

Giovane Carreira Improta



Mestrando em Fisiologia, Instituto de Biociências da USP [Lattes](#)

O título de meu projeto de mestrado é “Sincronização por fotoperíodos naturais e artificiais, de verão e de inverno, em roedores subterrâneos (*Ctenomys aff. knighti*)”, orientado por Gisele Akemi Oda (IB - USP) e Verónica Sandra Valentinuzzi (CRILAR - CONICET), ambas do Laboratório Binacional (Brasil-Argentina) de Cronobiologia. Um dos objetivos do meu projeto é entender como esses animais, mesmo sendo subterrâneos, processam a informação fotoperiódica anual e transduzem essa informação em ajustes sazonais, em especial, o ajuste da duração da atividade em roda (quando em condições de cativeiro).

Meus exercícios

Página com meus exercícios resolvidos (e, com sorte, certos) [AQUI](#)

Propostas de Trabalho Final

Proposta A: Com que roupa? | `cqr()`

Contextualização

Seja para um domingo no parque, uma aula do curso do R que começa pela manhã e termina só no final da tarde ou até mesmo “pro samba que você me convidou”, muitas vezes nos perguntamos com que roupa devemos ir. Devo levar guarda chuva? Vale a pena levar uma bermuda ou blusa na mochila? Seja lá qual for a decisão que tomamos, muitas vezes estamos errados.



Na tentativa de minimizar esses erros, pensei nessa função que leva em conta a sensação térmica, a probabilidade de chuvas, a data e horário em que o usuário estará fora de casa (além da oportunidade de trocar de roupa no meio do dessa saída) que resultará em um objeto dizendo o que ele deve vestir no tronco/“ombro” [1], nas pernas/“joelho” [2], o que deve calçar/“pé” [3] e se deve ou não levar um guarda-chuva/capa [4].

Juntamente com os argumentos de entrada haverão padrões estabelecidos (porém personalizáveis) de temperaturas “de corte” que definem, por exemplo, se a pessoa deveria usar calça ou uma bermuda.

[Caso o desenvolvimento inicial dessa proposta seja fluido, pretendo incluir a possibilidade de se planejar vários dias seguidos, tendo como saída uma mala contendo vários itens de roupa (sendo que cada tipo de peça tem um padrão diferente de repetição (e.g. podemos usar uma calça jeans mais do que apenas um dia)). Apesar da ideia que a [Renata](#) deu quando procurei ajuda, ainda não consigo pensar em uma maneira simples de incluir itens que fogem do padrão “tronco/pernas/calçado”

(ombro/joelho/pé), como por exemplo vestidos. Dependendo da facilidade para se extrair os dados climáticos da internet, a localização também pode entrar como um argumento para a função]

Entrada: cqr (D, Df, H0, Hf, L, T, T01, T02, TJ1, TJ2, TP1, TP2)

- **D** Dia fornecido para início da análise (0: hoje, 1: amanhã, -1: ontem, etc.) é com base nessa data que a previsão do tempo será extraída da internet;
- [**Df** Dia final de chegada] ;
- **H0** Horário de saída;
- **Hf** Horário de chegada;
- [**L** Localidade geográfica de referência para a previsão do tempo];
- **T** Horário da troca de roupa pós saída (não obrigatório);
- **T01** Temperatura de corte para transição da peça de roupa mais arejada para a intermediária, na seção “ombro”;
- **T02** Temperatura de corte para transição da peça de roupa intermediária para a mais fechada, na seção “ombro”
- **TJ1** Temperatura de corte para transição da peça de roupa mais arejada para a intermediária, na seção “joelho”;
- **TJ2** Temperatura de corte para transição da peça de roupa intermediária para a mais fechada, na seção “joelho”
- **TP1** Temperatura de corte para transição da peça de roupa mais arejada para a intermediária, na seção “pé”;
- **TP2** Temperatura de corte para transição da peça de roupa intermediária para a mais fechada, na seção “pé”.



Figura 1. Diagrama esquemático das variáveis T'X'1 e T'X'2. Sem escala.

