

Escala

Darren Norris

2022-09-22

Sumário

1	Apresentação	2
2	Escala: breve definação	2
3	Codigo e R	2
3.1	Organização do codigo no tutorial	4
4	Primeiros passos com uma raster	5
4.0.1	Pergunta 1	7
5	Alterando a resolução	10
5.0.1	Pergunta 2	12
6	Escala espacial e desenho amostral	13
6.1	Obter e carregar dados (vectores)	13
6.2	Visualizar os arquivos (camadas vector)	15
6.3	Obter e carregar dados (raster)	16
6.4	Visualizar os arquivos (camadas raster e vector)	16
6.5	Reclassificação	18
7	Comparação multiescala	20
7.0.1	Pergunta 3	21
7.0.2	Pergunta 4	23
7.0.3	Pergunta 5	23
8	Próximos passos: repetindo para muitas amostras.	24

1 Apresentação

Nesta aula (<https://rpubs.com/darren75/escala>) vamos entender a importância de escala na ecologia da paisagem através cálculos com a proporção de floresta. Durante a aula 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 conteúdo baseado no Capítulo 2 do livro [Fletcher & Fortin. 2019. Spatial Ecology and Conservation Modeling: Applications with R](#) e “Tutorial Escala” do Dr. Alexandre Martensen.

2 Escala: breve definição

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 diminuiram e a disponibilidade de dados de alta resolução aumentou, a covariância entre resolução e extensão nas investigações diminuiu.

3 Código e R

- 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 “[cran introdução tutorial](#)”..... Além disso, existem grupos de ajuda, como por exemplo: [R Brasil](#) e [Stack Overflow em Português](#)
- O objetivo é de apresentar um tutorial mostrando as 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 código) para alternar entre tarefas de processamento, modelagem e visualização de dados geográficos e não geográficos. Além disso, como é possível

importar, modificar, analisar e visualizar dados espaciais no mesmo ambiente com script/código, 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](#) inclui análises usando R.

3.1 Organização do código no tutorial

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

```
codigo de R para executar
```

Para seguir 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 código de R dentro de cada caixa também precisam ser executados em sequência. O símbolo `#` é usado para incluir comentários sobre os passos no código (ou seja, linhas começando com `#` não é código 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
```

Além disso, os símbolos `#>` 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
```

4 Primeiros passos com uma raster

Uma raster é um matriz de valores com coordenadas geográficas. 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 estar 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 (também tem ligação com escala cartográfica!).

```
#Essa linha de comando gera a paisagem virtual (simulação)
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
```

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> .

O arquivo raster tem uma pequena amostra com a classificação da terra feito pela [MapBiomias](#), 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 várias funções para a análise e modelagem de dados geográficos. Nós podemos ler os dados de cobertura da terra com a função `rast`.

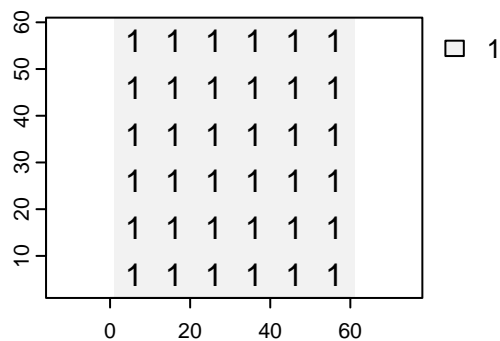


Figura 4.1: Paisagem simples de 36 pixels.

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

Plotar para verificar.

```
plot(ramostra)
```

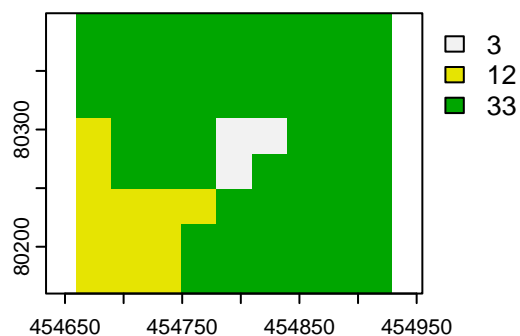


Figura 4.2: Mapbiomas 2020. Uma pequena amostra do Rio Araguari, perto de Porto Grande.

