Julia Dombroski

Mestre em Psicobiologia em ênfase em comportamento animal. Atua na área de bioacústica e conservação de cetáceos. Aceita para cursar o doutorado em biologia na Universidade de St. Andrews.

Neste link você encontra meu CV: cv_2016_academico_pt_03_2016.pdf

E aqui o link para meu Lattes: http://lattes.cnpq.br/8929859786091571

Quer saber um pouco mais sobre meu trabalho? Então acesse: http://www.rufford.org/projects/julia_dombroski e http://www.lab.bio.br/

Meus exercícios

Link para exec

Proposta de Trabalho Final

Principal

Função para elaboração de imagem com espectrograma, oscilograma e espectro de frequência de um som de interesse.

Na área da bioacústica a visualização de um som de interesse por meio de representações gráficas é fundamental e indispensável na maioria das publicações. Deste modo o autor deve gerar diversas imagens de representações gráficas para cada um dos sons que ele investigue. No caso de descrições de repertório acústico por exemplo, podem ser necessárias dezenas de imagens descritivas. Por se tratar de uma tarefa que empregará o mesmo conjunto de funções com variações de parâmetros específicos, será desenvolvida deste trabalho uma função para aumentar a eficiência os(as) autores(as) na elaboração de tais imagens.

A função proposta irá produzir uma imagem que possuirá as três representações gráficas básicas necessárias para a descrição de um som ou vocalização de maneira completa: espectrograma, oscilograma e espetro de frequência. As representações gráficas estarão organizadas de maneira intuitiva e condensada para que a imagem gerada possa ser destinada a publicação: o espectrograma ocupará a maior porcentagem da imagem; empregando a escala do eixo de tempo (x) do espectrograma, na porção inferior da janela, será criado um oscilograma; empregando a escalda de frequência (y) do espectrograma, na porção direita da janela será construído um espetro de frequência. Abaixo, esboço de como será a imagem retornada pela função.

Esta função utilizará ferramentas dos pacotes seewave, tuneR, signal e MASS. O objeto de entrada da função deverá conter o som de interesse a partir do qual serão geradas as representações gráficas. Os argumentos serão: tipo de arquivo de imagem a ser gerado; nome do arquivo retornado pela função; dimensões desejadas da imagem de retorno e palheta de cores será empregada na elaboração do espectrograma. A função irá solicitar ao usuário que insira, um a um, dados sobre o tipo de janela, comprimento de janela, e sobreposição das amostras para elaboração das representações gráficas; intervalo de frequência a ser representado, como ele será dividido e unidade (Hz ou kHz).

Para saber mais sobre as representações gráficas do som e seus parâmetros acesse: http://www.birds.cornell.edu/brp/Raven/Raven14UsersManual.pdf

> Muito boa e clara a sua proposta! Adorei o desenho! Minha única preocupação é que ela seja muito difícil. Não conheço os pacotes que você pretende usar, mas espero que eles produzam os dados que você precisa pra fazer esses gráficos com facilidade. Fazer o gráfico em si já vai ser bem trabalhoso, já que são três gráficos em um. Espero que você curta o desafio!
>
> —-*Mali*

> Ótima proposta! Também adorei o desenho ;) É legal deixar vários parâmetros a escolha do usuário, mas recomendo você pensar em argumentos default. Estou ansiosa para ver o resultado final.
>
> —-*Sara*

Plano B

Função para estimativa de nível sonoro na fonte.

O nível sonoro na fonte (SL source level) corresponde ao valor de intensidade de um som a uma distância de 1 metro da fonte. Este valor pode ser estimado por meio da equação dos sonares originalmente desenvolvida para avaliar a performance de sonares em ambientes aquáticos. Os termos dessa equação são valores expressos em dB e representam a perda de intensidade sonora por transmissão (TL – Transmission lost) e o nível do sinal recebido no sensor (RL – Received Level). Deste modo, o nível sonoro na fonte é igual a somatória do nível recebido e da perda por transmissão:

SL=RL+TL

Existem porém diferentes maneiras para calcular a perda por transmissão: pode-se considerar modelos de espalhamento geométrico, cilíndrico ou combinados (cada um com sua fórmula

específica) e cada um deles se comportará de maneira distinta de acordo com características como o tipo de fundo do local (arenoso ou rochoso), a profundidade do local, temperatura da água e salinidade. A função proposta possuirá como argumentos: profundidade do local, temperatura da água, salinidade e tipo de fundo e seu objeto de entrada deverá conter o valor de RL (dB). Então, a função perguntará ao usuário qual modelo para o cálculo da perda por transmissão ele deseja empregar (esférica, cilíndrica ou combinada), qual valor de absorção padrão e se deseja que o índice de direcionalidade do som seja levado em consideração ou não para o cálculo do SL e qual seria este índice (deverão ser empregados valores padrões para som omni, bi ou unidirecional. A função retornará então um vetor com o valor de SL e um gráfico mostrando a perda da intensidade desde a fonte até a localização do sensor, de acordo com o modelo escolhido, como na imagem abaixo:

Outra proposta interessante e clara. É bem complexa porque tem muitos elementos a serem considerados, mas você parece dominar bem a questão, e as dificuldades em R serão interessantes. Se você optar por esta, recomendo fazer uma versão com menos opções primeiro, e ir adicionando complexidade aos poucos. Só uma coisa: evite fazer perguntas ao usuário. Todas essas coisas podem ser transformadas em parâmetros de entrada da função. Não tena mêdo de ter parâmetros demais - só lembre-se de dar nomes claros para os parâmetros, e colocar valores default para aqueles parâmetros pouco usados.

—-*Mali*

As duas propostas são muito boas, e trabalhosas. A dificuldade maior da proposta principal imagino que será criar o gráfico, enquanto no Plano B a dificuldade virá de fazer os cálculos. Escolha a que te motiva mais, e boa sorte!

—-*Mali*

Concordo com Mali sobre os parâmetros na proposta 2 (e também comentei na 1). Ótimas propostas, mande bala na que você achar mais interessante e útil.

—-*Sara*

Mali e Sara, muito obrigada pelas dicas e pelo incentivo. Acho que vou ficar a primeira proposta: usarei muito, meus companheiros do Laboratório de Bioacústica usarão muito e como foi algo que me tirou o sono durante o mestrado, é questão de honra elaborar essa função! Já estou elaborando o script dos gráficos e depois vou partir para a elaboração da função. Novidades em breve!

Ju

Documentação da Função

```
sof.image {seewave}                                R Documentation

        Produzindo arquivo com imagem que reúne 3 representações gráficas do
som

Descrição:

Produz arquivo com imagem que possui as três representações gráficas básicas
necessárias para a descrição
de um som de maneira completa: espectrograma (f~t), oscilograma (a~t) e
espectro de frequência(a~f).

Uso:

sof.image(sound, fty, winsz, flim, espcolor, fscolor)

Argumentos:

sound
    objeto R da classe "Wave".

fty

    tipo que arquivo que se deseja criar: "jpeg", "tiff" ou "png"
winsz
    tamanho de janela em pontos fft.

flim

    limites mínimo e máximo do eixo de frequêcia (kHz).

espcolor

    cor do espectrograma: se colorido argumento deve ser "colors"; para tons
de cinza: "grey".

fscolor

    cor do espectro de frequência: "rd" para espectro de frequência em
vermelho e "bl" para espectro de frequência em preto.
```

```
Detalhes:


As representações gráficas da imagem resultante estarão organizadas da
seguinte maneira: o espectrograma ocupará a maior porcentagem da imagem;
empregando
a escala do eixo de tempo (x) do espectrograma, na porção inferior da
janela, será criado um oscilograma; empregando a escalda de frequência (y)
do espectrograma,
na porção direita da janela será construído um espetro de frequência.

Valor:

Um dispositivo gráfico de imagem é ativado mas nada é retornado ao usuário.

Aviso:

O objeto de entrada deve conter "f".
A duração total do objeto será representada graficamente.
Para todos os tipos de arquivo o tamanho da imagem é de 16cm de altura por
16cm de largura e resulução é de 600ppi.
Ambos espectrograma e espectro de frequência são calculados empregando-se
sobreposição de amostras de 50% e janela "Hanning".
O espectro de frequência é calculado por meio da função meanspec()

Autore(s)

Julia R. G. Dombroski
dombroski.julia@gmail.com

Referências:

Sueur J., T. Aubin, C. Simonis. 2007. Equipment review: seewave, a free
modular tool for sound analysis and synthesis. Bioacoustics 18(2) 212-226.


Exemplos:
##Elaborando uma imagem em formato jpeg com espectrograma colorido e
espectro de frequência em preto.
##Tamanho de janela de 512 pontos e o eixo de frenquência irá de 0 até 500Hz

whalecall=readWave("B.wav")
sof.image(whalecall, "jpeg", 512, c(0, 0.5), "colors", "bl")


##Elaborando, para o mesmo som de interesse (whalecall) imagem em tiff com
espetrograma em escala de cinza e espectro de frequência em vermelho.
####Tamanho de janela de 1024 pontos e o eixo de frenquência irá de 0 até
800Hz

sof.image(whalecall, "tiff", 1024, c(0, 0.8), "grey", "rd")
```

