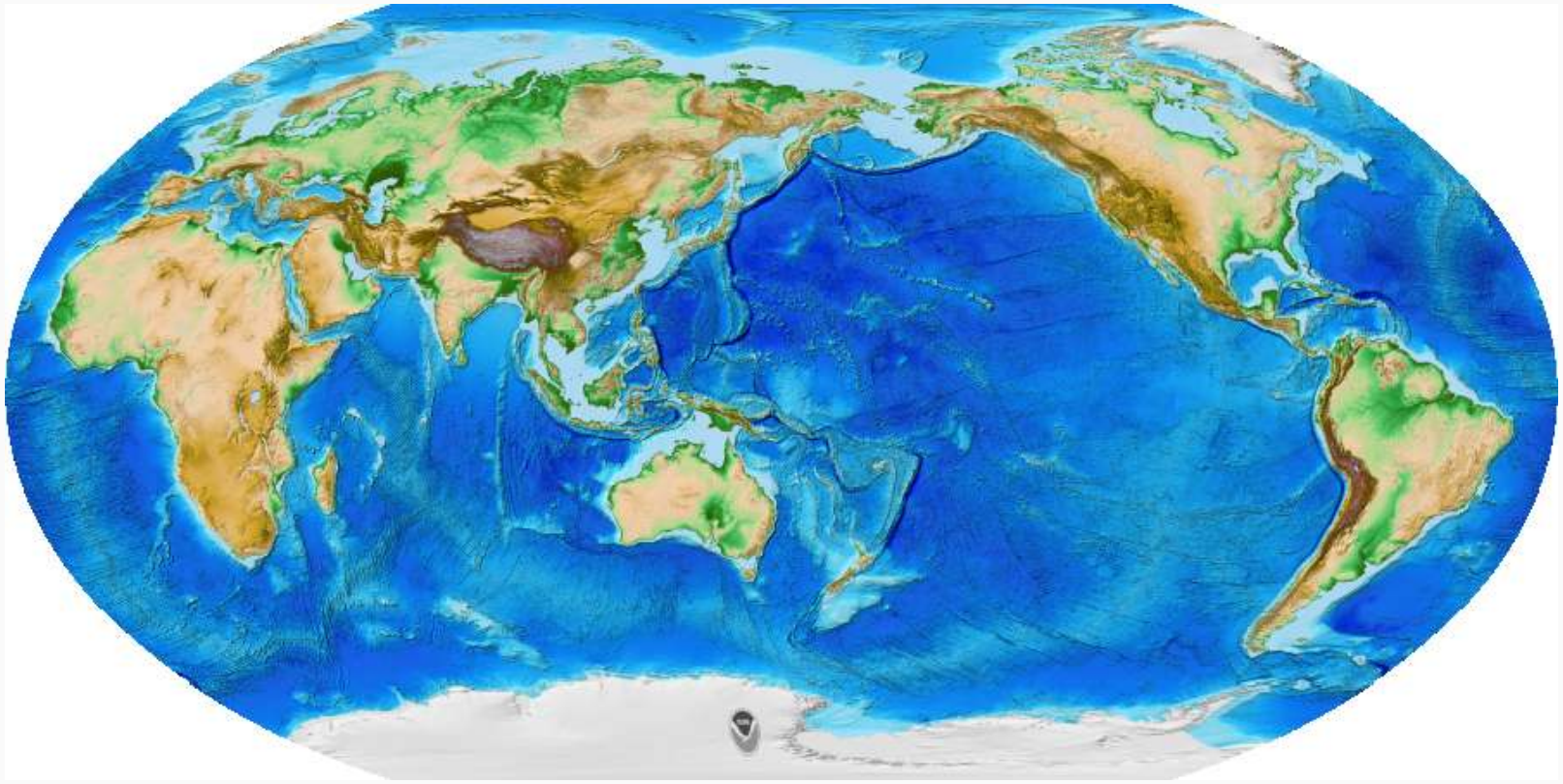
Introdução ao uso de dados geoespaciais no R