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

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

4.0.1 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 tamanho total da raster *ramostra* em hectares e quilômetros quadrados?

Vamos olhar o mapa de novo.

```
plot(ramostra)
```

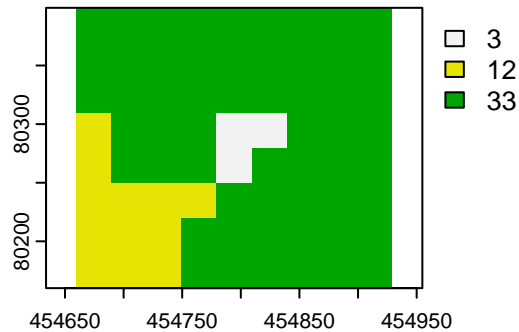


Figura 4.3: Mapbiomas 2020. Uma pequena amostra do Rio Araguari, perto de Porto Grande.

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” representam cobertura de “Formação Florestal”, “Formação Campestre”, e “Rio, Lago e Oceano”. Então podemos fazer um mapa com mais informações.

Daqui pra frente vamos aproveitar uma forma mais elegante de apresentar mapas e gráficos. Isso seria através 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>

Capítulo 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))
```

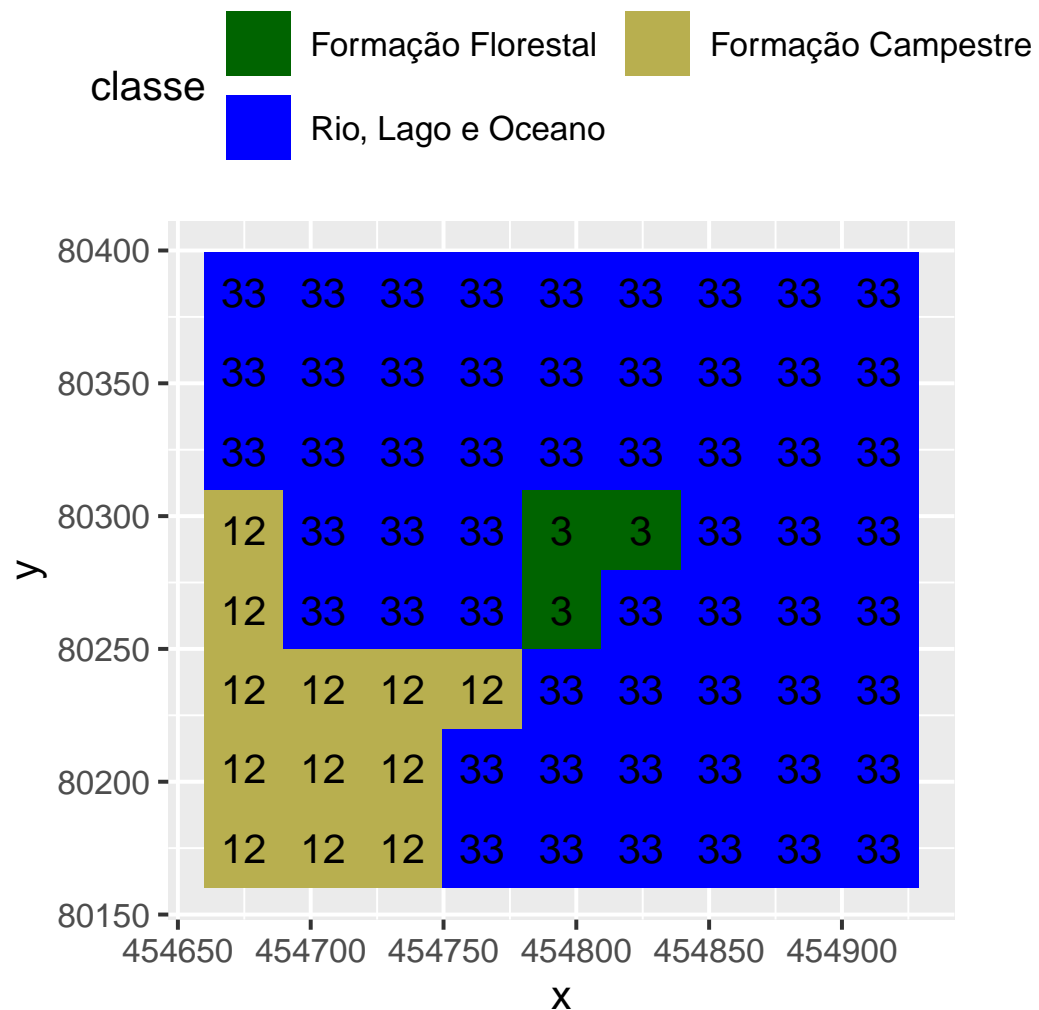


Figura 4.4: Paisagem com valores e classes de cobertura da terra. Mapbiomas 2020. Uma pequena amostra do Rio Araguari, perto de Porto Grande.

5 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))
```

