Ecologia de Paisagens com R

Darren Norris: darren.norris@unifap.br

12 maio, 2023

Sumário

Part I

Apresentação

Bem-vindos

Este é um trabalho em andamento do 1ª edição: **"Ecologia de Paisagens com R"**.

Versoes:

- web: https://darrennorris.github.io/epr/
- pdf: https://github.com/darrennorris/epr/blob/main/docs/epr.pdf

Este é um material introdutório destinado principalmente a estudantes de graduação e cursos de pós-graduação em ecologia e áreas correlatas.

O objetivo é de apresentar os capacidades e opções para desenvolver e integrar pesquisas na Ecologia da Paisagem no ambiente estatística de R.

Esperamos que ele seja utilizado tanto por quem quer se aprofundar em análises comumente utilizadas em Ecologia da Paisagem, mesmo por quem tem poucas habilidades quantitativas.

Agradecimentos

Este livro não é apenas o resulto dos autores. Mas é o resultado de muitas pessoas na comunidade R e Ecologia da Paisagem no Brasil. Muito obrigado!

Prefácio à primeira edição

Há quem diga que a velocidade com que a tecnologia e a ciência avançam tende a tornar livros e manuais sobre métodos rapidamente obsoletos. A evolução dos computadores pessoais e a ampliação do acesso a estes e à Internet têm transformado o jeito como aprendemos e ensinamos.

Portanto, um livro aberto e disponível livremente que as pessoas possam compartilhar e contribuir torna-se uma opção cada vez mais relevante.

O conteudo e organização dos capítulos esta separado em partes.

O primeiro grupo – que inclui dos capítulos x ao y, inclui os aspectos mais gerais da estrutura do livro, seus objetivos e sobre o funcionamento da linguagem R. No segundo grupo de capítulos (do x ao y), temos contato com análises específicas e atualmente usadas em Ecologia da Paisasgem, incluindo ....

Introdução

Ecologia da Paisagem é uma disciplina empolgante que permite transformar dados brutos em compreensão, insight e conhecimento.

O que você vai aprender

Ecologia da Paisagem é um campo vasto e não há como dominar tudo lendo um único livro. Este livro visa fornecer uma base sólida nas ferramentas mais importantes e conhecimento suficiente para encontrar os recursos para aprender mais quando necessário. Um modelo das etapas de um projeto típico de Ecologia da Paisagem se parece com. . . ..

Como este livro está organizado

A descrição anterior das ferramentas da Ecologia da Paisagem é organizada aproximadamente de acordo com a ordem em que você as usa em uma análise (embora, é claro, você as itere várias vezes). Em nossa experiência, no entanto, aprender a importar e organizar os dados primeiro não é o ideal, porque 80% do tempo é rotineiro e chato e, nos outros 20% do tempo, é estranho e frustrante. Esse é um péssimo lugar para começar a aprender um novo assunto! Em vez disso, começaremos com a visualização e transformação dos dados que já foram importados e organizados. Dessa forma, quando você ingerir e organizar seus próprios dados, sua motivação permanecerá alta porque você sabe que a dor vale o esforço.

Dentro de cada capítulo, tentamos aderir a um padrão consistente: comece com alguns exemplos motivadores para que você possa ver o quadro geral e depois mergulhe nos detalhes. Cada seção do livro é combinada com exercícios para ajudá-lo a praticar o que aprendeu. Embora possa ser tentador pular os exercícios, não há melhor maneira de aprender do que praticar em problemas reais.

O que você não vai aprender

Prerequisites

Fizemos algumas suposições sobre o que você já sabe para aproveitar ao máximo este livro. Você deve ser geralmente alfabetizado numericamente, e com conhecimento previsa de ecologia, geoprocessamento e uso de sistemas de informação geografica.

Você precisa de quatro coisas para executar o código deste livro: R, RStudio, uma coleção de pacotes R chamada **tidyverse** e um punhado de outros pacotes. Os pacotes são as unidades fundamentais do código R reproduzível. Eles incluem funções reutilizáveis, documentação que descreve como usá-los e dados de amostra.

R

Para fazer o download do R, acesse CRAN, a **c**omprehensive **R** **a**rchive **n**network, https://cloud.r-project.org. Uma nova versão principal do R é lançada uma vez por ano e há 2 a 3 versões secundárias a cada ano. É uma boa ideia atualizar regularmente. A atualização pode ser um pouco complicada, especialmente para as versões principais que exigem a reinstalação de todos os seus pacotes, mas adiar só piora as coisas. Recomendamos R 4.2.0 ou posterior para este livro.

RStudio

RStudio é um ambiente de desenvolvimento integrado, ou IDE, para programação R, que você pode baixar em https://posit.co/download/rstudio-desktop/. O RStudio é atualizado algumas vezes por ano e avisa automaticamente quando uma nova versão é lançada, para que não haja necessidade de verificar novamente. É uma boa ideia atualizar regularmente para aproveitar os melhores e mais recentes recursos. Para este livro, certifique-se de ter pelo menos o RStudio 2022.02.0.

O universo arrumado - tidyverse

Você também precisará instalar alguns pacotes do R. Um **pacote** do R é uma coleção de funções, dados e documentação que estende os recursos do R base. O uso de pacotes é a chave para o uso bem-sucedido do R. A maioria dos pacotes que você aprenderá neste livro faz parte do chamado tidyverse. Todos os pacotes no tidyverse compartilham uma filosofia comum de programação de dados e R e são projetados para trabalhar juntos.

Você pode instalar o tidyverse completo com uma única linha de código:

```
install.packages("tidyverse")
```

No seu computador, digite essa linha de código no console e pressione enter para executá-lo. R irá baixar os pacotes do CRAN e instalá-los em seu computador.

Você não poderá usar as funções, objetos ou arquivos de ajuda em um pacote até carregá-lo com `library()`. Depois de instalar um pacote, você pode carregá-lo usando a função `library()`:

```
library(tidyverse)
```

Isso diz a você que o tidyverse carrega nove pacotes: dplyr, forcats, ggplot2, lubridate, purrr, readr, stringr, tibble, alignr. Eles são considerados o **núcleo** do tidyverse porque você os usará em quase todas as análises.

Os pacotes no tidyverse mudam com bastante frequência. Você pode ver se há atualizações disponíveis executando `tidyverse_update()`.

Outros pacotes

Existem muitos outros pacotes excelentes que não fazem parte do tidyverse porque resolvem problemas em um domínio diferente ou são projetados com um conjunto diferente de princípios subjacentes. Isso não os torna melhores ou piores, apenas diferentes. Em outras palavras, o complemento do tidyverse não é o universo bagunçado, mas muitos outros universos de pacotes inter-relacionados. Ao lidar com mais projetos de Ecologia da Paisagem com R, você aprenderá novos pacotes e novas formas de pensar sobre os dados.

Usaremos outras pacotes de fora do tidyverse neste livro. Por exemplo, usaremos os seguintes pacotes porque eles fornecem conjuntos de funcões e dados interessantes para trabalharmos no processo de aprendizado de R:

```
install.packages(c("sp", "sf","raster", "mapview","tmap",
                   "terra", "kableExtra", "landscapemetrics"))
```

Part II

Escala e padrões

1 Escala

1.1 Apresentação

Nesta capitulo vamos entender a importância de escala na ecologia da paisagem através cálculos com a proporção de floresta. Durante o capitulo você aprenderá a

1. Alterar escala (resolução e extensão espacial),

2. Calcular a área de uma classe de habitat,

3. Desenvolve uma comparação multiescala.

É muito importante ficar claro para você o que é escala (e o que não é!) e qual a importância desse conceito na elaboração do desenho amostral, na coleta de dados, nas análises e na tomada de decisão. Nesse tutorial usaremos conteudo baseado no Capítulo 2 do livro (Fletcher and Fortin 2018) e "Tutorial Escala" do Dr. Alexandre Martensen.

1.1.1 Escala: breve definação

Todos os processos e padrões ecológicos têm uma dimensão temporal e espacial. Assim sendo, o conceito de escala não somente representar essas dimensões, mas também, ajudar nos apresentá-los de uma forma que facilite o entendimento sobre os processos e padrões sendo estudados.

Na ecologia o termo escala refere-se à dimensão ou domínio espaço-temporal de um processo ou padrão. Na ecologia da paisagem, a escala é frequentemente descrita por sua componentes: resolução e extensão.

- **Resolução**: menor unidade espacial de medida para um padrão ou processo.
- **Extensão**: descreve o comprimento ou tamanho de área sob investigação.

Resolução e extensão tendem a covariar – estudos com maior extensão tendem a ter resolução maiores também. Parte dessa covariância é prática: é difícil trabalhar em grandes extensões com dados coletados em tamanhos de resolução finos. No entanto, parte dessa covariância também é conceitual: muitas vezes em grandes extensões, podemos esperar que processos operando em resolução muito finos forneçam somente "ruído" e não dados/informações relevantes sobre os sistemas. Como os desafios computacionais diminuíram e a disponibilidade de dados de alta resolução aumentou, a covariância entre resolução e extensão nas investigações diminuiu.

1.2 Pacotes e dados

Agora vamos olhar um exemplo do mundo real. Uma pequena amostra do Rio Araguari, perto de Porto Grande. O ponto central da raster é de longitude: -51.406312 latitude: 0.726236. Para visualizar o ponto no Google Earth: https://earthengine.google.com/timelapse#v=0.72154,-51.41543,11.8,latLng&t=2.24&ps=25&bt=19840101&et=20201231&startDwell=0&endDwell=0 .

Em geral é necessário baixar alguns pacotes para que possamos fazer as nossas análises. Precisamos os seguintes pacotes, que deve esta instalado antes:

- tidyverse,
- sf,
- mapview,
- tmap.

Carregar pacotes:

```
library(tidyverse)
library(sf)
library(terra)
library(mapview)
library(tmap)
```

O arquivo raster tem uma pequena amostra com a classificação da terra feito pela MapBiomas, que produz mapeamento anual da cobertura e uso da terra no Brasil desde 1985.

Baixar arquivo com os dados (formato ".tif"), link: https://github.com/darrennorris/gisdata/blob/master/inst/raster/amostra_mapbiomas_2020.tif . Lembrando-se de salvar o arquivo ("amostra_mapbiomas_2020.tif") em um local conhecido no seu computador.

Carregar o arquivo trabalhamos com o pacote terra. O pacote tem varios funçoes para a análise e modelagem de dados geográficos. Nós podemos ler os dados de cobertura da terra no arquivo ".tif" com a função `rast`.

```
#
ramostra <- rast(file.choose())
```

Plotar para verificar.

```
plot(ramostra)
```

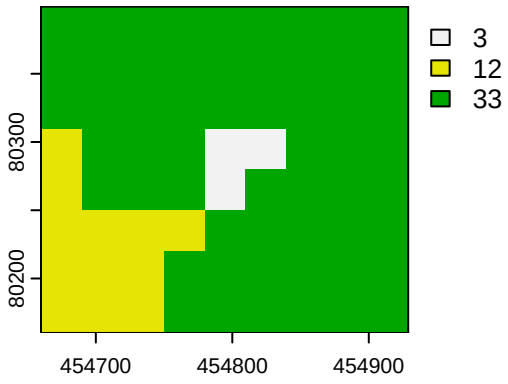


Figure 1: Mapbiomas 2020. Uma pequena amostra do Rio Araguari, perto de Porto Grande.

Podemos também verificar informações sobre o raster (metadados).

```
ramostra
```

```
## class       : SpatRaster
## dimensions  : 8, 9, 1  (nrow, ncol, nlyr)
## resolution  : 29.89281, 29.89281  (x, y)
## extent      : 454659.8, 454928.9, 80160.06, 80399.2  (xmin, xmax, ymin, ymax)
## coord. ref. : SIRGAS 2000 / UTM zone 22N (EPSG:31976)
## source      : amostra_mapbiomas_2020.tif
## name        : mapbiomas_2020
## min value   :              3
## max value   :             33
```

Isso nos mostra informações sobre escala espacial (resolução e extensão) e a sistema de coordenadas (SIRGAS 2000 / UTM zone 22N (EPSG:31976

$$https://epsg.io/31976$$

)). Além disso é possível obter informações específicas através de funções específicas.

```
# Obter informações sobre escala espacial
# resolução
res(ramostra)
# numero de colunas
ncol(ramostra)
# numero de linhas
nrow(ramostra)
```

1.2.0.1 Pergunta 1 **Com base nos resultados obtidos até agora em relação ao objeto raster ramostra, qual o tamanho do pixel em metros quadrados? Qual o tamnaho total da raster ramostra em hectares e quilometros quadrados?**

Vamos olhar o mapa de novo.

```
plot(ramostra)
```

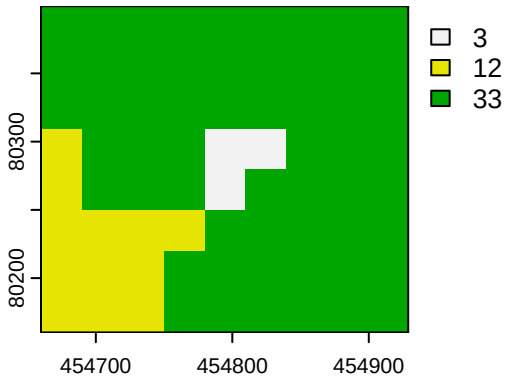


Figure 2: Mapbiomas 2020. Uma pequena amostra do Rio Araguari, perto de Porto Grande.

O mapa mostra três classes com valores de 3, 12 e 33. Lembrando, o objetivo principal não é de fazer mapas. Mas, a visualização dos dados é um passo importante para verificar e entender os padrões. Portanto, segue exemplo mostrando uma forma de visualizar o arquivo de raster como mapa.

Para entender o que os valores (3, 12, 33) representam no mundo real precisamos de uma referência (legenda). Para a MapBiomas Coleção 6, arquivo: Cod_Class_legenda_Col6_MapBiomas_BR.pdf. Existe também arquivos para fazer as mapas com cores corretas em QGIS ou ArcGIS.

Olhando a legenda (Cod_Class_legenda_Col6_MapBiomas_BR.pdf), sabemos que "3", "12" e "33" representem cobertura de "Formação Florestal", "Formação Campestre", e "Rio, Lago e Oceano". Então podemos fazer um mapa mostrando tais informações.

Daqui pra frente vamos aproveitar uma forma mais elegante de apresentar mapas e gráficos. Isso seria atraves a função "ggplot" (pacote ggplot2), que faz parte do "tidyverse". Mais exemplos no R cookbook : http://www.cookbook-r.com/Graphs/ .

E com mais exemplos de mapas e dados espaciais no R: sf e ggplot2 : https://www.r-spatial.org/r/2018/10/25/ggplot2-sf.html

Capitulo 9 no livro Geocomputation with R : https://geocompr.robinlovelace.net/adv-map.html

Primeiramente precisamos incluir as informações relevantes da legenda. Ou seja, incluir os nomes para cada valor de classe.

```
# legenda e cores na sequencia correta
classe_valor <- c(3, 12, 33)
classe_legenda <- c("Formação Florestal",
                    "Formação Campestre", "Rio, Lago e Oceano")
classe_cores <- c("#006400", "#B8AF4F", "#0000FF")
```

Agora podemos fazer o mapa com as classes e os cores seguindo o padrão recomendado pela MapBiomas para Coleção 6.

```
# Passo necessario para mostrar os valores
ramostra_df <- as.data.frame(ramostra, xy = TRUE)

ggplot(ramostra_df, aes(x=x, y=y)) +
```

```
geom_raster(aes(fill = factor(mapbiomas_2020))) +
scale_fill_manual("classe",
                    values = classe_cores,
                  labels = classe_legenda) +
coord_equal() +
geom_text(data = ramostra_df, aes(x = x, y = y,
              label = mapbiomas_2020)) +
theme(legend.position="top") +
guides(fill=guide_legend(nrow=2,byrow=TRUE))
```

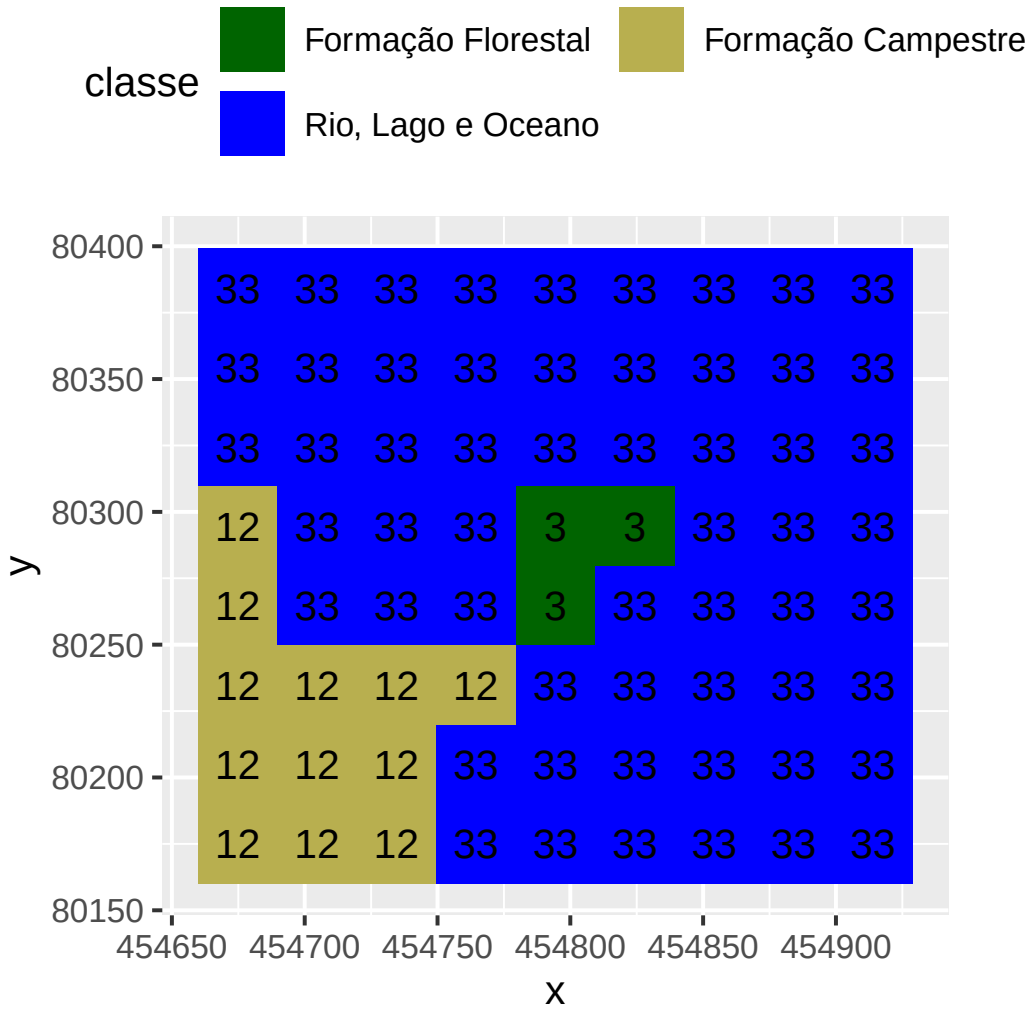


Figure 3: Paisagem com valores e classes de cobertura da terra. Mapbiomas 2020. Uma pequena amostra do Rio Araguari, perto de Porto Grande.

1.3 Alterando a resolução

Alterando a resolução serve como exemplo mostrando como os passos/etapas/cálculos mude dependendo o tipo de dados. Ou seja, é preciso adotar metodologias diferentes para dados categóricos (por exemplo classificação de cobertura da terra) e dados contínuos (por exemplo distância até rio).

Alterando a resolução às vezes seria necessário, por exemplo, quando preciso padronizar dados/imagens oriundos de fontes diferentes com resoluções diferentes e/ou para reduzir a complexidade da modelagem. Lembrando - em cada nível de resolução, são observáveis processos e padrões que não podem necessariamente ser inferidos daqueles abaixo ou acima.

Agora iremos degradar a resolução desses dados, ou seja, iremos alterar o tamanho dos pixels. Como exemplo, iremos juntar (agregar) 3 pixels em um único pixel. Como você acha que podemos fazer isso? Quais valores esse pixel que vai substituir os 3 originais deve ter? Existem diversas maneiras de se fazer isso, uma das formas é através da média.

```
ramostra_media<-aggregate(ramostra, fact=3, fun="mean")
ramostra_media <- resample(ramostra, ramostra_media)
```

Visualizar. Os valores calculados pela função não fazem sentido para uma classificação categórica.

```
# Tidy
as.data.frame(ramostra_media, xy = TRUE) %>%
  mutate(mapbiomas_2020 = round(mapbiomas_2020,1)) -> ramostra_media_df
# Plot
ggplot(ramostra_media_df, aes(x=x, y=y)) +
  geom_raster(aes(fill = factor(mapbiomas_2020))) +
  scale_fill_discrete("valor") +
  coord_equal() +
  geom_text(data = ramostra_media_df, aes(x = x, y = y,
                label = mapbiomas_2020))
```

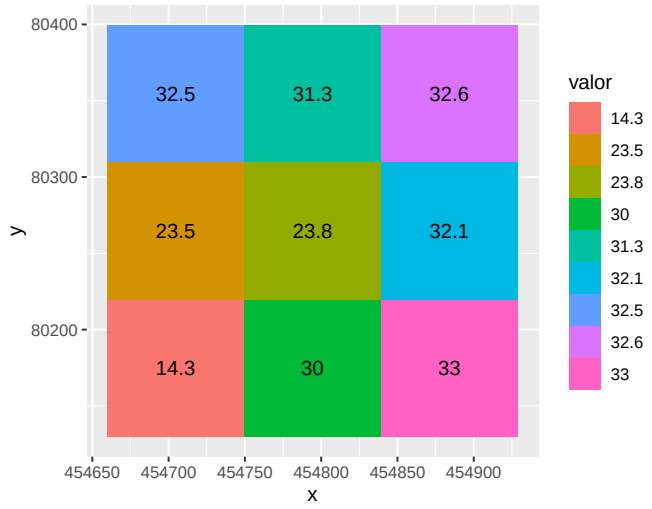


Figure 4: Agregação errado para dados categóricos. Uso da média cria valores categóricos errados e impossíveis.