Código da Função

```
##############################
##Antes de utilizar a função##
##############################

##Iniciar com a instalação dos pacotes que contém as ferramentas que vamos
utilizar.##

install.packages("seewave")
install.packages("MASS")
install.packeges("tuneR")
install.packeges("signal")

##Carregando as ferramentas dos pacotes préviamente instalados##

library(seewave)
library(tuneR)
library(signal)
library(MASS)

########################
##A função sof.image()##
########################

sof.image= function(sound, fty, winsz, flim, espcolor, fscolor) #Criando uma
funçao que se chama sof.image (Spectrogram, Oscilogram, Frequency spectrum).
Seus argumentos são sound (objeto R ao qual son foi atribuído), fty (tipo de
arquivo que será gerado), tjam (tamanho de janela), flim (limites do eixo de
frequência), espcolor (definiçao das cores a serem empregadas no
espectrograma) e fspec(cor do espectro de frequência).
{
  if(fty=="jpeg") #se o argumento fty for "jpeg" seguir por este caminho
  {
    jpeg(filename ="Figure%03d.jpeg", width=16, height=16, units="cm",
res=600) #criar um arquivo do tipo jpeg com 15 por 15 cm e resoluçao 600
ppi.
  }
  if (fty=="tiff") #se o argumento fty for "tiff" seguir por este caminho
  {
    tiff(filename="Figure%03d.tiff", width=16, height=16, units="cm",
res=600) #criar um arquivo do tipo tiff com 15 por 15 cm e resoluçao 600
ppi.
  }
  if(fty=="png") #se o argumento for fty "png" seguir por este caminho
  {
```

```
    png(filename ="Figure%03d.png", width=16, height=16, units="cm",
res=600) #criar um arquivo do tipo png com 15 por 15 cm e resoluçao 600 ppi.
  }
layout(matrix(c(1,2,3,0), nc=2, byrow=TRUE), width=c(2,1), height=c(3,1)) #
Divisão do espaço de acordo com a matriz para receber os gráficos. A matriz
possui 2 colunas e será organizada completando as linhas. largura das
colunas serão 2 e 1 e altura das linhas 3 e 1
par(mar=c(0,4.5,1,0), family="serif") #estabelecimento das margens do
primeiro gráfico em relaçao ao espaço determinado na linha acima; fonte do
gráfico -serif- determinada pelo argumento family.
  if(espcolor=="colors") #se o argumento espcolor for "colors" seguir por
este caminho
  {
    spectro(sound, flim=flim,  tcl=-0.15, axisY=TRUE, axisX=FALSE, wl=winsz,
wn="hanning", ovlp=50, scale=FALSE, tlab="", grid=F, cex.axis=1.2,
cex.lab=1.3, palette=temp.colors, collevels=seq(-65,0,0.5))
  } #calculo do espectrograma com janela "hanning", sobreposiçao de 50%
entre as amostras, tamanho de janela e limites de frequência conforme os
argumentos da funçao e paletta de cores quentes.
  if(espcolor=="grey")#se o argumento espcolor for "grey" seguir por este
caminho
  {
    spectro(sound, flim=flim,  tcl=-0.15, axisY=TRUE, axisX=FALSE, wl=winsz,
wn="hanning", ovlp=50, scale=FALSE, tlab="", grid=F, cex.axis=1.2,
cex.lab=1.3, palette=reverse.gray.colors.2, collevels=seq(-65,0,0.5))
  }#calculo do espectrograma com janela "hanning", sobreposiçao de 50% entre
as amostras, tamanho de janela e limites de frequência conforme os
argumentos da funçao e paletta de cores em escala de cinza.
par(mar=c(0,0,1,5), family="serif")
  if (fscolor=="rd") #se o argumento fspec for idêntico a "red" seguir por
este caminho
  {
    meanspec(sound, plot=2, flab="", alab=" ", col="red", lwd=1.9, tcl=0.15,
yaxt="n", flim=flim, wl=winsz) #elaboraçao do espectro de frequência com
tamanho de janela e limites do eixo f de acordo com os argumentos da função.
cor da linha: vermelha
    mtext("Relative \n amplitude (dB)", side=1, cex=1, family="serif",
line=1.9, las=1) #colocaçao de texto no eixo "z" do espetro de frequência.
  }
  if (fscolor=="bl") #se o argumento fspec for idêntico a "bl" seguir por
este caminho
  {
    meanspec(sound, plot=2, flab="", alab=" ", col="black", lwd=1.9,
tcl=0.15, yaxt="n", flim=flim, wl=winsz) #elaboraçao do espectro de
frequência com tamanho de janela e limites do eixo f de acordo com os
argumentos da função. cor da linha: preta
    mtext("Relative \n amplitude (dB)", side=1, cex=1.1, family="serif",
line=2, las=1) #colocaçao de texto no eixo "z" do espetro de frequência.
  }
par(mar=c(4.5,4.5,0, 0), family="serif", cex.axis=1.2, cex.lab=1.3)
#estabelecimento das margens do espectrograma em seu espaço determinado. A
```

```
fonte empregada  é a mesma dos outros gráficos. cex diz respeito ao aumento
dos símbolos usandos nas indicaçoes do eixo (cex.axis) e no título do eixo
(cex.lab)
oscillo(sound, bty="o", tcl=0.25) #criaçao do oscilograma
par(mfrow=c(1,1)) #Retornando configuração do dispositivo gráfico.
dev.off() #encerrando o dispositivo gráfico para que o arquivo criado seja
salvo.
cat("Sua imagem está pronta! Verifique o diretório de trabalho atual.")
#mensagem que indica que a imagem já foi gerada e deve ter sido salva no
diretório de trabalho.
}

#######
##FIM##
#######
```