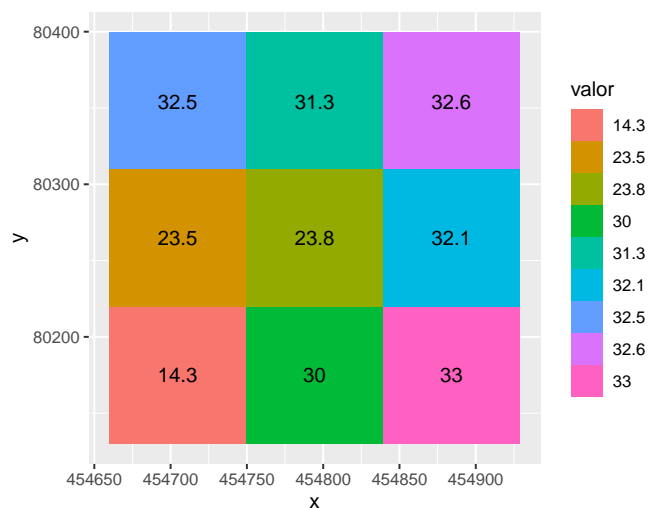


Figura 5.1: Agregação errado para dados categóricos. Uso da média cria valores categóricos errados e impossíveis.

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))
```

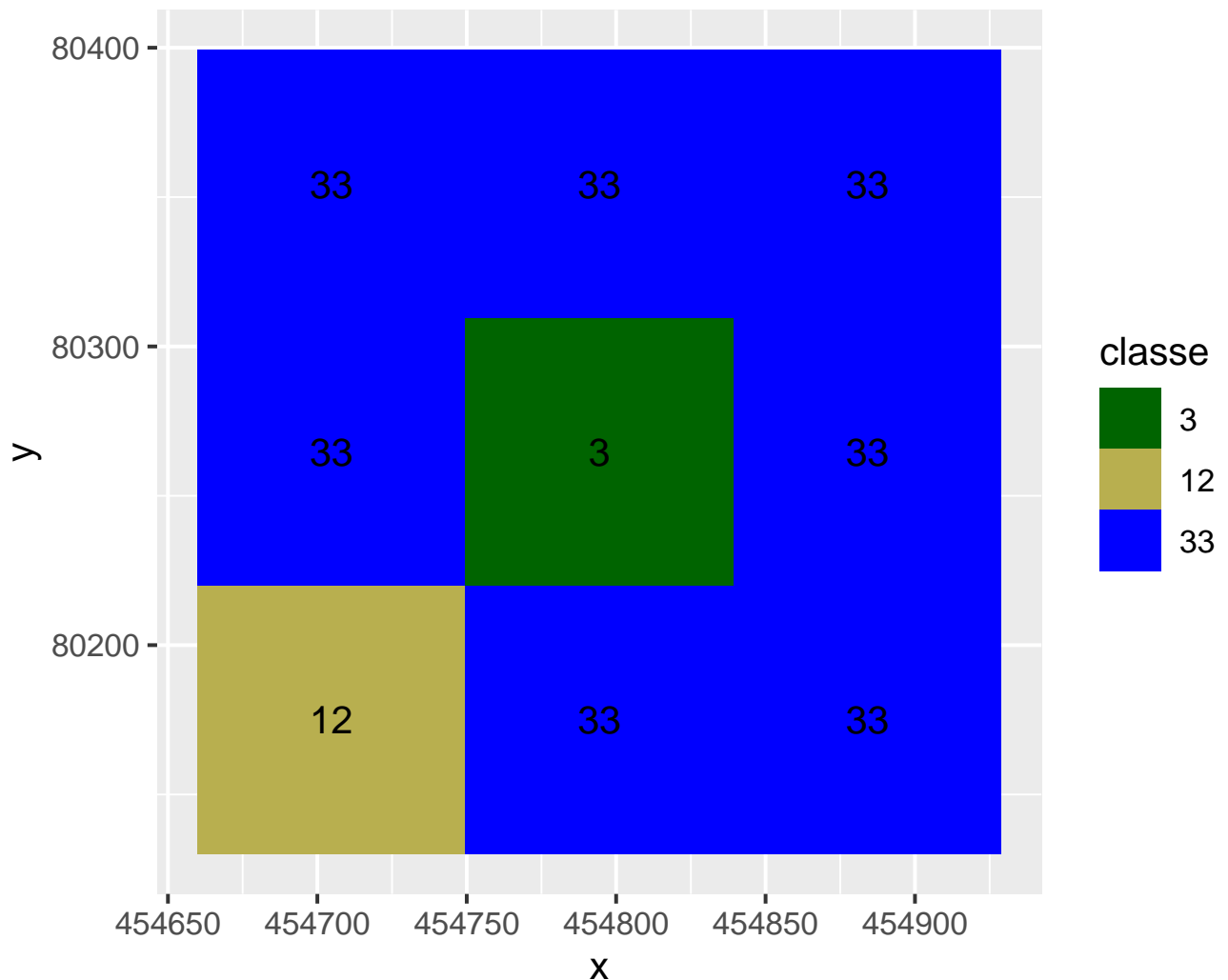


Figura 5.2: Agregação pela mais frequente.

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 pequena paisagem quando juntamos 3 pixels em um único (proporção mudou de 3/72 para 1/9). Ou seja, com cada passo mudamos a representação do mundo.

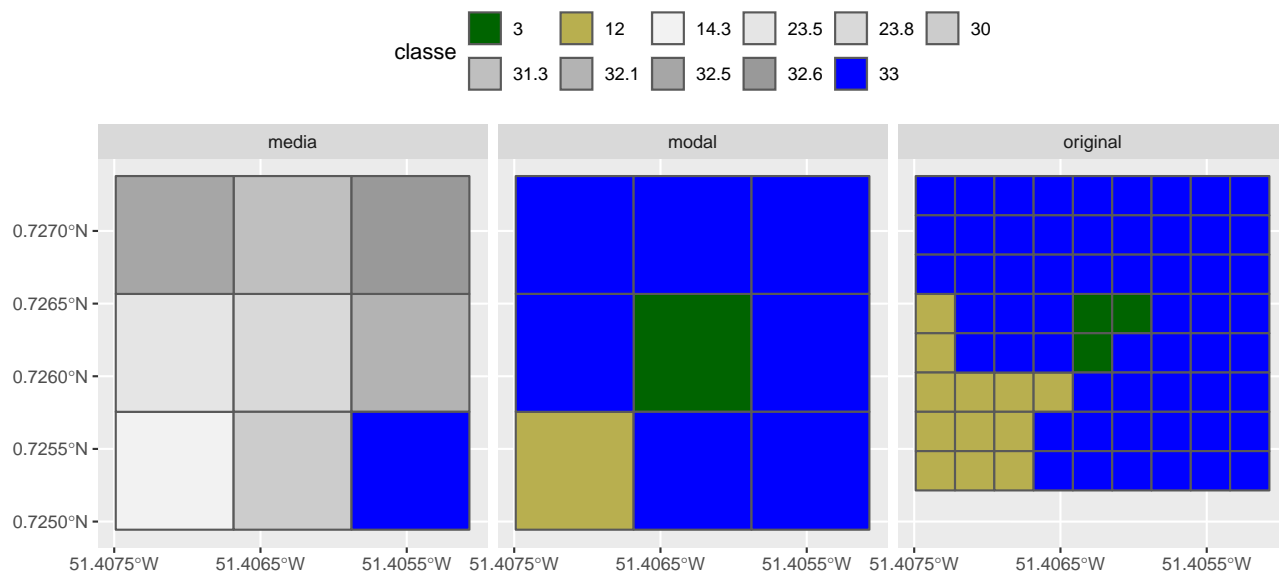


Figura 5.3: Mudanças causadas pela agregação.

5.0.1 Pergunta 2

Confira o código e os resultados obtidos anteriormente, quando mudamos a resolução da raster amostra (por exemplo figuras 5.1, 5.2 e 5.3). Explique o que aconteceu. Como e porque mudou os valores em cada caso (média e modal)?

6 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 fornecido algumas orientações importantes. As questões-chave incluem o tamanho da unidade de amostragem (resolução), o tipo de unidade de amostra e localizações da unidade de amostra, incluindo o espaçamento 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 técnica 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 sítios 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 vinculamos esses dados de cobertura da terra aos pontos de amostragem em rios.

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 54.8 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.

Agora 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. Existem camadas diferentes com pontos, linhas e polígonos:

```
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
```

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))
```

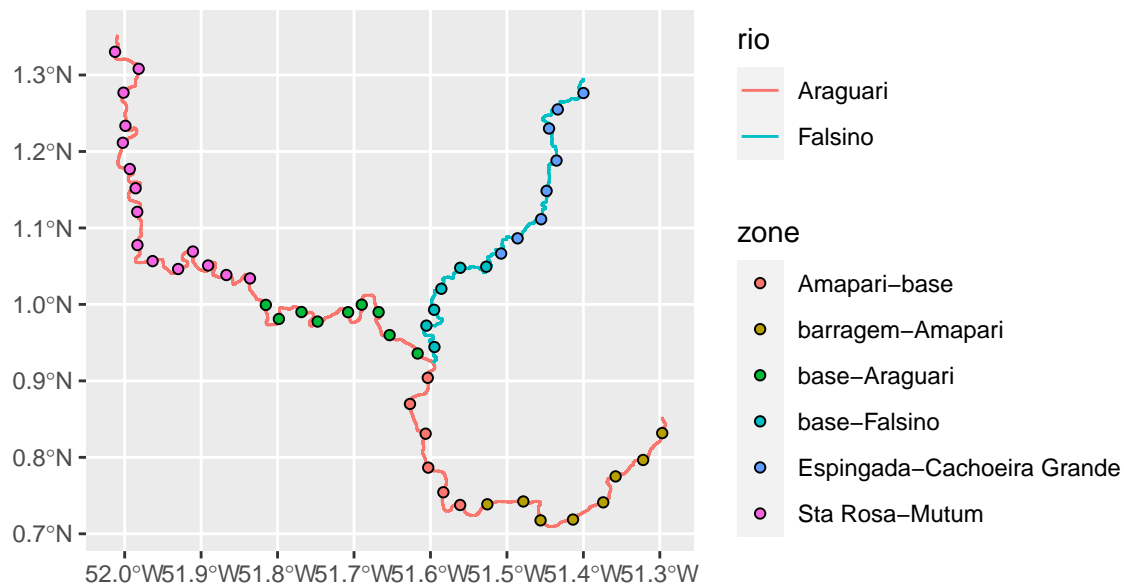


Figura 6.1: Pontos ao longo dos rios a montante das hidrelétricas no Rio Araguari.

