Nielson Pasqualotto

Doutorando da Ecologia Aplicada da ESALQ/CENA (USP), sob a orientação do Prof. Dr. Adriano G. Chiarello. Em minha tese vou investigar o efeito do ciclo das culturas agrícolas do nordeste paulista nos padrões espaciais de ocorrência da lebre europeia, espécie exótica e potencialmente invasora, assim como sua coocorrência com o único lagomorfo nativo que ocorre no Brasil, o tapiti.

I. Meus exercícios

Aqui você pode acessar meus Exercícios Resolvidos.

II. Trabalho Final

II.I Plano A: Horário de atividade de espécies por local e data

Contextualização

Distúrbios antropogênicos podem causar mudanças de comportamento em espécies silvestres. Recentemente, há um crescente número de estudos (Gaynor et al. 2018) investigando se as espécies estão aumentando sua noturnalidade em ambientes intensamente modificados pelo homem e, portanto, evitando estarem ativas quando a maior parte da população humana também está. Todavia, investigar essa pergunta pressupões um tratamento apropriado dos registros de atividade das espécies, classificando os em diurnos, noturnos e crepusculares em função da localização geográfica e data e hora dos registros. Isso se faz necessário já que a posição do sol vista de um planeta qualquer depende, essencialmente, da posição do observador em determinado tempo, assim como de variáveis astronômicas intrínsecas do planeta em questão (e.g., velocidade de deslocamento em sua órbita em torno do Sol, ângulo entre o eixo de rotação e plano de órbita, etc.).

Frente a esse contexto, a proposta de função aqui apresentada terá como objetivo classificar os registros de atividade de uma determinada espécie foco em diurnos, noturnos ou crepusculares para qualquer localização geográfica do globo terrestre. Para isso, será utilizado como dado de entrada um `data.frame` contendo necessariamente colunas com a data e hora do registro, latitude, longitude e nome da espécie. Neste `data.frame`, cada linha corresponderá a um registro diferente (i.e., uma data e hora diferente) das espécies, podendo certamente haver mais de um registro da mesma espécie. A função retornará, para a espécie de interesse, uma `lista` contendo dois objetos. Na primeira posição haverá um `data.frame` com todos os dados de entrada mais uma coluna nova contendo a classificação dos registros em quatro níveis (diurno, noturno, crepuscular matutino e crepuscular vespertino). Na segunda posição, por sua vez, haverá outro `data.frame` contendo as variáveis astronômicas calculadas (colunas) para cada registro (linhas) e necessárias para a essa classificação, caso o usuário necessite consultá-las.

Entrada: dayNightR (`dados`, `especie`, `fuso`)

- `dados`: `data.frame` contendo pelo menos quatro colunas:
  - `time`: data, hora e fuso de cada registro em um único objeto (classe: POSIX.ct).
  - `lat`: latitude de cada registro em graus decimais (classe: numeric).
  - `lon`: longitude de cada registro em graus decimais (classe: numeric).
  - `especie`: nome da espécie foco de cada registro com uma categoria (classe: character)
- `fuso`: vetor contendo o nome do fuso horário (*time zone*) de onde os registros foram coletados (classe: character).

Verificando os parâmetros:

- O objeto `dados` é um `data.frame`? Senão, retorne a mensagem: "O objeto dados precisa ser um data.frame)".
- O objeto `dados` contém as colunas data e hora dos registros (`time`), latitude (`lat`), longitude (`lon`) e espécie (`especie`)? Senão, retorne a mensagem: "O objeto dados deve conter, no mínimo, as colunas time, lat, lon e especie)".
- A coluna `time` é da classe `POSIXct`? Senão, retorne a mensagem: "O objeto time deve ser da classe POSIXct".
- A coluna `lat` é um vetor numérico? Senão, retorne a mensagem: "O objeto lat deve ser da classe numeric".
- A coluna `lon` é um vetor numérico? Senão, retorne a mensagem: "O objeto lon deve ser da classe numeric".
- A coluna `especie` é um vetor de caracteres? Senão, retorne a mensagem: "O objeto especie deve ser da classe character".