Verificando os parâmetros

- **D** (classe: integer)
- [**Df** Dia final da amostra (classe: integer), $Df > D$]
- **H0** horário (classe: numeric), $0 < H0 < 24$
- **Hf** horário de chegada (classe: numeric), $0 < Hf < 24$, $H0 < Hf$ no caso de serem no mesmo dia)
- [**L** ainda não sei como esse dado pode ser incorporado, talvez coordenadas de GPS]
- **T** (classe:numeric), $H0 < T < Hf$
- **T01** (classe: numeric)
- **T02** (classe: numeric), $T02 > T01$
- **TJ1** (classe: numeric)
- **TJ2** (classe: numeric), $TJ2 > TJ1$
- **TP1** (classe: numeric)
- **TP2** (classe: numeric), $TP2 > TP1$

Pseudo-código

1. A partir da data “D” [e local “L”], extrair da internet a previsão do tempo;
2. Calcular a média de sensação térmica e probabilidade de chuva dentro da janela entre H0 e Hf;
3. Usar T como referência se obter dois conjuntos de médias na janela temporal selecionada;

4. A partir dos intervalos dos conjuntos TO, TJ e TP selecionar as peças de acordo com os as faixas definidas.
5. Se a probabilidade de chuva for maior do que 65%, substituir o calçado ("pé") por bota e adicionar o aviso de guarda-chuva/capa de chuva.
6. Se a média de temperatura for menor do que 14°C recomendar, por meio de um aviso (message()), o uso de cachecol, luvas e gorro.
7. *[Se a incidência de UV estiver alta (>6) ou superior (<14) recomendar "óculos de Sol e filtro solar". Caso a base de dados forneça esse tipo de informação]*

Saída:

- Peça para o torso (inicial e pós-troca, caso haja)
- Peça para as pernas (inicial e pós-troca, caso haja)
- Calçado (inicial e pós-troca, caso haja)
- Aviso(s) no caso de chuva, frio e/ou UV extremo.
- Temperatura(s) média(s) (duas médias no caso de "troca")
- Probabilidade de chuva (duas médias no caso de "troca")

Comentários Felipe Ernesto

Fala Giovane!

Antes de tudo, bem massa a referência e o meme.

Sobre a proposta, achei interessante, principalmente se fosse pra aplicativo de celular. Acho que de modo geral atende ao que foi pedido quanto à complexidade e aplicabilidade. Porém que eu não sei direito o que pensar sobre um dos critérios: a viabilidade. Deixa eu explicar. O ponto principal da sua função é obter dados de clima da internet e depois processar esses dados de forma a responder a pergunta: "com que roupa?". Então, o passo de conseguir esses dados pode te causar problemas. Não estou dizendo que vai, pq vc não detalhou bem como pretende fazer isso, porém a chance é alta.

Ok, explicando mais a fundo. Eu pensaria em obter os dados de clima da internet de duas formas: 1) por meio de um pacote mágico do R que faz tudo pra mim; 2) procurando algum site da internet com url e layout estáveis pra que seja possível baixar o html e processá-lo. Quais são as dificuldades de cada método? 1) procurei rapidamente e achei um pacote que faria isso, porém não consegui instalá-lo, é possível que esteja descontinuado ou coisa do tipo. Além disso, não sei se esse pacote teria exatamente os dados que vc precisa; 2) Hoje em dia os sites são muito cheios de frescuras, o que faz os códigos html ficarem difíceis de processar. Por exemplo, olhei rapidamente o site do climatempo e não descobri nenhuma forma trivial de obter os dados necessários a partir do html.

Enfim, eu estou meio no escuro pq vc não detalhou muito esse passo, então pode ser que vc já tenha a solução. Mas caso não tenha pensado nisso minha dica é: só continue com o plano A caso vc encontre (o quanto antes) um pacote mágico ou um site fácil de processar, senão fuja. Digo isso pq esse nem é o foco da função e tem potencial de tomar todo o seu tempo. Vai ser bem ruim se chegar perto da data de entrega e vc nem tiver conseguido os dados de input pra poder trabalhar na sua função.

Também existe a possibilidade do usuário entrar com dados de clima e sua função processa eles. Ou seja, sem interação com a internet. Não sei se isso faria vc perder o interesse nessa proposta, mas estou pensando que caso a internet de fato venha a ser um problema esta pode ser a melhor opção (melhor até do que partir pro plano B). Aí se vc terminar a função em si rápido e ainda quiser continuar programando vc pode brigar com isso.

Alguns comentários sobre a função em si, caso decida ficar com ela:

1. No começo eu achei que a ideia era a função avisar um horário para a pessoa trocar de roupa. Porém acho que a ideia é entrar com um tempo T pré-determinado em que a pessoa sabe que vai trocar de roupa. Aí já que ela vai trocar de roupa naquele horário ela quer saber a melhor roupa. É isso?
2. Se sim, a ideia do passo 3 do pseudo-código é dividir os dados em T e depois calcular a média? Esse passo ficou meio confuso

É isso. Qualquer coisa posta aqui que eu vejo.

