Trabalho final

Olá Douglas,

Sou Gustavo A. Ballen, monitor encargado de dar um retorno sobre suas propostas. Vou comentar em geral aqui sobre as duas propostas.

Da para perceber que você conhece o R como ferramenta SIG e os pacotes para realizar tais análises; consequentemente, as duas se parecem bastante e apresentam saídas promisorias. Entretanto, em nenhuma das duas funções fica clara qual ira ser a sua contribuição como programador, ou por quê o procedimento desenvolvido pela função é original de alguma maneira. Em geral fica a impressão que a função só vai utilizar ferramentas preexistentes nos pacotes GIS indicados e integrá-los para gerar uma saída composta de diversas informações.

Por gentileza reformule as duas propostas dando enfase à sua contribuição em cada caso, e se for o caso deixe bastante claro qual ira ser o papel dos pacotes citados dentro da função. Se os pacotes forem os responsáveis principais pelos procedimentos que você precisa implementar na função, então acho melhor pensar numa terceira proposta ou então rever as que já estão disponíveis. Até então nenhuma das duas pode ser escolhida como função de trabalho final.

Gustavo A. Ballen

Olá Gustavo, muito obrigado pelo feedback.

Como conversamos por e-mail, editei a proposta A e deixei o pseudocódigo mais claro, aguardo seu retorno.

Proposta A - “RoLA” – Roadkill Landscape Analysis

Contextualização

O atropelamento de fauna é um dos grandes problemas na perda de indivíduos da fauna silvestre em todo o mundo, no Brasil estima-se que mais de 14 milhões de vertebrados silvestres morram todos os anos em atropelados em rodovias. Esses números podem variar de acordo com as estimativas, mas de modo geral são baseados em números de estudos que buscam monitorar sistematicamente rodovias em todo o país em busca de carcaças, para entender onde, quais e como esses animais morrem.

Figura: *Cerdocyon thous* atropelado.

Diversos fatores podem influenciar a mortalidade de animais em rodovias, desde o grupo taxonômico, o hábito alimentar, a hora de atividade no dia, o comportamento gregário ou solitário e até o tipo de habitat. Um dos fatores mais estudados no atropelamento de fauna é a paisagem, pois é nela que os animais se movimentam apresentando preferências ou repulsa por alguns tipos de vegetação ou uso do solo, e deste modo, entender a paisagem pode ajudar a entender o movimento da fauna.

Um dos principais objetivos dos estudos de mortalidade de fauna em rodovias é a proposição de medidas mitigadoras, como redutores de velocidade, cercamento, passagens de fauna, entre outros, que visam diminuir o número de colisões entre veículos e a fauna, diminuindo assim a mortalidade e a perda de espécimes silvestres e aumentando a segurança dos usuários da rodovia. Deste ponto de vista, analisar o tipo de paisagem onde os atropelamentos ocorrem podem nos ajudar a tomar decisões de onde e como implementar essas medidas de prevenção e mitigação. Após o monitoramento e o levantamento da fauna atropelada numa rodovia, podemos utilizar os pontos georreferenciados dos atropelamentos e analisar a paisagem circundante ao evento, permitindo entender quais fatores naquela paisagem podem estar influenciando o atropelamento de fauna.

Para esse tipo de análise, em geral, os pesquisadores e consultores ambientais, usam ferramentas de geoprocessamento, criando um buffer de influência a partir do ponto de atropelamento e utilizando um mapa de uso do solo para determinar qual a paisagem na região. Mas imagine que você tem um branco de dados do monitoramento de uma rodovia, com centenas de ocorrências, e você gostaria de olhar para a paisagem em todos estes pontos, talvez você tenha que criar muitos buffers e extrair muitas vezes as informações do uso do solo. Sendo assim, seria útil uma ferramenta que faça isso automaticamente, carregando o ponto, o tamanho do raio de influência e o mapa de uso do solo, te fornecendo as quantidades relativas de uso envolta de cada um dos atropelamentos.