Pseudocódigo:

1. Checa se o pacote `suncalc` está instalado. Se sim, carrega o pacote. Do contrário, para a função e retorna a mensagem de erro "pacote `suncalc` não encontrado, favor instalar".
2. Cria o objeto `dados.sp` após restringir o objeto `dados` em função da espécie de interesse.
3. Cria a coluna `date` (classe `Date`) no objeto `dados.sp` a partir da coluna `time`.
4. Cria o `data.frame` `sunNighTimes`, por meio da função `getSunlightTimes` do pacote `suncalc`, utilizando as colunas `date`, `lat` e `lon` do objeto `dados.sp`.
    1. A função `getSunlightTimes` calculará, para cada registro do objeto `dados.sp`, as variáveis astronômicas:
        - `sunriseEnd` (horário em que o sol terminou de nascer em determinado dia e local; fim do crepúsculo matutino e **início do dia**).
        - `sunset` (horário em que o sol não pode mais ser visto no horizonte em determinado dia e local; fim do dia e **início do crepúsculo vespertino**).
        - `night` (horário em que está escuro suficiente para observações astronômicas ao ar livre em determinado dia e local; fim do crepúsculo vespertino e **início da noite**).
        - `nightEnd` (horário em que não mais está escuro suficiente para observações astronômicas ao ar livre em determinado dia e local; fim da noite e **início do crepúsculo matutino**).
    2. O `data.frame` `sunNighTimes` terá, portanto, além das colunas `date`, `lat` e `lon`, colunas contendo as variáveis astronômicas acima descritas e necessárias para classificar os registro do objeto `dados.sp`.
5. Cria o vetor `dayNightTwil` contendo, em cada posição, a classificação de cada registro do objeto `dados.sp` em noturno, diurno, crepuscular vespertino ou crepuscular matutino. Para

isso:

1. Se o valor da coluna `times` do objeto `dados.sp` (data e hora do registro da espécie) é maior ou igual `nightEnd` e menor que `sunriseEnd`, classifica o registro como **crepúsculo matutino**;
2. Se o valor da coluna `times` do objeto `dados.sp` (data e hora do registro da espécie) é maior ou igual `sunriseEnd` e menor que `sunset`, classifica o registro como **diurno**;
3. Se o valor da coluna `times` do objeto `dados.sp` (data e hora do registro da espécie) é maior ou igual `sunset` e menor que `night`, classifica o registro como **crepúsculo vespertino**;
4. Do contrário, classifica cada um dos demais registros como **noturno**.

6. Armazena esse vetor como uma nova coluna do objeto `dados.sp`.

Saída:

- Uma `lista` com duas posições:
  - `data.frame`= dados de entrada com a coluna nova de classificação dos registros.
  - `data.frame`= variáveis utilizadas para a classificação dos registros.

Referências:

- Gaynor, K. M., Hojnowski, C. E., Carter, N. H. & Brashares, J. S. The influence of human disturbance on wildlife nocturnality. Science 360, 1232–1235 (2018).

II.II Plano B: Métricas de paisagem em múltiplas escalas espaciais

Contextualização

Planejamento de ações de conservação e manejo de vida silvestre tem maior chance de sucesso quando se conhece, dentre muitos outros aspectos, os requerimentos mínimos de habitat de cada espécie foco. Modelos de habitat são frequentemente utilizados para investigar tais requisitos, estimando a força de associação entre abundância e/ou ocorrência de determinada espécie com variáveis preditoras ambientais. Estudos prévios sugerem que a quantificação dessas preditoras em escalas espaciais apropriadas (i.e., escala de efeito; escala cuja força de associação entre variável preditora e resposta é maior) é determinante para se obter estimativas realistas e confiáveis (Jackson and Fahrig 2015; Miguet et al. 2016).

Fonte: Miguet et al. 2016.