Mapa interativo (funcione somente com internet) Mostrando agora com fundo de mapas “base” (Open-StreetMap/ESRI etc)

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

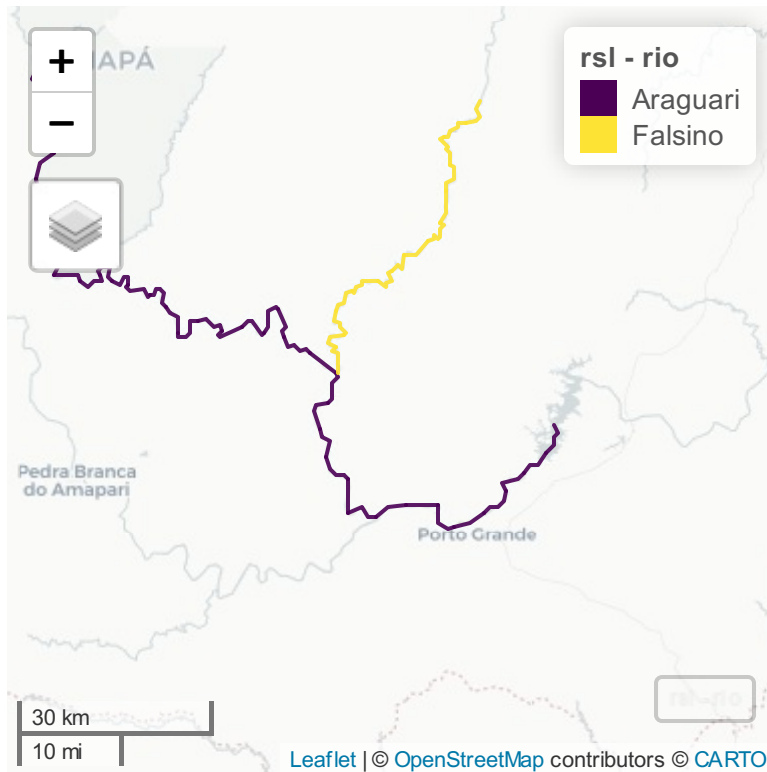


Figura 6.2: Linhas dos rios a montante das hidrelétricas no Rio Araguari.

6.3 Obter e carregar dados (raster)

Mais uma vez vamos aproveitar os dados de MapBiomias. 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)
```

6.4 Visualizar os arquivos (camadas raster e vector)

Visualizar para verificar. É possível de visualizar camadas de raster e vetor juntos com funções 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")