Figura: Exemplo de *buffer* de influência e uso do solo ao redor de atropelamentos

Além disso, uma ferramenta assim pode ser útil para outros tipos de estudos ambientais em que se tem um ponto georreferenciado e se quer saber o uso do solo envolta deste ponto, como por exemplo um ponto de ocorrência de algum animal ou planta, onde queremos saber qual o habitat e a matriz onde ele foi observado; o ponto de uma nascente de rio, para saber quanto por cento da APP da nascente está conservada; um ponto de localização de colar de GPS de um animal, para saber por onde ele andou; o ponto de implantação de uma armadilha fotográfica e até mesmo o ponto de uma rodovia em que pretendemos implementar uma passagem de fauna.

Planejamento da função:

Pacotes necessários:

“raster” “sp”

Entrada:

```
RoLA (y, x, buffer, graph = TRUE, landscape.data)
```

• `y:` 'objeto' ou 'vetor' com a latitude do(s) ponto(s) (em graus decimais);

• `x:` 'objeto' ou 'vetor' com a longitude do(s) ponto(s) (em graus decimais);

• `graph: default =` TRUE, retorna o mapa gráfico do buffer;

• `landscape.data:` arquivo raster (.tiff) contendo o uso do solo na região estudada;

Pseudocódigo:

1. verifica se `landscape.data` é um arquivo .tif – caso não, retorna uma mensagem de erro;
2. guarda num `data.frame` os valores e as frequências de valores do arquivo raster;
3. guarda a projeção do arquivo raster em um objeto (será utilizada para criação dos pontos);
4. guarda o valor - em metros - do buffer escolhido pelo usuário em um objeto;
5. cria um `data.frame` com uma coluna de ID, uma coluna de latidude, uma coluna de longitude e uma coluna para cada categoria de uso presente no raster;
6. função `for`, para cada valor de `x` e `y`;

- 6.1. preenche o ID, lat e long nas colunas respectivas;
- 6.2. cria um ponto com as coordenadas ordenadas de `x` e `y` - na mesma projeção do raster;
- 6.3. à partir do ponto cria um circulo com raio estipulado pelo `buffer` fornecido pelo usuário;
- 6.4. utiliza o círculo criado para cortar o arquivo `landscape.data`;
- 6.5. guarda as frequências relativas àquela região (buffer) em um objeto `data.frame`;
- 6.6. função `if`: caso o usuário tenha pedido um mapa gráfico, produz um mapa para cada ponto, com a área de influência e os usos presentes nele (garante que todos os mapas atribuam as mesmas cores aos respectivos usos);
- 6.7. cria e reinicia contadores;
- 6.8. função `for`, contador `j` preenche as colunas do `data.frame` final com as frequências relativas de cada categoria de uso;
  - 6.8.1. função `if`: caso o buffer possua a categoria presente no arquivo raster, preenche o respectivo valor no `data.frame`;
  - 6.8.2. função `else`: caso o buffer não possua a categoria presente no arquivo raster, preenche com zero;

- retorna o laço para o `for` para realizar o procedimento para as coordenadas `x` e `y` seguintes;

Saída:

1. retorna um `data.frame` com os valores em porcentagem para cada valor de pixel e cada ponto; (Atenção: os valores do pixel para cada uso, precisam ser conhecidos pelo usuário, como no exemplo à seguir, que são os valores de uso por pixel do MapBiomas: exemplo);
2. caso o argumento `graph` seja verdadeiro, retorna um mapa gráfico com o buffere cada uso, todos na mesma paleta de cores pros valore iguais (ou seja, se há um valor “3” em um mapa, o mesmo valor “3” estará com a mesma cor no mapa de outro ponto).

O que é produzido por pacotes de GIS e o que é programado?