Assim, o objetivo da proposta de função aqui apresentada é criar variáveis preditoras para análises de escala de efeito. As escalas espaciais serão representadas aqui por círculos concêntricos (i.e., polígonos circulares de diferentes raios), gerados ao redor de cada unidade amostral (centro dos círculos). Para isso, será utilizado como dado de entrada 1) mapa vetorial de cobertura do solo, georreferenciado, contendo a identificação de cada polígono em relação a uma das classes de cobertura (mapa vetorial de polígono; `SpatialPolygonsDataFrame`); 2) mapa vetorial,

georreferenciado no mesmo sistema de referência do mapa de cobertura, contendo a localização geográfica e o nome das unidades amostrais (mapa vetorial de ponto; `SpatialPointsDataFrame`). O usuário definirá o tamanho, dado pelo raio (i.e., 309 m e 798 m), e o nome (i.e., 30 ha e 200 ha) das escalas espaciais de interesse. Para cada raio diferente será gerado um `SpatialPolygonsDataFrame` diferente referente a cada escala espacial. Assim, em cada `SpatialPolygonsDataFrame` de cada escala haverá polígonos circulares de mesma área. Após a intersecção do mapa de cobertura do solo com cada `SpatialPolygonsDataFrame`, gerados pelos diferentes raios, será calculada a área ocupada por cada classe de cobertura do solo em relação a cada unidade amostral. A função deve retornar um `data.frame` contendo, para cada unidade amostral (linhas), a quantidade de área ocupada, em hectares, por cada classe de cobertura do solo de interesse, em cada escala espacial de interesse (colunas).

Entrada: multQuantR (`map`, `amost`, `raio.esc`, `nome.esc`, `classe.uso`)

- `map`: um mapa de uso e cobertura do solo, georreferenciado, contendo polígonos com suas respectivas classes de cobertura (classe: `SpatialPolygonsDataFrame`).
    - `Classe`: coluna do `data.frame` do objeto `map` contendo o nome da classe de cobertura do solo de cada polígono (classe: `character`).
- `amost`: mapa vetorial de ponto, georreferenciado, contendo a localização das unidades amostrais e seus respectivos nomes (classe: `SpatialPointsDataFrame`).
    - `Amostra`: coluna do `data.frame` do objeto `amost` contendo o nome de cada unidade amostral (classe: `character`).
- `raio.esc`: raios das diferentes escalas espaciais de interesse (classe: `numeric`).
- `nome.esc`: nome das escalas espaciais de interesse, apresentados na mesma ordem em que os raios foram colocados no argumento `raio.esc` (classe: `character`).
- `classe.uso`: nome das classes de cobertura do solo de interesse (classe: `character`).

Verificando os parâmetros:

- O objeto `map` é um `SpatialPolygonsDataFrame`? Senão, retorne a mensagem: “O objeto map precisa ser um SpatialPolygonsDataFrame”.
- O `data.frame` do objeto `map` contém uma coluna com o nome `Classe`? Senão, retorne a mensagem: “O data.frame do objeto map precisa ter uma coluna Classe”.
- A coluna `Classe` do `data.frame` do objeto “map” pertence à classe `character`? Senão, retorne a mensagem: “O objeto map precisa ter uma coluna Classe da classe character”.
- O objeto `amost` é um `SpatialPointsDataFrame`? Senão, retorne a mensagem: “O objeto map precisa ser um SpatialPointsDataFrame”.
- O `data.frame` do objeto `amost` tem uma coluna com o nome `Amostra`? Senão, retorne a mensagem: “O data.frame do objeto amost precisa ter uma coluna Amostra”.
- A coluna `Amostra` do `data.frame` do objeto “amost” pertence à classe `character`? Senão, retorne a mensagem: “O objeto amost precisa ter uma coluna Amostra da classe character”.
- Os sistemas de coordenadas dos objetos `map` e `amost` são os mesmos? Senão, retorne a mensagem: “Os sistemas de coordenadas devem ser idênticos”.
- O objeto `raio.esc` é da classe `numeric`? Senão, retorne a mensagem: “O objeto raio.esc precisa ser da classe numeric”.
- O objeto `nome.esc` é da classe `character`? Senão, retorne a mensagem: “O objeto nome.esc precisa ser da classe character”.
- O objeto `classe.uso` é da classe `character`? Senão, retorne a mensagem: “O objeto classe.uso precisa ser da classe character”.