Abs

Resposta Giovane

Oi Felipe, tudo bem?

Obrigado de novo pela atenção e pelas dicas! Como falei na caixa de comentários da proposta B, acabei optando por um “pacote mágico no R” que vai pegar esses dados meteorológicos para mim, mas ele vai usar uma base de dados chamada “Weather Underground”, para isso, tive que pedir acesso ao API deles, me registrando no site e solicitando acesso ao “Anvil Plan” (por conta do recurso “Hourly 10-day forecast”), na categoria Developer, que é grátis, mas limita os acessos da minha API Key à 500 pedidos por dia, e 10 por minuto. Mas acredito que isso não vai ser algo realmente limitante para o propósito atual. Outra coisa que tive que me comprometer nos termos de uso, é a não divulgar para terceiros a minha chave de uso, mas acho meio inviável todo usuário ter que colocar a própria chave antes de rodar a função, por isso, incluirei a chave direto na função.

(Se for simples, talvez eu anexe um .txt com dados já extraídos para testar a função offline/sem chave API)

Talvez realmente fosse mais fácil uma solução que retirasse esses dados direto de algum site usando o html, mas infelizmente não consegui achar nada desse tipo também. Embora tenha achado [um pessoal que usou dados em XML do INPE](#), mas pelo que entendi, era no Python.

Em resposta:

1. Sim, a ideia é o usuário já saber se (e quanto) vai ter um horário de troca de roupa (por exemplo, no banho depois de uma atividade esportiva, ou um “checkpoint” na casa de alguém antes de sair para uma festa), aí a função calcularia as melhores opções de roupa dentro de cada um desses dois intervalos.
2. Sim, a ideia é cortar os dados em duas partes, aí em cada uma dessas partes calcular duas médias (uma de temperatura/sensação térmica e outra de probabilidade de chuva. Embora pensando agora, talvez seja mais válido trabalhar com a chance absoluta de chuva por que a “média” dessa probabilidade não faz sentido, se tem 0% de chance às 13h, e 100% às 14h, não significa que um guarda-chuva só seja “50% recomendado”)

Mais uma vez, obrigado Felipe Valeu!

Comentários Felipe Ernesto

Fala Giovane!

Vou responder aqui os dois comentários. Bem massa esse pacote, parece que faz tudo que vc precisa. Parece que tem tudo pra dar certo, pode mandar bala. Só lembre que se algo der errado é melhor focar na função em si e resolver isso depois, mas duvido que seja o caso.

Sobre o lance da chave, acho que qualquer uma das duas soluções é válida. E se vc passar a chave acho que não tem muito como eles perceberem, não vai ser muita gente que vai rodar ela. Mas pensando bem, o código da função será postado aqui né? Hm, talvez postar online não seja muito legal. Talvez vc possa passar por email qnd a gente for corrigir, aí no site vc deixa linkado o github caso alguém se interesse e queira rodar.

Esse link do viva o linux faz exatamente a outra solução que eu comentei, a única diferença é que está em python. Mas tem uma linha com um url do inpe. Entrei (tem que tirar o final do url) e leva pra uma página que explica como usar, basicamente vc consegue definir o local no url e cai numa página bem simples com a previsão. Parece que o climatempo usa essas páginas tb. O problema é que não tem dados de hora em hora :/

O pacote que eu tinha comentado era um oficial do R, o weatherr. Tentei instalar dnv e tb não consegui. Fui atrás e parece que uma dependência dele foi descontinuada. Uma pena, pq ele parecia bem bom tb. Acho que ainda é possível instalar, mas aí vai ser mais chatinho.

De qualquer forma esse pacote que vc achou está bom.

E caso vc precise ir pro plano B (duvido), não se assuste com manipular strings, não é mt complicado, alguns posts no fórum e vc vai ter tudo o que precisa haha

Sobre os tópicos:

1. Ok!

2. Ok. Sobre a chuva, acho que se em um dado período a probabilidade passar de um limiar pra pelo menos uma das horas do período, aí vc já pode recomendar o guarda-chuva.

Acho que é isso. Qualquer coisa só falar.

Abs

Proposta B: Língua do Pê no Érre | Ingp()

Contextualização

Uma das coisas que eu achava (e ainda acho) mais irritantes é quando as crianças começam a falar na “língua do Pê”, mesmo na variante mais básica, sempre me embananei e não entendia direito e nem conseguia “converter” português para o “peportupeguespe”.

