

Adriana Moriguchi Jeckel



Doutoranda do Programa de Pós-Graduação em Zoologia do Instituto de Biociências da USP. Faço parte do Laboratório de Anfíbios do Departamento de Zoologia, orientada pelo Prof. Taran Grant. Tenho interesse na ecologia e evolução da defesa química em anfíbios. O meu projeto de doutorado tem como objetivo principal compreender alguns dos mecanismos que permitem que as rãs de veneno (em inglês, *poison frogs*) sequestrem alcaloides da sua alimentação.

[Laboratório de Anfíbios](#)

Meus exercícios

[exec](#)

Propostas Trabalho Final

Plano Principal

Construção de Matriz a partir de data frames de tamanhos de linhas diferentes

O objetivo desta função é construir uma matriz a partir de data frames de duas colunas com números diferentes de linha. A primeira coluna dos data frames deverá ser a variável de interesse, e a segunda coluna os valores de observação de cada caso. A função irá organizar os dataframes levando em conta a variável de interesse. Se em algum dos data frames alguma das linhas não estiver presente, ele automaticamente colocará "0". Estes data frames devem estar num diretório exclusivo para eles, já que a função irá chamar todos os data frames direto de arquivos .txt. Como default, os argumentos serão FALSE para binary= e TRUE para transpose=, ou seja, a matriz gerada conterá os valores apresentados no data frame, e transporá a matriz, para que as linhas se tornem as colunas. Caso especificado, binary=TRUE construirá matriz de presença (1) e ausência (0), e transpose=FALSE, manterá a matriz no formato original. Caso nada for especificado no argumento file=, a função irá devolver uma matriz na área de trabalho; se especificado, um arquivo .txt será gerado neste mesmo diretório.

Atualização 23/06/2017

Durante a execução da função, percebi que não precisaria ser exatamente uma matriz, mas poderia manter a classe do objeto final como `data.frame`. Então, como pode ser mais útil, decidi manter o objeto final como `data.frame`. Assim, a função final vai modificar um pouco do aprovado pela Melina.

Plano B

Diferenças entre grupos por método Monte Carlo para amostras pequenas

Na minha área de estudo, temos que analisar estatisticamente muitos parâmetros de diversidade de toxinas da pele de sapos, porém, não temos um N suficiente para testar qualquer distribuição. Nesta função, quero automatizar as análises de diferença de média que geralmente fazemos, mas aplicando o método de Monte Carlo. A função irá receber dois vetores, um numérico e um fatorial. A função irá identificar se o vetor fatorial tem 2 ou mais fatores. Caso tenha 2 fatores, ele fará o teste de diferença entre duas amostras utilizando uma distribuição nula gerada utilizando os próprios dados. Caso o vetor tenha mais de 2 fatores, ele executará um teste ANOVA, também utilizando um cenário nulo a partir dos dados. Como saída, a função irá descrever na área de trabalho o número de fatores que foi utilizado para fazer a análise e o p-valor.

Comentários Melina

Olá Adriana. Eu gostei mais da sua proposta A, ela tem argumentos úteis e parece um bom desafio. Minha dúvida é: a sua função vai ler todos os arquivos de dados txt de uma pasta ou você vai ter que ficar especificando arquivo por arquivo que será agrupado na matriz? Espero que você consiga fazer com que sua função leia todos de uma vez para evitar que o usuário que tenha por exemplo 50 arquivos de dados (data frames) precise dizer o nome de todos eles na função. Uma outra dúvida é sobre o "0" que será colocado na matriz caso não haja a variável de interesse no data frame. Acho que será importante tomar cuidado com a diferença entre dados faltantes e zero, pois se o usuário tiver valores zero "verdadeiros" em outras linhas do data frame, isso será confundido com esse zero inserido posteriormente na matriz.

Achei o plano B menos interessante, pois pelo que entendi é muito parecido com a função `simula.R`, ou seja, teste de hipótese por amostragem aleatória.

Eu sugiro você tentar o plano A mesmo e ir em frente!

Oi Melina, A minha ideia é que a função leia todos os arquivos de dados que estiverem na pasta. Então, a pessoa teria que ter cuidado para não ter outros arquivos dentro desta pasta que ela colocaria para ser lida na função. Sobre o 0, é que como na minha cabeça, a ideia é que seja tipo uma matriz de diversidade (ex. espécie X localidade), o zero, significa que aquela espécie não está naquele local, então estaria OK colocar o zero. Porém, posso deixar como default NA nos dados faltantes, e se a pessoa quiser transformar o NA em 0, pode. O que acha? Já estou finalizando a função. Como posso fazer para te mostrar para ver se está tudo OK, e que posso seguir em frente com o help?

Obrigada pelas sugestões!

