Giovana de Assis Garcia

Sobre

Mestranda em Oceanografia Biológica, Instituto Oceanográfico, USP.

Minha tese de mestrado tem como objetivo estudar a reprodução do robalo-peva, *Centropomus parallelus*, estimando comprimentos e idades de maturação gonadal de ambos os estados morfo-fisiológicos (machos e fêmeas funcionais) e de inversão sexual de forma a subsidiar o manejo pesqueiro e contribuir para a preservação da espécie. Orientada pela professora Dra. June Ferraz Dias, responsável pelo laboratório de Ecologia da Reprodução e Recrutamento de Organismos Marinhos (ECORREP).

Exercícios

Página de exercícios resolvidos: Exercícios

Trabalho Final

Proposta A: Checando coordenadas geográficas

Contextualização

Na ciência oceanográfica frequentemente trabalhamos com dados não coletados por nós mesmos, devido aos elevados custos de saídas de campo e cruzeiros oceanográficos. Por este motivo, muitas vezes a qualidade do conjunto de dados não é a que esperamos ou gostaríamos. Durante as coletas, as informações de cada estação oceanográfica são anotadas manualmente em fichas e posteriormente digitadas para planilhas Excel. Nestas etapas são bastante comuns erros de anotação ou digitação, em virtude de condições do mar (e consequentemente, indisposição física do responsável pela ficha), qualidade de caligrafia do responsável pela ficha, ou até mesmo esbarrar em teclas durante a digitação. As coordenadas de cada estação oceanográfica são indispensáveis e imprescindíveis para posterior análise de dados, e não é incomum nos depararmos com coordenadas erradas que indicam estações oceanográficas no continente, do outro do oceano ou até mesmo em outro oceano.

A ideia da função insere-se neste contexto, de forma a operacionalizar e facilitar a conferência de um conjunto de coordenadas geográficas. Assim, a função lê um dataframe inserido, e compara com as coordenadas de linha de costa mundial de um pacote (Ocedata) ou com um shapefile inserido na função, com opcional de mapa dos pontos de coleta.

A função: ''conf.coord(datafile, base, fonte, mapa)''

Entrada:

> `datafile` = dataframe com coordenadas geográficas (latitude, longitude em graus) > `base` = logical. Default é TRUE , e usa como base os continentes do pacote `Ocedata`. Se FALSE, deve haver fonte. > `fonte` = shapefile de fonte a ser conferida. (só existe se `base` é FALSE) > `mapa` = default é FALSE. Se TRUE, mostra mapa dos pontos de coleta.

Verificações de premissas:

> `datafile` é dataframe de 2 colunas com “latitude” na primeira (com range entre -90 e +90) e “longitude” na segunda (com range entre -180 e +180). Se não, imprime erro “Datafile não está no formato correto, verifique no Help.”

> `fonte` só existe se base é FALSE. Se inserir fonte com outra base TRUE, imprime erro “fonte só é parâmetro se base é FALSE”

Saída:

> pares de coordenadas que não estão contidas na base em um dataframe.

Pseudo-código

- carrega pacotes (`ocedata` e `sp`)
- condicional se `base` == TRUE:
- 1. carrega linha de costa do pacote e guarda os pares de coordenadas em um dataframe chamado `cline`
- 2. separa pares de coordenadas por continente - no dataframe `cline` estão separados por `NA` - e guarda em uma lista de listas chamada `cont` com um loop baseado no número de NAs (com `wich(is.na())`)
- 3. cria polígonos com os continentes
- 4. confere se os pares de coordenadas em datafile estão dentro do polígono (com a função `over`)
- 5. guarda em um dataframe chamado `fora` os pares de coordenadas que estão fora dos continentes.
- condicional `base` == FALSE
- 1. lê shapefile de fonte em dataframe `ft` e guarda os pares de coordenadas
- 2. cria polígono de `ft` e verifica se os pontos de datafile estão dentro do polígono
- 3. guarda em um dataframe chamado `fora` os pares de coordenada que estão fora da fonte
- se `mapa`==TRUE
- 1. cria mapa da região com pontos de coleta deixando em vermelho aqueles que estão em `fora`
- retorna e imprime dataframe `fora`

Comentários Lucas Camacho

Oi Giovana, certinho?

Achei essa sua ideia simples e tem uma certa generalidade, pois como você mesmo disse na descrição existem várias fontes de erro na hora da coleta e ter uma forma de arrumar isso automaticamente é interessante. Porém eu tenho um comentário e uma dúvida (a dúvida é porque eu sou leigo em coleta de dados oceanográficos mesmo)

1. O output final são os dados que devem estar errados na planilha coletada correto? Talve fosse legal você deixar o output mais “user friendly” gerando uma nova planilha contendo todos os dados coletados e os dados errados corrigidos ou identificados nessa planilha. Seria algo que a pessoa teria que fazer na mão e você já daria esse passo no finalzinho da função.

1. Essa base de dados OceData tem dados de todas as bases oceanográficas existentes? Fiquei pensando o que aconteceria se eu coletasse em algum lugar e não tivesse a localização dela no OceData. Se isso tem chance de acontecerm talvez fosse legal colocar um warning a mais pra isso.