Outra opção é utilizar o valor mais comum da área, o que é particularmente adequado quando temos um mapa categórico, como por exemplo floresta/não-floresta. Segue exemplo com o valor mais frequente (modal).

```
ramostra_modal<- aggregate(ramostra, fact=3, fun="modal")
ramostra_modal <- resample(ramostra, ramostra_modal, method="near")
```

Visualizar. Os valores calculados pela função são consistentes com o original e fazem sentido.

```
# Tidy
ramostra_modal_df <- as.data.frame(ramostra_modal, xy = TRUE)
# Plot
ggplot(ramostra_modal_df, aes(x=x, y=y)) +
  geom_raster(aes(fill = factor(mapbiomas_2020))) +
  scale_fill_manual("classe", values = classe_cores) +
  coord_equal() +
geom_text(data=ramostra_modal_df, aes(x=x, y=y, label=mapbiomas_2020))
```

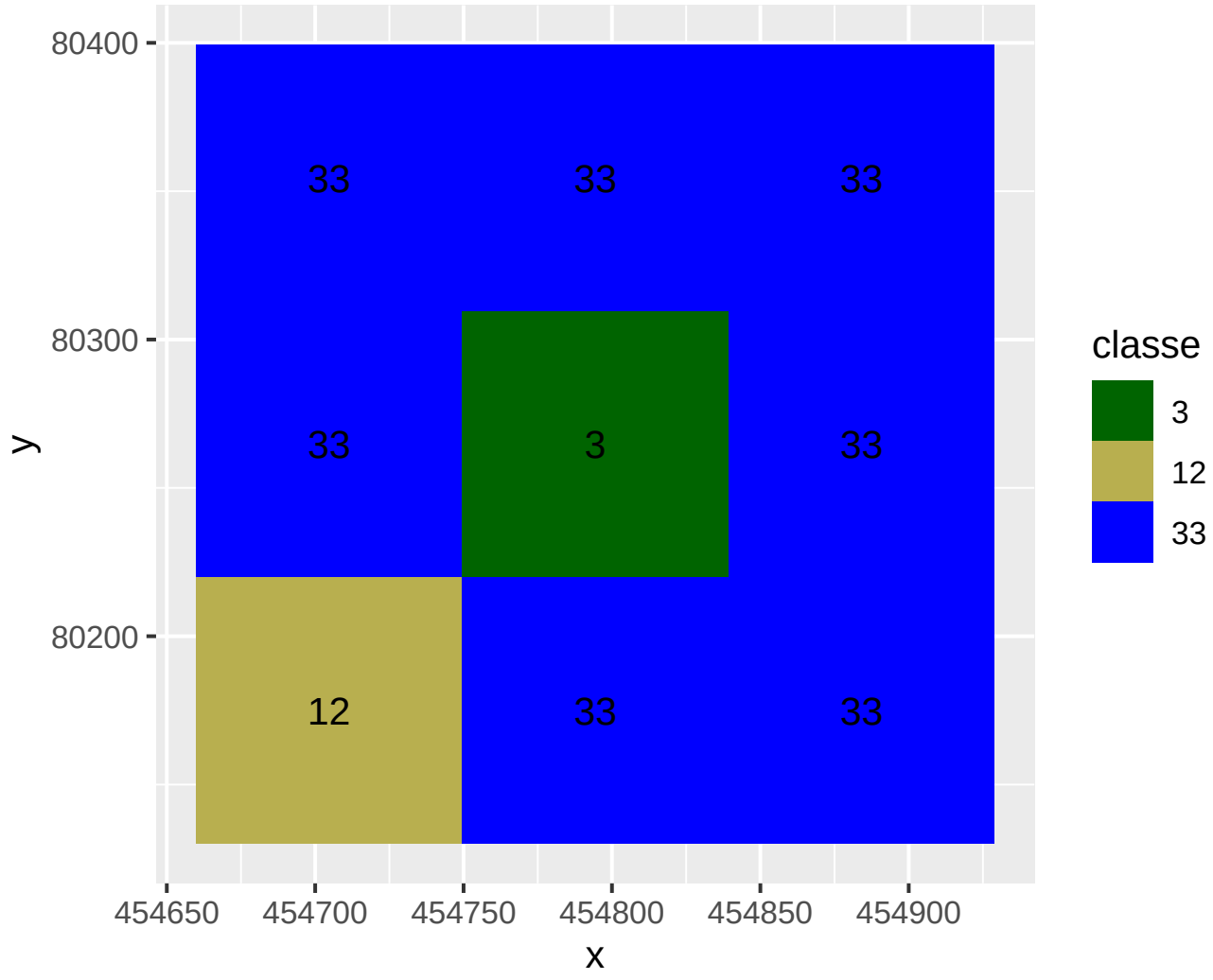


Figure 5: Agregação pela mais frequente.

Em cada nível de resolução, são observáveis processos e padrões que não podem necessariamente ser inferidos daqueles abaixo ou acima. Aqui por exemplo, mudamos a proporção de cobertura florestal em nossa pequeno paisagem quando juntamos 3 pixels em um único: a proporção de floresta moudou de 4% (3/72) para 11% (1/9). Ou seja, com cada passo mudamos a representção do mundo.

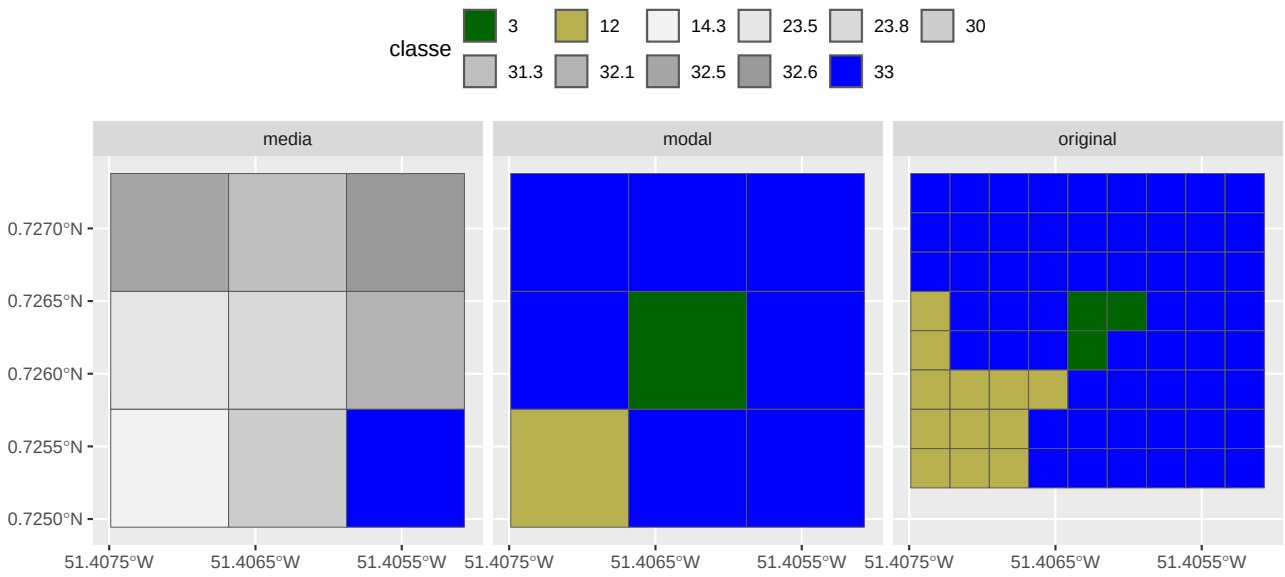


Figure 6: Mudanças causadas pela agregação.

1.3.0.1 Pergunta 2 **Confira o codigo e os resultados obtidos anteriormente, quando mudamos a resolução da ratser ramostra (por exemplo figuras 5.1, 5.2 e 5.3). Explique o que aconteceu. Como e porque moudou os valores em cada caso (média e modal)?**

1.4 Escala espacial e desenho amostral

Dado o papel que a escala pode desempenhar em nossa compreensão dos padrões e processos ecológicos, como escala deve ser considerada no desenho do estudo? Claramente, a resposta a esta pergunta irá variar dependendo dos fenômenos de interesse, mas ecologistas e estatísticos têm forneceu algumas orientações importantes. As questões-chave incluem o tamanho da unidade de amostragem (resoloção), o tipo de unidade de amostra e localizações da unidade de amostra, incluindo o espacamento entre as amostras (distância entre as amostras) e o tamanho da área de estudo.

Com a disponibilidade de imagens de satélite é possível responder questões importantes relacionadas ao desenho do estudo antes de qualquer trabalho de campo. Uma tecnica de geoprocessamento (bordas - Buffers) é um dos mais frequentemente adotados para quantificar escala espacial na ecologia da paisagem.

O objetivo é criar buffers circulares de diferentes extensões ao redor dos sitios de amostragem (pontos, pixels, manchas, transetos lineares etc). Aqui, vamos entender a escala em que a cobertura de floresta muda ao redor dos rios. Para isso, quantificamos a quantidade de floresta que ocorre em várias distâncias em pontos ao longo dos rios a montante das hidrelétricas no Rio Araguari. Para ilustrar esta abordagem geral, usamos o banco de dados MapBiomas Coleção 6 de 2020, e vincule esses dados de cobertura da terra aos pontos de amostragem em rios.

1.4.1 Obter e carregar dados (vectores)

Precisamos carregar os dados para rios e pontos de amostragem. Baixar arquivo (vector) com os dados (formato "GPKG", tamanho 54.9 MB). Este arquivo contém diferentes camadas vetoriais usadas para avaliar impactos de barragens hidroelétricas em tracajas ((Bárcenas-García et al. 2022)) e ariranhas ((Raffo et al. 2022)).

Mais sobre dados vetoriais. O formato aberto GeoPackage é um contêiner que permite armazenar dados SIG (feições/camadas) em um único arquivo. Por exemplo, um arquivo GeoPackage pode conter vários dados (dados vetoriais e raster) em diferentes sistemas de coordenadas. Todos esses recursos permitem que você compartilhe dados facilmente e evite a duplicação de arquivos.

Baixar o arquivo Link: https://github.com/darrennorris/gisdata/blob/master/inst/vector/rivers.gpkg . Lembrando-se de salvar o arquivo ("rivers.gpkg") em um local conhecido no seu computador.

O formato "GPKG" e diferente de "tif" (raster), o processo de importação é, portanto, diferente. Primeira, avisar R sobre onde ficar o arquivo. O código abaixo vai abrir uma nova janela, e você deve buscar e selecionar o arquivo "rivers.GPKG":

```
meuSIG <- file.choose()
```

Agora vamos olhar o que tem no arquivo. Depois que voces rodar o codigo **st_layers(meuSIG)** , o resultado mostra que o arquivo rivers.GPKG inclui camadas diferentes com pontos ("Point"), linhas ("Line String") e polígonos ("Polygon"). Alem disso, a coluna "crs_name" mostrar que a sistema de coordenadas é geográfica (WGS84, (EPSG: 4326)

$$https://epsg.io/4326$$

, e é diferente do arquivo raster:

```
sf::st_layers(meuSIG)
```

```
## Driver: GPKG
## Available layers:
##                   layer_name geometry_type features fields crs_name
## 1                  centerline   Line String       52     15   WGS 84
## 2                  forestloss         Point   276086     12   WGS 84
## 3                   canalpoly       Polygon        3      6   WGS 84
## 4              extentpoly50km       Polygon        1      0   WGS 84
## 5                   midpoints         Point       52     17   WGS 84
```

```
## 6            midpoints_hansen         Point       52     37   WGS 84
## 7         cachoeira_caldeirao         Point        1      2   WGS 84
## 8                porto_grande         Point        1      1   WGS 84
## 9                 icmbio_base         Point        1      1   WGS 84
## 10              direct_affect       Polygon        1      2   WGS 84
## 11 midpoints_hansen_distances         Point       52     43   WGS 84
## 12       midpoints_hansen_ffr         Point       52     82   WGS 84
## 13     midpoints_hansen_ffril         Point       52     91   WGS 84
## 14         direct_affect_line   Line String        1      2   WGS 84
```

Nós só precisamos de duas dessas camadas. O código abaixo vai carregar as camadas que precisamos e criar os objetos "rsm" e "rsl". Assim, agora temos dados com: pontos cada 5 km ao longo os rios ("rsm") e a linha central de rios ("rsl").

```
# pontos cada 5 km
rsm <- sf::st_read(meuSIG, layer = "midpoints")
```

```
## Reading layer `midpoints' from data source
##   `C:\Users\user\Documents\Articles\gis_layers\gisdata\inst\vector\rivers.gpkg'
##   using driver `GPKG'
## Simple feature collection with 52 features and 17 fields
## Geometry type: POINT
## Dimension:     XY
## Bounding box:  xmin: -52.01259 ymin: 0.7175827 xmax: -51.29688 ymax: 1.330365
## Geodetic CRS:  WGS 84
```

```
# linha central de rios
rsl <- sf::st_read(meuSIG, layer = "centerline")
```

```
## Reading layer `centerline' from data source
##   `C:\Users\user\Documents\Articles\gis_layers\gisdata\inst\vector\rivers.gpkg'
##   using driver `GPKG'
## Simple feature collection with 52 features and 15 fields
## Geometry type: LINESTRING
## Dimension:     XY
## Bounding box:  xmin: -52.01443 ymin: 0.7094595 xmax: -51.2924 ymax: 1.352094
## Geodetic CRS:  WGS 84
```

1.4.2 Visualizar os arquivos (camadas vector)

Visualizar para verificar. Mapa com linha central e pontos de rios em trechos de 5km.

```
ggplot(rsl) +
  geom_sf(aes(color=rio)) +
  geom_sf(data = rsm, shape=21, aes(fill=zone))
```

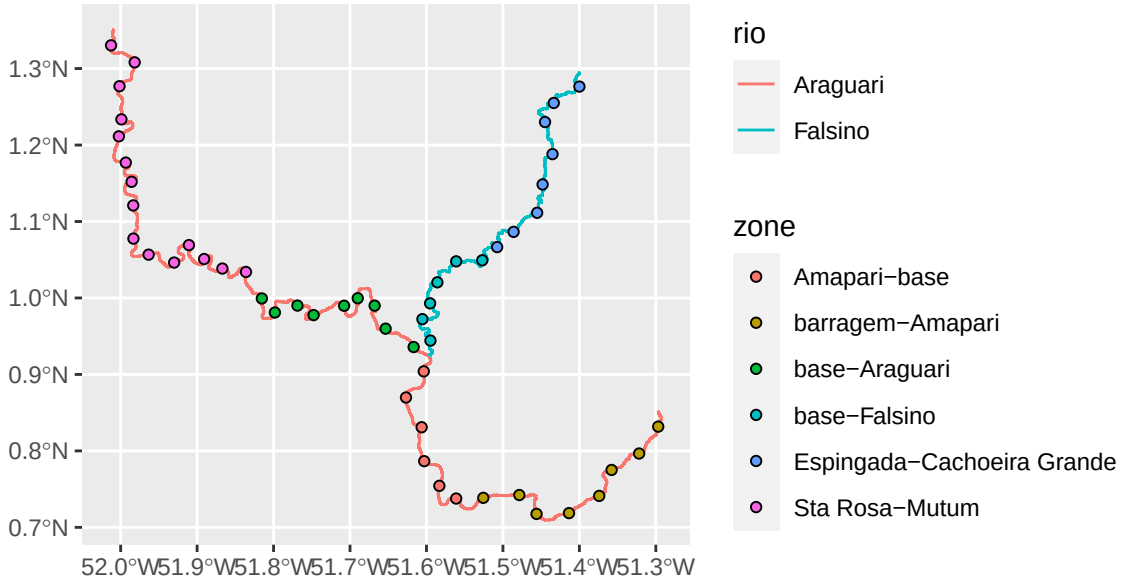


Figure 7: Pontos ao longo dos rios a montante das hidrelétricas no Rio Araguari.

Mapa interativo (funcione somente com internet) Mostrando agora com fundo de mapas "base" (OpenStreetMap/ESRI etc)

```
#
mapview(rsl, zcol = "rio")
```

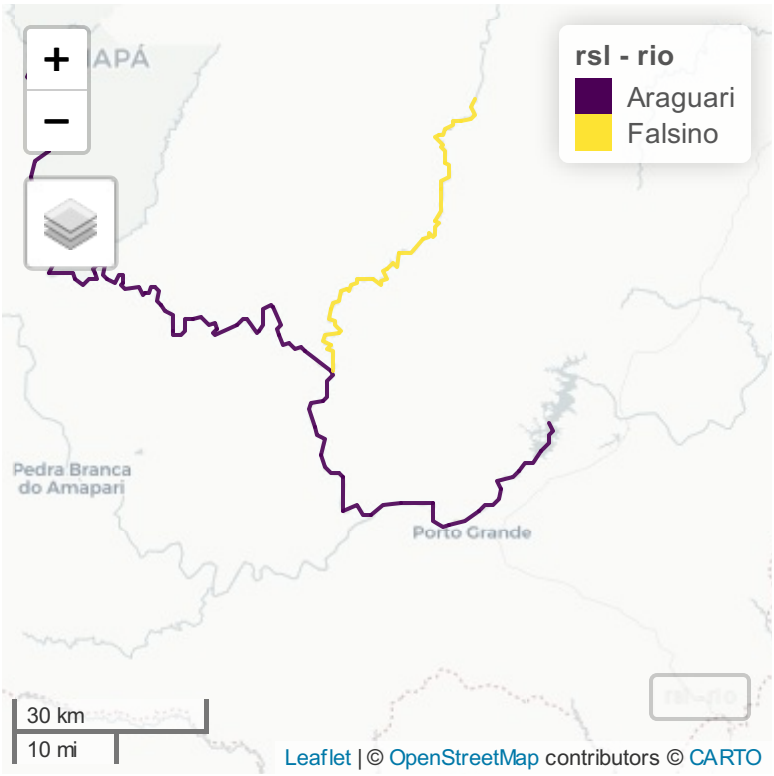


Figure 8: Linhas dos rios a montante das hidrelétricas no Rio Araguari.

1.4.3 Obter e carregar dados (raster)

Mais uma vez vamos aproveitar os dados de MapBiomas. Agora baixar arquivo raster com cobertura de terra no entorno dos rios em 2020, (formato ".tif", tamanho 3.3 MB). Link: https://github.com/darrennorris/gisdata/blob/master/inst/raster/mapbiomas_AP_utm_rio/utm_cover_AP_rio_2020.tif . Lembrando-se de salvar o arquivo ("utm_cover_AP_rio_2020.tif") em um local conhecido no seu computador. Agora avisar R sobre onde ficar o arquivo. O código abaixo vai abrir uma nova janela, e você deve buscar e selecionar o arquivo "utm_cover_AP_rio_2020.tif":

```
meuSIGr <- file.choose()
```

O código abaixo vai carregar os dados e criar o objeto "mapbiomas_2020".

```
mapbiomas_2020 <- rast(meuSIGr)
```

1.4.4 Visualizar os arquivos (camadas raster e vector)

Visualizar para verificar. É possível de visualizar camadas de raster e vetor juntos com funcões no pacote Tmap (https://r-tmap.github.io/tmap-book/index.html).

```
# Passo necessario para agilizar o processamento
mapbiomas_2020_modal<-aggregate(mapbiomas_2020, fact=10, fun="modal")
# Plot
tm_shape(mapbiomas_2020_modal) +
  tm_raster(title = "Classe", style = "cat", palette = "Set3") +
tm_shape(rsl) +
  tm_lines(col="blue") +
tm_shape(rsm) +
  tm_dots(size = 0.2, col = "yellow") +
```

```
tm_compass(position=c("left", "top")) +
tm_scale_bar(breaks = c(0, 25, 50), text.size = 1,
             position=c("left", "bottom")) +
tm_layout(legend.position = c("right","top"), legend.bg.color="white")
```

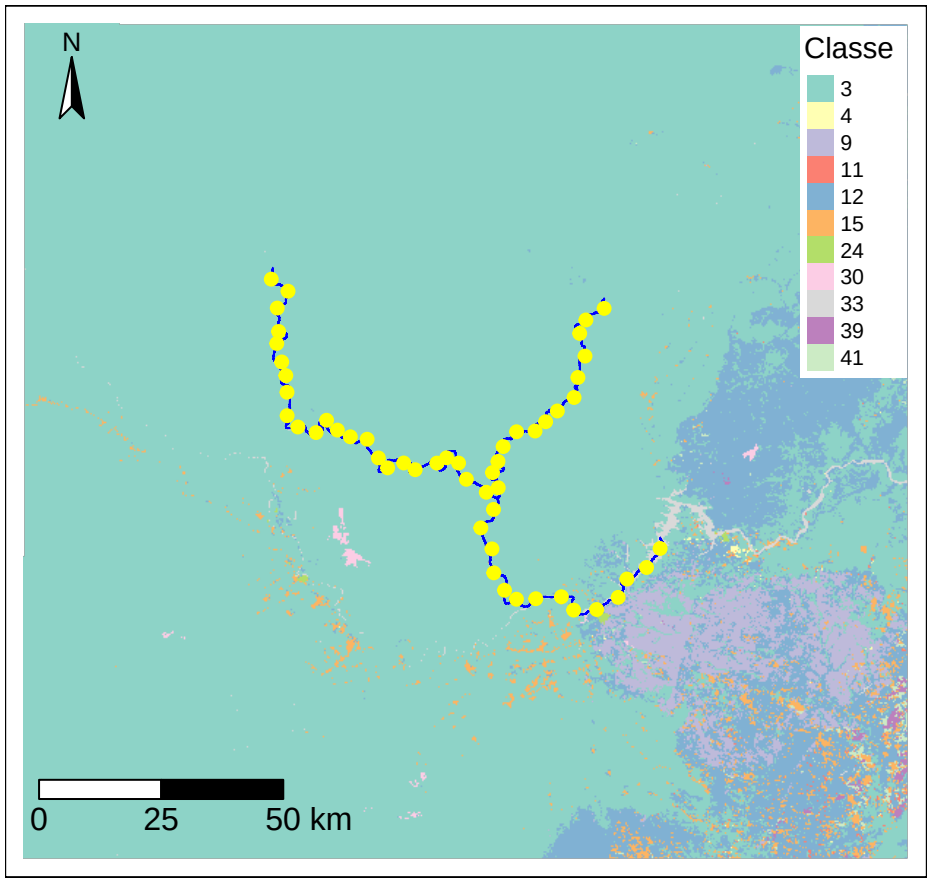


Figure 9: Cobertura da terra ao redor do Rio Araguari em 2020. Mostrando os pontos de amostragem (pontos amarelas) cada 5 quilômetros ao longo do rio.