8 Estrutura e manipulação de dados matriciais

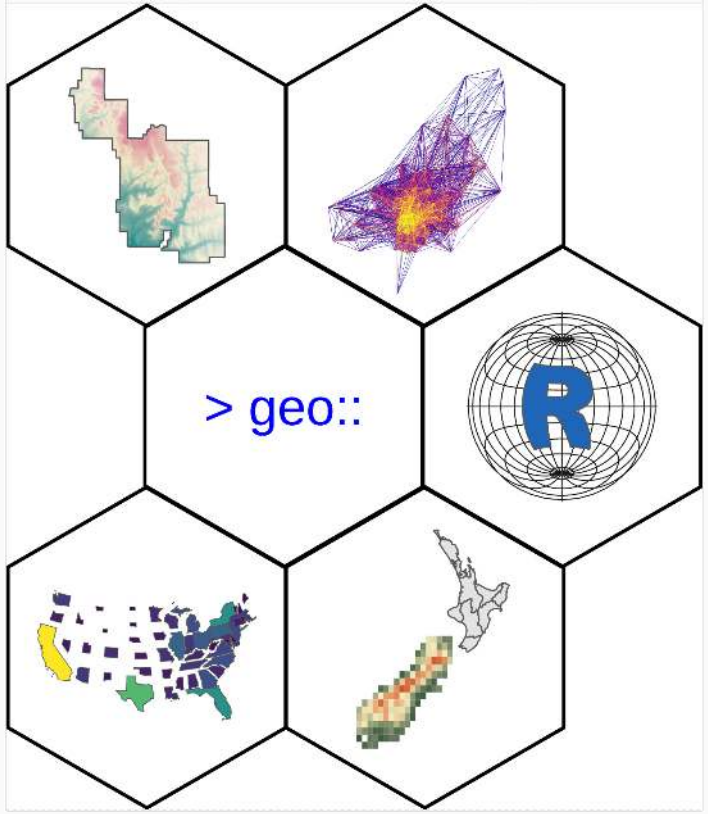
Maurício H. Vancine
Milton C. Ribeiro
UNESP - Rio Claro
Laboratório de Ecologia Espacial e Conservação (LEEC)
25/10/2021-05/11/2021



8 Estrutura e manejo de dados raster

Tópicos

1. Principais pacotes
2. Classes raster
3. Importar dados matriciais
4. Descrição de objetos raster
5. Converter CRS de objetos raster
6. Operações de objetos raster
7. Interações raster-vetor
8. Conversões raster-vetor
9. Índices espectrais
10. Exportar objetos raster