A “língua do pê” consiste em adicionar a sílaba “pê” no começo de todas as palavras e após toda sílaba terminada em vogal.

Exemplo: Gato → PêGapêtopê

A ideia dessa proposta é fazer um conversor de português para a língua do pê.

Entrada: lngp (f)

- f “frase”

Verificando os parâmetros

- f Deve conter caracteres formando palavras e estar entre aspas.

Pseudo-código

- Dividir a frase “f” em sílabas, por meio de uma função derivada da função [syllables \(\)](#), usada para contar o número de sílabas em inglês;
- Unir as sílabas novamente com a função `paste ()` usando `sep=pe` como um dos argumentos;
- Retornar o objeto formado.

Saída:

- Frase “f”, na língua do pê básica.

Comentários Felipe Ernesto

Fala Giovane,

Achei a proposta legal, porém é um pouco simples. Na verdade não tenho tanta certeza pq não sei qual é essa função derivada da `syllables`. Suponho que ela receba uma palavra e devolva as sílabas da palavra. Aí basicamente o seu código consiste em usar essa função e depois a `paste` pra todas as palavras da frase. Não é extremamente simples pq vai envolver manipular strings (que eu acho que vcs não viram no curso) e vai ter um loop e alguns ifs pra ver se termina com vogal, mas mesmo assim dá pra resolver em poucas linhas. Além disso não têm muitas opções pro usuário. Ele só entra com um texto e recebe outro.

Enfim, isso não quer necessariamente dizer que não vai dar pra usar o plano B caso seja necessário. Basta adicionar mais complexidade. Pensei em coisas do tipo:

1. O usuário tem a opção de entrar com alguma língua a sua escolha com um argumento `lang` que pode receber qualquer sílaba, pra gerar por exemplo a língua do Tê;
2. Traduzir da língua do Pê pro português. Ok isso pode deixar as coisas mais difíceis e pode acabar sendo uma má ideia, principalmente pelo fato de que não hora de tirar os Pês vc pode acabar tirando os da própria palavra, como em apêndice. Pensei em algumas formas de evitar isso, mas teria que pensar melhor

Enfim, acho melhor ficar com a proposta A. Se der problema o lance da internet vc começa sem isso pra se garantir e depois adiciona caso sobre tempo.

Resposta Giovane

Boa Noite Felipe, Muito obrigado pelos comentários e dicas nas duas propostas! Eu já estava para responder a proposta A, mas achei que a resposta aqui seria mais rápida já que pretendo continuar com a proposta A mesmo.

Na minha busca por funções que trabalhassem com texto/divisão silábica, achei “syllables” que devolve apenas o número de sílabas de um input, mas achei que talvez por fazer essa contagem, ela saiba identificar possíveis “pontos de quebra” para, com as devidas modificações, as sílabas serem separadas. Além disso, suspeito fortemente que ela só interpreta tudo como língua inglesa, o que só seria uma solução temporária e não ideal para o começo dessa função.

Desde o começo achei a proposta A mais interessante que a B, até por isso a proposta está menos complexa. E agora lembrando dessas complicações para dividir sílabas e o medo já batendo quando você disse em ter que manipular strings, fiquei ainda menos interessado nela hahahaha

Tanto na proposta A, quanto na proposta B, os meus pontos de partida estavam nebulosos porque eu ainda não tinha uma resposta para os dados climáticos e para a divisão de sílabas, mas achei um [link no GitHub](#) detalhando como eu poderia retirar dados climáticos do “Weather Underground API”, depois de me inscrever no site e solicitar uma chave API para uso daqueles dados. Me parece que está tudo funcionando muito bem, e que eu vou ter dados previsão de hora em hora de 10 dias futuros, mas não cheguei a trabalhar com os dados como objetos no R, mas com a minha chave 27bb942b36a2f0a8 já consegui uns [dados como os do exemplo](#) nesse [vídeo do YouTube](#)

Depois de instalar o pacote “rwunderground”, eu consegui aplicar minha chave sem erros aparentes usando `rwunderground::set_api_key(“27bb942b36a2f0a8”)` então acho que sanei o problema de obter os dados, mas talvez ainda tenha que ajustar pra celsius e coisas do tipo, por isso, com muita vantagem na frente da B, a proposta A é a que eu escolhi para fazer. 😊

TRABALHO FINAL: Função `cqr()`

Considerações e mudanças do trabalho final em relação a “Proposta A”

Houveram algumas modificações em relação à proposta inicial da função principalmente porque, na ocasião da proposta, eu não fazia ideia de como/onde extrair os dados meteorológicos.