O pacote de GIS cria pontos, lê rasters e cria shape.files, a programação consistirá em:

1. calcular as frequências relativas do arquivo raster;
2. produzir controles de fluxo, garantindo que independente da quantidade de coordenadas x e y que o usuário entre o programa leia e extraia os dados do arquivo raster;
3. permitir que para cada ponto se calcule as proporções de uso na área de influência (indicada pelo buffer);
4. permitir que para cada categoria de uso presente no raster haja uma coluna respectiva no fata.frame final;
5. preencher corretamente cada par de coordenadas (linhas) com as categorias de uso (colunas), independente da área de influência possuir, uma, duas ou todas as categorias do raster original;
6. criar objetos e contadores para o preenchimento adequado da tabela;
7. criar mapas gráficos, com coordenadas, legendes e cores correspondentes entre si, independente do tamanho do buffer ou independente do número de categorias de uso;

As principais funções utilizadas pelos pacotes GIS serão:

- `raster` - lê um arquivo raster;
- `SpatialPoint` - cria um ponto a partir de coordenadas (é necessário identificar e indicar o tipo de projeção cartográfica);
- `mask` e `crop`, cortam um arquivo espacial à partir de outro;

As principais funções de programação em R serão:

- `for` - para percorrer todos os pontos, independente da quantidade deles e para percorrer toda a tabela a ser gerada ao final do procedimento;
- `if` - para verificar o tamanho e tipo dos arquivos fornecidos pelo usuário; para verificar dentro do `for` se a categoria de uso é correspondente antes de preencher a tabela; para verificar se o usuário quer ou não um resultado gráfico;
- `data.frame` - cria e manipula data frames;
- `plot` e `par` - cria e manipula gráficos, e nesse caso garantirá que todos os gráficos tenham exatamente o mesmo padrão de cores, de legendas e tamanho, independente do número de categorias que possuir;
- contadores - irá utilizar diversos contadores para garantir o preenchimento e extração adequada dos dados;]
- `as.character`; `as.numeric`, e outros manipuladores de dados como `[]`, `sum`, `freq`, `c()`, ==; & ; etc - para garantir a extração adequada de dados.