1.4.5 Reclassificação

Para simplificar nossa avaliação de escala, reclassificamos a camada mapbiomas_2020 em uma camada binária de floresta/não-floresta. Essa tarefa de geoprocessamento pode ser realizada anteriormente usando SIG (QGIS). Aqui vamos reclassificar as categorias de cobertura da terra (agrupando diferentes áreas de cobertura florestal tipos) usando alguns comandos genéricos do R para criar uma nova camada com a cobertura de floresta em toda a região de estudo. Para isso, criamos um mapa do mesmo resolução e extensão, e então podemos redefinir os valores do mapa. Neste caso, queremos agrupar a cobertura da terra categorias 3 e 4 (Formação Florestal e Formação Savânica, respectivamente).

```
# criar uma nova camada de floresta
floresta_2020 <- mapbiomas_2020
# Com valor de 0
values(floresta_2020) <- 0
# Atualizar categorias florestais agrupados com valor de 1
floresta_2020[mapbiomas_2020==3 | mapbiomas_2020==4] <- 1
```

Vizualizar para verificar.

```
# Passo necessario para agilizar o processamento
floresta_2020_modal<-aggregate(floresta_2020, fact=10, fun="modal")
# Plot
tm_shape(floresta_2020_modal) +
  tm_raster(style = "cat",
            palette = c("0" = "#E974ED", "1" ="#129912"), legend.show = FALSE) +
  tm_add_legend(type = "fill", labels = c("não-floresta", "floresta"),
    col = c("#E974ED", "#129912"), title = "Classe") +
tm_shape(rsl) +
  tm_lines(col="blue") +
tm_shape(rsm) +
  tm_dots(size = 0.2, col = "yellow") +
tm_scale_bar(breaks = c(0, 25, 50), text.size = 1,
             text.color = "white", position=c("left", "bottom")) +
tm_layout(legend.position = c("right","top"),legend.bg.color = "white")
```

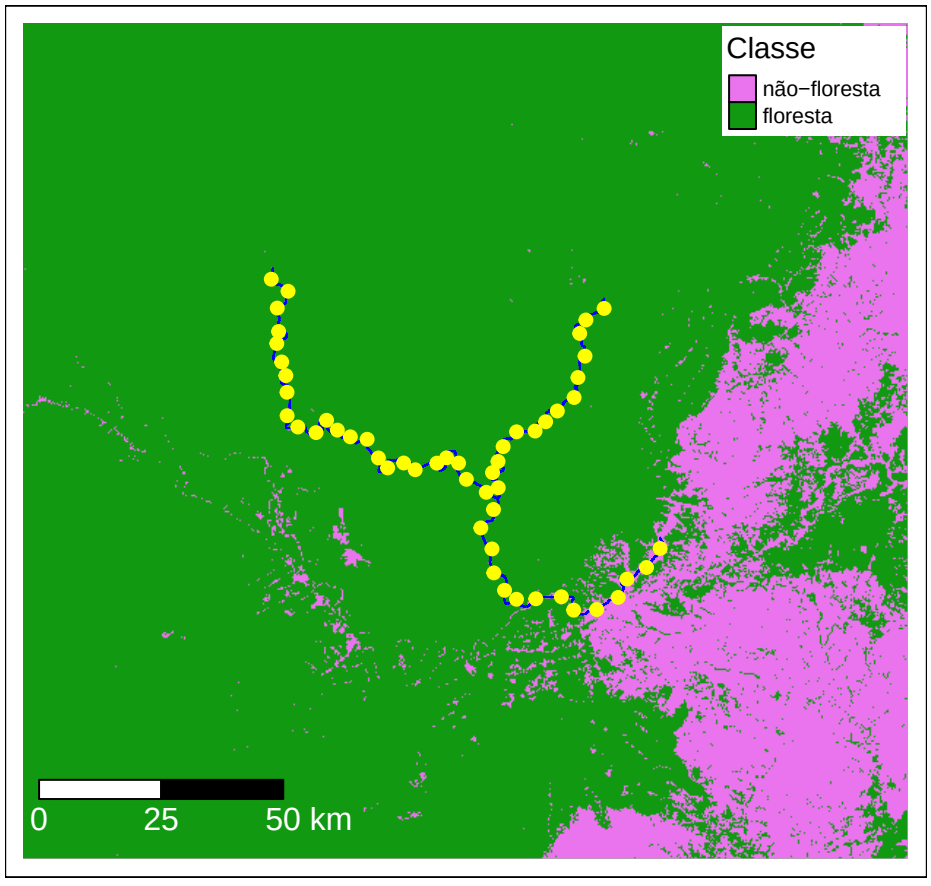


Figure 10: Floresta ao redor do Rio Araguari. MapBiomas 2020 reclassificado em floresta e não-floresta. Mostrando os pontos de amostragem (pontos amarelas) cada 5 quilômetros ao longo do rio.

1.5 Comparação multiescala

Em seguida, com as coordenadas dos pontos das localizações das amostras calculamos a quantidade de floresta que circunda cada local de amostragem em diferentes extensões.

```
rsm_31976 <- st_transform(rsm, 31976)
# Buffer
rsm_31976_b1000 <- st_buffer(rsm_31976[1, ], dist = 1000)

# Recorte com buffer de 1000 metros (mudando a extensão).
buffer.forest1.1km <- crop(floresta_2020, snap="out", rsm_31976_b1000)
# Máscara para que os pixels fora do polígono sejam nulos.
buffer.forest1.1km <- mask(buffer.forest1.1km, rsm_31976_b1000, touches=TRUE)
names(buffer.forest1.1km) <- "forest_2020_1km"
```

Vizualizar para verificar.

```
# Plot
tm_shape(buffer.forest1.1km) +
  tm_raster(style = "cat",
            palette = c("0" = "#E974ED",
                        "1" ="#129912"), legend.show = FALSE) +
tm_shape(rsm_31976[1, ]) +
  tm_symbols(shape =21, col = "yellow",
             border.col = "black", border.lwd = 0.2, size=0.5) +
tm_shape(rsm_31976_b1000) +
  tm_borders(col = "black", lwd = 4, lty = "dashed") +
tm_add_legend(type = "fill",  labels = c("não-floresta", "floresta"),
    col = c("#E974ED", "#129912"), title = "Classe") +
tm_compass(position=c("left", "top")) +
tm_scale_bar(breaks = c(0, 0.5, 1), text.size = 1,
             position=c("left", "bottom")) +
tm_layout(legend.position = c("right","top"), legend.bg.color = "white")
```

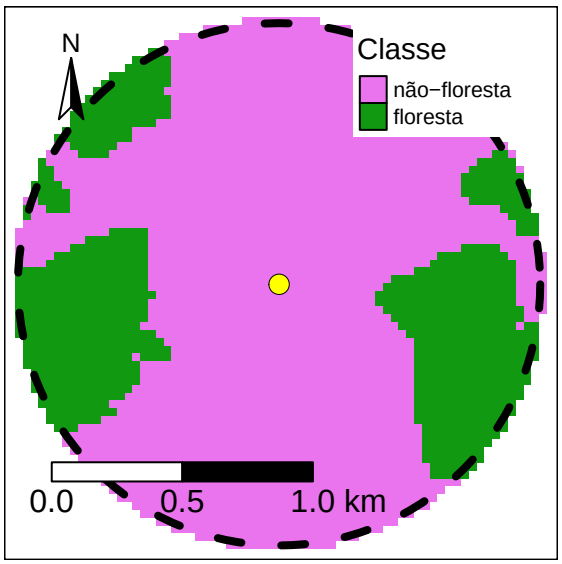


Figure 11: Ilustração da determinação da quantidade de habitat ao redor de um ponto. Para um determinada extensão, o habitat de interesse é isolado. Um buffer (linha tracejada) é colocado ao redor de um ponto (amarela) e o número de células (pixels) que contém o habitat é somado e multiplicado pela área de cada pixel.

1.5.1 Pergunta 3

Qual é a extensão em número de pixels desse recorte (buffer.forest1.1km)?

Temos valores de 0 (não-floresta) e 1 (floresta). Então, para saber a aréa de floresta podemos somar o número de células (pixels) que contém o habitat e multiplica pela área de cada pixel conforme o codigo:

```
# 1) Somatório.
# No caso igual o numero de pixels de floresta.
# Para todo a paisagem, somatorio "global".
# Não deve incluir pixels nulos, então use "na.rm = TRUE".
```

```r
soma_floresta <- global(buffer.forest1.1km, "sum", na.rm = TRUE)
soma_floresta
```

```
##                  sum
## forest_2020_1km 943
```

```r
# 2) Área de cada pixel.
# Sabemos o sistema de coordenadas (EPSG = 31976).
# EPSG 31976 é uma sistema projetado com unidade em metros.
buffer.forest1.1km
```

```
## class       : SpatRaster
## dimensions  : 68, 68, 1  (nrow, ncol, nlyr)
## resolution  : 29.89281, 29.89281  (x, y)
## extent      : 465959.3, 467992, 90921.47, 92954.18  (xmin, xmax, ymin, ymax)
## coord. ref. : SIRGAS 2000 / UTM zone 22N (EPSG:31976)
## source(s)   : memory
## name        : forest_2020_1km
## min value   :               0
## max value   :               1
```

```r
# Portanto, o tamanho de cada pixel é igual.
area_pixel_m2 <- 29.89281 * 29.89281
area_pixel_m2
```

```
## [1] 893.5801
```

```r
# 3) Calculos de aréa.
# Aréa de floresta m2
area_floresta_m2 <- soma_floresta * area_pixel_m2
area_floresta_m2
```

```
##                    sum
## forest_2020_1km 842646
```

```r
# Aréa de floresta hectares
area_floresta_ha <- area_floresta_m2 / 10000
area_floresta_ha
```

```
##                     sum
## forest_2020_1km 84.2646
```

Para uma comparação multiescala, vamos repetir o mesmo processo, mas agora com distancias de 250, 500, 1000, 2000 e 4000 metros, doubrando a escala (extensão) em cada passo.

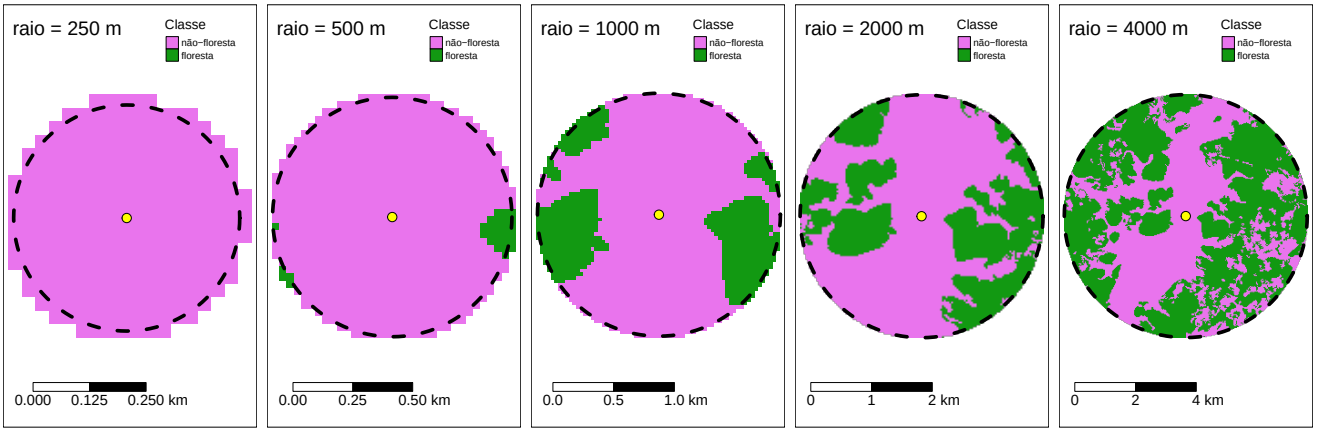


Figure 12: Cobertura florestal em extensões diferentes ao redor de um local de amostragem.

Aspectos quantitativos das paisagens mudam fundamentalmente com a escala. Por exemplo, nesse caso, parece que a proporção de floresta aumenta à medida que a extensão aumenta de 500 para 4000 metros. Esta percepção visual é confirmada pelos valores calculados, onde as áreas são:

- raio 250 m = 0 hectares de floresta
- raio 500 m = 6,3 hectares de floresta
- raio 1000 m = 84,3 hectares de floresta
- raio 2000 m = 502.6 hectares de floresta
- raio 4000 m = 3351.0 hectares de floresta

1.5.2 Pergunta 4

Usando os valores listadas acima de raio e área de floresta para os diferentes buffers circulares, calcule a proporção de floresta em cada uma das diferentes extensões de buffer. Apresente 1) os resultados incluindo cálculos. 2) um gráfico com valores de extensão no eixo x e proporção da floresta no eixo y. 3) Em menos de 200 palavras apresente a sua interpretação do gráfico.

1.5.3 Pergunta 5

A modelagem multiescala quantifica as condições do ambiente em múltiplas escalas alterando o resolução ou a extensão da análise e, em seguida, avaliando qual das escalas consideradas explica melhor um padrão ou processo. Escolha 1 espécie aquático e 1 espécie terrestre que ocorram na região a montante das hidrelétricas no Rio Araguari. Com base nas diferenças entre extensões (indicados no exemplo anterior) e as características funcionais das espécies (por exemplo área de vida), escolher as extensões mais adequadas para um estudo multiescala de cada espécie.

1.6 Próximos passos: repetindo para muitas amostras.

Neste exemplo comparamos a área de floresta em torno de um único ponto de amostragem. Para calcular o mesmo para todos os 52 pontos, seriam necessárias varias repetições (52 pontos x 5 extensões = 260 repetições).

Poderíamos escrever código para executar esse processo automaticamente. Felizmente, alguém já escreveu funções para fazer isso e muito mais. O próximo tutorial sobre métricas de paisagem mostrará exemplos usando o pacote "landscapemetrics" (https://r-spatialecology.github.io/landscapemetrics/).

2 Métricas da paisagem

2.1 Apresentação

As métricas da paisagem nos ajudam a entender as mudanças na paisagem de diferentes perspectivas (visual, ecológica, cultural).

Asssim sendo, análises com métricas de paisagem é um atividade fundamental na ecologia da paisagem. Nesta capitulo aprenderemos sobre como analisar a cobertura da terra com métricas de paisagem em R. As tecnicas será ilustrada através cálculos usando a cobertura florestal ao redor do Rio Araguari. Ao longo do caminho, revisaremos modelos lineares e não lineares, aprenderemos sobre manipulação de dados em R e aprenderemos como criar gráficos com o pacote ggplot2.

No capitulo você aprenderá a:

- Importar e plotar dados raster em R e mapear locais de amostragem com os pacotes terra, sf e tmap.
- Calcular métricas de paisagem com o pacote landscapemetrics.
- Calcular métricas de paisagem em locais de amostragem e dentro de um buffer ao redor deles (comparação multiescala).
- Construir gráficos com o pacote ggplot2.
- Comparação de padrões lineares e não-lineares.

2.2 Métricas da paisagem e pacote "landscapemetrics"

As métricas de paisagem são a forma que os ecólogos de paisagem usam para descrever os padrões espaciais de paisagens para depois avaliar a influência destes padrões espaciais nos padrões e processos ecológicos.

landscapemetrics tem funções para calcular métricas de paisagem em paisagems categóricos (onde tem uma classificação de cobertura de terra/habitat - modelo mancha-corredor-matriz), em um fluxo de trabalho organizado. O pacote pode ser usado como um substituto do FRAGSTATS (McGarigal et al. 1995, https://doi.org/10.2737/PNW-GTR-351), pois oferece um fluxo de trabalho reproduzível para análise de paisagem em um único ambiente (Professor McGarigal se aposentou, então FRAGSTATS não é mais apoiado). landscapemetrics também permite cálculos de quatro métricas teóricas de complexidade da paisagem: entropia marginal, entropia condicional, entropia conjunta e informação mútua (Nowosad e Stepinski 2019 https://doi.org/10.1007/s10980-019-00830-x).

2.2.1 Pacotes

Além do "landscapemetrics", precisamos carregar alguns pacotes a mais para facilitar a organização e apresentação de dados espaciais (vector e raster) e os resultados.

Carregar pacotes (que deve esta instalado antes):

```
library(tidyverse)
library(sf)
library(raster)
library(terra)
library(tmap)
library(gridExtra)
library(kableExtra)
library(mgcv)
```

Agora, digite o código abaixo e veja o resultado. Leia com atenção e preste particular atenção na organização da página de ajuda.

```
library(landscapemetrics)
?landscapemetrics
```

No final da página você vai encontrar a palavra "Index". Clique nela e você verá todas as funções do pacote. Desca até as lsm_. . . e clique em algumas delas ali. Explorar! Para listar todas as métricas disponíveis, você pode usar a função `list_lsm()` . A função também permite mostrar métricas filtradas por nível, tipo ou nome da métrica. Para obter mais informações sobre as métricas, consulte os arquivos de ajuda correspondentes ou https://r-spatialecology.github.io/landscapemetrics.

Digite o código abaixo e veja o resultados, mostrando exemplos das métricas diferentes.

```
# métricas de agregação, nível de fragmento
landscapemetrics::list_lsm(level = "patch", type = "aggregation metric")
# métricas de agregação, nível de classe
landscapemetrics::list_lsm(level = "class", type = "aggregation metric")
#
landscapemetrics::list_lsm(metric = "area")
# ajudar com opções da função
?landscapemetrics::list_lsm
```

Nesse pacote o formato geral para uma função é o seguinte "lsm_nível_métrica":

- A primeira parte é sempre lsm_ ("landscapemetric"), seguinda do "nível_" e por fim a "métrica". Ou seja, todas as funções que calculam métricas começam com lsm_ .....

- Daí você deve incluir o nível da análise “p” para patch (ou seja, para a mancha/fragmento), “c” para classe e “l” para landscape ou seja, métricas para a paisagem como um todo.
- E daí existem inúmeras métricas, como por exemplo a `cpland` que é o percentual de área central - “core area” na paisagem, como vimos na aula teórica. Assim sendo, a função `lsm_c_cpland` vai calacular a métrica porcentagem da área central em cada classe. Lembrando existem metricas que podem se calculados nos trés niveis, e metricas que so pode se calculados somente para um nivel espcifco.

Digite o código abaixo e veja o resultados, mostrando exemplos das métricas diferentes.

```
# ajudar com opções para uma função especifica
?landscapemetrics::lsm_c_cpland
```

2.2.1.1 Pergunta 1 **Descreva brevemente 2 métricas de cada nível usando ajudar (usando ? e/ou list_lsm) e/ou a leitura disponivel no Google Classroom (Base teórica 4 Dados, métricas, analises). Incluindo na descrição - o nome, porque serve, unidades de medida, e relevância ecológica.**

2.3 Dados

Existem varios formas de importar e exportar dados geoespaciais. Aqui, precisamos o arquivo com os dados de MapBiomas "utm_cover_AP_rio_2020.tif", que voces baixaram no tutorial anterior (tutorial Escala https://rpubs.com/darren75/escala).

Link: https://github.com/darrennorris/gisdata/blob/master/inst/raster/mapbiomas_AP_utm_rio/utm_cover_AP_rio_2020.tif

Lembrando-se de salvar o arquivo ("utm_cover_AP_rio_2020.tif") em um local conhecido no seu computador. Agora, nós podemos carregar os dados de cobertura da terra "utm_cover_AP_rio_2020.tif" com a função rast.

```
# Selecionar e carregar arquivo "utm_cover_AP_rio_2020.tif"
mapbiomas_2020 <- rast(file.choose())
# Reclassificação -
# Criar uma nova camada de floresta (novo objeto de raster copiando mapbiomas_2020,
# assim para ter os mesmos coordenados, resolução e extensão)
floresta_2020 <- mapbiomas_2020
# Todos os pixels com valor de 0
values(floresta_2020) <- 0
# Atualizar com valor de 1 quando pixels originais são de floresta (classe 3 e 4)
floresta_2020[mapbiomas_2020==3 | mapbiomas_2020==4] <- 1
```

Plotar para verificar, incluindo nomes e os cores para classes de floresta (valor = 1) e não-floresta (valor = 0).

```
# Passo necessario para agilizar o processamento
floresta_2020_modal<-aggregate(floresta_2020, fact=10, fun="modal")
# Plot
tm_shape(floresta_2020_modal) +
  tm_raster(style = "cat",
            palette = c("0" = "#E974ED", "1" ="#129912"), legend.show = FALSE) +
  tm_add_legend(type = "fill", labels = c("não-floresta", "floresta"),
    col = c("#E974ED", "#129912"), title = "Classe") +
tm_layout(legend.bg.color = "white")
```

Se esta todo certo, voces devem ter uma imagem assim:

2.3.1 Exibir dados raster e sobreposição com locais de amostragem

Agora temos a paisagem, precisamos tambem os pontos de amostra. Por isso, precisamos carregar os dados de rios e pontos de amostragem que usamos no tutorial Escala - arquivo "rivers.GPKG". Vamos carregar as camadas que voces baixaram no tutorial anterior. Baixar o arquivo Link: https://github.com/darrennorris/gisdata/blob/master/inst/vector/rivers.GPKG . Lembrando-se de salvar o arquivo ("rivers.GPKG") em um local conhecido no seu computador.

Agora, com o proximo bloco de codigo, podemos selecionar o arquivo "rivers.GPKG", e carregar primeiramente a camada "midpoints" e depois "centerline".