Proposta B: Consumo diário ideal de água

Contextualização

Todos sabemos que precisamos ingerir idealmente certa quantidade de água diariamente. No entanto, tal quantidade varia conforme idade? Sexo? Intensidade de atividade física? Tendo isto em vista, a função tem como proposta definir a quantidade ideal de água consumida diariamente para um usuário, e período de consumo de diferentes quantidades, a partir de informações dadas: idade, sexo, intensidade, período e duração de atividade física realizada. Tendo como referência “Food and Nutrition Board, Institute of Medicine.Dietary Reference Intakes for Water, Potassium, So-dium, Chloride, and Sulfate. Washington, DC: Na-tional Academies Press; 2004.” Disponível no site http://www.nap.edu/books/0309091691/html.

Como base são considerados os níveis adequados de insumo (AI) definidos por sexo e idade (Tabela 1), e a partir dos dados usados para modelo (gráfico retirado da referência) de taxas de perda de água por nível de atividade física (baseado em calorias perdidas) em diferentes temperaturas, em mL/h. Estes dados estão disponíveis no Anexo C - Tabela C-1 (https://www.nap.edu/read/10925/chapter/14#489) da referência previamente citada.

Tabela 1: níveis adequados de insumo por dia

Gráfico do modelo de taxa de perda de água por nível de atividade física e temperatura.

A função: ''cons.agua(idade, sexo, atv.fisica, per, dur)''

Entrada:

> idade = idade do usuário (número inteiro > 0)

> sexo = sexo do usuário (caractere: “F” - Female, ou “M” - Male)

> atv.fisica = intensidade de atividade física diária (caracteres: “sedentario”, “baixa”, “media” ou “alta”)

> per = período de atividade física (caracteres: “n”, “manha”, “tarde”, ou “noite”)

> dur = duração média da atividade física em horas (numérica > 0)

Verificações de premissas:

> idade é número inteiro maior que zero? Se não, imprime erro “Idade deve ser número inteiro maior que zero.”

> sexo é caractere “F” ou “M”? Se não, imprime erro “Sexo deve ser F para mulheres (female) ou M para homens (male).”

> atv.fisica é “sedentario”, “baixa”, “media” ou “alta”? Se não, imprime erro “atv. fisica deve ser uma das categorias: sedentario, baixa, media, ou alta.”

> dur é númerico >= 0? Se não, imprime erro “dur deve ser o tempo de duração média da atividade física em horas, número maior que zero. Se atv.fisica for sedentario, dur será zero.”

> se atv.fisica é “sedentario” e per não é “n” ou dur > 0, imprime warning “Para esta categoria de intensidade de atividade física, o período não é considerado (per=“n”) e duração é dada como zero (dur=0).”

Saída:

> Quantidade total de água a ser consumida diariamente, e a composição em cada um dos períodos do dia (manhã, tarde e noite).

Pseudo-código

1. cria tabela de idadeXsexo num dataframe de nome base de dimensões [8,2]
2. cria dataframe de dados para modelo
3. constrói modelo exponencial
4. constrói equações de modelo para cada intensidade de atividade física
5. lê idade e sexo, e por indexação do dataframe base, guarda em objeto AI, a quantidade de água a ser consumida (independente de atividade física)
6. lê atividade física e com condicionais (if), define a equação a ser usada
7. dentro dos condicionais que definem a equação:
8. 1.lê o período de atividade (se manhã ou noite considera temperatura de 10 a 25ºC, tarde de

25 a 40ºC)
9. 2. calcula a média de mL perdidos por hora, guarda em objeto `media.add`
10. 3. multiplica `dur` por `media.add` para obter a média de mL perdidos no tempo de duração da atividade física, e guarda em objeto `add.atv`
11. 4. soma `AI` com `add.atv` em objeto chamado `total`
12. cria dataframe chamado `per.tot` de dimensões [1,4] com colnames (“manhã”, “tarde”, “noite” e “total”)
13. em `per.tot[1,4] ← total`
14. em `per.tot[1,1] = per.tot[1,2] = per.tot[1,3] = AI/3` (divide o consumo diário independente de atividade física pelos períodos)
15. com condicional de período, soma na coluna correspondente de `per.tot`, `add.atv` (adicional por atividade física) no período de atividade física definido pelo usuário
16. retorna e imprime `per.tot`

Comentários Lucas Camacho

Essa ideia é mais complexa que a primeira e com certeza tem mais passos, mais conferência de argumentos, etc. Pelo que eu entendi você vai dizer o quanto a pessoa precisa beber de água no dia dependendo de vários fatores.

Porém, você já tem um pseudo-código para as duas, parece que ambas a ideias estão estruturadas e você já possui uma noção de como fazer. Com isso, acho que as duas propostas estão aptas a serem o seu trabalho final, mas eu ficaria com a primeira pela simplicidade dela em relação a segunda.

Qualquer dúvida ou desespero pode me mandar e-mail que eu respondo quando possível (lucas.camacho@usp.br)

Abração

Projeto final

- Página da função: Conferindo pares de coordenadas geográficas
- Página de ajuda da função: Help