(n.total))!=0,TRUE,FALSE))
#n.total e um numero inteiro e maior que 0?
  stop("n.total precisa ser um numero inteiro e maior que 0") #Se nao para e
indica o erro

if(length(n.total)!= length(n.mark)|ifelse(sum(n.total < n.mark)>0,TRUE,
FALSE))#n.total tem o mesmo n? de registros e um total de individuos maior
que n.mark?
  stop("n.total é inválido. O argumento precisa ter o mesmo numero de
elementos que n.mark e valores maiores ou iguais aos do vetor n.mark") #Se
nao para e indica o erro

if(n.sim<0| as.integer(n.sim)!=(n.sim))# n.sim e um numero inteiro e maior
que 0?
  {
  n.sim=2000 #reconfigura o valor para o padrao de 2000 simulacoes
  warning("n.total precisa ser um numero inteiro e maior que 0. O resultado
sera apresentado usando o padrao 2000 simulacoes.")#Se nao para e indica o
erro
   }

if(teste != "chisq.test" & teste !=  "anova" & teste != "ambos")# teste e
igual a  chisq.test, anova ou ambos?
   {
   teste="ambos" #reconfigura o valor para o padrao ambos
  warning("o teste indicado nao foi encontrado, o resultado sera apresentado
usando o padrao ambos") #Se nao indica o erro e alerta para sequencia com o
valor -ambos- padrao
   }

if(nc<0|nc>=1) #o nivel de confianca e positivo e menor ou igual a 1?
  {
  nc="0.95" #Se nao: reconfigura o valor para o padrao 0.95
  warning("o nivel de confianca nao foi indicado corretamente, o resultado
```

```
sera apresentado usando 0.95")#Se indica o erro e alerta para o uso do valor
padrao: 0.95
   }

#############################################################################
##
 #  controle de fluxo encaminha ao teste de aderencia indicado pelo usuario
#
#############################################################################
##

  #Cria os objetos que receberão os resultados dos testes.
  ##Caso o teste não seja realizado a seguir o objeto é incluido com valor
NULL na lista de resultados, sem retorno de erros.
  anova=NULL #cria o objeto anova
  qui.quadrado=NULL #cria o objeto qui.quadrado

if(teste== "anova"|teste== "ambos")# verifica se o usuario solicitou anova
como teste de aderencia e encaminha para o teste
{
    anova<- anova(lm(n.mark~site)) #realiza anova usando como grupo os
sitios de reavistamento. A hipostese testada é que os sitios de
reavistamento não tem diferenças entre si (H0)

    if(anova[[5]][1]<(1-nc)) #testa se há agregação entre os dados, ou seja,
se H0 não foi rejeitada no nivel de confiança indicado.
      warning("Anova indicou agregação entre seus dados (p-valor: ",
anova[[5]][1],") e a estimativa populacional com o socialmr pode ser
incorreta") #apresenta a mensagem de alerta para agregação entre os dados
quando p-valor é menor que 1- nivel de confiança indicado
}

  if(teste== "chisq.test"|teste== "ambos") #verifica se o usuario solicitou
teste qui-quadrado como teste de aderencia e encaminha para o teste.
{
    #cria a tabela de contigencia por sitio de reavistamento para os dados
de entrada
    qui.mark<- tapply( n.mark, #objeto com os numeros de individuos marcados
                       INDEX = site, #indexacao pelos sitios para criacao
databela de contingencia
                       sum) # indica a aplicacao da funcao soma ao numero de
registros marcados por sitio

    #realiza anova usando como grupo os sitios de reavistamento. A hipotese
testada é que os sitios de reavistamento não tem diferenças entre si (H0):
    qui.quadrado <- chisq.test(qui.mark,#usa a tabela de contingência gerada
acima e considerando p igual para todos os grupos (default do chisq.test())
retorna
                               simulate.p.value = T, # Faz os cálculos do p-
valor por simulações de monte carlo
```

```
                                B= n.sim) #faz o mesmo numero de simulações
que o usuário configurar



    if(qui.quadrado$p.value <=(1-nc))#testa se há agregação entre os dados,
ou seja, se H0 não foi rejeitada no nivel de confiança indicado.
      warning(" Qui-quadrado indicou agregação entre seus dados (p-valor: ",
qui.quadrado$p.value,") e a estimativa populacional com o socialmr pode ser
incorreta") #apresenta a mensagem de alerta para agregação entre os dados