No exemplo, usamos %>%, que estabelece a ligação entre os passos do processo. Ou seja, %>% passa o objeto resultante automaticamente para a próxima função como primeiro argumento. Primeiramente carregamos os dados e em seguida converter (reprojeção) as coordenadas para o mesmo sistema de referência que o arquivo raster (com a função st_transform).

```
#  Selecionar o arquivo "rivers.GPKG",
meuSIG <- file.choose()
# Carregar pontos cada 5 km, camada midpoints
rsm_31976 <- sf::st_read(meuSIG, layer = "midpoints") %>%
  st_transform(31976)
```

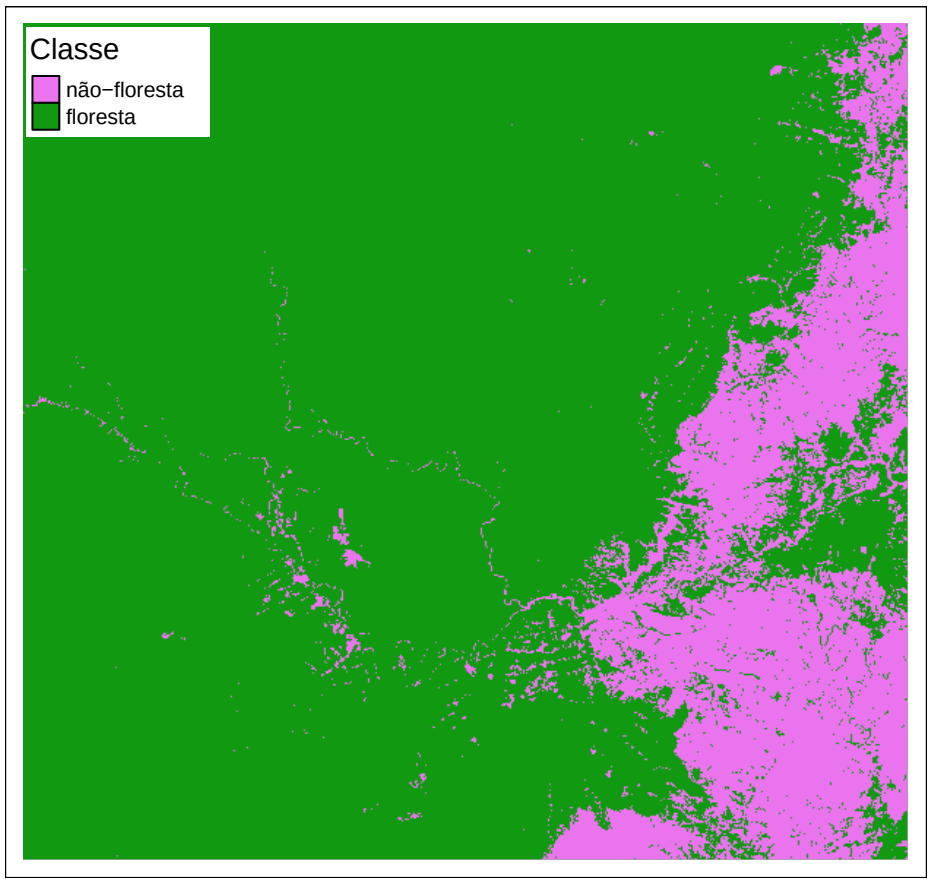


Figure 13: Floresta ao redor do Rio Araguari. MapBiomas 2020 reclassificado em floresta e não-floresta.

```r
# Carregar linha central de rios, camada centerline
rsl_31976 <- sf::st_read(meuSIG, layer = "centerline") %>%
  st_transform(31976)
```

Visualizer para verificar.

```
# Passo necessario para agilizar o processamento
floresta_2020_modal<-aggregate(floresta_2020, fact=10, fun="modal")
# Plot
tm_shape(floresta_2020_modal) +
  tm_raster(style = "cat",
            palette = c("0" = "#E974ED", "1" ="#129912"), legend.show = FALSE) +
  tm_add_legend(type = "fill", labels = c("não-floresta", "floresta"),
    col = c("#E974ED", "#129912"), title = "Classe") +
tm_shape(rsl_31976) +
  tm_lines(col="blue") +
tm_shape(rsm_31976) +
  tm_dots(size = 0.2, col = "yellow") +
tm_layout(legend.bg.color="white")
```

Depois de executar ("run") o código acima, você deverá ver a figura a seguir.

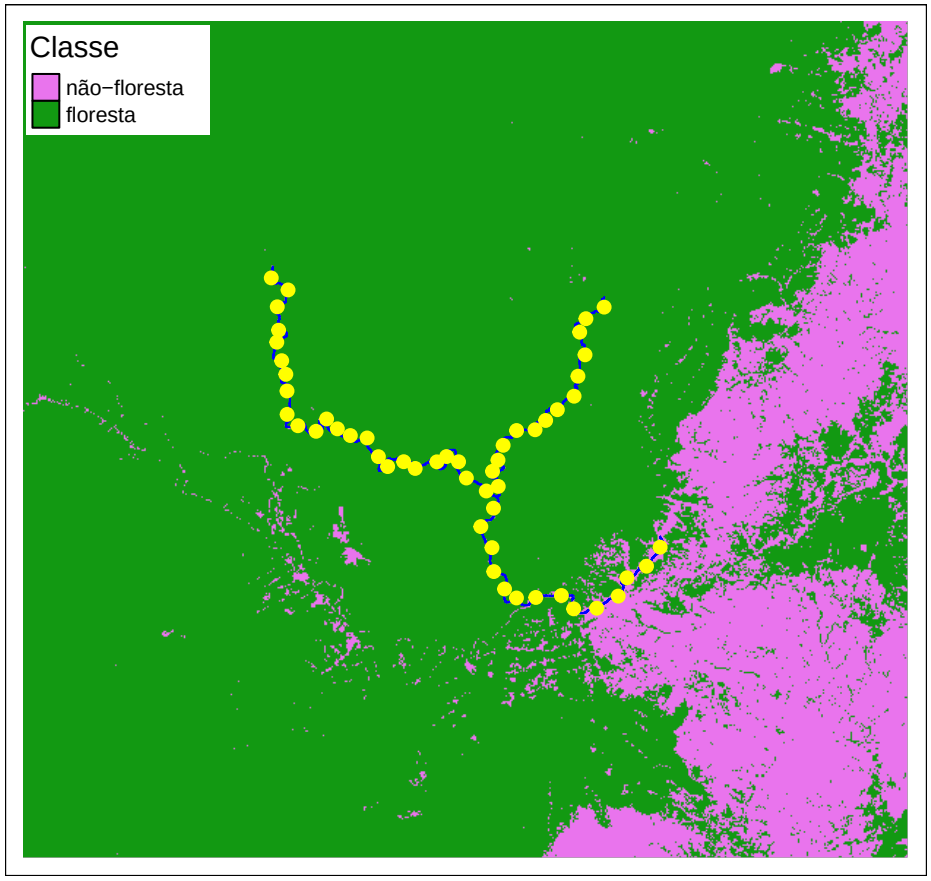


Figure 14: Cobertura da terra ao redor do Rio Araguari em 2020. Mostrando os pontos de amostragem (pontos amarelas) cada 5 quilômetros ao longo do rio (linha azul).

2.4 Calculo de métricas

Para ilustrar como rodar as funções e cálculos com landscapemetrics, vamos calcular a área central na paisagem. Vamos estudar uma classe (floresta), portanto vamos incluir as métricas para nível de classe. Além disso, as métricas de paisagem em nível de classe são mais eficazes na definição de processos ecológicos (Tischendorf, L. Can landscape indices predict ecological processes consistently?. Landscape Ecology 16, 235–254 (2001). https://doi.org/10.1023/A:1011112719782.).

Para calcular as métricas de paisagem dentro de um certo buffer em torno de pontos de amostra, existe a função `sample_lsm()` . Através da função `sample_lsm()` podemos calcular mais de 50 métricas da paisagem, dentro de extensões (raios/distancias) diferentes.

2.4.1 Ponto único, raio único, métrica única

Métricas de área central ("core area") são consideradas medidas da qualidade de hábitat, uma vez que indica quanto existe realmente de área efetiva de um fragmento, após descontar-se o efeito de borda. Vamos calcular a percentual de área central ("core area") no entorno de um ponto de amostragem. Isso seria, a percentual de áreas centrais (excluídas as bordas de 30 m) de cada classe em relação à área total da paisagem.

Para a função `sample_lsm()` funcionar, precisamos informar (i) a paisagem (arquivo de raster), (ii) ponto (arquivo vector), (iii) raio, (iv) forma do buffer (círculo ou quadrado) e por final (v) a métrica desejada. Cada opção tem especificações particulares assim para que a função pode receber dados em formatos diferentes e produzir resultados confiorme necessidades de diversos casos.

```
minha_amostra_1000 <- sample_lsm(landscape = floresta_2020,
                                 y = rsm_31976[1, ],
                                size = 1000, shape = "circle",
                                metric = "cpland",
                                edge_depth = 1)
```

Depois que executar ("run"), podemos olhar os dados com o codigo a seguir. Os dados deve ter os valores (coluna value) da métrica (coluna metric) de cada classe (coluna class):

```
minha_amostra_1000
```

layer	level	class	id	metric	value	plot_id	percentage_inside
1	class	0	NA	cpland	66.94191	1	99.9608
1	class	1	NA	cpland	19.07745	1	99.9608

2.4.1.1 Pergunta 2 **O modelo mancha-corredor-matriz é frequentemente adotado na ecologia da paisagem. Com base nas aulas teóricas e usando os valores no objeto minha_amostra_1000 apresentados na tabela acima, identificar qual classe representar a matriz na paisagem. Há alguma informação faltando que limita a sua capacidade de identificar qual classe representar a matriz? Se sim, o que precisa ser adicionado? Justifique as suas respostas de forma clara e concisa.**

2.4.2 Ponto único, distâncias variados, métrica única

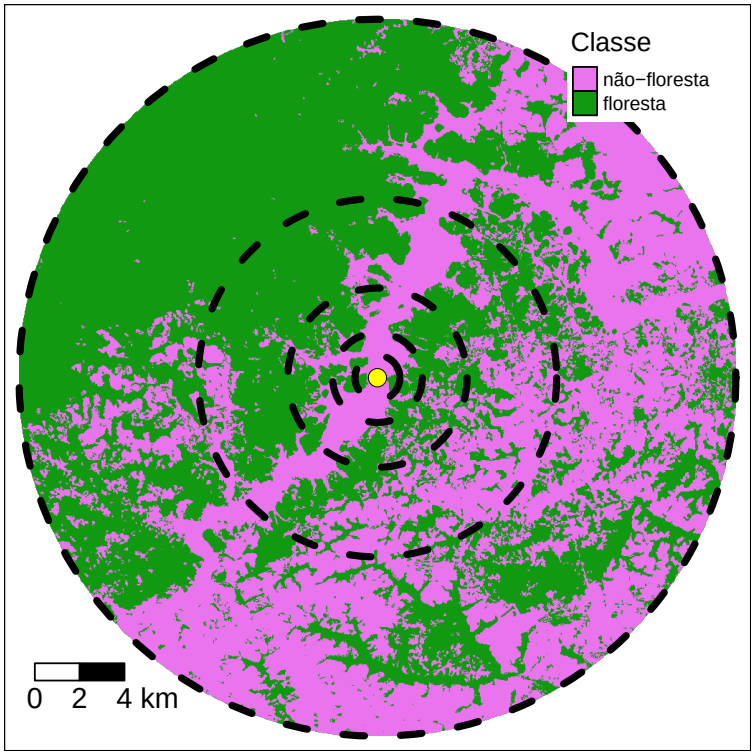


Figure 15: Cobertura florestal em extensões diferentes ao redor de um ponto de amostragem.

Para uma comparação multiescala, vamos calcular a mesma métrica, no mesmo ponto, mas agora com extensões diferentes. Continuando o exemplo no tutorial anteriio (Escala), vamos repetir o mesmo processo, mas agora com raios de 250, 500, 1000, 2000, 4000, 8000 e 16000 metros, doubrando a escala (extensão) em cada passo.

Para obter resultados com extensões diferentes, precisamos primeiramente repetir o código, ajustando para cada extensão, e depois juntar os resultados. O código a seguir calculará a mesma métrica para as diferentes distâncias. No exemplo, usamos %>%, que estabelece a ligação entre os passos do processo. Neste caso, para incluir uma coluna nova (raio) para manter o valor das diferentes distâncias.

```
# raio 250 metros
sample_lsm(floresta_2020, y = rsm_31976[1, ],
           size = 250, shape = "circle",
           metric = "cpland") %>%
  mutate(raio = 250) -> minha_amostra_250
# raio 500 metros
sample_lsm(floresta_2020, y = rsm_31976[1, ],
           size = 500, shape = "circle",
           metric = "cpland")  %>%
  mutate(raio = 500) -> minha_amostra_500
# raio 1 km (1000 metros)
sample_lsm(floresta_2020, y = rsm_31976[1, ],
           size = 1000, shape = "circle",
           metric = "cpland")  %>%
  mutate(raio = 1000) -> minha_amostra_1000
# raio 2 km
sample_lsm(floresta_2020, y = rsm_31976[1, ],
           size = 2000, shape = "circle",
           metric = "cpland") %>%
  mutate(raio = 2000) -> minha_amostra_2000
# raio 4 km
sample_lsm(floresta_2020, y = rsm_31976[1, ],
           size = 4000, shape = "circle",
           metric = "cpland")  %>%
  mutate(raio = 4000) -> minha_amostra_4000
# raio 8 km
sample_lsm(floresta_2020, y = rsm_31976[1, ],
           size = 8000, shape = "circle",
           metric = "cpland")  %>%
  mutate(raio = 8000) -> minha_amostra_8000
# raio 16 km
sample_lsm(floresta_2020, y = rsm_31976[1, ],
           size = 16000, shape = "circle",
           metric = "cpland")  %>%
  mutate(raio = 16000) -> minha_amostra_16000
```

E agora, o código a seguir juntará os resultados das diferentes extensões.

```
bind_rows(minha_amostra_250,
          minha_amostra_500,
          minha_amostra_1000,
          minha_amostra_2000,
          minha_amostra_4000,
          minha_amostra_8000,
          minha_amostra_16000) -> amostras_metrica
```

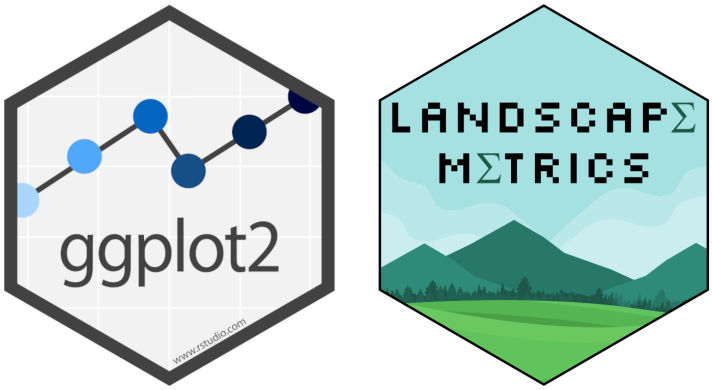
Depois que executar ("run"), podemos olhar os dados "amostras_metrica" com o codigo a seguir.

```
amostras_metrica
```

Os dados deve ter os valores (coluna value) da métrica (coluna metric) de cada classe (coluna class) para cada distância (coluna raio):

layer	level	class	id	metric	value	plot_id	percentage_inside	raio
1	class	0	NA	cpland	79.4	1	99	250
1	class	0	NA	cpland	86.9	1	100	500
1	class	1	NA	cpland	0.7	1	100	500
1	class	0	NA	cpland	66.9	1	100	1000
1	class	1	NA	cpland	19.1	1	100	1000
1	class	0	NA	cpland	57.6	1	100	2000
1	class	1	NA	cpland	26.9	1	100	2000
1	class	0	NA	cpland	36.2	1	100	4000
1	class	1	NA	cpland	42.2	1	100	4000
1	class	0	NA	cpland	35.8	1	100	8000
1	class	1	NA	cpland	45.4	1	100	8000
1	class	0	NA	cpland	37.1	1	100	16000
1	class	1	NA	cpland	46.3	1	100	16000

2.4.2.1 Faça um gráfico



Uma imagem vale mais que mil palavras. Portanto, gráficos/figuras/imagens são uma das mais importantes formas de comunicar a ciência. Os dados apresentados em uma tabela podem ser difíceis de entender. Portanto, a primeira pergunta que você deve se fazer é se você pode transformar aquela tabela (chata e feia) em algum tipo de gráfico. Lembrando, sempre pode incluir a tabela como anexo.

Aqui, vamos fazer um grafico com os dados amostras_metrica, usando o pacote ggplot2.

O ggplot2 faz parte do conjunto de pacotes tidyverse, e é um pacote de visualização de dados. "gg" se refere a uma gramática de gráficos. A ideia principal é criar um gráfico como se fosse uma frase, onde cada elemento do gráfico seria uma palavra, organizados em uma sequencia logica para construir uma frase completo (gráfico final). Você fornece os dados, informa ao ggplot2 como mapear variáveis para estética, quais tipos/formatos gráficas usar e ele cuida dos detalhes.

Isto nos permite construir gráficos tão complexos quanto quisermos. Os gráficos criados com ggplot2 são, em geral, mais elegantes do que os gráficos tradicionais do R. Para mais exemplos e tutoriais com mais detalhes veja os capitulos sobre ggplot2 nos livros:

- Ciência de Dados com R

- Análises Ecológicas no R
- No livro em ingles R Graphics Cookbook .
- E sempre pode buscar exemplos no Google, por exemplo digitando: ggplot2 grafico de barra no Google, tem mais de 50 mil resultados com paginas de imagens, codigo pronto e exemplos no YouTube.

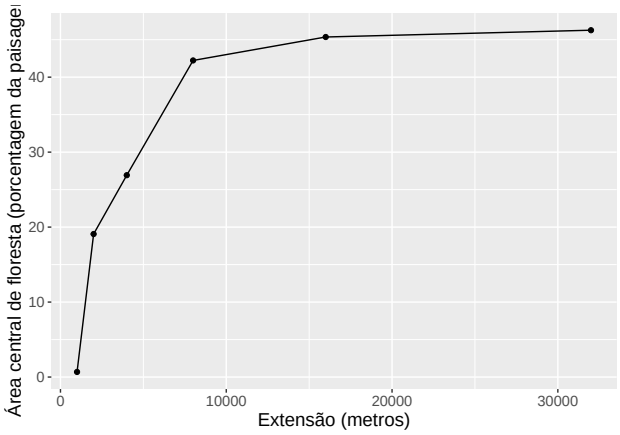
O ggplot2 exige que os dados a serem plotados estejam em um "dataframe" (tabela de dados). Ou seja, sempre teremos que transformar os dados para dataframe ou construir um dataframe com os dados que possuímos. Dataframe é um formato comum e fácil de trabalhar. Por exemplo, se você importar uma planilha de dados, o resultado seria como dataframe (para mais detalhes veja Estrutura e manipulação de objetos e lendo dados). Além disso, o resultado dos funções de landscapemetrics é sempre um dataframe, ou seja os resultados da função `sample_lsm()` são prontos para um grafico.

O principal função a ser utilizado é `ggplot()`. Para ggplot, precisamos os dados (dataframe), e depois cria o "mapeamento" das variáveis, normalmente usando `aes` (de aesthetics). Ou seja, você especifica quais são as variáveis dos eixos x e y dentro de `aes()`. Através dele vamos definir qual é a variável preditora/explanatora (eixo x) e qual é a variável resposta (eixo y) em nosso conjunto de dados. Depois da função `ggplot()`, na sequencia no codigo nós especificamos qual tipo de grafico com um "geom". Por exemplo, `geom_point()` para plotar pontos, `geom_boxplot()` para um boxplot, etc. Para a lista completa de geoms e todas as outras opções do pacote, visite a página do projeto ggplot2 https://ggplot2.tidyverse.org/index.html .

Aqui vamos fazer um gráfico com valores de extensão no eixo x e proporção da floresta central no eixo y. Assim sendo, com o codigo a segir, vamos informar (i) os dados, selecionando classe de floresta atraves de um filtro e acresentando uma coluna nova ("ext_m") com a extensão em metros, (ii) as colunas para os eixos x e y, (iii) tipo de grafico (grafico de pontos - `geom_point()` e grafico de linha - `geom_line()`), (iv) nomes para os eixos. No exemplo, usamos %>%, que estabelece a ligação entre os passos do processo, ligando os dados (amostras_metrica) e o grafico `ggplot`. Note que no codigo a seguir, adicionamos um geom com um "+". No ggplot2, nós criamos gráficos em camadas, e adicionamos camada a camada com um "+". Assim, é posivel ajustar qualquer elemento do grafico.

```
# arrumar os dados
amostras_metrica %>%
  filter(class==1) %>% mutate(ext_m = 2*raio) %>%
# fazer o grafico
  ggplot(aes(x=ext_m, y=value)) +
  geom_point() +  geom_line() +
  labs(x = "Extensão (metros)",
       y = "Área central de floresta (porcentagem da paisagem)")
```

Depois de executar ("run") o código acima, você deverá ver o grafico a seguir.