8 Estrutura e manejo de dados raster

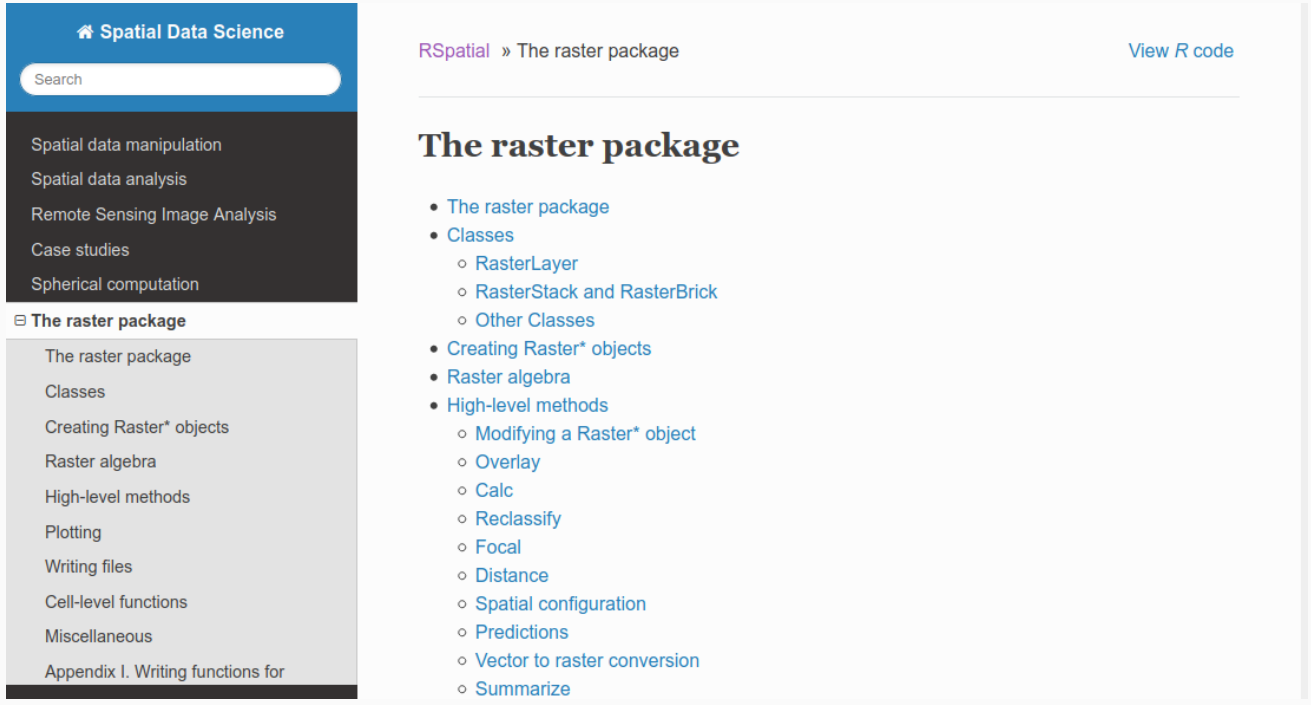
Script

`08_script_intro_geoespacial_r.R`

1. Principais pacotes

Pacote raster

```
# raster
install.packages("raster")
library(raster)
```

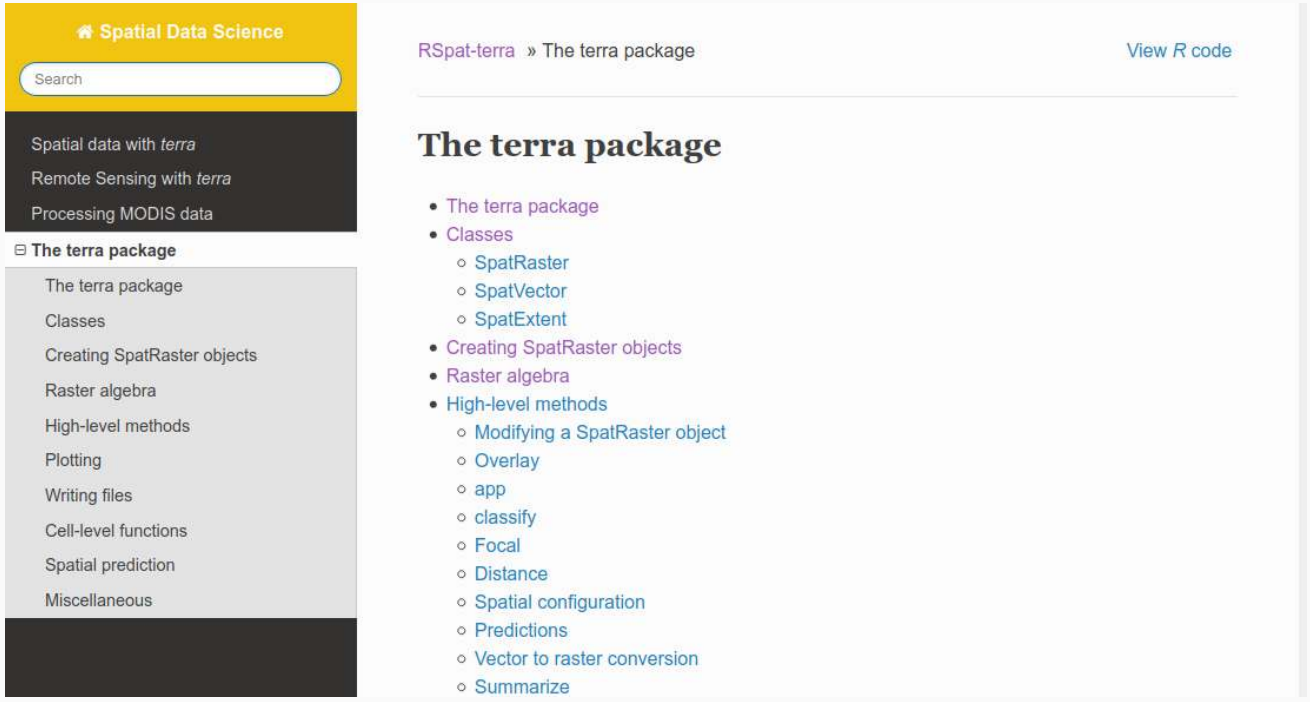


The raster package

1. Principais pacotes

Pacote terra

```
# raster
install.packages("terra")
library(terra)
```

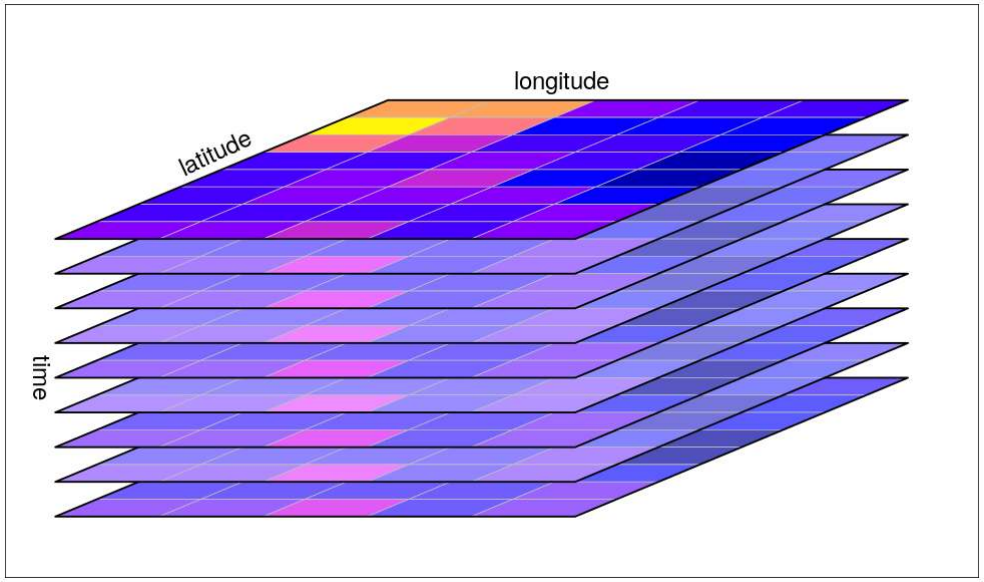


The terra package

1. Principais pacotes

Pacote stars

```
# raster
install.packages("stars")
library(stars)
```

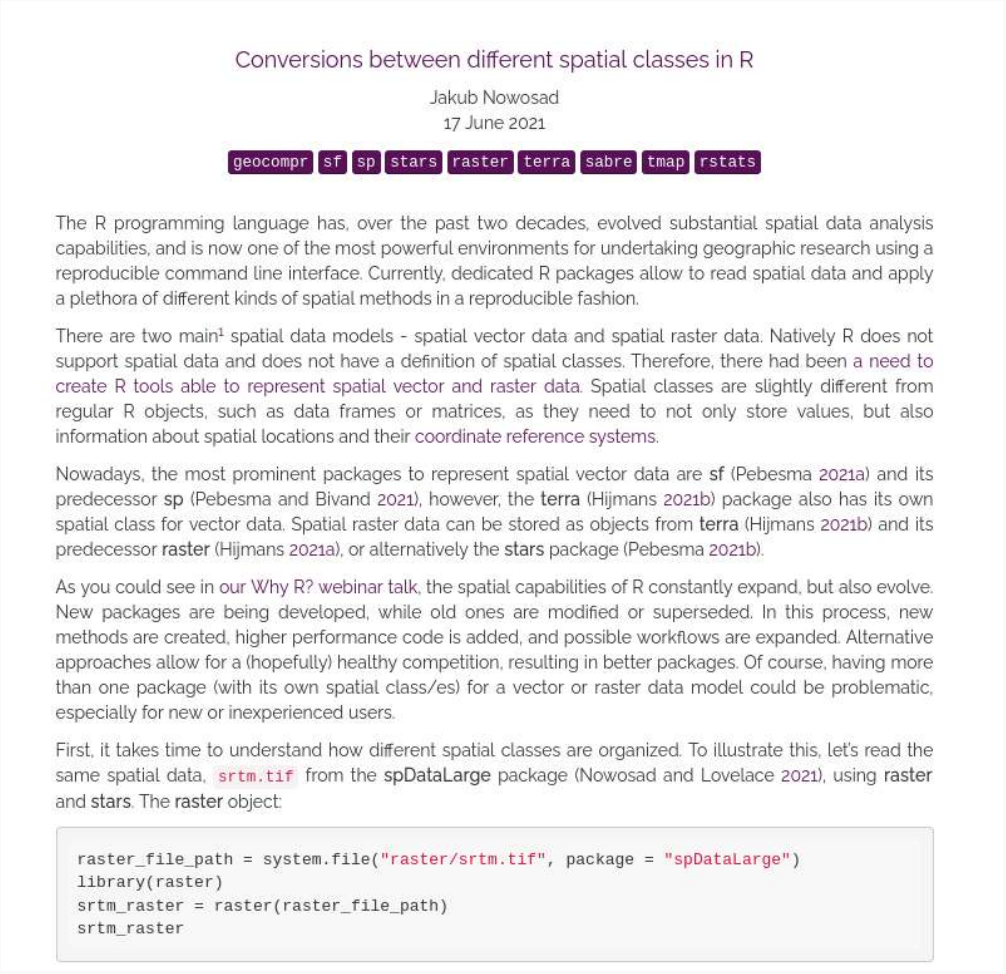


stars

Qual usar?

1. Principais pacotes

Conversões



Conversions between different spatial classes in R

Jakub Nowosad
17 June 2021

`geocompr` `sf` `sp` `stars` `raster` `terra` `sabre` `tmap` `rstats`