FUNÇÃO FINAL - A

FUNÇÃO: Função RoLA - Final

```
#######################################################
####FUNÇÃO RoLA - Roadkill Landscape Analysis########
#######################################################

RoLA <- function (landscape.data, lat, long, buffer.radius,  map.graph=T,
pie.graph=F) #nome da função e seus argumentos;
{
```

```
  x<-long #atribui o vetor de longitudes a um objeto x
  y<-lat  #atribui o vetor de latitudes a um objeto y
  if(length(x)!=length(y)){
    stop('Os vetores de Latitude e Longitude devem ter o mesmo tamanho e ser
ordenados')  #verifica o tamanho dos vetores e pára caso não sejam do mesmo
tamenho
  }
  if (require("raster")!=TRUE)
  {stop('É necessário instalar o pacote "raster" ')} #verifica se o usuário
possui o pacote "raster"
  if (require("sp")!=TRUE)
  {stop('É necessário instalar o pacote "sp" ')}  #verifica se o usuário
possui o pacote "sp"
  land.data <- raster(landscape.data) #atribui o arquivo raster à um objeto
  land.data[land.data==0]<-NA  #substitui os zeros por NA's no arquivo
raster
  projection <- as.character(land.data@crs) #guarda a projeção do arquivo
raster em um objeto
  freq_land.data <- as.data.frame(freq(land.data, useNA="no")) #mede as
frequências de cada categoria de uso presentes no mapa raster
  m <- as.numeric(length(freq_land.data$value)) #guarda o número de
categorias do mapa raster num objeto
  k <- freq_land.data$value  #guarda em um vetor com todos os valores de uso
do mapa raster
  col=(terrain.colors(m)) #cria um objeto com as cores padronizadas de
acordo com o número de categorias no mapa raster
  names(col) = c(k) #coloca o nome de cada uso do raster em cada respectiva
cor criada
  buffer.size <- buffer.radius #guarda o tamanho do buffer fornecido em um
objeto
  n.points <- as.numeric(length(x)) #guarda o número de pontos ordenados
(lat, long) em um objeto
  tabmatrix <-matrix(ncol=3+m, nrow=n.points) #cria uma matriz de dados com
o número de colunas igual aos usos do raster + 3, e com o numero de linhas
igual o de pontos
  tab.fin <- as.data.frame (tabmatrix) #transforma a matriz em um data.frame
  colnames(tab.fin) = c("ID", "lat", "long", c(k)) #renomeia as colunas do
data.frame - as 3 primeiras com Id, lat e long
  for (i in 1:n.points){  #inicia um loop que vai de 1 até o número de
coordenadas (lat,long)
    j <- 0 # zera o contador j
    l <- 4 #atribui 4 ao contador l (que preencherá as colunas)
    p <- 1 #contador p para a posição das frequencias no buffer
    tab.fin[i,1]=i #atribui o ID da linha
    tab.fin[i,2]=y[i]    #atribui a latitude da linha
    tab.fin[i,3]=x[i]    #atribui a longitude da linha
    ponto<-SpatialPoints(cbind(x[i], y[i]), proj4string= CRS(projection))
#cria um ponto com a coordenada lat, long ordenada
    influencia <- buffer(ponto, width=buffer.size) #cria um buffer circular
com um raio determinado à partir do ponto
    cort <- mask(land.data, influencia) #corta o mapa raster com o buffer a
```

```
partir do ponto
    uso<-crop(cort, influencia) #recorta o mapa do buffer ppelo tamanho
exato do buffer
    frequencias <- as.data.frame(freq(uso, useNA="no")) #guarda um
data.frame com os valores de uso(categorias) e as frequências dentro do
buffer
    z <- (frequencias[,1]) #guarda um vetor com as categorias de uso
    f <- (frequencias[,2]) #guarda um vetor com as frequências de uso
    n <-as.numeric(length(frequencias$value)) #guarda a quantidade de
categorias de uso no buffer
    tot_val=sum(freq(uso, useNA="no")) #guarda o tamanho total do buffer em
área, com todas as categorias
    if (map.graph == T){ #se o argumento gráfico é verdadeiro
      x11() #abre uma janela gráfica
      par(xpd=F, mar=c(5,4,4,4.5)) #atribui os parâmetros de margem
      cl = col[as.character(c(z))] #atribui as cores do mapa gráfico de
acordo com o arquivo raster e o buffer
      plot(uso, col=col, breaks=c(1,c(k)),legend=F, main=c(y[i], x[i]))
#plota um mapa da região do buffer
      par(xpd=T)
      legend(par()$usr[2],par()$usr[4], legend=c(z),fill=cl) #aficiona a
legenda no canto superior direito do mapa
      par(new=T)
      plot(ponto, col="red") #plota o ponto (lat, long)
      if (pie.graph == T){ #se o argumento gráfico é verdadeiro
        x11() #abre uma janela gráfica
        par(xpd=F, mar=c(5,4,4,4.5)) #atribui parâmetros de margem
        cl = col[as.character(c(z))] #atribui cores do gráfico de acordo com
o arquivo raster
        pie(frequencias[,2],labels =
round((frequencias[,2]/tot_val*100),2),col = cl, radius = 1, main = c(y[i],
x[i])) #gera um gráfico de pizza com as porcentagens
        par(xpd=T)
        legend(par()$usr[2],par()$usr[4], legend=c(z),fill=cl) #adiciona uma
legenda por cores
      }
    }
    for (j in k){  #para cada j com os valores do raster original
      if (j == z[p] & j <= z[n]){  #verifica se o tamanho do vetor é menor
ou igual ao contador p e se o contador j é igual ao valor de z
        tab.fin[i, l] = round((f[p]/tot_val*100),4)  #atribui a frequência
do uso à respectiva coluna do data.frame final
        p=p+1 #soma 1 no contador
        l=l+1 #soma 1 no contador
      }
      else{
        tab.fin [i,l] = 0 #adiciona zero à coluna do uso que não aparece no
buffer
        l=l+1 #soma 1 ao contador
      }
```

```
    }
  }
  return(tab.fin)
}
```