Pseudocódigo:

1. Checa se o pacote `rgdal` está instalado. Se sim, carrega o pacote. Do contrário, para a função e retorna a mensagem de erro “pacote rgdal não encontrado, favor instalar”.
2. Checa se o pacote `rgeos` está instalado. Se sim, carrega o pacote. Do contrário, para a função e retorna a mensagem de erro “pacote rgeos não encontrado, favor instalar”.
3. Checa se o pacote `raster` está instalado. Se sim, carrega o pacote. Do contrário, para a função e retorna a mensagem de erro “pacote raster não encontrado, favor instalar”.
4. Checa se o pacote `reshape` está instalado. Se sim, carrega o pacote. Do contrário, para a função e retorna a mensagem de erro “pacote reshape não encontrado, favor instalar”.
5. Seleciona do objeto `map` apenas os polígonos das classes de cobertura de interesse definidas pelo argumento `classe.uso`.
6. Cria o objeto `lista`, da classe `list`, vazio.
7. Executa dentro de um `for` (inicia em 1 e vai até o comprimento de `raio.esc`):
    1. Cria, em cada ciclo, um objeto `buf` (`SpatialPolygonsDataFrame`) para cada valor de raio definido no argumento `raio.esc`. Cada objeto `buf` contém polígonos circulares de mesma área criados em torno das unidades amostrais.
    2. Cria o objeto `inters` (`SpatialPolygonsDataFrame`) contendo a intersecção do mapa de cobertura com o objeto `buf`.
    3. Cria a coluna `Area_ha` no objeto `inters`, contendo o cálculo da área em hectares de cada polígono.
    4. Cria o `data.frame` `sum.area` agregando por soma a área dos polígonos (coluna `Area_ha`) em função das unidades amostrais (coluna `Amostra`) e das classes de cobertura do solo (coluna `Classe`).
    5. Utiliza as funções `melt` e `cast` do pacote `reshape` para criar o `data.frame` `df.cast` que terá, para cada unidade amostral (coluna `Amostra`) a área ocupada pelos polígonos de cada classe de cobertura do solo (colunas com os respectivos nomes das classes).
    6. Reomeia as colunas do objeto `df.cast` para que incluam, junto ao nome das classes de cobertura (e.g. **cerrado**), o nome da escala espacial quantificada (e.g. **10ha**), facilitando a organização dos dados (e.g. **cerrado_10ha**).
    7. Guarda cada `df.cast` de cada ciclo (i.e., gerado para cada raio e respectivo nome de escala) no objeto `lista`.
    8. Fecha o `for`.
8. Adiciona um `data.frame` ao objeto `lista` contendo apenas o nome de todas as unidades amostrais (coluna `Amostra`).
9. Cria o objeto `df.merg`, por meio da função `merge_recurse`, contendo a fusão de todos os `data.frames` do objeto `lista`. Como todos eles possuem a coluna `Amostra`, a fusão será feita com base nessa coluna.
10. Cria o `data.frame` `df.merg.lin.ord` reordenando alfabeticamente as linhas do objeto `df.merg` com base na coluna `Amostra`.
11. Cria o `data.frame` `df.merg.col.sort` reordenando alfabeticamente as colunas do objeto `df.merg.lin.ord`, mas mantendo a coluna `Amostra` na primeira posição.
12. Substitui os valores `NA` do objeto `df.merg.col.sort` por zero (0).

Saída:

- Retorna um `data.frame` contendo, na primeira coluna, o nome das amostras e, nas demais, a área ocupada, em hectares, por cada classe de cobertura de interesse, em cada escala espacial de interesse.

Referências:

- Jackson, H. B. & Fahrig, L. Are ecologists conducting research at the optimal scale? Glob. Ecol. Biogeogr. 24, 52–63 (2015).
- Miguet, P., Jackson, H. B., Jackson, N. D., Martin, A. E. & Fahrig, L. What determines the spatial extent of landscape effects on species? Landsc. Ecol. 31, 1177–1194 (2016).

