

Juliana Fernandes Primon



Mestranda em Zoologia, Instituto de Biociências, USP.

Minha pesquisa é sobre impactos da invasão acústica de *Lithobates catesbeianus* em comunidades nativas de anuros, sob orientação do Prof. Taran Grant.

Meus Exercícios

Link para página com meus [Exercícios](#) resolvidos

Trabalho Final

Proposta A: Seleção de arquivos

Contextualização

No contexto da linha de pesquisa da bioacústica, uma quantidade muito alta de arquivos é gerada a partir da utilização de gravadores automáticos, que são capazes de armazenar até 1TB de dados durante meses em campo. O resultado é uma lista extensa de arquivos de áudio de duração pré estabelecida (variando de 1 minuto até 1 hora). Agora imagine um trabalho de monitoramento acústico de longo prazo, com gravadores programados para gravar 1 minuto a cada 10 minutos, durante 24 horas por dia, durante 6 meses? Isso resulta em quase 26 mil arquivos por gravador!

Logo, o objetivo dessa função é facilitar a busca de arquivos de interesse para realização de pesquisas que utilizem um número muito alto de arquivos, como por exemplo a bioacústica. A ideia é permitir que o pesquisador restrinja os arquivos de interesse e assim consiga analisar parte dos seus dados de forma mais rápida e eficaz. Além disso, a função será útil também para aleatorizar os arquivos que serão utilizados, supondo que o pesquisador queira selecionar uma quantidade X de arquivos para serem analisados. Os dados para rodar a função deverão estar organizados em um

arquivo .txt ou .csv, como mostrado na figura 1.

```
DATE, TIME, LAT, , LON, , POWER (V) , TEMP (C) , #FILES, MIC0 TYPE, MIC1 TYPE
2019-Mar-26,16:40:00,-23.56472,s,46.73055,w,4.5,39.25,1,IN,IN
2019-Mar-26,16:51:00,-23.56472,s,46.73055,w,4.5,37.50,1,IN,IN
2019-Mar-26,17:01:00,-23.56472,s,46.73055,w,4.6,37.75,1,IN,IN
2019-Mar-26,17:12:00,-23.56472,s,46.73055,w,4.6,37.50,1,IN,IN
2019-Mar-26,17:23:00,-23.56472,s,46.73055,w,4.6,37.00,1,IN,IN
2019-Mar-26,17:34:00,-23.56472,s,46.73055,w,4.6,36.50,1,IN,IN
2019-Mar-26,17:45:00,-23.56472,s,46.73055,w,4.6,36.00,1,IN,IN
2019-Mar-26,17:56:00,-23.56472,s,46.73055,w,4.6,35.25,1,IN,IN
2019-Mar-26,18:07:00,-23.56472,s,46.73055,w,4.6,33.75,1,IN,IN
2019-Mar-26,18:18:00,-23.56472,s,46.73055,w,4.5,32.25,1,IN,IN
2019-Mar-26,18:29:00,-23.56472,s,46.73055,w,4.6,30.75,1,IN,IN
2019-Mar-26,18:40:00,-23.56472,s,46.73055,w,4.5,29.25,1,IN,IN
2019-Mar-26,18:51:00,-23.56472,s,46.73055,w,4.5,28.25,1,IN,IN
2019-Mar-26,19:02:00,-23.56472,s,46.73055,w,4.5,27.25,1,IN,IN
2019-Mar-26,19:13:00,-23.56472,s,46.73055,w,4.5,26.50,1,IN,IN
2019-Mar-26,19:24:00,-23.56472,s,46.73055,w,4.5,25.75,1,IN,IN
2019-Mar-26,19:35:00,-23.56472,s,46.73055,w,4.5,25.25,1,IN,IN
2019-Mar-26,19:46:00,-23.56472,s,46.73055,w,4.5,24.50,1,IN,IN
2019-Mar-26,19:57:00,-23.56472,s,46.73055,w,4.5,24.00,1,IN,IN
2019-Mar-26,20:08:00,-23.56472,s,46.73055,w,4.5,23.50,1,IN,IN
2019-Mar-26,20:19:00,-23.56472,s,46.73055,w,4.5,23.25,1,IN,IN
2019-Mar-26,20:30:00,-23.56472,s,46.73055,w,4.5,22.75,1,IN,IN
2019-Mar-26,20:41:00,-23.56472,s,46.73055,w,4.5,22.50,1,IN,IN
2019-Mar-26,20:52:00,-23.56472,s,46.73055,w,4.5,22.00,1,IN,IN
2019-Mar-26,21:03:00,-23.56472,s,46.73055,w,4.5,21.75,1,IN,IN
2019-Mar-26,21:14:00,-23.56472,s,46.73055,w,4.5,21.50,1,IN,IN
```

Figura 1. Exemplo de arquivo .txt que pode der utilizado nessa função.