Arquivos da função sof.image()

Arquivo de som em formato wave para exemplo:

Arquivo da Função: sofimage_arquivo.r

Documentação da Função: documentacao_sofimage.txt

— *Julia Ribeiro Guimaraes Dombroski*

—-

> Olá Julia! Sua função está funcionando bem, mas queria fazer alguns comentários para melhorar a usabilidade dela, já que você pretende usar a função e compartilhar com seus colegas no futuro:
>
> - O arquivo da função está com erros de digitação na parte dos “install.packages” (ainda bem que é numa parte do código que é tão pouco importante)
>
> - A função forma gráficos esdrúxulos se eu cometo algum erro bobo na hora de escrever os argumentos, por exemplo, “color” em vez de “colors”, ou “red” em vez de “rd”. É importante que uma função saiba lidar com esse tipo de erro, porque alguém sempre erra... No seu caso o problema é que existir um “if(espcolor==“grey”)” e um “if(espcolor==“colors”)”, mas o que acontece se nenhum desses “if”s é satisfeito? Uma solução é reconhecer quando nenhum dos ifs foi satisfeito usando if-elses encadeados, e, ou lançar uma mensagem de erro, ou escolher uma opção default.
>
> - Falando em defaults, nenhum dos seus argumentos tem opção default, mas praticamente todos poderiam ter, especialmente os últimos, que só determinam as cores do gráfico. Por exemplo, por que não tornar default o gráfico ser em png, colorido, e com a linha

vermelha? E deixar a pessoa só editar esses argumentos quando for importante?

- Outro detalhe, mas você pode adicionar um “\n” no final da sua mensagem no final pra incluir uma quebra de linha, isto é, fazer o prompt aparecer na próxima linha. Por exemplo “Verifique o diretório de trabalho atual.\n”. Experimente fazer essa mudança.

Não dá mais pra editar a função aqui na wiki, mas se quiser conversar sobre isso me mande um e-mail.

—-*Mali*