Vitor Rios

Nielson, interessantes suas propostas, e me parecem ser bastante úteis. Faltam só alguns detalhes pra decidir qual vc deve seguir. Uma coisa importante: sua função não deve nunca, jamais, instalar nem atualizar nada no computador do usuário sem permissão, isso pode gerar problemas sérios de compatibilidade com outros scripts. Dependências de pacotes devem ser informadas no help, e vc pode testar se os pacotes estão presentes usando a função require: if(!require(xxx)){stop("pacote xxx não encontrado, favor instalar")

A sua proposta A,precisa ser mais bem elaborada, não dá pra entender como a classificação será feita. Parabéns pela coragem de mexer com valores data e hora, isso dá um trabalho do cão pra limpar os dados, principalmente dados de terceiros. Nos item 4 e 5, especifique as variáveis astronomicas de interesse, e como elas vão ser usadas nos cálculos. O que acontece se o bicho tiver observações em horários diferentes? Vc cita colunas lat e lon mas elas não aparecem em momento nenhum no seu pseudocódigo

Na sua proposta B, ao invés de passar os arquivos nos parâmetros, é melhor o usuário carregar os arquivos em objetos antes, e passar esses objetos para a função, assim ele pode fazer as edições e manipulações necessárias. "buf.raio" deve ser um vetor de raios? Não fica claro o que vc quer dizer com "nome da escala" no nome dos objetos. Me parece que sua função vai apenas chamar funções já existentes no pacote sp e juntar os resultados num objeto de saída, é isso? Se não for isso, explicite melhor os cálculos que você vai fazer, e as estruturas de controle que vai usar.

Quando vc tiver feito as modificações, me envie um email para eu olhar novamente Vitor Rios

Nielson Pasqualotto

Olá, Vitor.

Antes de mais nada, muitíssimo obrigado pelas sugestões. Elas foram muito úteis para que eu pudesse ajustar as propostas. De maneira geral, seguindo sua sugestão, removi a instalação de pacotes e adicionei apenas um controle de fluxo para checar se eles já estão instalados. Além disso, busquei elaborar detalhadamente

ambas propostas para que fique mais claro a sequência de modificações que pretendo executar.

Em relação à proposta A, adicionei as variáveis astronômicas a partir do item 4 do pseudocódico e mencionei explicitamente as colunas lat e lon. Sobre o bicho (espécie?) tiver observações em horários diferentes, não há problema, já que a proposta é classificar a atividade da espécie que pode apresentar variações (hora registrada no final da tarde, hora à noite, etc.) em função do grau de antropização das unidades amostrais. Especificamente, o data.frame dados pode conter, de fato, várias observações (linhas) de uma mesma espécie, as quais serão classificadas normalmente. Além disso, gostaria de destacar que não planejei a função para executar "limpeza" de dados de data e hora. A função está pensada para receber, dentre outros dados, um objeto POSIXct (coluna time) já manipulado/configurado, restando ao usuário informar o fuso via argumento da função. Tudo bem quanto a isso?

Da mesma forma, após sua sugestão, a proposta B terá como dados de entrada mapa vetoriais já carregados no R. Não mais carregarei esses mapas dentro da função, possibilitando que o usuário os edite/manipule antes de rodar a função. Sobre o argumento "buf.raio", ele é sim um vetor com o tamanho dos raios. Alterei seu nome para raio.esc (raio da escala espacial), pois achei mais informativo. Melhorei a definição desse argumento, assim como do argumento nome.escala (agora editado para nome.esc) e inclui uma melhor descrição do controle de fluxo e de modificações que ocorrerão dentro da função.

Aguardo seu retorno para prosseguir.

Muito obrigado.

Vitor Rios

Nielson, suas propostas estão ótimas. Você pode seguir com a que preferir. Eu sugeriria vc seguir com a proposta B, mais por uma questão de gosto pessoal mesmo. Vitor Rios

III. Links para acessar o trabalho final

Para o trabalho final escolhi desenvolver o Plano B (Métricas de paisagem em múltiplas escalas espaciais) criando a função `multQuantR`. A função, como já mencionado, realiza a quantificação da área (em hectares) ocupada por classe de cobertura do solo de interesse em múltiplas escalas espaciais circulares de diferentes tamanhos (raios).

- Link para a acessar o código da minha função: multQuantR
- Link para a acessar o help da minha função: Help