Planejamento da função

- Entrada: arquivo(data, horario, lat, long, aleat, pasta)

1. data = data de interesse (exemplo: 2019-Mar-26)
2. horario = horário de interesse, no seguinte formato HH:MM:SS
3. lat = latitude, em decimais
4. log = longitude, em decimais
5. aleat = seleciona um número X de arquivos ≥ 1 (se não for especificado, a função retorna todos os arquivos disponíveis baseada nos demais parâmetros)
6. pasta = nome da pasta onde os arquivos selecionados serão armazenados

- Verificando os parâmetros

1. data: verificar se o dado é válido, ou seja, dia entre 1-31 (classe = integer)
2. horario: verificar se o dado é válido, ou seja, horário $\geq 00:00$ e $\leq 23:59$ (classe = numeric)

3. lat: classe = numeric
4. log: classe = numeric
5. aleat: 0 para não utilizar a aleatorização, ≥ 1 para escolher a quantidade de arquivos (classe: integer)
6. pasta: classe = character

- Pseudocódigo

1. Entrar no diretório onde se encontra os arquivos de interesse
2. Carregar o arquivo .txt ou .csv
3. Verificar o arquivo carregado
4. Definir os valores dentro de todos os 6 argumentos
5. Verificar se os dados são válidos
6. Se "aleat" $\neq 0$, aleatorizar X arquivos dentro do intervalo de tempo definido nos outros argumentos
7. Criar pasta com o nome especificado no argumento "pasta"
8. Copiar os arquivos selecionados para a nova pasta
9. Retornar a lista com o nome dos arquivos selecionados

- Saída

1. Arquivos de interesse em uma pasta separada
2. Uma lista com o nome dos arquivos selecionados

Proposta B: Busca bibliográfica

Contextualização

Fazer uma busca bibliográfica de qualidade pode ocupar uma boa parte do (pouco) tempo disponível dos pesquisadores e estudantes. Assim, essa proposta de função tem como objetivo facilitar esse processo, ao automatizar a busca de artigos de interesse disponíveis em um banco de dados (por exemplo, Google Scholar) a partir de palavras provenientes do título e/ ou do resumo, palavras chave, ano de publicação e autor. A ideia é que a partir de um artigo X, a função seja capaz de importar os artigos que foram citados por X (artigos Y) e os artigos que foram citados pelos artigos Y (artigos Z) e assim por diante, até que o limite de artigos definidos pelo usuário da função seja atingido (Figura 1).