Os dados meteorológicos foram conseguidos por meio do [API do WEATHER UNDERGROUND](#), o que fez a **minha função ter como requisito a instalação do pacote “rwunderground”**. A chave gratuita limita meus pedidos de dados para 10 vezes por minuto e 500 pedidos por dia, mas não acredito que isso será um empecilho. Minha chave API de uso e a abertura do pacote já estão embutidos da função, mas a [instalação deve ser feita por fora](#) ou usando o comando `install.packages(“rwunderground”)` (há uma mensagem de erro que indica a necessidade da instalação.)

O argumento “T” (troca) foi separado em dois novos argumentos: **Dt** (dia da troca), **Ht** (horário da troca), seguindo o mesmo padrão de D e H para as datas inicial e final da previsão. O motivo dessa mudança é que T sendo apenas um “horário” não forneceria a informação de qual dia ocorre a troca de roupa em previsões que vão de um dia a outro.

O argumento L foi incluído, e deve seguir os padrões do pacote “rwunderground”. **Embora seja possível mudá-lo manualmente, configurei a função para buscar a localização com base no IP do usuário, o que é EXTREMAMENTE recomendado** (além de mais prático e exato), dado problemas para o input manual de algumas cidades (pelo menos nos dias dos testes), em especial São Paulo (e não foi por conta do til e nem do espaço). Caso

queira usar a cidade de São Paulo de maneira manual, recomendo utilizar uma estação meteorológica fixa, no caso, [a Estação Meteorológica Santa Cecilia, proxima ao cemiterio da Consolação](#), por meio do argumento `L = PWS_id="IS0PAUL0272"`. O uso de coordenadas de latitude e longitude também não se mostrou muito confiável nos testes.

Como dito na proposta inicial, caso o andamento da projeto fosse fluido (o que não aconteceu), eu iria incluir um *recurso para planejar uma mala de viagem*. Esse recurso acabou ficando de fora do projeto final por conta de tempo perdido em algumas complicações (especialmente, de problemas com a timezone e a localização "manual" dentro do `rwunderground`). Embora não tenha incluído na função, um dos modos que pensei que poderiam viabilizar essa ideia era uma série de objetos que funcionem como os pares de argumentos `Dt` e `Ht` (troca da roupa), e que subdividam a janela temporal definida entre o início e o fim da previsão. A partir dessa divisão em várias pequenas janelas de troca de roupa, e tendo um objeto "guarda-roupa", contendo as peças de roupa disponíveis para serem amostradas conforme a indicação da previsão do tempo, e contabilizar quanto de cada peça seria necessário (levando em conta que calças e blusas são socialmente (e higienicamente) mais "repetíveis" do que camisetas, por exemplo, simplificando a mala para não ter necessariamente uma roupa "nova" por dia).

Não achei alguma referência confiável de temperaturas para cada tipo de roupa para adultos ([mas para bebês, sim](#)) então usei meus próprios parâmetros, que podem diferir dos seus. Recomendo discricão no uso dos padrões pré-estabelecidos para os argumentos `T`"seção do corpo"1 e 2.

Bom uso, Giovane ;)

Argumentos de entrada: `cqr` (`D`, `Df`, `H`, `Hf`, `L`, `Dt`, `Ht`, `T01`, `T02`, `TJ1`, `TJ2`, `TP1`, `TP2`)

Padrões (como a função roda como `cqr()`):

- **D:** Dia que será daqui a 1 hora;
- **Df:** Dia que será daqui a 5 horas;
- **H:** Daqui a 1 hora;
- **Hf:** Daqui a 5 horas (4h depois de H);
- **L:** autoip - localização automática pelo IP;
- **Dt:** (ausente, sem troca);
- **Ht:** (ausente, sem troca);
- **T01:** 18 °C;
- **T02:** 32 °C;
- **TJ1:** 24 °C;
- **TJ2:** 36 °C;
- **TP1:** 10 °C;
- **TP2:** 32 °C.

Script da função `cqr()`

Acesse: [Script](#)

Help da função cqr()

Acesse: [Help](#)

From:

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

Permanent link:

http://ecor.ib.usp.br/doku.php?id=05_curso_antigo:r2018:alunos:trabalho_final:giovane.improta:start 

Last update: **2020/08/12 06:04**