The R programming language has, over the past two decades, evolved substantial spatial data analysis capabilities, and is now one of the most powerful environments for undertaking geographic research using a reproducible command line interface. Currently, dedicated R packages allow to read spatial data and apply a plethora of different kinds of spatial methods in a reproducible fashion.

There are two main[1] spatial data models - spatial vector data and spatial raster data. Natively R does not support spatial data and does not have a definition of spatial classes. Therefore, there had been a need to create R tools able to represent spatial vector and raster data. Spatial classes are slightly different from regular R objects, such as data frames or matrices, as they need to not only store values, but also information about spatial locations and their coordinate reference systems.

Nowadays, the most prominent packages to represent spatial vector data are **sf** (Pebesma 2021a) and its predecessor **sp** (Pebesma and Bivand 2021), however, the **terra** (Hijmans 2021b) package also has its own spatial class for vector data. Spatial raster data can be stored as objects from **terra** (Hijmans 2021b) and its predecessor **raster** (Hijmans 2021a), or alternatively the **stars** package (Pebesma 2021b).

As you could see in our Why R? webinar talk, the spatial capabilities of R constantly expand, but also evolve. New packages are being developed, while old ones are modified or superseded. In this process, new methods are created, higher performance code is added, and possible workflows are expanded. Alternative approaches allow for a (hopefully) healthy competition, resulting in better packages. Of course, having more than one package (with its own spatial class/es) for a vector or raster data model could be problematic, especially for new or inexperienced users.

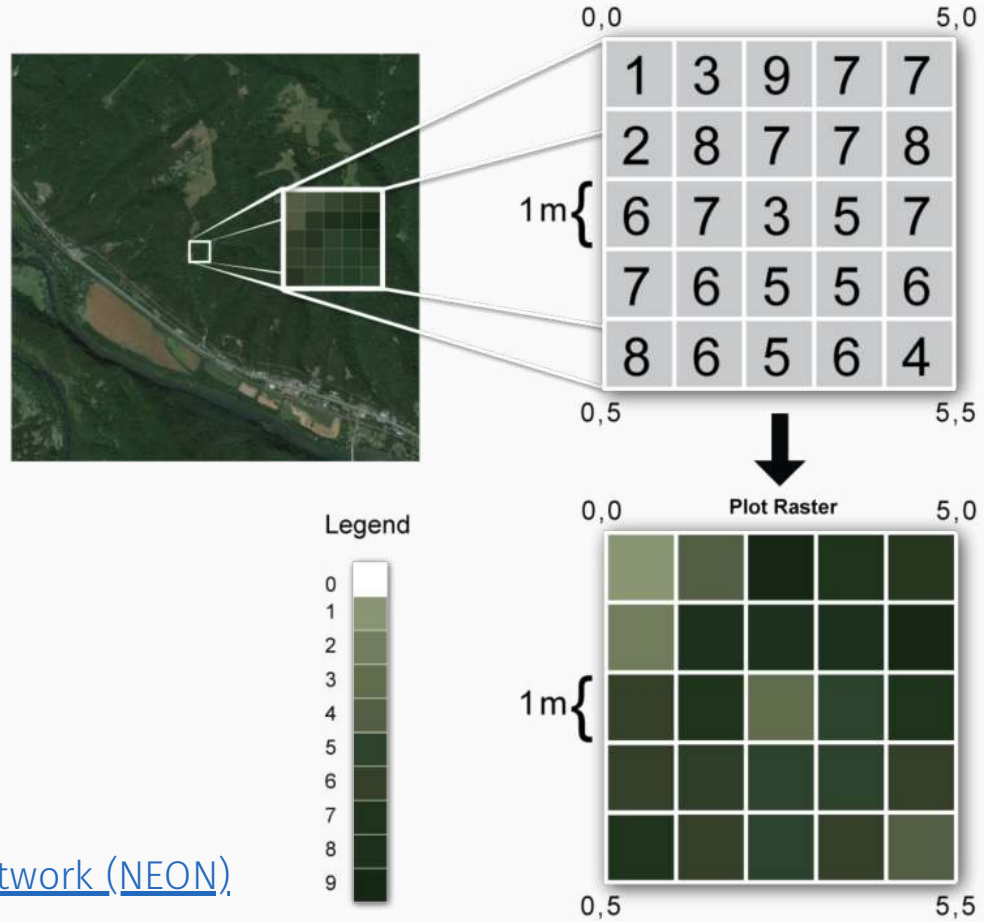
First, it takes time to understand how different spatial classes are organized. To illustrate this, let's read the same spatial data, `srtm.tif` from the **spDataLarge** package (Nowosad and Lovelace 2021), using **raster** and **stars**. The **raster** object:

```
raster_file_path = system.file("raster/srtm.tif", package = "spDataLarge")
library(raster)
srtm_raster = raster(raster_file_path)
srtm_raster
```