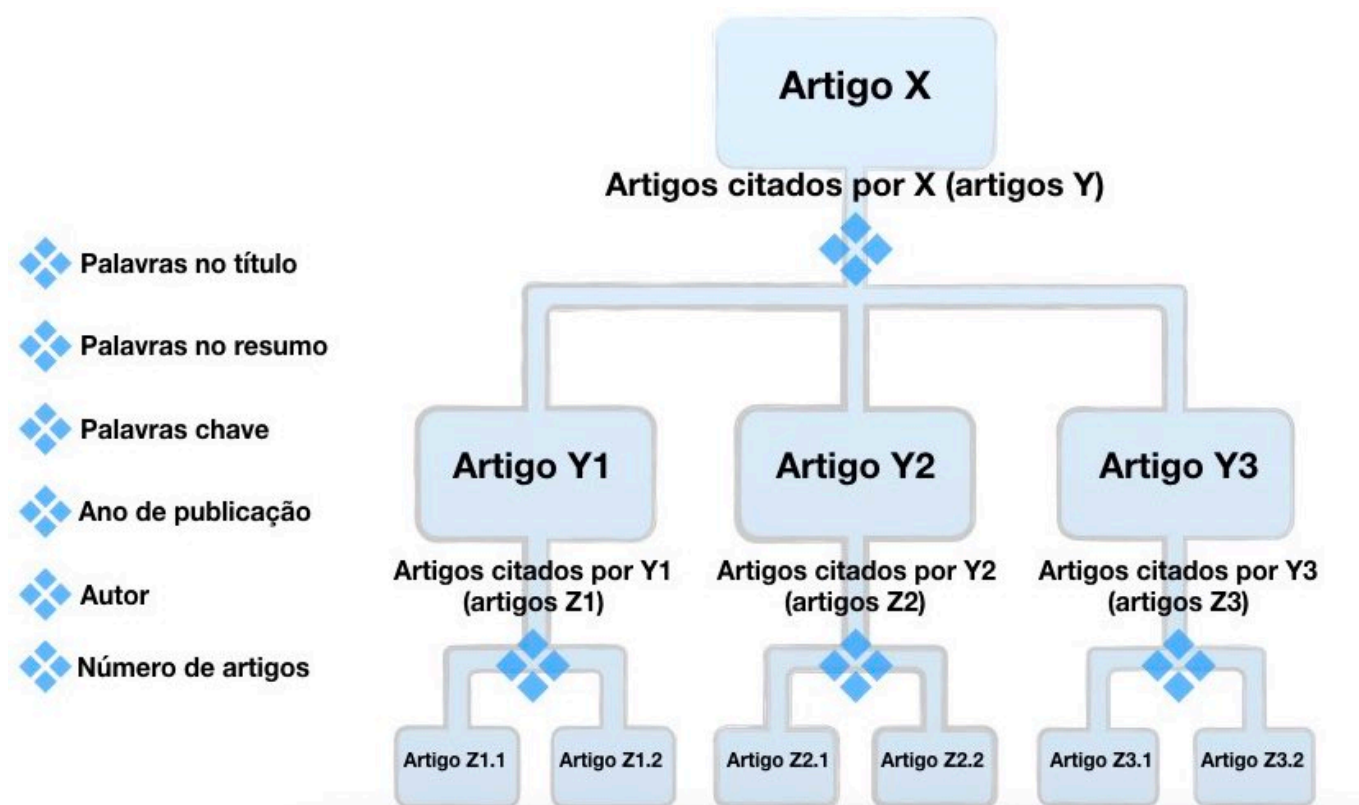


Figura 1. Esquema para representar o uso da função proposta.

Planejamento da função

- Entrada: busca(titulo, resumo, palavra.chave, ano, autor, nivel, n.artigos)
1. titulo = palavras que devem conter no título (não obrigatório)
 2. resumo = palavras que devem conter no resumo (não obrigatório)
 3. palavra.chave = palavras chave de interesse (não obrigatório)
 4. ano = ano de publicação dos artigos de interesse ou intervalo de tempo (não obrigatório)
 5. autor = sobrenome do autor dos artigos de interesse (não obrigatório)
 6. nivel = a bibliografia do artigo X (inicial) será importada, dando origem ao grupo de artigos Y, configurando nível=1. Se o usuário decidir importar a bibliografia dos artigos do grupo Y, ele irá receber artigos do grupo Z, configurando então nível=2 e assim por diante.
 7. n.artigos = número máximo de artigos

Verificando os parâmetros

1. titulo: classe = character
2. resumo: classe = character
3. palavra.chave: classe = character
4. ano: classe = numeric
5. autor: classe = character
6. nivel: deve ser ≥ 1 (classe = integer)
7. n.artigos: deve ser ≥ 1 (classe = integer)

Pseudo código

1. Carregar o artigo X (inicial)
2. Definir os argumentos opcionais e obrigatórios
3. Retornar os artigos resultantes
4. Mensagem de aviso especificando artigos não disponíveis para download

Saída

1. Arquivos dos artigos resultantes da busca bibliográfica
2. Mensagem para arquivos não disponíveis para download gratuito

[Ainda não sei exatamente como fazer para conectar o Google Scholar ao R para que a função seja capaz de extrair os arquivos da internet]

Olá Juliana. Eu achei que a sua proposta A está demasiadamente simples. Mesmo sem a sua função, com poucas linhas de comando uma pessoa consegue fazer o que a função se propõe. A ausência de controle de fluxo e modificações sistematicas na função dá uma ideia de função simples.

Acho que a sua proposta B tem mais futuro. Entendi que o seu problema é como conectar o google scholar ao R. Entretanto, o Isi web of science que, é a maior ferramenta de busca da ciência, permite que você baixe um artigo em txt contendo todas as informações dos artigos de uma determinada busca. Para cada artigo que apareceu na sua busca ele devolve informações como título, abstract, número de citações, autor, quais outros artigos esse artigo citou.

Você pode tentar usar esse arquivo. O desafio aqui vai ser conseguir jogar todas as informações desse arquivo no R de forma organizada de forma que você consiga acessar essas informações e devolver alguma coisa útil.

Alternativamente se você quiser continuar com a proposta A, eu sugiro que você tente deixar ela um pouco mais elaborada. Com contendo controle de fluxo (if) ou modificações sistemáticas de informações (for)

Se você optar pela sugestão que eu dei para a proposta B. Me mande um e-mail ou me procure na sala 243 da ecologia que eu te explico mais sobre como extrair esses arquivos do web of science.

— [Bruno Travassos](#) 2019/06/14 13:10

Função final: Seleção e leitura de arquivos de áudio

A função final passou por uma série de alterações. Não consegui trabalhar com os argumentos de horário, porém para compensar a falta deles, inclui a parte de geração de espectograma com o auxílio de pacotes próprios para isso.

[funcao.r](#)

[help.r](#)

Para testar a função, abaixo existe uma tabela que pode ser usada, assim como o link para uma pasta no google drive com uma amostra de arquivos de áudio.

[s4a08400_a_summary.txt](#)

<https://drive.google.com/drive/folders/1w06WI3woWG0pKqMQgi75MvIOH2fZC5Bg?usp=sharing>

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

Permanent link:
http://ecor.ib.usp.br/doku.php?id=05_curso_antigo:r2019:alunos:trabalho_final:juliana.primon:start

Last update: **2020/08/12 16:34**