2.4.2.2 Pergunta 3 **Em vez de extensão, você preciso incluir o tamanho (área do circulo) correspondente a cada raio. Incluir uma cópia do código ajustado para produzir uma figura com tamanho (área em quilômetros quadrados) no eixo x.**

2.4.3 Faça um gráfico elegante

Podemos ajustar qualquer elemento do grafico com ggplot2. Agora, vamos mudar as unidades de metros para quilometros, aumentar o tamanho dos pontos, incluir uma linha reta para ilustrar a tendência geral, colocar o titulo longo do eixo y em duas linhas, e aumentar o tamanho da fonte para o texto ficar mais claro.

```
# arrumar os dados
amostras_metrica %>%
  filter(class==1) %>%
  mutate(ext_m = 2*raio,
         ext_km = (2*raio)/1000) %>%
# fazer o grafico
  ggplot(aes(x=ext_km, y=value)) +
  geom_point(size = 4) +
  geom_line() +
  stat_smooth(method = "lm", se = FALSE, color = "green",
              linetype = "dashed") +
  labs(x = "Extensão (quilômetros)",
       y = "Área central de floresta\n(porcentagem da paisagem)") +
  theme(text = element_text(size = 18))
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

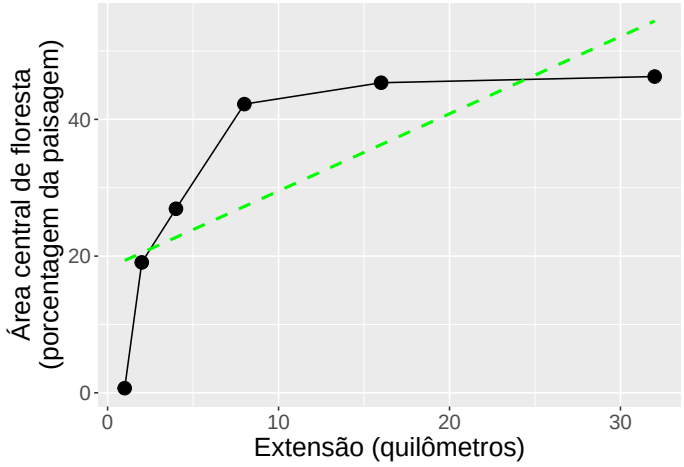


Figure 16: Comparação da área central de floresta em diferentes extensões.

2.4.3.1 Pergunta 4 **Em menos de 200 palavras apresente a sua interpretação do gráfico em figura 5.2.**

2.4.4 Modelos linear e não linear

Um dos desafios mais frequentes é como melhor representar os dados observados para gerar evidências científicas robustas e informações confiáveis. Nós vimos que as mudancas na métrica porcentagem de área central de floresta não segue uma linha reta em relação de escala (extensão). Para ir além de uma descrição simplista dos padrões observados, na ecologia da paisagem uma variedade de modelos estatísticos são usados.Não vamos rodar modelos (ainda), mas é importante entender algumas das opções disponíveis ao interpretar os gráficos.

Por exemplo, modelos de regressão são amplamente usados em diversas aplicações para descrever a relação entre uma variável resposta Y e uma variável explicativa x. Os modelos lineares são uma generalização dos testes de hipótese clássicos mais simples (Modelos linear e Modelos lineares). Uma regressão linear, só pode ser aplicada para dados em que tanto a variável preditora quanto a resposta são contínuas, enquanto uma análise de variância é utilizada quando a variável preditora/explicativa é categórica. Os modelos lineares generalizados não têm essa limitação, podemos usar variáveis contínuas ou categóricas indistintamente (Modelos Lineares Generalizados).

Mas, no caso de padroes ecologicas, sera que um modelo linear é o melhor modelo para representar a relação que explica "y" em função de "x"? Um número crescente de pesquisadores compartilham o sentimento de que as relações entre variáveis biológicas/ecologicas são melhores descritas por funções não lineares. Processos ecologicas (como por exemplo crescimento, mortalidade, dispersão, e competição) raramente são relacionadas linearmente às variáveis explicativas.

A principal vantagem do modelo não linear sobre o linear é que 1) sua escolha está associada à conhecimento prévio sobre a relação a ser modelada e 2) geralmente apresenta interpretação prática para os parâmetros. Em modelos não-linear dados observados de uma variável resposta são descritos por uma função de uma ou mais variáveis explicativas que é não linear seus parâmetros. Assim como nos modelos lineares o objetivo é identificar e estabelecer a relação entre variáveis explicativas e resposta. Entretanto, enquanto os modelos lineares definem, em geral, relações empíricas/teoricas, os modelos não-lineares são, em grande parte das vezes, motivados pelo conhecimento do tipo de relação entre as variáveis. Desta forma, as aplicações surgem nas diversas áreas onde relações físicas, biológicas, cinéticas, químicas, fisiológicas, dentre outras, são estabelecidas por funções não lineares que devem ter coeficientes (parâmetros) identificados (estimados) a partir de dados observados, dados experimentais e/ou dados simulados.

Como as mudanças na estrutura da paisagem caracterizam-se por serem não-lineares, para desenvolver análises estatísticas robustos pode (i) aplicar uma transformação (por exemplo, "log") ou (ii) adotar modelos não-lineares.

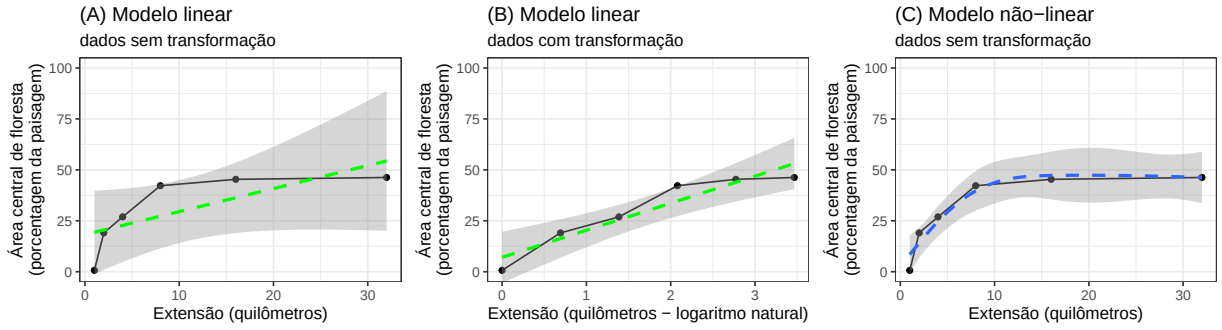


Figure 17: Comparação de padrões lineares e não-lineares.

2.4.4.1 Pergunta 5 **Comparar os resultados apresentados nas figuras com modelos lineares e não-lineares. Como podemos estabelecer qual seria o melhor modelo? Qual modelo seria mais adequado para identificar limiares no padrão de área central de floresta?**

2.5 Ponto único, distâncias variados, métricas variadas

No exemplo anterior comparamos uma métrica da paisagem em torno de um único ponto de amostragem. Mas sabemos que uma combinação de várias métricas de paisagem diferentes é necessária para entender os padrões na paisagem. Aqui mostraremos como incluir cálculos de diferentes métricas de paisagem ao mesmo tempo.

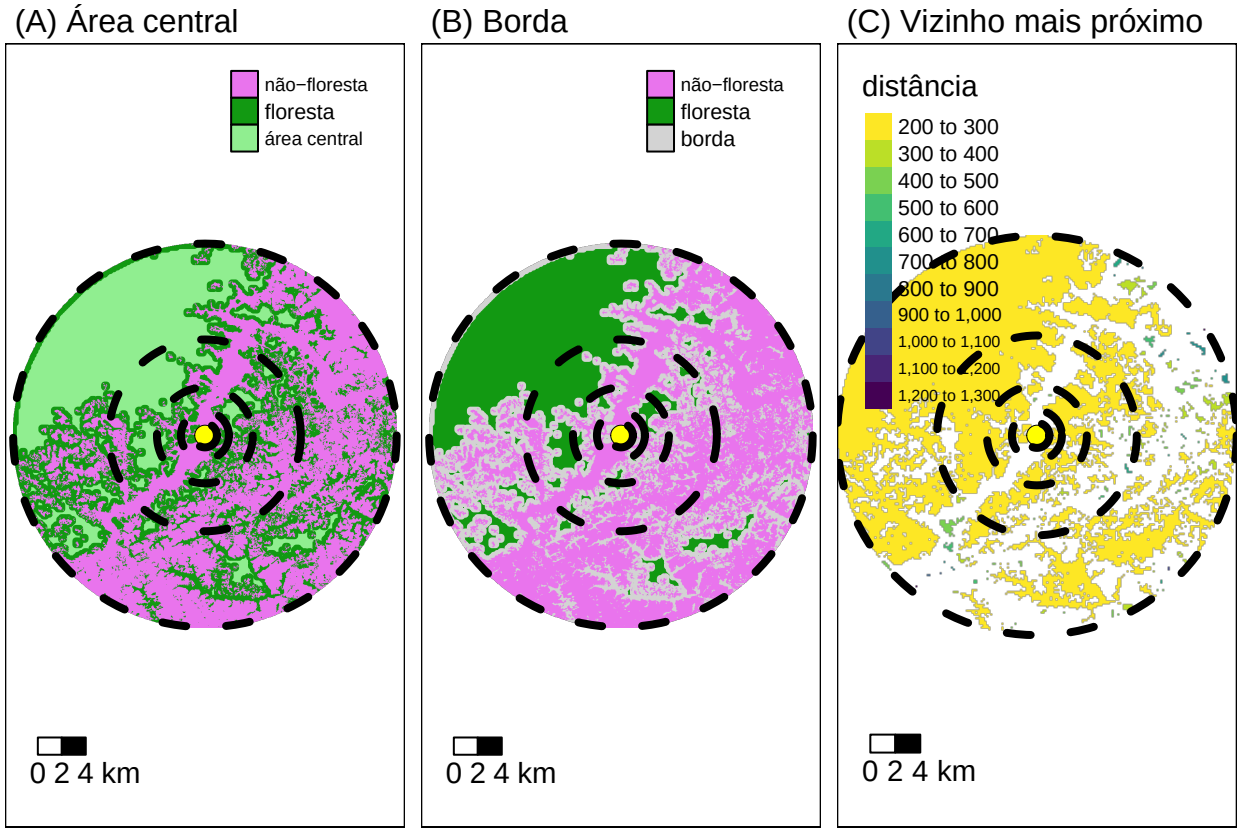


Figure 18: Ilustração da determinação de métricas da paisagem diferentes ao redor de um ponto. Exemplo com a estrutura da paisagem representado com três caracteristicas (A) Área central, (B) Borda e (C) Vizinho mais próximo. O habitat de interesse (classe) é isolado. Um buffer (linha tracejada) é colocado ao redor de um ponto (amarela) e as métricas calculadas. E em seguida o processo é repetido em diferentes extensões.

Não deve calcular todas as métricas disponíveis, mas sim, escolher aquelas que podem ser realmente adequadas para sua pergunta de pesquisa.

Calculando todas as métricas se chama um "tiro no escuro", algo cujo resultado se desconhece ou é imprevisível. Isso não é recomendado. Para fazer uma escolha melhor (mais robusta), seguindo princípios básicos da ciência, precisamos ler os estudos anteriores (artigos) para obter as métricas mais relevantes para nosso objetivo, pergunta e/ou a hipótese a ser testada. Aqui, como exemplo illustrativa vamos calcular alguns das métricas mais comums. Mas, isso não representa necessariamente as métricas mais adequados ou recommendadas.

- Métricas de área e borda (area and edge metrics). Quantificam a composição da paisagem:
  - `pland` = percentage of landscape. Percentagem da paisagem. Porcentagem de cobertura da classe na paisagem.
  - `ed` = edge density . Densidade de borda que é igual à soma dos comprimentos (m) de todos os segmentos de borda que envolvem o fragmento, dividida pela área total da paisagem (m^2), sendo posteriormente convertido em hectares.
  - `cpland` = core area percentage of landscape. Percentual de área central ("core") na paisagem. Percentual de áreas centrais (excluídas as bordas de 30 m) em relação à área total da paisagem. O termo "Core area" foi traduzido como área central ou área núcleo. Aqui vamos adotar área central.
- Métricas de agregação. Quantificam a configuração da paisagem:
  - `enn` = euclidian nearest neighbour distance. Distância euclidiana do vizinho mais próximo.
  - `enn_cv` = Coefficient of variation of euclidean nearest-neighbor distance. Coeficiente de variação da distância euclidiana do vizinho mais próximo. A métrica resume cada classe como o Coeficiente de variação das distâncias euclidianas do vizinho mais próximo entre as manchas pertencentes à classe. O valor de enn_cv = 0 se a distância euclidiana do vizinho mais próximo for idêntica para todas as manchas. Aumenta, sem limite, à medida que a variação do ENN aumenta.
  - `enn_sd` = Standard deviation of euclidean nearest-neighbor distance. Desvio padrão da distância euclidiana do vizinho mais próximo.
  - `pd` = Patch density. Densidade das manchas.
  - `cohesion` = Cohesion index. Índice de coesão das manchas.

Para incluir cálculos de diferentes métricas de paisagem ao mesmo tempo, precisamos acrescentar somente uma nova linha de codigo. Uma nova linha, que cria um objeto com os nomes das funções para as métricas que queremos calcular..... Também precisamos usar a opção "what" na função para aceitar os nomes das funções.

```
# Objeto com os nomes das funções para calcular as métricas desejadas.
minhas_metricas <- c("lsm_c_pland", "lsm_c_ed", "lsm_c_cpland",
                     "lsm_c_enn_mn", "lsm_c_enn_sd", "lsm_c_enn_cv",
                     "lsm_c_pd","lsm_c_cohesion")

# 8 Métricas calculadas para cada extensão
# raio 250 metros
sample_lsm(floresta_2020, y = rsm_31976[1, ],
           size = 250, shape = "circle",
           what = minhas_metricas) %>%
  mutate(raio = 250) -> metricas_amostra_250
# raio 500 metros
sample_lsm(floresta_2020, y = rsm_31976[1, ],
           size = 500, shape = "circle",
           what = minhas_metricas)  %>%
   mutate(raio = 500) -> metricas_amostra_500
# raio 1 km (1000 metros)
sample_lsm(floresta_2020, y = rsm_31976[1, ],
           size = 1000, shape = "circle",
```

```
           what = minhas_metricas)  %>%
   mutate(raio = 1000) -> metricas_amostra_1000
# raio 2 km (2000 metros)
sample_lsm(floresta_2020, y = rsm_31976[1, ],
           size = 2000, shape = "circle",
           what = minhas_metricas) %>%
  mutate(raio = 2000) -> metricas_amostra_2000
# raio 4 km (4000 metros)
sample_lsm(floresta_2020, y = rsm_31976[1, ],
           size = 4000, shape = "circle",
           what = minhas_metricas)  %>%
  mutate(raio = 4000)  -> metricas_amostra_4000
# raio 8 km (8000 metros)
sample_lsm(floresta_2020, y = rsm_31976[1, ],
           size = 8000, shape = "circle",
           what = minhas_metricas)  %>%
  mutate(raio = 8000) -> metricas_amostra_8000
# raio 16 km (16000 metros)
sample_lsm(floresta_2020, y = rsm_31976[1, ],
           size = 16000, shape = "circle",
           what = minhas_metricas)  %>%
  mutate(raio = 16000) -> metricas_amostra_16000
```

E agora, o código a seguir juntará os resultados das diferentes extensões.

```
bind_rows(metricas_amostra_250,
          metricas_amostra_500,
          metricas_amostra_1000,
          metricas_amostra_2000,
          metricas_amostra_4000,
          metricas_amostra_8000,
          metricas_amostra_16000) -> amostras_metricas
```

Depois que executar ("run"), podemos olhar os dados com o codigo a seguir.

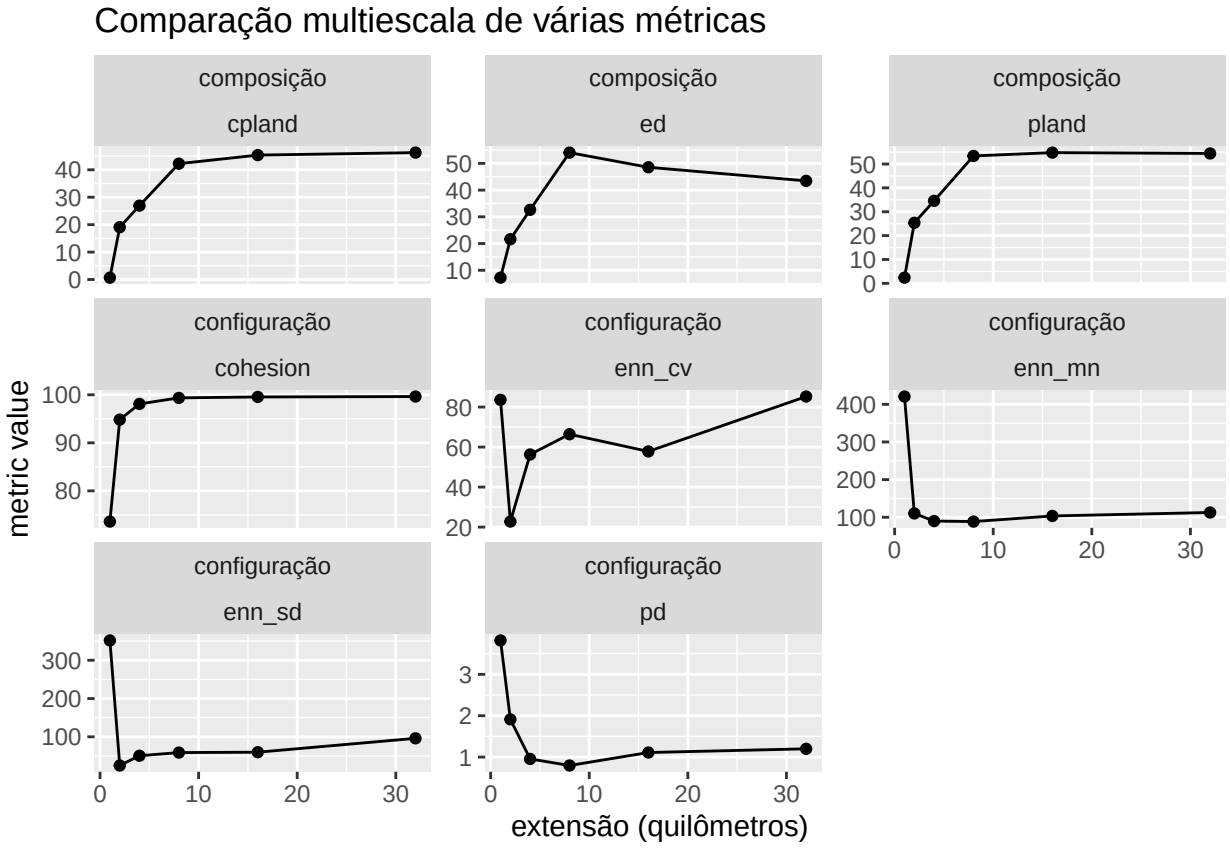
```
amostras_metricas
```

Os dados deve ter os valores (coluna value) das métricas (coluna metric) de cada classe (coluna class) para cada distância (coluna raio):

layer	level	class	id	metric	value	plot_id	percentage_inside	raio
1	class	0	NA	cohesion	100.00	1	99.3	250
1	class	0	NA	cpland	79.36	1	99.3	250
1	class	0	NA	ed	0.00	1	99.3	250
1	class	NA	NA	enn_cv	NA	1	99.3	250
1	class	NA	NA	enn_mn	NA	1	99.3	250
1	class	NA	NA	enn_sd	NA	1	99.3	250
1	class	0	NA	pd	5.13	1	99.3	250
1	class	0	NA	pland	100.00	1	99.3	250
1	class	0	NA	cohesion	99.96	1	100.1	500
1	class	1	NA	cohesion	73.61	1	100.1	500

Agorta, vamos fazer um grafico com os dados amostras_metricas, usando o pacote ggplot2. Para ajudar na visualização incluímos quais métricas são para composição e configuração e nomes que são mais fáceis de entender.

```
metricas_composicao <- c("pland", "ed", "cpland")
# arrumar dados
amostras_metricas %>%
  filter(class==1) %>%
  mutate(ext_km = (2*raio)/1000,
         met_cat = if_else(metric %in% metricas_composicao,
                           "composição", "configuração")) %>%
# fazer grafico
  ggplot(aes(x=ext_km, y=value)) +
  geom_point() +
  geom_line() +
  facet_wrap(met_cat~metric, scales = "free_y") +
  labs(title = "Comparação multiescala de várias métricas",
    x = "extensão (quilômetros)",
    y = "metric value")
```



2.5.0.1 Pergunta 6 **Com base nos resultados apresentados (figura e tabela) caracterisar as mudanças na paisagem em função de extensões diferentes. Olhando os graficos prever como seria o padrão para extensões maiores (lembrando que valores são doubrados - por exemplo raio de 250 metros gerar uma extensão de 500 metros). Seria relevante repetir incluindo calculos para extensões maiores (por exemplo 64 km e 128 km)? Justifique sua caracterização e previsões de forma clara e concisa, apoie sua escolha com exemplos da literatura científica.**

2.5.0.2 Pergunta 7 **Usando como base o conteudo das aulas, leitura disponivel no Google Classroom (Base teórica 4 Dados, métricas, analises), e/ou exemplos apresentados aqui no**

tutorial, selecione pelo menos seis métricas de nível classe para caracterizar a paisagem de estudo e objectivos da sua projeto. Justifique sua seleção de forma clara e concisa, apoie sua escolha com exemplos da literatura científica.

Part III

Exemplos de caso

3 Garimpo do Lourenço

3.1 Apresentação

Mudanças na paisagem ao redor do Garimpo do Lourenço. Changes in the landscape surrounding the Lourenço gold mine.

Código de R e dados para calcular métricas de paisagem associadas com a exploração de recursos minerários.

O objetivo é calcular métricas de paisagem e descrever a composição e a configuração da paisagem no entorno do Garimpo do Lourenço.

As métricas de paisagem são a forma que os ecólogos de paisagem usam para descrever os padrões espaciais de paisagens para depois avaliar a influência destes padrões espaciais nos padrões e processos ecológicos.

Nesta exemplo (https://rpubs.com/darren75/lourenco) aprenderemos sobre como analisar a cobertura da terra com métricas de paisagem em R.

Este exemplo tem como base teórica o modelo "mancha-corredor-matriz" - uma representação da paisagem em manchas de habitat (fragmentos).

3.2 Pacotes necessarios:

```
library(tidyverse)
library(readxl)
library(terra)
library(sf)
library(landscapemetrics)
library(mapview)
library(knitr)
library(gridExtra)
```

3.3 Área de estudo

Para alcançar o objetivo de caracterizar a paisagem no entorno do Garimpo do Lourenço, precisamos estabelecer a extensão da área de estudo. Isso seria estabelicida com base nos objetivos e estudos anteriores. Sabemos que atividades asociados com a mineração pode aumentar a perda da floresta até 70 km além dos limites do processo de mineração: Sonter et. al. 2017. Mining drives extensive deforestation in the Brazilian Amazon https://www.nature.com/articles/s41467-017-00557-w

Para visualizar um exemplo com a Extração de bauxita na Flona Saracá-Taquera: https://earthengine.google.com/timelapse/#v=-1.70085,-56.45017,8.939,latLng&t=2.70

E aqui com o Garimpo do Lourenço: https://earthengine.google.com/timelapse#v=2.2994,-51.68423,11.382,latLng&t=0.03

3.4 Dados

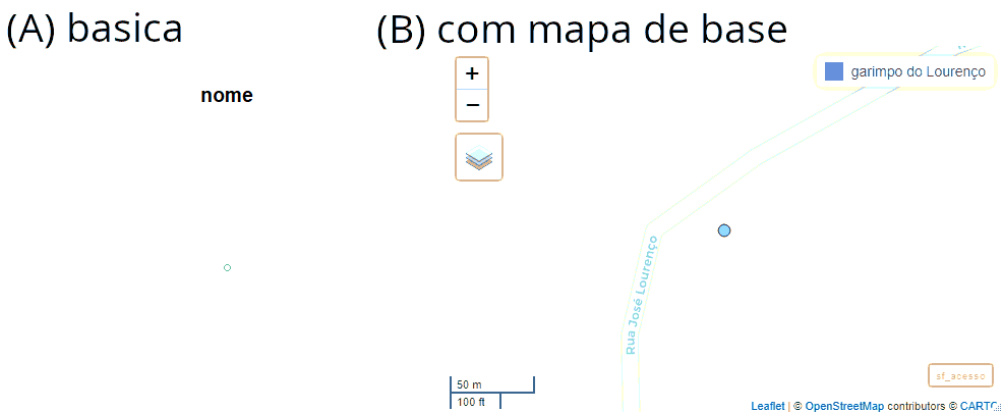
3.4.1 Ponto de referência (EPSG: 4326)

Aqui vamos incluir um raio de 20 km além do ponto de acesso para o Garimpo do Lourenço em 1985. Isso representa uma área quadrada de 40 x 40 km (1600 km2).

```
# Tabela de dados com coordenados de acesso em 1985.
acesso <- data.frame(nome = "garimpo do Lourenço",
           coord_x = -51.630871,
           coord_y = 2.318514)

# Converter para objeto espacial, com sistema de coordenados geográfica.
sf_acesso <- st_as_sf(acesso,
               coords = c("coord_x", "coord_y"),
            crs = 4326)
```

Visualizar para verificar.