Nowosad (2021)

2. Classes raster

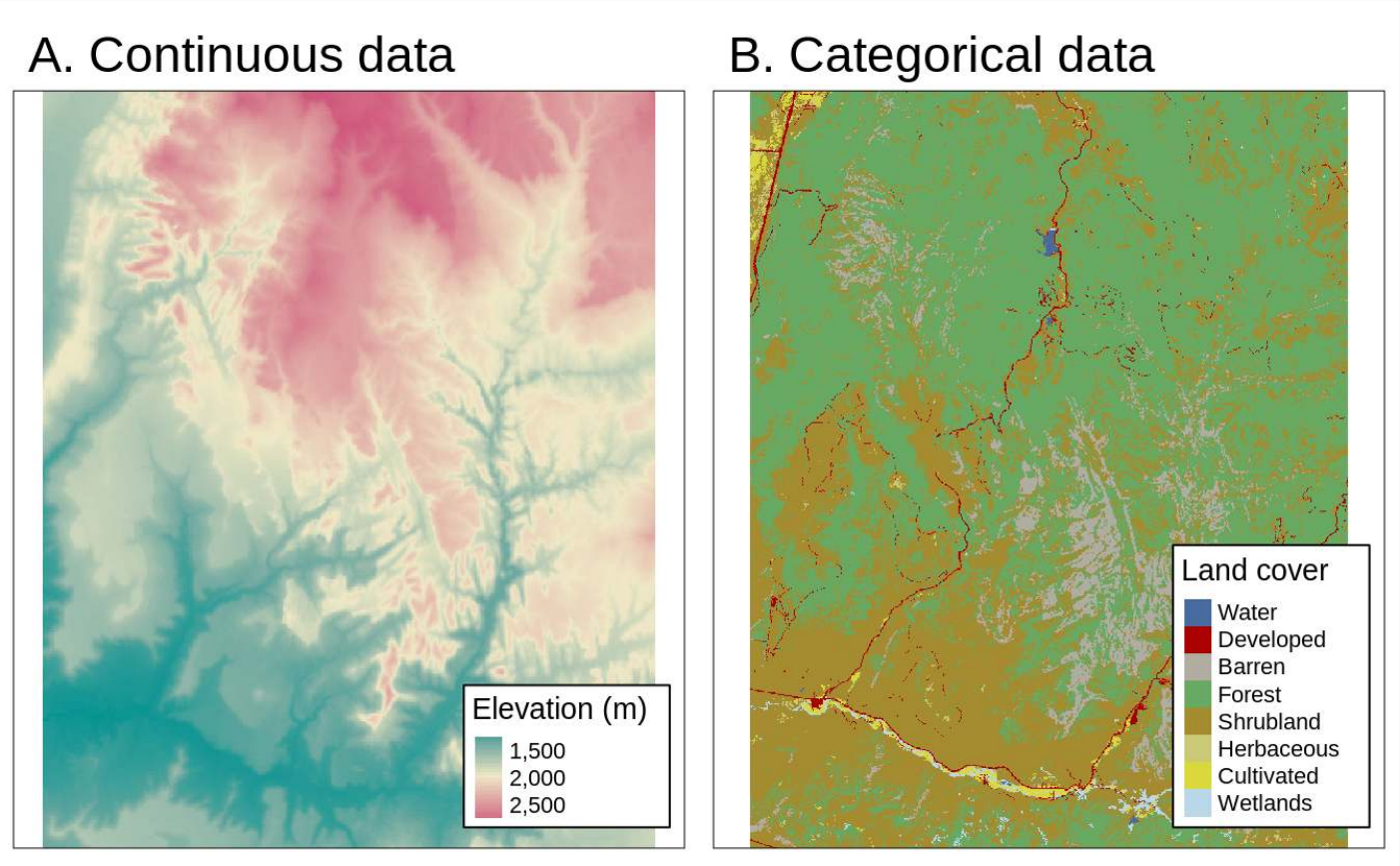
Raster ou Gride ou Dado Matricial (matriz)