DOCUMENTAÇÃO: R Documentation - RoLA

```
                              R documentation

RoLA – Roadkill Landscape Analysis

Description:
Gera um data.frame com as frequências relativas –em porcentagem-  de uso do
solo dentro de um buffer de tamanho e ponto central determinado pelo
usuário;

Usage:

       RoLA(landscape.data"raster.tif", lat=y, long = x, buffer.radius=4000,
map.graph = TRUE, pie.graph = FALSE)



Arguments:

        landscape.data: um arquivo/mapa do tipo raster (formato .tif)
contendo os valores de uso do solo na região
        de estudo;

        lat: um objeto numérico ou vetor contendo a(s) latitude(s) do(s)
ponto(s) a ser(em) utilizado(s) - em GRAUS
        DECIMAIS;

        long: um objeto numérico ou vetor contedo a(s) longitude(s) do(s)
ponto(S) a ser(em) utilizado(s) - em
        GRAUS DECIMAIS;

        buffer.radius: tamanho – em metros – do raio do buffer a ser gerado
à partir do(s) ponto(s) central(is);
        map.graph: default=TRUE – argumento lógico TRUE/FALSE, determina se
a saída será acompanhada de um mapa da
        região com os usos do solo;

        pie.graph: default=FALSE – argumento lógico TRUE/FALSE, determina se
a saída será acompanhada de uma
        gráfico tipo pizza com as porcentagens relativas de cada uso dentro
do buffer;
```

```
Details:

       O buffer será gerado no tamanho informado à partir do(s) ponto(s)
dado(s), nas coordenadas ordenadas de
       latitude e longitude. O arquivo raster será a base da informação do
uso do solo, fornecido pelo usuário. A
       saída é um data.frame com as frequências relativas de cada valor
presente no arquivo raster fornecido, deste
       modo o usuário deve conhecer o significado de cada valor de seu
arquivo raster.

       O argumento map.graph = TRUE gera um mapa, com um círculo de raio
igual ao buffer.size com todos os usos
       presentes dentro dessa área e uma legenda;

       O argumento pie.graph = TRUE gera um gráfico de pizza, com as
porcentagens relativas de cada uso dentro do
       buffer;
       As cores de cada uso no mapa e no gráfico de pizza, são equivalentes
entre si.



Value:

      *Um data.frame contendo na primeira coluna um ID numérico; na segunda
coluna a latitude do ponto; na terceira
       coluna a longitude do ponto; à partir da quarta coluna a porcentagem
relativa do uso do solo obtida à partir
       do arquivo raster fornecido;

       *Caso o argumento map.graph seja verdadeiro, gera um mapa, delimitado
pelo buffer, contendo os usos do solo,
       com legenda e referência geográfica;

       *Caso o argumento pie.graph seja verdadeiro, gera um gráfico de pizza
com as porcentagens relativas de cada
       uso;



Warning:

      Para o funcionamento adequado da função é necessário ter instalado os
pacotes “sp” e “raster”;

      Todos os pontos de coordenadas geográficas devem ser fornecidos em
GRAUS DECIMAIS;

      Garanta que seus pontos de coordenadas se sobrepõem à mesma região que
```

```
o raster que você forneceu;

      Arquivos raster são arquivos pesados, sua função pode demorar para
rodar, aguarde o R terminar o
      processamento!



Author(s):
    Douglas W. Cirno douglaswcirino@hotmail.com



See also:

       SatialPoints, mask, crop, buffer



Examples:

       #para os arquivos de raster disponíveis no link:

       Ex. 1 – Atropelamento no Rodoanel de SP:
       # RoLA(landscape.data="SP_2017.tif", lat=-23.805440, long=-46.694391,
buffer.radius=2500, map.graph = TRUE,
       pie.graph = TRUE)


       Ex. 2 – Atropelamentos no MS:

       #lat1<- c(-20.28403, -20.19575, -20.13389, -20.45842, -20.5501,
-20.6926, -21.1363, -21.42647)
       #long1<- c(-56.302455, -56.44354, -56.67615, -55.2395, -55.5366,
-55.7840, -55.8203, -56.46465)

       #RoLA(landscape.data="MS_2016.tif", lat=c(lat1), long=c(long2),
buffer.radius=2260)


       #Ex. 3 – APPS de nascentes de Itu-SP (Mapa FBDS – resolution:5m)

       #nasc_lat = c(-23.273701, -23.254287, -23.272466, -23.355888,
-23.353230, -23.317253, -23.258457)
       #nasc_long = c( -47.269918, -47.252207, -47.359948, -47.414180,
-47.358264, -47.415235,  -47.366648)

       #APP_ITU<- RoLA ("Itu_WGS84.tif", lat = c(nasc_lat), long =
c(nasc_long), 100, map.graph = T, pie.graph = T)
```

```
      #Ex. 4 – APPS de nascentes de Itu-SP (Mapa MApBiomas -
resolution:30m)

      #nasc_lat = c(-23.273701, -23.254287, -23.272466, -23.355888,
-23.353230, -23.317253, -23.258457)
      #nasc_long = c( -47.269918, -47.252207, -47.359948, -47.414180,
-47.358264, -47.415235,  -47.366648)

      #APP_ITU<- RoLA ("ITU_2017.tif", lat = c(nasc_lat), long =
c(nasc_long), 500, map.graph = F, pie.graph = T)
```