```
#
plot(sf_acesso) # teste basica
mapview(sf_acesso) #verificar com mapa de base (OpenStreetMap)
```



3.4.2 Ponto de referência (EPSG: 31976)

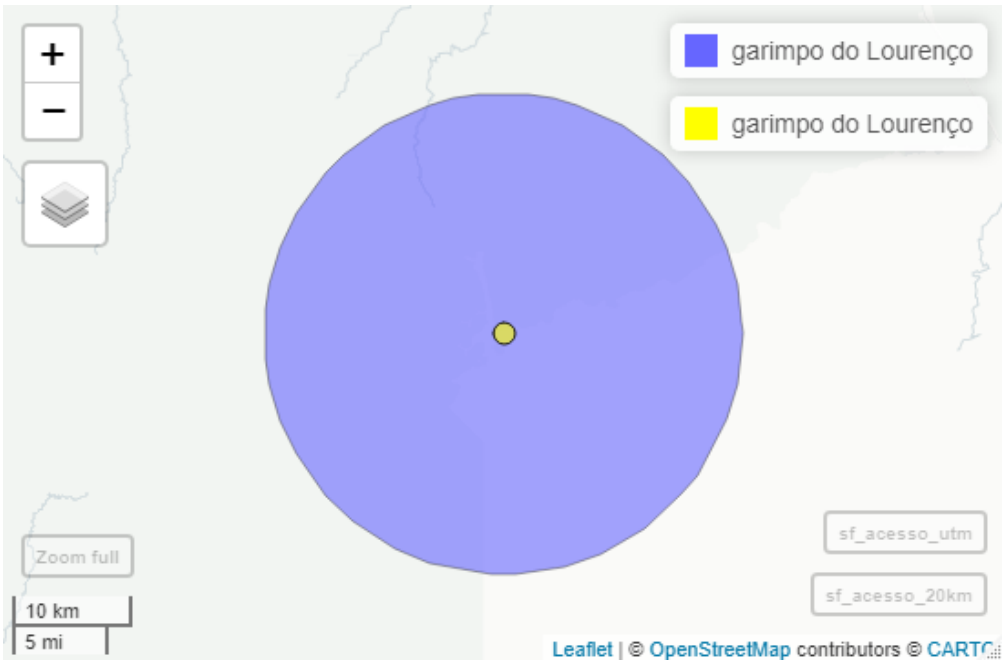
As análises da paisagem com o modelo "mancha-corredor-matriz" depende de uma classificação categórica. Portanto, deve optar para uma sistema de coordenados projetados, com pixels de área igual e com unidade em metros. Temos um raio de 20 km, que é um area geografica onde o retângulo envolvente é menor que um fuso UTM. Assim sendo, vamos adotar a sistema de coordenados projetados de datum SIRGAS 2000, especificamente EPSG:31976 (SIRGAS 2000/UTM zone 22N).

Precisamos então reprojetar o objeto original (em coordenados geográficas) para a sistema de coordenados projetados. Em seguida, vamos produzir um polígono com raio de 20 km no entorno do ponto.

```
# Reprojetar o ponto.
sf_acesso_utm <- st_transform(sf_acesso, crs = 31976)
# Polígono com raio de 500 metros no entorno do ponto.
sf_acesso_500m <- st_buffer(sf_acesso_utm, dist=500) %>%
  mutate(raio_km = 0.5)
# Polígono com raio de 1 km no entorno do ponto.
sf_acesso_1km <- st_buffer(sf_acesso_utm, dist=1000) %>%
  mutate(raio_km = 1)
# Polígono com raio de 2 km no entorno do ponto.
sf_acesso_2km <- st_buffer(sf_acesso_utm, dist=2000) %>%
  mutate(raio_km = 2)
# Polígono com raio de 4 km no entorno do ponto.
sf_acesso_4km <- st_buffer(sf_acesso_utm, dist=4000) %>%
  mutate(raio_km = 4)
# Polígono com raio de 20 km no entorno do ponto.
sf_acesso_20km <- st_buffer(sf_acesso_utm, dist=20000)

acesso_buffers <- bind_rows(sf_acesso_500m, sf_acesso_1km,
                            sf_acesso_2km, sf_acesso_4km)
```

3.4.3 Verificar com mapa de base (OpenStreetMap).

```
#
mapview(sf_acesso_20km) +
  mapview(sf_acesso_utm, color = "black", col.regions = "yellow")
```



3.4.4 Dados: MapBiomas cobertura da terra

Agora vamos olhar cobertura e uso da terra no espaco que preciso (área de estudo). Para isso, vamos utilizar um arquivo de raster do projeto MapBiomas com cobertura de terra ao redor do Garimpo do Lourenço em 1985. Este arquivo no formato raster, tem apenas valores inteiros, em que cada célula/pixel representa uma área considerada homogênea, como uso do solo ou tipo de vegetação. Arquivo ".tif" disponível aqui: utm_cover_AP_lorenco_1985.tif

Não vamos construir mapas, portanto os cores nas visualizações não corresponde ao mundo real (por exemplo, verde não é floresta). Para visualizar em QGIS preciso baixar um arquivo com a legenda e cores para Coleção¨6 (https://mapbiomas.org/codigos-de-legenda) e segue tutoriais: https://www.youtube.com/watch?v=WtyotodHK8E .

Este vez, a entrada de dados espaciais seria atraves a importação de um raster (arquivo de .tif). Lembre-se, para facilitar, os arquivos deve ficar no mesmo diretório do seu código (verifique com getwd()). Como nós já sabemos a sistema de coordenados desejadas, o geoprocessamento da raster foi concluído antes de começar com as análises da paisagem.

```
r1985 <- rast("utm_cover_AP_lorenco_1985.tif")
r1985

#class       : SpatRaster
#dimensions  : 1341, 1341, 1  (nrow, ncol, nlyr)
#resolution  : 29.87713, 29.87713  (x, y)
#extent      : 409829.5, 449894.7, 236241.1, 276306.3  (xmin, xmax, ymin, ymax)
#coord. ref. : SIRGAS 2000 / UTM zone 22N (EPSG:31976)
#source      : utm_cover_AP_lorenco_1985.tif
#name        : classification_1985
#min value   :                   3
#max value   :                  33
```

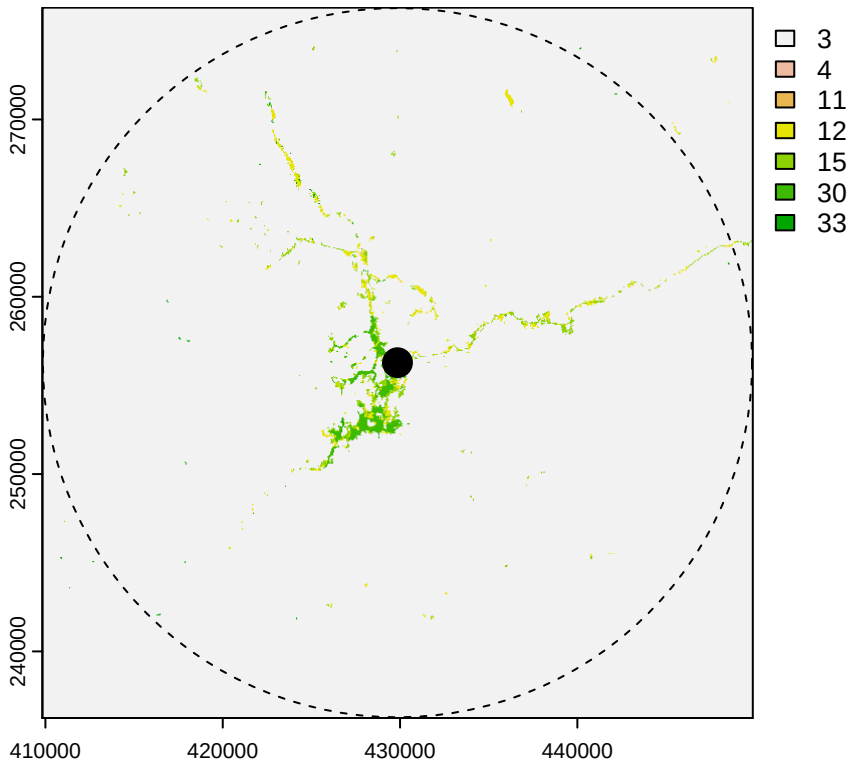
Ou use o função file.choose(), que faz a busca para arquivos.

```
r1985 <- rast(file.choose())
r1985
```

```
##            used  (Mb) gc trigger  (Mb)  max used   (Mb)
## Ncells  5794335 309.5   13604237 726.6  13604237  726.6
## Vcells 12676611  96.8  107057221 816.8 141606949 1080.4
```

Agora que o arquivo foi importado, podemos visualizá- lo.

```
# Visualizar para verificar
# Gradiente de cores padrão não corresponde
# ao mundo real (por exemplo verde não é floresta)
plot(r1985, type="classes")
plot(sf_acesso_20km, add = TRUE, lty ="dashed", color = "black")
plot(sf_acesso_utm, add = TRUE, cex = 2, pch = 19, color = "black")
```



3.5 Calculo de métricas

Vamos olhar alguns exemplos de métricas para cada nível da análise:

- landscape (métricas para a paisagem como um todo).
- class (métricas por classe ou tipo de habiat).
- patch (para a mancha ou fragmento).

Primeiro, pecisamos verificar se o raster está no formato correto.

```
check_landscape(r1985)
```

```
##   layer       crs units   class n_classes OK
## 1     1 projected     m integer         7  v
```

```
#  layer crs    units   class n_classes OK
#  1  projected   m   integer         7  v
```

Tudo certo (veja a coluna do "OK")!

3.5.1 Métricas para a paisagem

Vamos começar avaliando a área total da paisagem (área) de estudo.

```
area.total <- lsm_l_ta(r1985)
area.total #160264 Hectares
```

```
## # A tibble: 1 x 6
##   layer level     class    id metric   value
##   <int> <chr>     <int> <int> <chr>    <dbl>
## 1     1 landscape    NA    NA ta     160264.
```

Agora vamos ver a distância total de borda (te= "total edge").

```
##            used  (Mb) gc trigger  (Mb)  max used   (Mb)
## Ncells  5798853 309.7   13604237 726.6  13604237  726.6
## Vcells 12678376  96.8   85645777 653.5 141606949 1080.4
```

```
te <- lsm_l_te(r1985)
te # 547140 metros
```

```
## # A tibble: 1 x 6
##   layer level     class    id metric   value
##   <int> <chr>     <int> <int> <chr>    <dbl>
## 1     1 landscape    NA    NA te     547140.
```

Total de borda mede a configuração da paisagem porque uma paisagem altamente fragmentada terá muitas bordas. No entanto, a borda total é uma medida absoluta, dificultando comparações entre paisagens com áreas totais diferentes. Mas pode ser aplicado para comparar a configuração na mesma paisagem em anos diferentes.

Agora vamos ver a densidade de Borda ("Edge Density"). Densidade de Borda mede a configuração da paisagem porque uma paisagem altamente fragmentada terá valores mais altas. "Densidade" é uma medida adequado para comparacoes de paisagens com áreas totais diferentes.

```
ed <- lsm_l_ed(r1985)
ed #3.41 metros por hectare
```

```
## # A tibble: 1 x 6
##   layer level     class    id metric value
##   <int> <chr>     <int> <int> <chr>  <dbl>
## 1     1 landscape    NA    NA ed      3.41
```

3.5.2 Métricas para as classes

Area de cada classe em hectares.

```
lsm_c_ca(r1985)
```

```
## # A tibble: 7 x 6
##   layer level class    id metric     value
##   <int> <chr> <int> <int> <chr>      <dbl>
## 1     1 class     3    NA ca     158582.
## 2     1 class     4    NA ca          1.70
```

```
## 3     1 class    11    NA ca          1.79
## 4     1 class    12    NA ca        548.
## 5     1 class    15    NA ca        563.
## 6     1 class    30    NA ca        526.
## 7     1 class    33    NA ca         41.4
```

Como tem varios classes é dificil de interpretar os resultados porque os numeros (3, 4, 11.....) não tem uma referncia do mundo real. Para entender os resultados, precisamos acrescentar nomes para os valores. Ou seja incluir uma coluna de legenda com os nomes para cada classe. Para isso precisamos outro arquivo com os nomes.

Baixar a arquivo de legenda: mapbiomas_6_legend.xlsx.

Agora carregar o arquivo com o codigo a seguir.

```
class_nomes <- read_excel(file.choose())
```

Agora rodar de novo, com os resultados juntos com a legenda de cada classe. Nos resultados acima, os valores na coluna "class" são as mesmas que tem na coluna "aid" no objeto "class_nomes", onde também tem os nomes . Assim, podemos repetir, mas agora incluindo os nomes para cada valor de class, com base na ligação (join) entre as colunas.

```
# Área de cada classe em hectares, incluindo os nomes para cada classe
lsm_c_ca(r1985) %>%
  left_join(class_nomes, by = c("class" = "aid"))

# Numero de fragmentos (manchas)
lsm_c_np(r1985) %>%
  left_join(class_nomes, by = c("class" = "aid"))

# Maior numero de manchas em classes de cobertura classificadas como
# pasto (pasture) e formação campestre (grassland).

#  layer level class    id metric value class_description  group_description
#     1 class     3    NA np        28 Forest Formation     Natural forest
#     1 class     4    NA np         2 Savanna Formation    Natural forest
#     1 class    11    NA np         7 Wetlands             Natural non fore
#     1 class    12    NA np       246 Grassland            Natural non fore.
#     1 class    15    NA np       262 Pasture              Farming
#     1 class    30    NA np        35 Mining               Non vegetated
#     1 class    33    NA np        50 River,Lake and Ocean Water
```

3.5.3 Métricas para as manchas

Vamos calcular o tamanho de cada mancha agora.

```
mancha_area <- lsm_p_area(r1985) # 630 manchas
mancha_area
```

Agora queremos saber o tamanho da maior mancha em cada class, e portanto o tamanho da maior mancha de mineração.

```
mancha_area %>%
group_by(class) %>%
summarise(max_ha = max(value))
# 30.8 hectares (class 15 = mineração)
```

3.5.4 Quais métricas devo escolher?

A decisão deve ser tomada com base em uma combinação de fatores. Incluindo tais fatores como: base teórica, considerações estatísticas, relevância para o objetivo/hipótese e a escala e heterogeneidade na paisagem de estudo.

Queremos caracterizar áreas de mineração na paisagem, e aqui vamos olhar somente uma paisagem, em um momento do tempo. Então as métricas para a paisagem como todo não tem relevância.

Estamos olhando uma classe (mineração), portanto vamos incluir as métricas para classes. Alem disso, as métricas de paisagem em nível de classe são mais eficazes na definição de processos ecológicos (Tischendorf, L. Can landscape indices predict ecological processes consistently?. Landscape Ecology 16, 235–254 (2001). https://doi.org/10.1023/A:1011112719782.).

```
# métricas de composição para a paisagem por classes
list_lsm(level = "class", type = "area and edge metric")

# métricas de configuração para a paisagem por classes
list_lsm(level = "class", type = "aggregation metric")
```

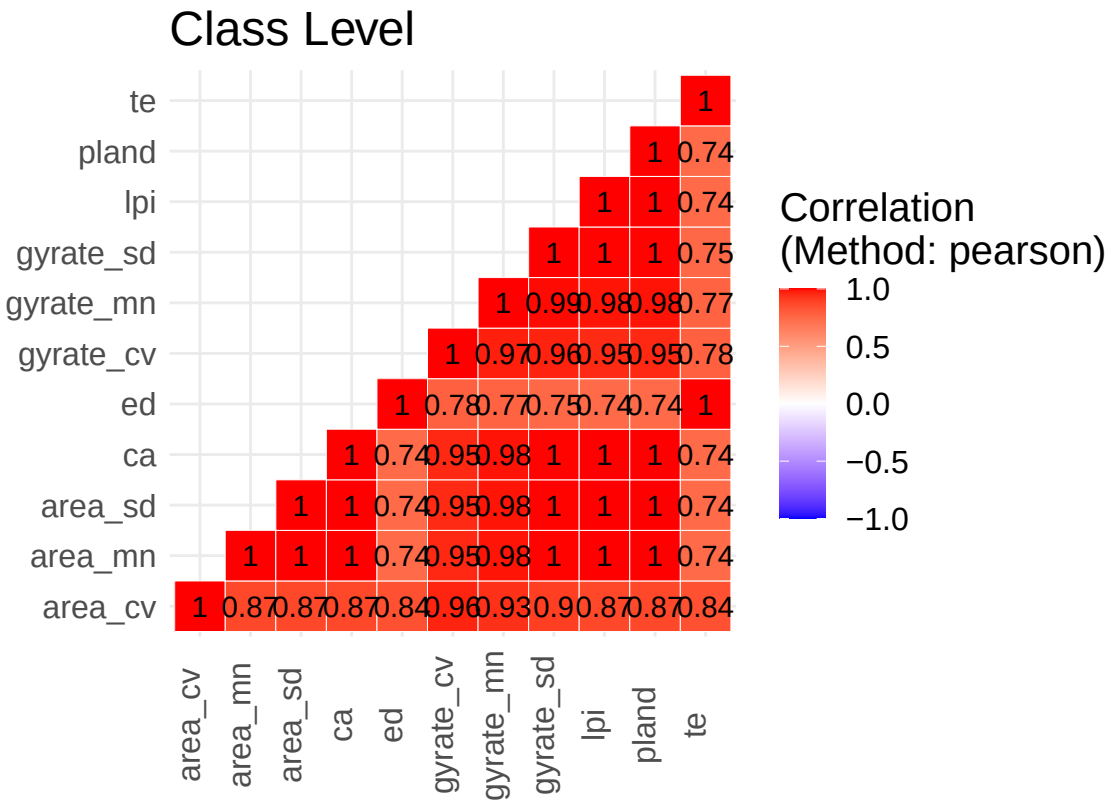
3.5.5 Métricas por classe de mineração

Aqui vamos calcular todos as métricas por classe (função calculate_lsm())).

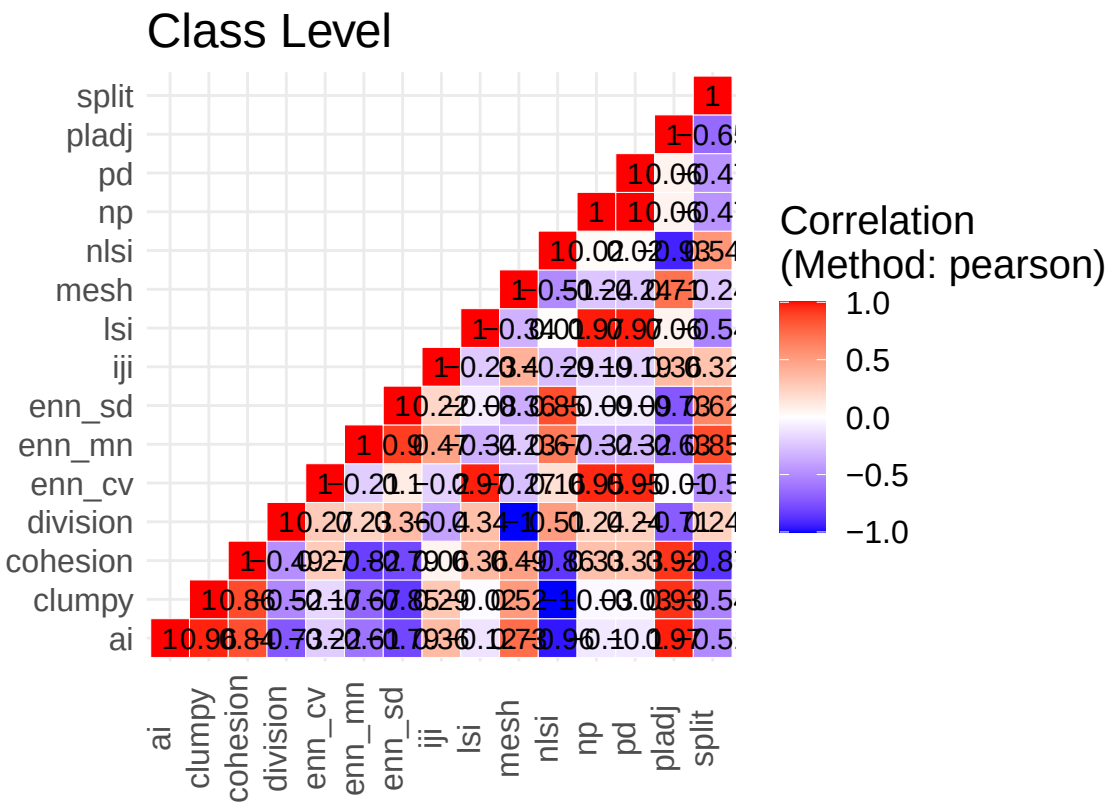
```
# métricas de composição para a paisagem por classes
metrics_comp <- calculate_lsm(r1985, level = "class", type = "area and edge metric")

# métricas de configuração para a paisagem por classes
metrics_config <- calculate_lsm(r1985, level = "class", type = "aggregation metric")
```

E aqui, calcular correlações entre todos as métricas por classe (função show_correlation())).

```
show_correlation(data = metrics_comp, method = "pearson", labels = TRUE)
```




```
show_correlation(data = metrics_config, method = "pearson", labels = TRUE)
```



Temos muitos valores e muitas métricas. Este se chama um "tiro no escuro", algo cujo resultado se desconhece ou é imprevisível. Isso não é recomendado. Para fazer uma escolha melhor (mais robusta), seguindo princípios básicos da ciência, precisamos ler os estudos anteriores (artigos) para obter as métricas mais relevantes para nosso objetivo e a hipótese a ser testada. Com base em os estudos anteriores e os objetivos vamos incluir 8 métricas nos resultados.

3.5.6 Exportar as métricas

O próximo passo é comunicar os resultados obtidos. Para isso precisamos resumir e apresentar as métricas selecionadas em tabelas e figuras. Agora já fizemos os cálculos, as tabelas e figuras podem ser feitas no R (figuras), tanto quanto em aplicativos diferentes (por exemplo tabelas atraves ["tabelas dinamicas"] no Microsoft Excel ou LibreOffice calc). Mas por isso, primeiramente precisamos exportar os resultados (veja mais exemplos aqui: Introdução ao R import-export.

O arquivo vai sai no mesmo diretório do seu código (verifique com getwd()).

```
bind_rows(metrics_comp, metrics_config) -> metricas_1985
write.csv2(metricas_1985, "metricas_lourenco_1985.csv", row.names=FALSE)
```

3.6 Preparando os resultados

A entrada de dados seria com as métricas da paisagem calculados anteriormente.

Vocês devem baixar o arquivo de Excel metricas_lourenco_1985.xlsx. Lembre-se, para facilitar, os dados deve ficar no mesmo diretório do seu código (verifique com getwd()).

No caso de um arquivo de Excel simples, a importação poderia ser feita através menu de "Import Dataset" na janela/panel "Environment" de Rstudio. Ou com linhas de código:

```
metricas_1985 <- read_excel("metricas_lourenco_1985.xlsx")
metricas_1985

#  layer level class id    metric     value
#   <dbl> <chr> <dbl> <chr> <chr>      <dbl>
#     1 class     3 NA    area_cv  529.
#     1 class     4 NA    area_cv   22.3
#     1 class    11 NA    area_cv   71.2
```

Ou use o função file.choose(), que faz a busca para arquivos.