Função Trabalho Final

Código da Função join.frames

Arquivo da função: [join.frames](#)

```
#O objetivo desta função é construir uma matriz a partir de data frames com  
números diferentes de linha.  
#A primeira coluna dos data frames deverá ser a variável de interesse, e o  
restante das colunas, os valores de observação de cada caso.  
#A função irá organizar os dataframes levando em conta a variável de  
interesse.
```

```
join.frames<- function(path, na.zero=F, binary=F, transpose=T, file=F)
{
  filenames<- list.files(path) #Objeto com o nome dos arquivos da pasta  
indicada
  filelist<- lapply(filenames, read.delim) #Objeto tipo lista com todos os  
data.frames que estão na pasta indicada
  frames<-Reduce(function(x,y) merge(x,y, all=TRUE), filelist) #Junta  
todos os arquivos de data.frame num arquivo só. Onde não houver dado de uma  
determinada linha em um dos data.frames, ele vai transformar colocar um NA.  
#Verificar se existe algum nome repetido na variável de interesse
  rownames(frames)<- frames[,1] #Coloca a variável de interesse que está  
na primeira linha como o nome das linhas.
  frames<- frames[,-1] #Retira a primeira coluna, para deixar o data.frame  
apenas numérico
  #Se o argumento na.zero=F, ele vai deixar NA. Se o na.zero=T, ele vai  
transformar os NAs em zero.
  if(na.zero==T)
  {
    frames[is.na(frames)]<- 0 #Transforma NA em 0
  }
  #Se o argumento binary=F, ele vai deixar os valores que estão nos  
data.frames. Se binary=T, ele vai transformar os NAs zero, deixar onde tem  
zero, e transformar todas as posições que tem valor em 1.
  if(binary==T)
  {
    frames[is.na(frames)]<- 0 #Transforma NA em 0
    frames[frames>0] <- 1 #Transforma todos os valores acima de 0 em 1
  }
  #Se o argumento transpose=T, ele vai transpor o data.frame. Caso  
transpose=F, ele vai deixar o data.frame do jeito que está.
  if(transpose==T)
  {
    frames<-t(frames) #Transpõe o data.frame
  }
  #Se o argumento file=NULL, ele vai liberar o data.frame na área de
```

trabalho. Se file="path" ou "file.txt", gravará um arquivo .txt no arquivo indicado ou no diretório de trabalho ativo.

```
if(file==F)
{
    return(frames) #Retorna na área de trabalho o data.frame gerado
}else
{
    write.table(frames, file, sep="\t") #Salva o data.frame em um
arquivo .txt no diretório ativo
}
}
```

Página do Help: join.frames

join.frames

package:unknown

R Documentation

Cria um único **data frame** a partir de outros **data frames**

Description:

Esta função cria um **data.frame** a partir de outros **data.frames** com diferentes números de linha de um diretório específico. A primeira coluna de cada **data.frame** deve ser a variável de interesse, e as outras colunas, as observações de cada caso. Esta função irá organizar o **data.frame** baseado na primeira coluna de cada **data.frame**, que deverá ser a variável de interesse. Se em algum dos **data.frames** algum caso da variável de interesse não estiver presente, será colocado um NA.

Usage:

```
join.frames(path, na.zero=F, binary=F, transpose=F, file=F)
```

Arguments:

path diretório onde estão os arquivos .txt que serão lidos na função. Os arquivos serão lidos com a função **read.delim()**.
na.zero se TRUE, transforma os NAs em 0.
binary FALSE - os valores de cada observação serão mantidos como nos arquivos originais. Se TRUE, transforma o **data.frame** em dados de presença (1) e ausência (0).
transpose se FALSE, as colunas e linhas do **data.frame** serão mantidos como os arquivos de **data.frames** originais.
file se FALSE, o **data.frame** final será lançado no área de trabalho. Se **file="file.txt"**, o **data.frame** será salvo no diretório atual como arquivo .txt, separado por tabulação.

Details:

Note que o diretório especificado no **"path"** deve ser exclusivo para os arquivos que serão trabalhados na função, já que todos os arquivos do diretório serão lidos.

Value:

`join.frames` gera o **data.frame** final na área de trabalho. Se for especificado algum nome para o arquivo `.txt`, será gerado um arquivo `.txt` separado por tabulação com o **data.frame** no diretório ativo.

Warning:

A primeira coluna, da variável de interesse, será transformado em nome de linhas em um determinado momento. Caso exista alguma observação repetida ou nome repetido nesta coluna, o `rownames()` gerará uma mensagem de erro impossibilitando a continuidade da função.

Note:

Os arquivos serão lidos do diretório especificado pela função `read.delim()`. Certifique-se que os arquivos estão no formato correto.

Author(s):

A. M. Jeckel

See Also:

`read.delim()`
`rownames()`

Examples:

```
#Para executar o exemplo, o arquivo Exemplo_Funcao.zip deve ser baixado  
(wikialunos de Adriana Moriguchi Jeckel:  
http://ecologia.ib.usp.br/bie5782/doku.php?id=01_curso_atual:alunos:trabalho  
_final:amjeckel:func ), e colocado em um diretório exclusivo, o qual será o  
seu "path"  
#Assim que baixado, coloque esta pasta como diretório ativo  
getwd() #Verifique se o diretório está correto  
dir() #Verifique se apenas os arquivos do exemplo estão neste diretório  
join.frames(getwd(), na.zero=T, binary=F, transpose=F,  
file="example.txt")
```

Link para o exemplo_funcao.zip

[Exemplo_Funcao.zip](#)

From:

<http://ecor.ib.usp.br/> - **ecoR**

Permanent link:

http://ecor.ib.usp.br/doku.php?id=05_curso_antigo:r2017:alunos:trabalho_final:amjeckel:start

Last update: **2020/09/23 20:12**