EXEMPLOS E ARQUIVOS PARA TESTE

Exemplo 1

Atropelamento no Rodoanel de SP

Arquivo raster (mapa de SP): SAO_PAULO_2017.tif Legenda do arquivo raster: MapBiomas 3.1 - Legenda

Script R:

```
atrop_rodoanel <- RoLA(landscape.data="SAO_PAULO_2017.tif", lat=-23.805440,
long=-46.694391, buffer.radius=2500, map.graph = TRUE, pie.graph = TRUE)
```

Exemplo2

Atropelamentos no Pantanal

Arquivo raster (mapa de uma região do MS): MS_2016.tif Legenda do arquivo Raster: MAPBiomas 3.0 - Legenda

Script R:

```
lat1<- c(-20.28403, -20.19575, -20.13389, -20.45842, -20.5501, -20.6926,
-21.1363, -21.42647)
long1<- c(-56.302455, -56.44354, -56.67615, -55.2395, -55.5366, -55.7840,
-55.8203, -56.46465)

atrop_pantanal <- RoLA ("2016_MS.tif", lat = c(lat1), long = c(long1), 2260,
map.graph = T)
```

Exemplo 3

APP's de nascentes de Itu com Mapa de 5m de resolução

Arquivo Raster (uso do Solo do FBDS para Itu): ITU_WGS84.tif Legenda do arquivo Raster: Legenda FBDS - Itu

Script R:

```
nasc_lat = c(-23.273701, -23.254287, -23.272466, -23.355888, -23.353230,
-23.317253, -23.258457)
nasc_long = c( -47.269918, -47.252207, -47.359948, -47.414180, -47.358264,
-47.415235,  -47.366648)

APP_ITU<- RoLA ("Itu_WGS84.tif", lat = c(nasc_lat), long = c(nasc_long),
300, map.graph = T, pie.graph = T)
```

Exemplo 4

APP's de Nascentes de Itu com mapa de 30m de resolução

Arquivo Raster (uso do solo MapBiomas para Itu): ITU_2017.tif Legenda do arquivo raster: MapBiomas 3.1 - Legenda

Script R:

```
nasc_lat = c(-23.273701, -23.254287, -23.272466, -23.355888, -23.353230,
-23.317253, -23.258457)
nasc_long = c( -47.269918, -47.252207, -47.359948, -47.414180, -47.358264,
-47.415235,  -47.366648)

APP_ITU<- RoLA ("ITU_2017.tif", lat = c(nasc_lat), long = c(nasc_long), 500,
map.graph = F, pie.graph = T)
```

Proposta B - "HFFF" - How Far from a Forest?