```

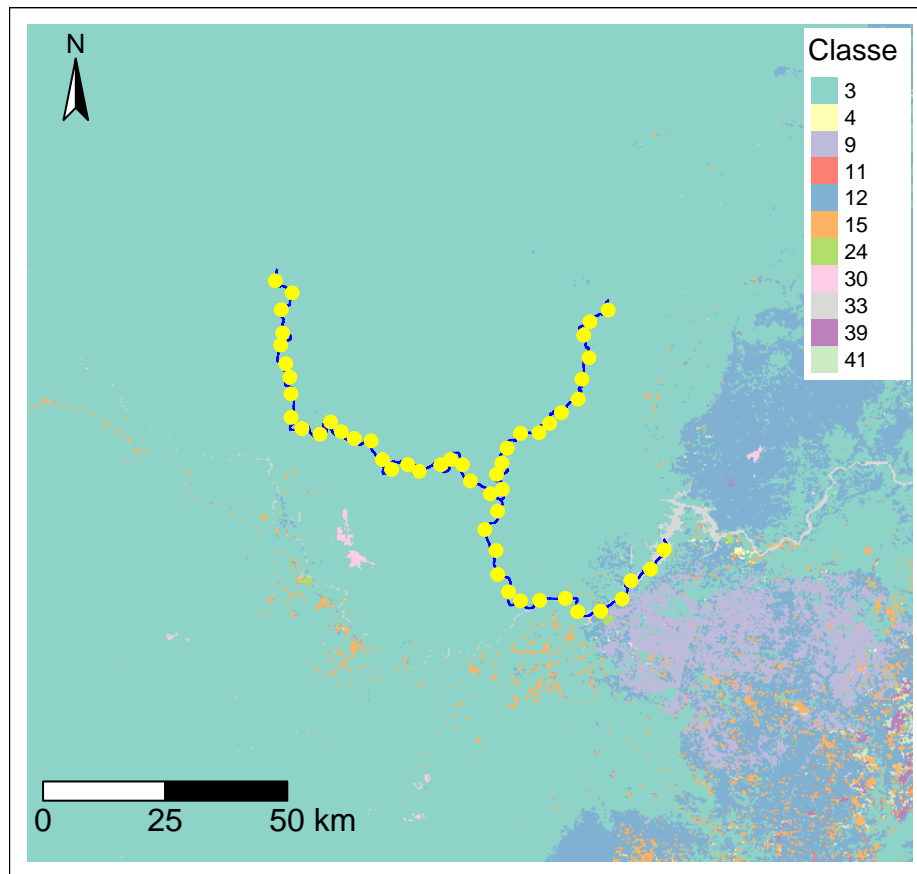


Figura 6.3: Cobertura da terra ao redor do Rio Araguari em 2020. Mostrando os pontos de amostragem (pontos amarelas) cada 5 quilômetros ao longo do rio.

6.5 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")
```

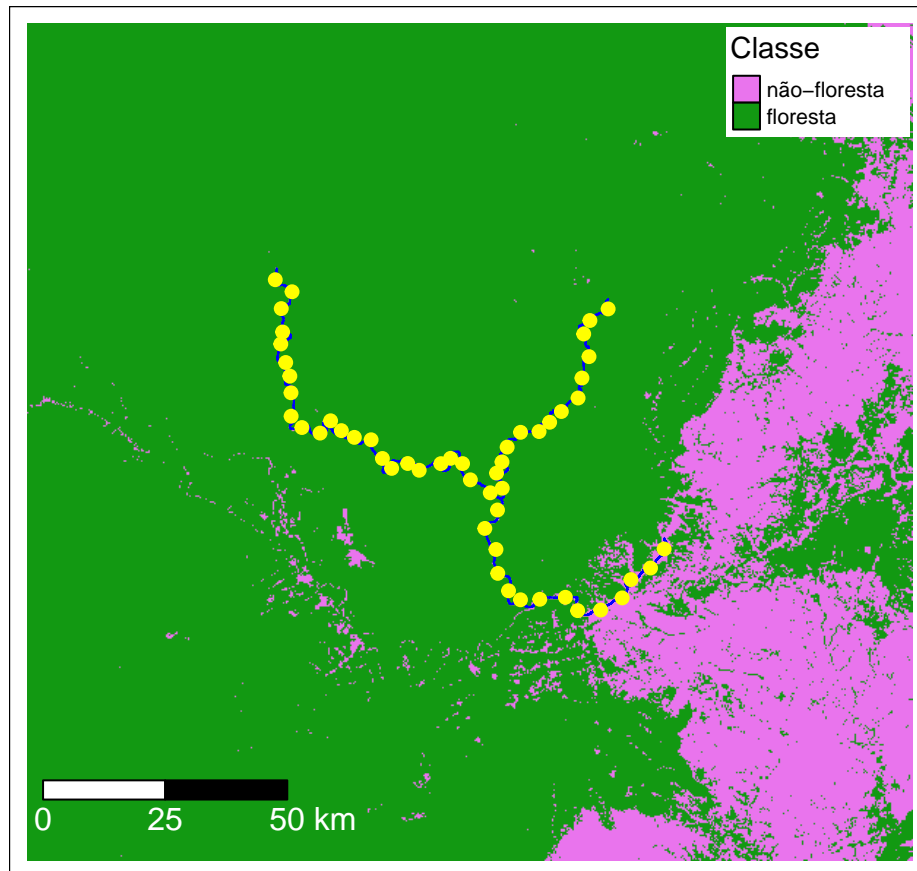


Figura 6.4: Floresta ao redor do Rio Araguari. MapBiomass 2020 reclassificado em floresta e não-floresta. Mostrando os pontos de amostragem (pontos amarelos) cada 5 quilômetros ao longo do rio.

7 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")
```