}

 ###################################
 #     Calculo das estimativas     #
 ###################################

pop.sim<-rep(NA, n.sim)  #Cria um objeto pop.sim para receber valores de
populacao da simulacao
mark.sim<- rep(NA, n.sim)  #Cria um objeto mark.sim com mesmo comprimento de
n.mark para receber os valores de marcaco das simulacoes
total.sim<- rep(NA, n.sim) #Cria um objeto total.sim com mesmo comprimento
de n.total para receber os valores de total de individuos nas simulaces
pop.sim[1]<-marked*(sum(n.total)+1)/(sum(n.mark)+1) #Salva na primeira
posicao de pop.sim o resultado da estimativa populacional para os dados
reais, dado por: marked vezes a somatoria de n.total, dividido pela
somatoria de n.mark
total.sim[1]<- sum(n.total)# inclui na posicao 1 do vetor o total de
individuos observado nos dados
mark.sim[1]<- sum(n.mark) #inclui na posicao 1 do vetor o total de
individuos marcados observado nos dados

for (i in 2:n.sim) #Entra num ciclo de repeticoes indicado pelo argumento
n.sim
     {
       boot.indice<- sample(1:length(n.mark), replace=TRUE) #Sorteia
posicoes aleatorias de 1:length(n.mark) com reposc e salva em um objeto
boot.indice
       mark.sim[i]<- sum(n.mark[boot.indice]) #Reamostra n.mark nas posicoes
sorteadas em boot.indice e salva em mark.sim
       total.sim[i]<- sum(n.total[boot.indice]) #Reamostra n.total nas
posicoes sorteadas em boot.indice e salva em total.sim
       pop.sim[i]<- marked*(total.sim[i]+1)/(mark.sim[i]+1) #Salva pop.sim
na posicoo i o resultado da estimativa populacional para a simulacao
      }

media.pop<-mean(pop.sim)#Cria o objeto media.pop que recebe a media das
estimativas geradas em
ic.pop<- round(quantile(pop.sim, probs = c(0,nc)+((1-nc)/2), na.rm=T),4) #
criando objeto com quantis referentes ao indicado por nc
sd.pop<- sd(pop.sim) #Cria o objeto sd que recebe o valor de desvio padrao
```

```
de pop.sim
par(mfrow=c(1,1))# ajusta os parametros de grafico para uma janela de 1x1,
ou seja todas as plotagens estarão no mesmo quadro
hist(pop.sim, #Mostrar pop.sim em um histograma
     prob= T, #Eixo y com valores de probabilidade
     xlab= "Populacao estimada", #muda a legnda do eixo x
     ylab= "Probabilidade", #muda a legenda do eixo y

     main= "") #retira o titulo
abline(v=pop.sim[1], col="red")#Indicar o valor da estimativa de populacao
gerada pelos dados (pop.sim[1]) em uma reta vertical no grafico
    if(hist.curv==TRUE) #verifica se o usuario deseja a plotagem da curva no
histograma
    {
      lines(density(pop.sim),col="dark grey")#inclui uma curva de densidade
dos valores simulados no histograma
    }

return(list(media.pop=media.pop,
            sd.pop=sd.pop,
            ic.pop=ic.pop,
            anova=anova,
            qui.quadrado=qui.quadrado)) #apresenta os resultados em forma de
lista contendo a media, o desvio-padrao, o intervalo de confianca das
estimativas populacionais ao nivel de confianca indicado. E o resultado
do(s) teste(s) de aderencia, que indica se houve mistura aletoria dos
animais na populacao


###........................ Fim da
funcao.................................####

}
```

Ajuda da Função ''socialmr()''

```
socialmr                package:nenhum             R Documentation

~~Estimativa de populacao de animais sociais usando metodo de marcacao e
recaptura ~~

Description:

     ~~ A funcao faz a estimativa populacional usando a razao individuos
marcados/total de registros da especie. A funcao tambem faz um teste da
aderencia dos dados a um modelo de distribuicao binomial com proporcao que
indica se
```