```
metricas_1985 <- read_excel(file.choose())
metricas_1985
```

```
## # A tibble: 182 x 6
##    layer level class id    metric     value
##    <dbl> <chr> <dbl> <chr> <chr>      <dbl>
##  1     1 class     3 NA    area_cv  529.
##  2     1 class     4 NA    area_cv   22.3
##  3     1 class    11 NA    area_cv   71.2
##  4     1 class    12 NA    area_cv  169.
##  5     1 class    15 NA    area_cv  175.
##  6     1 class    30 NA    area_cv  276.
##  7     1 class    33 NA    area_cv   74.2
##  8     1 class     3 NA    area_mn 5664.
##  9     1 class     4 NA    area_mn    0.848
## 10     1 class    11 NA    area_mn    0.255
## # i 172 more rows
```

Os dados são padronizados (“tidy”), mas ainda não parece adequados para apresentação em tabelas ou figuras. Temos muitos valores e muitas métricas (listadas na coluna “metric”). Com base em os estudos anteriores e os objetivos vamos incluir 8 métricas (4 de composição e 4 de configuração).

Métricas de composição:

- mean patch area (lsm_c_area_mn) Área médio das manchas por classe.
- SD patch area (lsm_c_area_sd) Desvio padrão das áreas dos manchas por classe.
- class area percentage of landscape (lsm_c_pland) Porcentagem de área na paisagem por classe.
- largest patch index (lsm_c_lpi) Índice de maior mancha (proporção da paisagem).

Métricas de configuração:

- aggregation index (lsm_c_ai) Índice de agregação.
- patch cohesion index (lsm_c_cohesion) Índice de coesão das manchas.
- number of patches (lsm_c_np) Número de manchas.
- patch density (lsm_c_pd) Densidade de manchas.

Escolheremos (atraves um filtro) as métricas que queremos para obter uma tabela de dados. Mantendo os dados originais, assim sendo para acresentar mais métricas nos resultados, preciso somente acrescentar mais no codigo.

```
# Arquivo com os nomes das classes
class_in <- "C:\\Users\\user\\Documents\\Articles\\gis_layers\\gisdata\\inst\\raster\\mapbiomas_cover_l
class_nomes <- read_excel(class_in)

# Especificar métricas desejados
met_comp <- c("pland", "lpi", "area_mn", "area_sd")
met_conf <- c("ai", "cohesion", "np", "pd")
```

```
met_todos <- c(met_comp, met_conf)

# Escholer métricas desejados do conjunto completo
metricas_1985 %>%
filter(metric %in% met_todos) %>%
left_join(class_nomes, by = c("class" = "aid")) -> metricas_nomes
```

3.7 Uma tabela versatil

Mas, ainda não tem uma coluna com os nomes das métricas. Portanto, solução simples é de exportar no formato de .csv e finalizar/editar no Excel / calc.

Outra opção que pode facilitar, particularmente quando pode há mudanças e revisões, é produzir a tabela no R. Aqui vamos repetir no R os passos que vocês conhecem com as ferramentas de Excel (arraste e solte, copiar-colar, filtro, tabela dinâmica).

3.8 Reorganização

Escolhendo as colunas desejadas (select), reorganizando para as métricas ficam nas colunas (pivot_wider) e colocando as colunas novas na sequência desejada (select).

```
metricas_nomes %>%
# Escholer métricas desejados do conjunto completo de métricas.
dplyr::select(c(type_class, classe_descricao, hexadecimal_code,
metric, value)) %>%
# reorganizando
pivot_wider(names_from = metric, values_from = value) -> metricas_tab
```

3.9 Uma figura elegante

Uma imagem vale mais que mil palavras. Portanto, gráficos/figuras/imagens são uma das mais importantes formas de comunicar a ciência.

Como exemplo ilustrativo, aqui vamos produzir gráficos comparando métricas de composição e configuração da paisagem ao redor do Garimpo do Lourenço.

É uma boa ideia gastar bastante tempo para tornar figuras científicas as mais informativas e atraentes possíveis. Escusado será dizer que a precisão empírica é primordial. E por isso, o que fica excluído/omitido é tão importante quanto o que foi incluído. Para ajudar, você deve se perguntar o seguinte ao criar uma figura: eu apresentaria essa figura em uma apresentação para um grupo de colegas? Eu o apresentaria a um público de não especialistas? Eu gostaria que essa figura aparecesse em um artigo de notícias sobre meu trabalho? É claro que todos esses locais exigem diferentes graus de precisão, complexidade e estética, mas uma boa figura deve servir para educar simultaneamente públicos muito diferentes.

Tabelas versus gráficos — A primeira pergunta que você deve se fazer é se você pode transformar aquela tabela (chata e feia) em algum tipo de gráfico. Você realmente precisa dessa tabela no texto principal? Você não pode simplesmente traduzir as entradas das células em um gráfico de barras/colunas/xy? Se você pode, você deve. Quando uma tabela não pode ser facilmente traduzida em uma figura, na maioria das vezes a provavelmente pertence às Informações Suplementares/Anexos/Apêndices.

3.9.1 Gráfico de barra

Primeiramente, vamos produzir uma gráfico de barra comparando a proporção que cada classe representa na paisagem.

```
# Inclundo cores conforme legenda da Mapbiomas Coleção 6
# Legenda nomes ordem alfabetica
classe_cores <- c("Campo Alagado e Área Pantanosa" = "#45C2A5",
"Formação Campestre" = "#B8AF4F",
"Formação Florestal" = "#006400",
"Formação Savânica" = "#00ff00",
"Mineração" = "#af2a2a",
"Pastagem" = "#FFD966",
"Rio, Lago e Oceano" = "#0000FF")
```

E agora o grafico.....

```
# Grafico de barra basica
metricas_tab %>%
mutate(class_prop = pland) %>%
ggplot(aes(x = classe_descricao, y = class_prop)) +
geom_col()


# Agora com ajustes
# Agrupando por tipo (natural e antropico)
# Com cores conforme legenda da Mapbiomas Coleção 6
# Corrigindo texto dos eixos.
# Mudar posição da leganda para o texto com nomes longas encaixar.
metricas_tab %>%
mutate(class_prop = pland) %>%
ggplot(aes(x = type_class, y = class_prop,
fill = classe_descricao)) +
scale_fill_manual("classe", values = classe_cores) +
geom_col(position = position_dodge2(width = 1)) +
```

```
coord_flip() +
labs(title = "MapBiomas cobertura da terra",
subtitle = "Entorno do Garimpo do Lorenço 1985",
y = "Proporção da paisagem (%)",
x = "") +
theme(legend.position="bottom") +
guides(fill = guide_legend(nrow = 4))
```

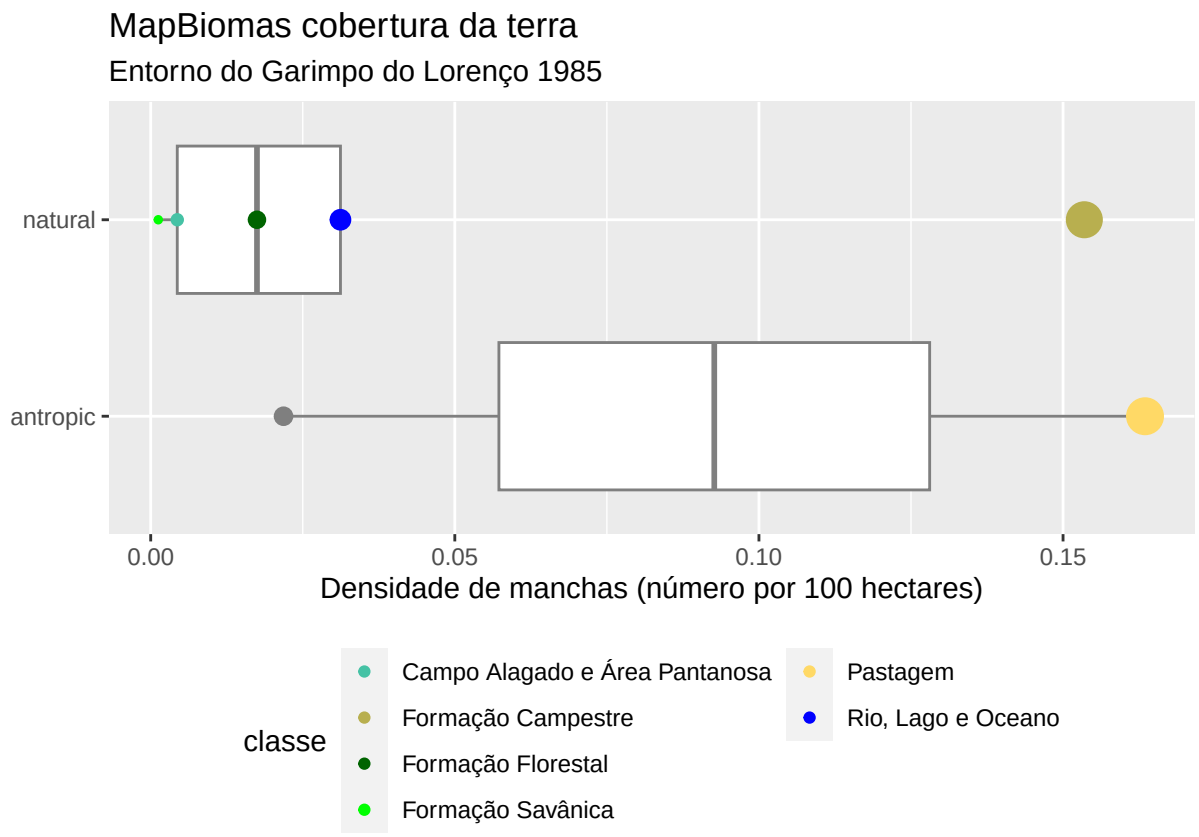
Uma imagem vale mais que mil palavras:

Mas existe uma separação grande na faixa de valores e ainda é difícil de ver todas as classes. Temos uma distribuição com valores muito mais altos comparada com os outros. extremos. Uma solução seria uma transformação (por exemplo "log"), assim os valores ficarem mais proximos.

3.9.2 Gráfico de boxplot

Agora com uma métrica de configuração:

```
# Métrica de configuração: Densidade de manchas (coluna "pd").
# Agrupando por tipo (natural e antrópico)
# Incluindo boxplot indicando tendência central (mediano)
# Com cores conforme legenda da Mapbiomas Coleção 6
# Tamanho dos pontos proporcional o numero de manchas
# Corrigindo texto dos eixos.
# Mudar posição da leganda para o texto com nomes longas encaixar.
metricas_tab %>%
ggplot(aes(x = type_class, y = pd)) +
geom_boxplot(colour = "grey50") +
geom_point(aes(size = np, colour = classe_descricao)) +
scale_color_manual("classe", values = classe_cores) +
scale_size(guide = "none") +
coord_flip() +
labs(title = "MapBiomas cobertura da terra",
subtitle = "Entorno do Garimpo do Lorenço 1985",
y = "Densidade de manchas (número por 100 hectares)",
x = "") +
theme(legend.position="bottom") +
guides(col = guide_legend(nrow = 4))
```



3.10 Comparação entre anos

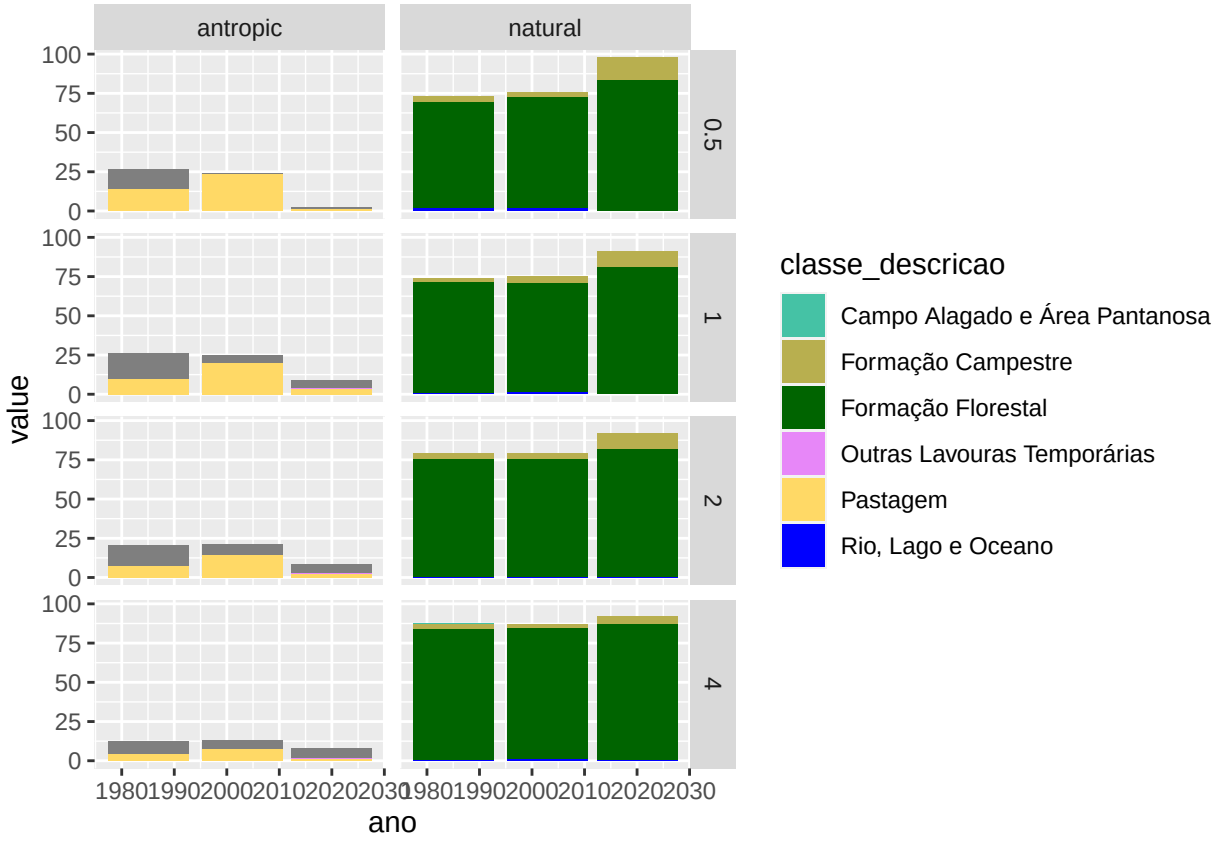
Calcular as métricas para 3 anos.

```
# metricas desejados
what_metricas <- c("lsm_c_pland", "lsm_c_lpi", "lsm_c_area_mn", "lsm_c_area_sd",
                   "lsm_c_ai", "lsm_c_cohesion", "lsm_c_np", "lsm_c_pd")
# rodar
metricas_anos <- sample_lsm(landscape = mapbiomas_85a20,
                            y = acesso_buffers,
                           plot_id = data.frame(acesso_buffers)[, 'raio_km'],
                             what = what_metricas,
                           edge_depth = 1)

# Organizar dados
# Dados rferentes os buffers
resultados_anos <- acesso_buffers %>%
  left_join(metricas_anos %>%
              dplyr::mutate(value = round(value,2),
        ano = case_when(layer==1 ~1985,
                        layer==2~2003,
                        layer==3~2020)) %>%
              dplyr::select(ano, plot_id, class, metric, value),
            by=c("raio_km"="plot_id"))
# Dados referentes os classes
resultados_anos <- resultados_anos %>%
```

```
left_join(class_nomes, by = c("class" = "aid"))
```

Agora grafico de barra com varios anos

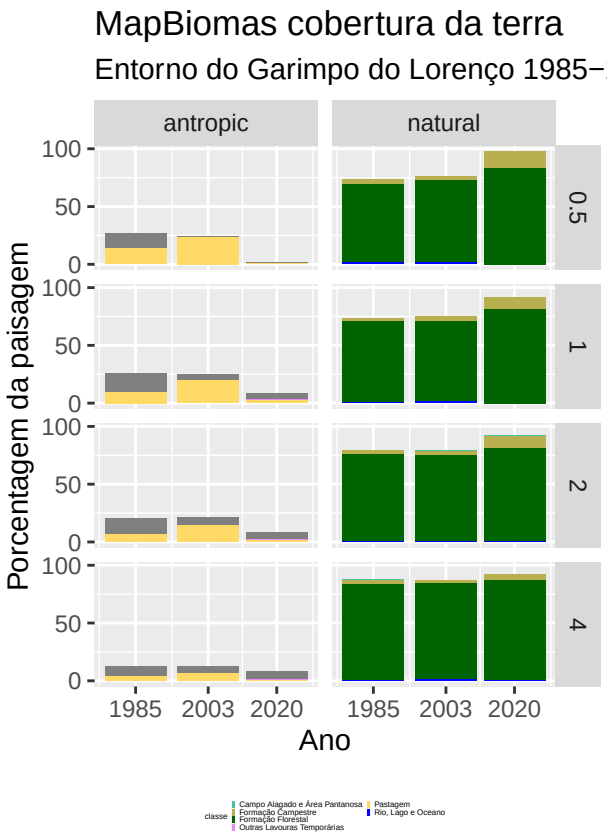
```
# It's recommended to use a named vector
# Legenda nomes ordem alfabetica
classe_cores <- c("Campo Alagado e Área Pantanosa" = "#45C2A5",
"Formação Campestre" = "#B8AF4F",
"Formação Florestal" = "#006400",
"Formação Savânica" = "#00ff00",
"Mineração" = "#af2a2a",
"Pastagem" = "#FFD966",
"Rio, Lago e Oceano" = "#0000FF",
"Outras Lavouras Temporárias" = "#e787f8")

resultados_anos %>%
  mutate(rcor = paste("#", hexadecimal_code, sep="")) %>%
  filter(metric=="pland") %>%
  ggplot(aes(x=ano, y=value)) +
  geom_col(position="stack", aes(fill=classe_descricao)) +
  scale_fill_manual(values = classe_cores) +
  facet_grid(raio_km~type_class)
```

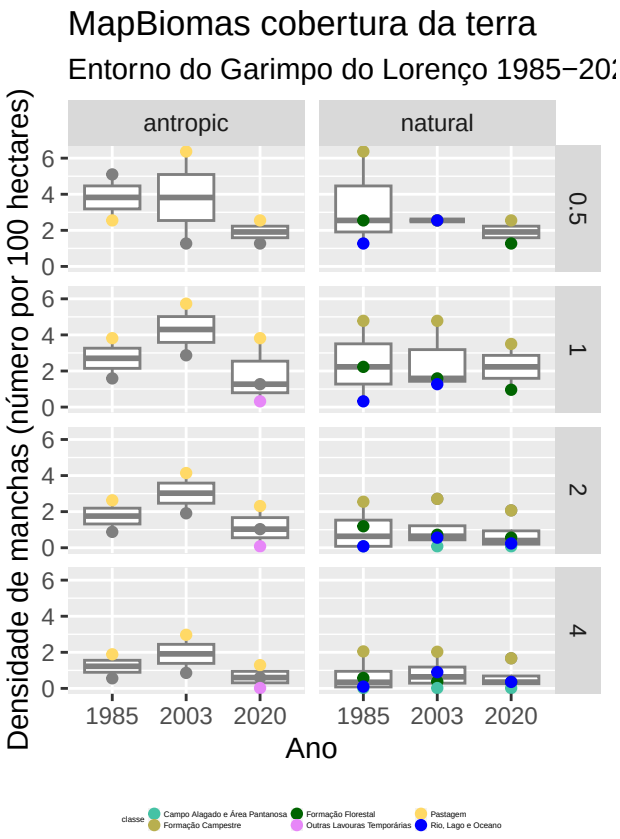


Agora com "pland" e densidade de manchas juntos.