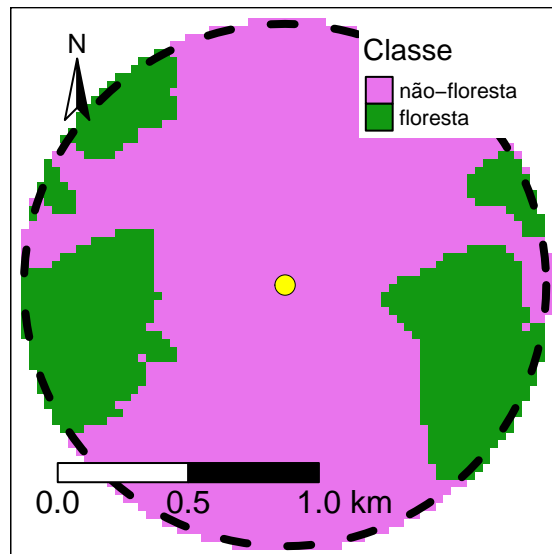


Figura 7.1: Ilustração da determinação da quantidade de habitat ao redor de um ponto. Para uma determinada extensão, o habitat de interesse é isolado. Um buffer (linha tracejada) é colocado ao redor de um ponto (amarelo) e o número de células (pixels) que contém o habitat é somado e multiplicado pela área de cada pixel.

7.0.1 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 área de floresta podemos somar o número de células (pixels) que contém o habitat e multiplicar pela área de cada pixel conforme o código:

```

# 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".
soma_floresta <- global(buffer.forest1.1km, "sum", na.rm = TRUE)
soma_floresta
#>               sum
#> forest_2020_1km 943

# 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          : memory
#> name            : forest_2020_1km
#> min value       : 0
#> max value       : 1

# Portanto, o tamanho de cada pixel é igual.
area_pixel_m2 <- 29.89281 * 29.89281
area_pixel_m2
#> [1] 893.5801

# 3) Calculos de área.
# Área de floresta m2
area_floresta_m2 <- soma_floresta * area_pixel_m2
area_floresta_m2
#>               sum
#> forest_2020_1km 842646
# Área 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, dobrando a escala (extensão) em cada passo.

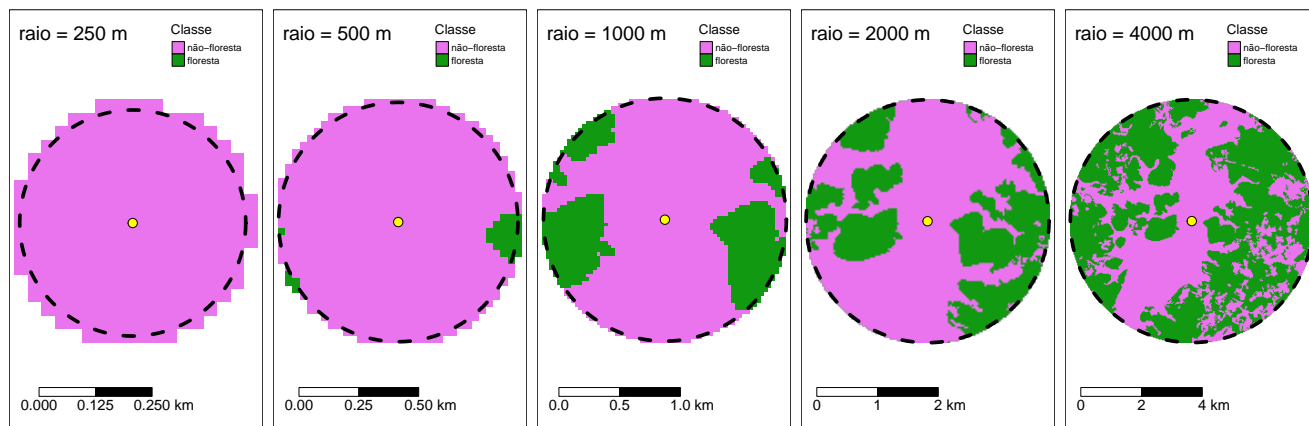


Figura 7.2: Cobertura florestal em extensões diferentes ao redor de um local de amostragem.

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

7.0.2 Pergunta 4

Usando os valores listados 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.

7.0.3 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.

8 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/>).