Contextualização

O desmatamento é um dos principais problemas ecológicos que vivenciamos e está associado a diversos problemas ambientais de nosso cotidiano, desde a proliferação de doenças, o descontrole de populações de pragas, a má qualidade da água e principalmente as mudanças climáticas.

Figura: Desmatamento na Amazônia de 1984 a 2016

As florestas são responsáveis por diversos serviços ecossistêmicos em nossa dia-a-dia, desde o controle da temperatura e a produção de chuvas, até mesmo efeitos que nem percebemos, como o bem-estar e a saúde, sobretudo mental e cardiovascular. Parte disso se dá pelo que Edward O. Wilson chamou de Biofilia, que seria a atração pela natureza e atributos naturais. Por outro lado, a cada minuto que se passa nos tornamos mais distantes das florestas, seja pela vida agitada seja pela população humana estar em sua maioria em centros urbanos.

Devemos buscar saber qual é a floresta, por menor que ela seja, que está mais perto de nós, da nossa casa, do nosso trabalho. E precisamos ter a consciência de quanto de floresta é perdido no mundo a cada minuto de nossas vidas. Através da entrada de sua data de nascimento e de sua coordenada geográfica de localização, conseguimos calcular o quanto que você está longe de uma floresta nativa e estimar o quanto de desflorestamento aconteceu do dia que você nasceu até o presente. Essa é uma forma de conscientizarmos as pessoas de que precisamos urgentemente nos preocupar com as questões ambientais. Mas nem tudo são pesadelos, muitas florestas no mundo todo, inclusive no Brasil estão se regenerando aos poucos, apesar do número absoluto ser maior de desmatamento do que de regeneração, algumas ações ambientais podem mudar esse cenário em algumas décadas, então podemos saber também quanto de florestas recuperamos desde o dia em que pisamos nesse mundo! O Brasil é o país com maior quantidade de florestas tropicais no mundo e detém a maior biodiversidade do planeta, portanto estimar o quanto o nosso país está perdendo e ganhando de florestas é essencial para entendermos como vai nosso planeta e nossa biodiversidade.

Figura: Perda de florestas no Brasil (Fonte: MapBiomas)

Planejamento da função

Pacotes necessários

“maptools” “raster” “rgdal” “GISTools”

Arquivos necessários:

Arquivo de uso do solo no Brasil (MapBiomas);

Entrada:

HFFF(forests, birth, x, y)

• forests: nome do arquivo .tif com florestas salvo no repositório;

• birth: dia do seu nascimento (na estrutura dd/mm/aaaa)

• y: latitude do ponto onde você está ou quer medir a distância de uma floresta (em graus decimais);

• x: longitude do ponto onde você está ou quer medir a distância de uma floresta (em graus decimais);

Pseudocódigo

1. Abre o arquivo raster `forests` a partir do diretório do usuário – previamente baixado.
2. Verifica se `x` e `y` estão em graus decimais – caso contrário retorna uma mensagem de erro;
3. Verifica se a data de nascimento `birth` está no formato correto – caso contrário retorna uma mensagem de erro;
4. Mede a distância do ponto informado até o pixel de floresta mais próximo;
5. Plota um mapa em raster com as florestas do Brasil;
6. Insere uma linha do ponto dado até o pixel de floresta mais próximo;
7. Calcula o número de dias que passaram do nascimento até o dia atual;
8. estima um valor em hectares de florestas desmatadas no Brasil, segundo estimativas do MapBiomas desde o dia do nascimento;
9. estima um valor de regeneração florestal do dia do nascimento até o presente dia;

Saída

1. Retorna uma lista com:
    - valor numérico em hectares de quanto de floresta foi perdido no Brasil desde o dia de seu nascimento;
    - valor numérico em hectares de regeneração florestal no Brasil desde o dia do seu nascimento;
    - valor numérico em quilômetros do ponto até a floresta mais próxima e o tamanho dessa floresta;
2. Retorna o plot de um mapa das florestas, com o ponto e uma linha reta até a floresta mais próxima;