```
resultados_anos %>%
  mutate(rcor = paste("#", hexadecimal_code, sep="")) %>%
```

```
  filter(metric=="pland") %>%
  ggplot(aes(x=ano, y=value)) +
  geom_col(position="stack", aes(fill=classe_descricao)) +
  scale_fill_manual("classe", values = classe_cores) +
  scale_y_continuous(breaks=c(0,50,100)) +
  scale_x_continuous(breaks=c(1985,2003, 2020)) +
  facet_grid(raio_km~type_class) +
  labs(title = "MapBiomas cobertura da terra",
subtitle = "Entorno do Garimpo do Lorenço 1985-2020",
y = "Porcentagem da paisagem",
x = "Ano") +
theme(legend.position="bottom",
      legend.title = element_text(size = 3),
              legend.text  = element_text(size = 3),
              legend.key.size = unit(0.1, "lines")) +
guides(fill = guide_legend(nrow = 4)) -> fig_pland

# Densidade de manchas
resultados_anos %>%
  mutate(rcor = paste("#", hexadecimal_code, sep="")) %>%
  filter(metric=="pd") %>%
ggplot(aes(x = factor(ano), y = value)) +
geom_boxplot(colour = "grey50") +
geom_point(aes(colour = classe_descricao)) +
scale_color_manual("classe", values = classe_cores) +
scale_size(guide = "none") +
facet_grid(raio_km~type_class) +
labs(title = "MapBiomas cobertura da terra",
subtitle = "Entorno do Garimpo do Lorenço 1985-2020",
y = "Densidade de manchas (número por 100 hectares)",
x = "Ano") +
theme(legend.position="bottom",
      legend.title = element_text(size = 3),
              legend.text  = element_text(size = 3),
              legend.key.size = unit(0.1, "lines")) +
guides(fill = guide_legend(nrow = 4)) -> fig_pd

grid.arrange(fig_pland, fig_pd, nrow=1)
```


MapBiomas cobertura da terra
Entorno do Garimpo do Lorenço 1985−2
antropic
natural
0.5
1
2
4
Porcentagem da paisagem
100
50
0
1985
2003
2020
Ano
classe
Campo Alagado e Área Pantanosa
Formação Campestre
Formação Florestal
Outras Lavouras Temporárias
Pastagem
Rio, Lago e Oceano


MapBiomas cobertura da terra
Entorno do Garimpo do Lorenço 1985−20
antropic
natural
0.5
1
2
4
Densidade de manchas (número por 100 hectares)
6
4
2
0
1985
2003
2020
Ano
classe
Campo Alagado e Área Pantanosa
Formação Campestre
Formação Florestal
Outras Lavouras Temporárias
Pastagem
Rio, Lago e Oceano

3.11 Conclusões e próximos passos

Os resultados apresentados na figura anterior não segam os resultados esperados que a cobertura de classes antrópicos ia aumentar ao longo do tempo. Para entender melhor os padores, precisamos:

- Verficar padrões nas outras metricas calculados
- Verficar padrões com mais anos
- Verficar padrões usando poligonos/pontos dos processos de mineração (dados no SIGMINE https://www.gov.br/anm/pt-br e https://app.anm.gov.br/dadosabertos/SIGMINE/PROCESSOS_MINERARIOS/)

Alem disso, seria relevante buscar complementar os dados de MapBiomas com uma classificação supervisionado usando imagens de Sentinel-2 (exemplo com QGIS Semi-Automatic Classification plugin aqui: https://fromgistors.blogspot.com/2016/09/basic-tutorial-2.html). Assim para aumentar a precisão dos resultados.

Uma forma alternativa para visualização as mudanças entre anos seria um diagrama "Sankey"/"Alluvial". Como exemplo, veja figura 3 no artigo "Rapid land use conversion in the Cerrado has affected water transparency in a hotspot of ecotourism, Bonito, Brazil" https://doi.org/10.1177/19400829221127087 .

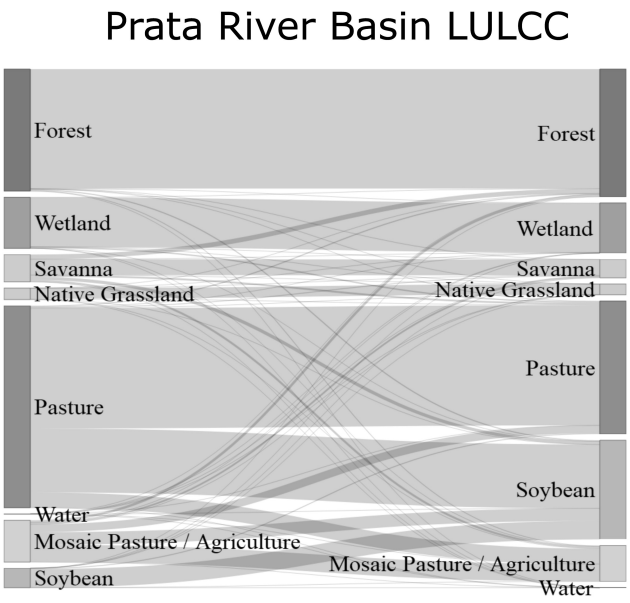


Figure 19: Diagrama Sankey mostrando a mudança de uso da terra na bacia do rio Prata entre 2010 (lado esquerdo) e 2020 (lado direto). Fonte: Figura 3. Chiaravalloti et. al. 2022. Tropical Conservation Science. doi:10.1177/19400829221127087

No R pode fazer com o pacote "networkD3" segue tutoriais:

- https://www.displayr.com/sankey-diagrams-r/
- https://epirhandbook.com/en/diagrams-and-charts.html#alluvialsankey-diagrams
- https://rpubs.com/droach/CPP526-codethrough

Part IV

R: Codigo

4 Codigo no livro

- Objetivo não é de apresentar detalhes sobre os cálculos/métodos estatísticas ou os funções no R. Existem diversos exemplos disponíveis "Ciência de Dados com R–Introdução.....": e com google "r cran introdução tutorial"..... Alem disso, existem grupos de ajuda, como por exemplo: R Brasil e Stack Overflow em Português
- O objetivo é de apresentar um livro mostrado os capacidades e opções para desenvolver e integrar pesquisas na ecologia da paisagem no ambiente estatística de R

Obviamente, todas as tarefas de geoprocessamento podem ser desenvolvidas anteriormente em um SIG (QGIS). Então porque use R? R tem a capacidade (baseada em codigo) para alternar entre tarefas de processamento, modelagem e visualização de dados geográficos e não geográficos. Alem disso, como é possível importar, modificar, analisar e visualizar dados espaciais no mesmo ambiente com script/codigo, o R permite fluxos de trabalho transparentes e reproduzíveis (A Ciência Aberta).

Aliás, atualmente a grande maioria dos artigos científicos publicados na revista Landscape Ecology incluir análises usando R.

4.1 Organização do codigo no livro

O capitulo está organizado em etapas de processamento, com blocos de código em caixas cinzas:

```
codigo de R para executar
```

Para segue os passos, os blocos de código precisam ser executados em sequência. Se você pular uma etapa, ou rodar fora de sequência o próximo bloco de código provavelmente não funcionará.

As linhas de codigo de R dentro de cada caixa tambem preciso ser executados em sequência. O simbolo # é usado para incluir comentarios sobre os passos no codgio (ou seja, linhas começando com # não é codigo de executar).

```
# Passo 1
codigo de R passo 1 # texto e numeros tem cores diferentes
# Passo 2
codigo de R passo 2
# Passo 3
codigo de R passo 3
```

Alem disso, os simbolos #> e/ou [**1**] no início de uma linha indica o resultado que você verá no console de R.

```
# Passo 1
1+1
```

[1] 2

```
# Passo 2
x <- 1 + 1
# Passo 3
x
```

[1] 2

```
# Passo 4
x + 1
```

[1] 3

5 Primeiros passos com uma raster

Uma raster é um matriz de valores com coordenadas geográficos. Cada pixel de uma raster representa uma região geográfica, e o valor do pixel representa uma característica dessa região (mais sobre dados raster).

Em geral é necessário baixar alguns pacotes para que possamos fazer as nossas análises. Precisamos os seguintes pacotes, que deve esta instalado antes:

- tidyverse,
- terra,
- sf,
- mapview,
- tmap.

Carregar pacotes:

```
library(tidyverse)
library(terra)
library(sf)
library(mapview)
library(tmap)
```

Inicialmente iremos gerar uma raster representando uma paisagem bem simples, de 6 por 6 pixels. Você já deve saber que pixel é a unidade básica de uma imagem (lembra da camera do seu celular, 10Mb ou algo assim?!). Vocês devem ter visto sobre pixels e resolução no mesmo em aulas de geoprocessamento. Aqui podemos tratar o pixel como a resolução. Vamos dizer que temos um pixel de 10 metros (res=10 no bloco de código), ou seja, uma quadrado de 10 por 10m, sendo essa, a menor unidade mapeável. Assim sendo, a resolução tambem tem ligação com escala cartográfica!

Vamos gerar e plotar uma paisagem simples em 4 passos. Primeiramente, a função `rast` cria um objeto do tipo raster. E depois a função `values` atribui valores, e na sequencia vamos visualisar os valores com `plot` e `text` .

```
#Função "rast" gera a paisagem virtual (paisagem simulado)
pai_sim <- rast(ncols=6, nrows=6,
                xmin=1, xmax=60,
                ymin=1, ymax=60,
                res=10)
#E essa atribui valores ("values") para os pixels criados acima
values(pai_sim) <- 1
plot(pai_sim) #Essa plota
text(pai_sim) #Essa coloca os valores dos pixels
```

5.0.1 Obter e carregar dados (raster)

Mais uma vez vamos aproveitar os dados de MapBiomas. Agora baixar arquivo raster com cobertura de terra no entorno dos rios em 2020, (formato ".tif", tamanho 3.3 MB). Link: https://github.com/darrennorris/gisdata/blob/master/inst/raster/mapbiomas_AP_utm_rio/utm_cover_AP_rio_2020.tif . Lembrando-se de salvar o arquivo ("utm_cover_AP_rio_2020.tif") em um local conhecido no seu computador. Agora avisar R sobre onde ficar o arquivo. O código abaixo vai abrir uma nova janela, e você deve buscar e selecionar o arquivo "utm_cover_AP_rio_2020.tif":

```
meuSIGr <- file.choose()
```

O código abaixo vai carregar os dados e criar o objeto "mapbiomas_2020".

```
mapbiomas_2020 <- rast(meuSIGr)
```

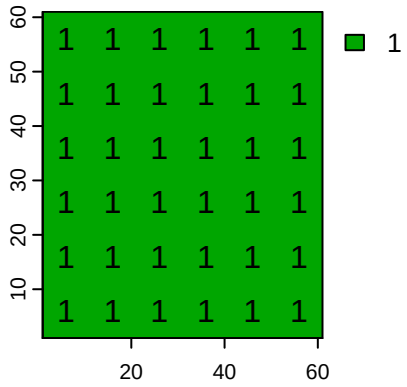


Figure 20: Paisagem simples de 36 pixels.

5.0.2 Reclassificação

Para simplificar nossa avaliação de escala, reclassificamos a camada mapbiomas_2020 em uma camada binária de floresta/não-floresta. Essa tarefa de geoprocessamento pode ser realizada anteriormente usando SIG (QGIS). Aqui vamos reclassificar as categorias de cobertura da terra (agrupando diferentes áreas de cobertura florestal tipos) usando alguns comandos genéricos do R para criar uma nova camada com a cobertura de floresta em toda a região de estudo. Para isso, criamos um mapa do mesmo resolução e extensão, e então podemos redefinir os valores do mapa. Neste caso, queremos agrupar a cobertura da terra categorias 3 e 4 (Formação Florestal e Formação Savânica, respectivamente).

```
# criar uma nova camada de floresta
floresta_2020 <- mapbiomas_2020
# Com valor de 0
values(floresta_2020) <- 0
# Atualizar categorias florestais agrupados com valor de 1
floresta_2020[mapbiomas_2020==3 | mapbiomas_2020==4] <- 1
```

Vizualizar para verificar.

```
# Passo necessario para agilizar o processamento
floresta_2020_modal<-aggregate(floresta_2020, fact=10, fun="modal")
# Plot
tm_shape(floresta_2020_modal) +
  tm_raster(style = "cat",
            palette = c("0" = "#E974ED", "1" ="#129912"),
            legend.show = FALSE) +
  tm_add_legend(type = "fill", labels = c("não-floresta", "floresta"),
    col = c("#E974ED", "#129912"), title = "Classe") +
tm_scale_bar(breaks = c(0, 25, 50), text.size = 1,
             text.color = "white", position=c("left", "bottom")) +
tm_layout(legend.position = c("right","top"),legend.bg.color = "white")
```

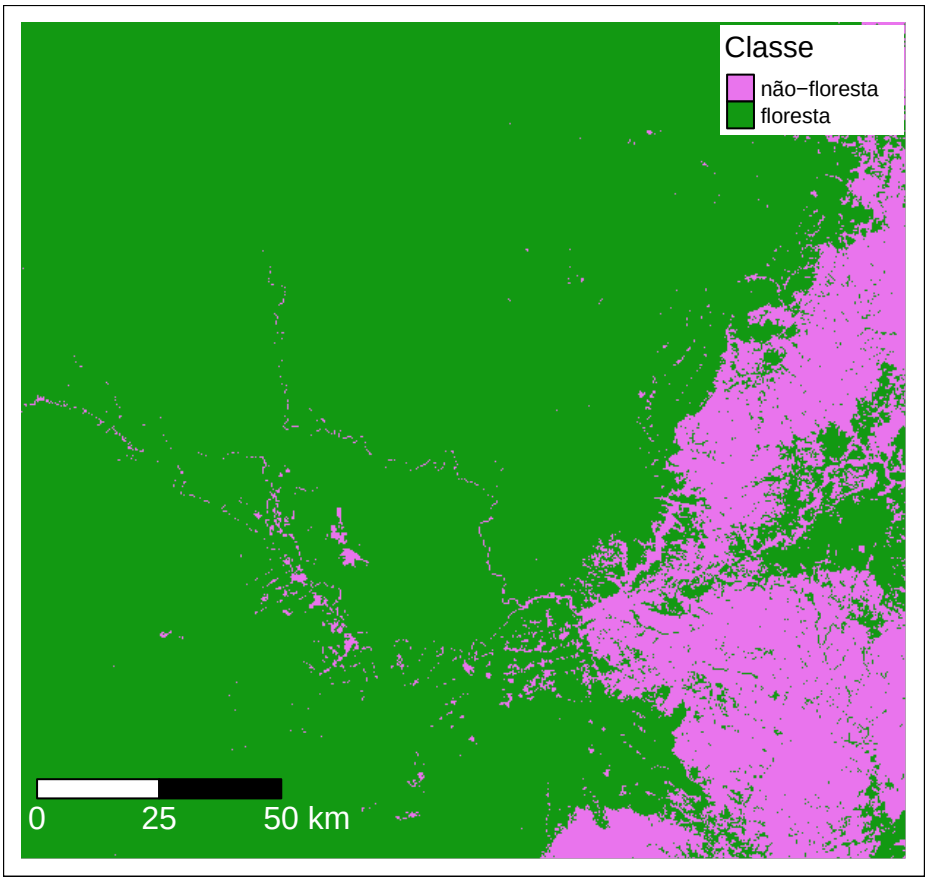


Figure 21: Floresta ao redor do Rio Araguari. MapBiomas 2020 reclassificado em floresta e não-floresta. Mostrando os pontos de amostragem (pontos amarelas) cada 5 quilômetros ao longo do rio.

6 Primeiros passos com vector

Em geral é necessário baixar alguns pacotes para que possamos fazer as nossas análises. Precisamos os seguintes pacotes, que deve esta instalado antes:

- tidyverse,
- sf,
- mapview,
- tmap.

Carregar pacotes:

```
library(tidyverse)
library(sf)
library(mapview)
library(tmap)
```

6.1 Obter e carregar dados (vectores)

Precisamos carregar os dados para rios e pontos de amostragem. Baixar arquivo (vector) com os dados (formato "GPKG", tamanho 54.9 MB). Mais sobre dados vetoriais. O formato aberto GeoPackage é um contêiner que permite armazenar dados SIG (feições/camadas) em um único arquivo. Por exemplo, um arquivo GeoPackage pode conter vários dados (dados vetoriais e raster) em diferentes sistemas de coordenadas. Todos esses recursos permitem que você compartilhe dados facilmente e evite a duplicação de arquivos.

Baixar o arquivo Link: https://github.com/darrennorris/gisdata/blob/master/inst/vector/rivers.gpkg . Lembrando-se de salvar o arquivo ("rivers.gpkg") em um local conhecido no seu computador.

O formato "GPKG" e diferente de "tif" (raster), o processo de importação é, portanto, diferente. Primeira, avisar R sobre onde ficar o arquivo. O código abaixo vai abrir uma nova janela, e você deve buscar e selecionar o arquivo "rivers.GPKG":

```
meuSIG <- file.choose()
```

Agora vamos olhar o que tem no arquivo. Depois que voces rodar o codigo **st_layers(meuSIG)** , o resultado mostra que o arquivo rivers.GPKG inclui camadas diferentes com pontos ("Point"), linhas ("Line String") e polígonos ("Polygon"). Alem disso, a coluna "crs_name" mostrar que a sistema de coordenadas é geográfica (WGS84, (EPSG: 4326)

$$https://epsg.io/4326$$

, e é diferente do arquivo raster:

```
sf::st_layers(meuSIG)
```

```
## Driver: GPKG
## Available layers:
##                    layer_name geometry_type features fields crs_name
## 1                  centerline   Line String       52     15   WGS 84
## 2                  forestloss         Point   276086     12   WGS 84
## 3                   canalpoly       Polygon        3      6   WGS 84
## 4               extentpoly50km       Polygon        1      0   WGS 84
## 5                   midpoints         Point       52     17   WGS 84
## 6            midpoints_hansen         Point       52     37   WGS 84
## 7         cachoeira_caldeirao         Point        1      2   WGS 84
## 8                porto_grande         Point        1      1   WGS 84
## 9                 icmbio_base         Point        1      1   WGS 84
## 10              direct_affect       Polygon        1      2   WGS 84
## 11 midpoints_hansen_distances         Point       52     43   WGS 84
```

```
## 12       midpoints_hansen_ffr         Point       52     82   WGS 84
## 13     midpoints_hansen_ffril         Point       52     91   WGS 84
## 14         direct_affect_line   Line String        1      2   WGS 84
```

Nós só precisamos de duas dessas camadas. O código abaixo vai carregar as camadas que precisamos e criar os objetos "rsm" e "rsl". Assim, agora temos dados com: pontos cada 5 km ao longo os rios ("rsm") e a linha central de rios ("rsl").

```
# pontos cada 5 km
rsm <- sf::st_read(meuSIG, layer = "midpoints")
```

```
## Reading layer `midpoints' from data source
##   `C:\Users\user\Documents\Articles\gis_layers\gisdata\inst\vector\rivers.gpkg'
##   using driver `GPKG'
## Simple feature collection with 52 features and 17 fields
## Geometry type: POINT
## Dimension:     XY
## Bounding box:  xmin: -52.01259 ymin: 0.7175827 xmax: -51.29688 ymax: 1.330365
## Geodetic CRS:  WGS 84
```

```
# linha central de rios
rsl <- sf::st_read(meuSIG, layer = "centerline")
```

```
## Reading layer `centerline' from data source
##   `C:\Users\user\Documents\Articles\gis_layers\gisdata\inst\vector\rivers.gpkg'
##   using driver `GPKG'
## Simple feature collection with 52 features and 15 fields
## Geometry type: LINESTRING
## Dimension:     XY
## Bounding box:  xmin: -52.01443 ymin: 0.7094595 xmax: -51.2924 ymax: 1.352094
## Geodetic CRS:  WGS 84
```

6.2 Visualizar os arquivos (camadas vector)

Visualizar para verificar. Mapa com linha central e pontos de rios em trechos de 5km.

```
ggplot(rsl) +
  geom_sf(aes(color=rio)) +
  geom_sf(data = rsm, shape=21, aes(fill=zone))
```

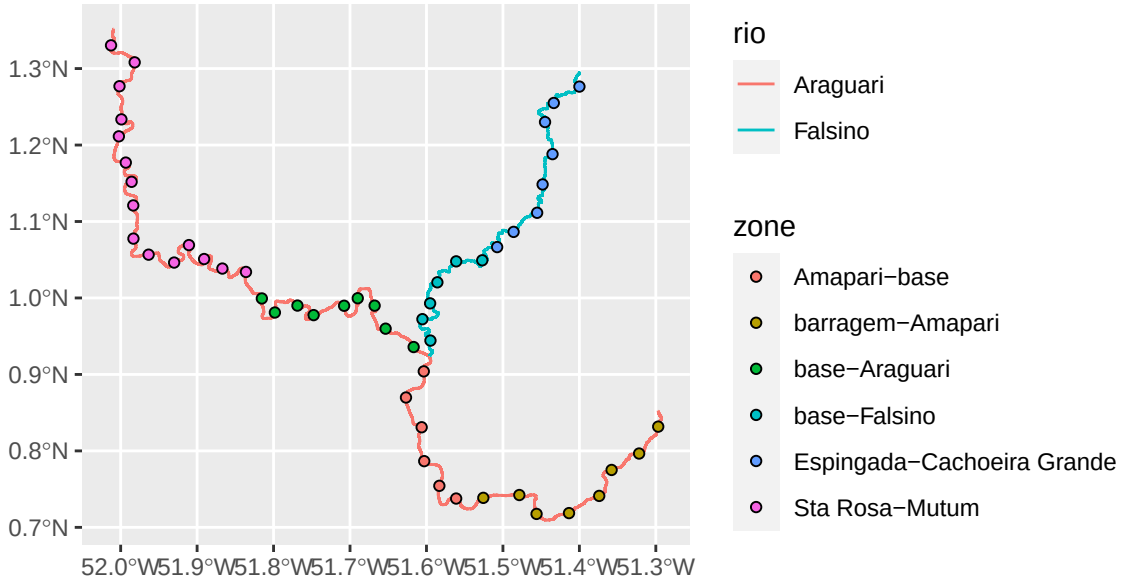


Figure 22: Pontos ao longo dos rios a montante das hidrelétricas no Rio Araguari.

Mapa interativo (funcione somente com internet) Mostrando agora com fundo de mapas "base" (OpenStreetMap/ESRI etc)

```
#
mapview(rsl, zcol = "rio")
```

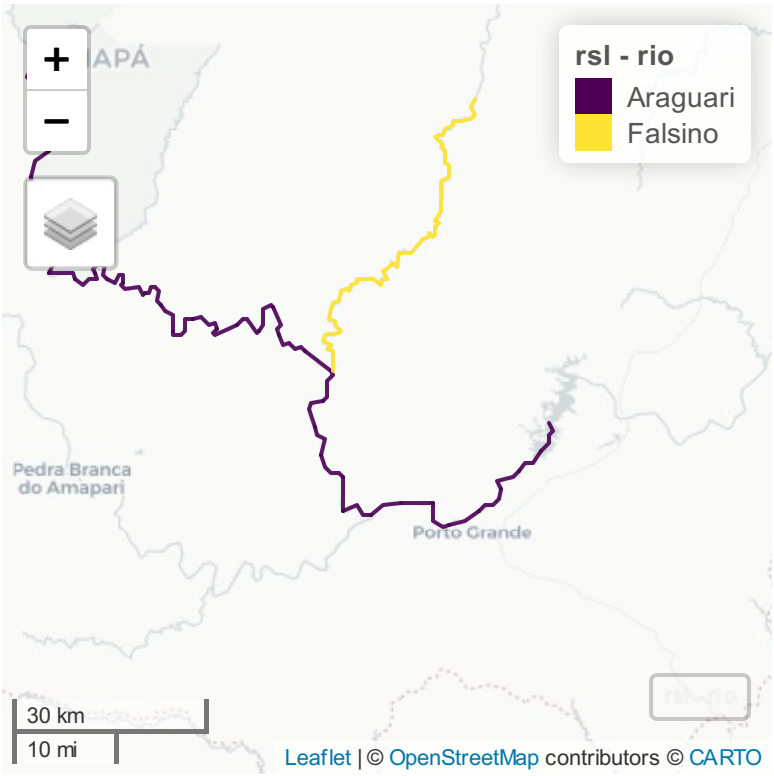


Figure 23: Linhas dos rios a montante das hidrelétricas no Rio Araguari.

Bibliografia

Bárcenas-García, Andrea, Fernanda Michalski, James P. Gibbs, and Darren Norris. 2022. "Amazonian Run-of-River Dam Reservoir Impacts Underestimated: Evidence from a Before–after Control–impact Study of Freshwater Turtle Nesting Areas." *Aquatic Conservation: Marine and Freshwater Ecosystems* 32 (3): 508–22. https://doi.org/10.1002/aqc.3775.

Fletcher, Robert, and Marie-Josée Fortin. 2018. *Spatial Ecology and Conservation Modeling*. Springer International Publishing. https://doi.org/10.1007/978-3-030-01989-1.

Raffo, Deborah C. Dávila, Darren Norris, Sandra Maria Hartz, and Fernanda Michalski. 2022. "Anthropogenic Influences on the Distribution of a Threatened Apex-Predator Around Sustainable-Use Reserves Following Hydropower Dam Installation." *PeerJ* 10 (October): e14287. https://doi.org/10.7717/peerj.14287.

A Soluções de exercícios