National Ecological Observatory Network (NEON)

2. Classes raster

Tipos de dados: contínuos ou categóricos



Lovelace et al. (2020)

2. Classes raster

Valores

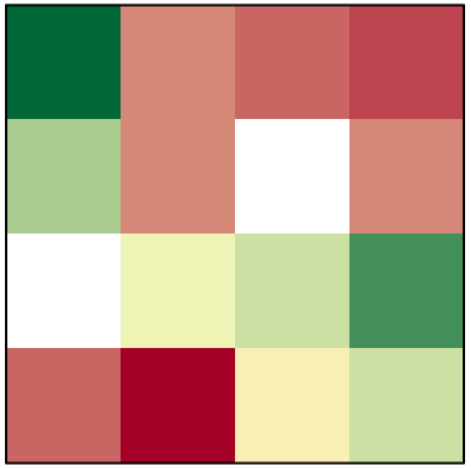
A. Cell IDs

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

B. Cell values

100	28	22	15
73	31	NA	30
NA	59	62	91
25	6	53	66

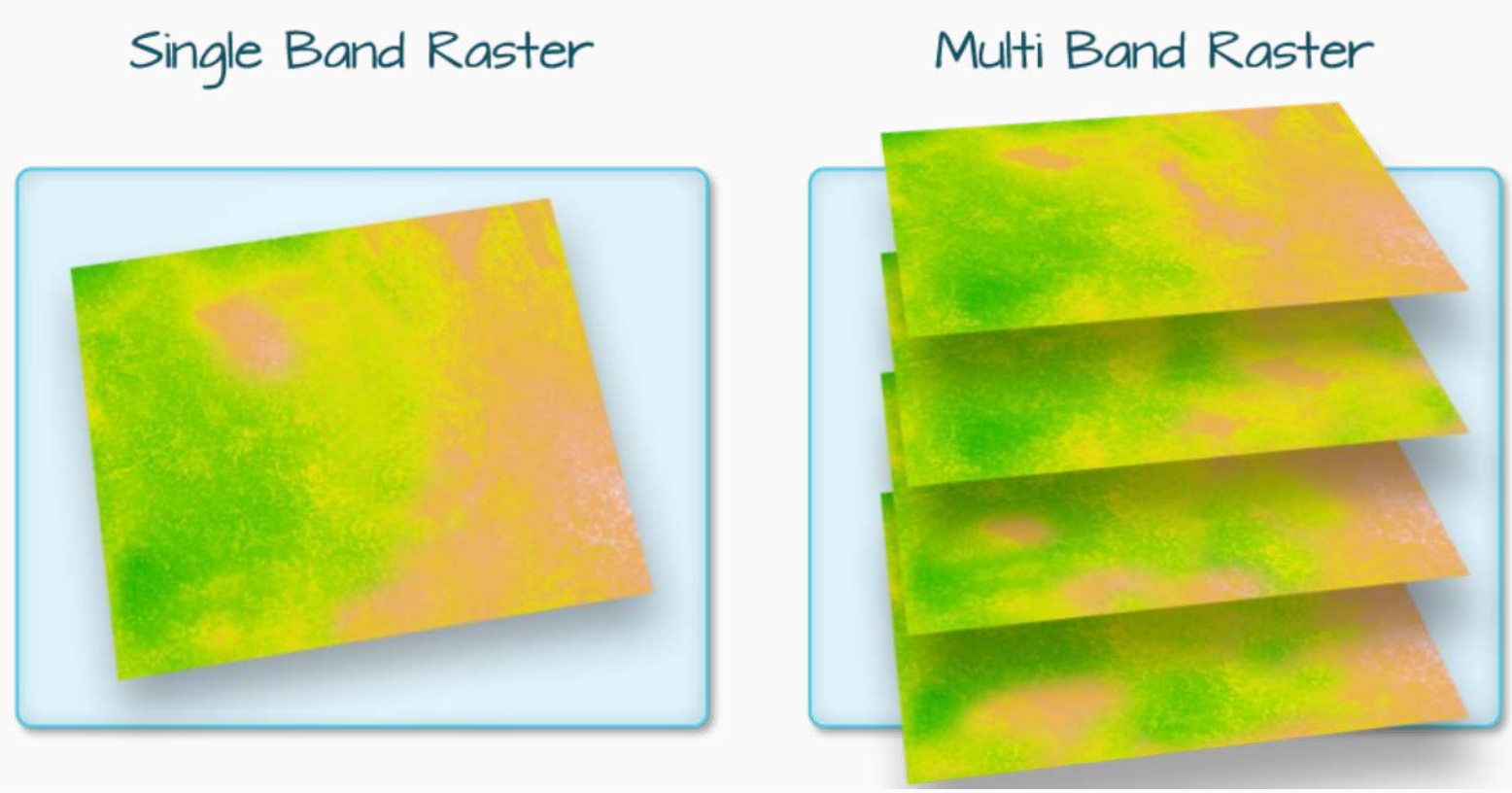
C. Colored values



Lovelace et al. (2020)

2. Classes raster

RasterLayer, RasterStack ou RasterBrick



National Ecological Observatory Network (NEON)

2. Classes raster

RasterLayer

A classe **RasterLayer** representa apenas uma camada raster

National Ecological Observatory Network (NEON)

2. Classes raster

RasterLayer

A classe **RasterLayer** representa apenas uma camada raster

```
# volcano
volcano
```

```
##       [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12] [,13] [,14] [,15] [,16] [,17] [,18] [,19] [,2
##  [1,]  100  100  101  101  101  101  101  100  100   100   101   101   102   102   102   102   103   104   103   1
##  [2,]  101  101  102  102  102  102  102  101  101   101   102   102   103   103   103   103   104   105   104   1
##  [3,]  102  102  103  103  103  103  103  102  102   102   103   103   104   104   104   104   105   106   105   1
##  [4,]  103  103  104  104  104  104  104  103  103   103   103   104   104   104   105   105   106   107   106   1
##  [5,]  104  104  105  105  105  105  105  104  104   103   104   104   105   105   105   106   107   108   108   1
##  [6,]  105  105  105  106  106  106  106  105  105   104   104   105   105   106   106   107   109   110   110   1
##  [7,]  105  106  106  107  107  107  107  106  106   105   105   106   106   107   108   109   111   113   114   1
##  [8,]  106  107  107  108  108  108  108  107  107   106   106   107   108   108   110   113   115   117   118   1
##  [9,]  107  108  108  109  109  109  109  108  108   107   108   108   110   111   113   116   118   120   123   1
## [10,]  108  109  109  110  110  110  110  109  109   108   110   110   113   116   118   120   122   125   127   1
## [11,]  109  110  110  111  111  111  111  110  110   110   112   114   118   121   123   125   127   129   133   1
## [12,]  110  110  111  113  112  111  113  112  112   114   116   119   121   124   127   129   133   138   143   1
## [13,]  110  111  113  115  114  113  114  114  115   117   119   121   124   126   129   133   140   145   150   1
## [14,]  111  113  115  117  116  115  116  117  117   119   121   124   126   128   132   137   143   151   156   1
```

2. Classes raster

RasterLayer

A classe **RasterLayer** representa apenas uma camada raster

```
# raster layer
ra_layer ← raster::raster(volcano)
ra_layer
```

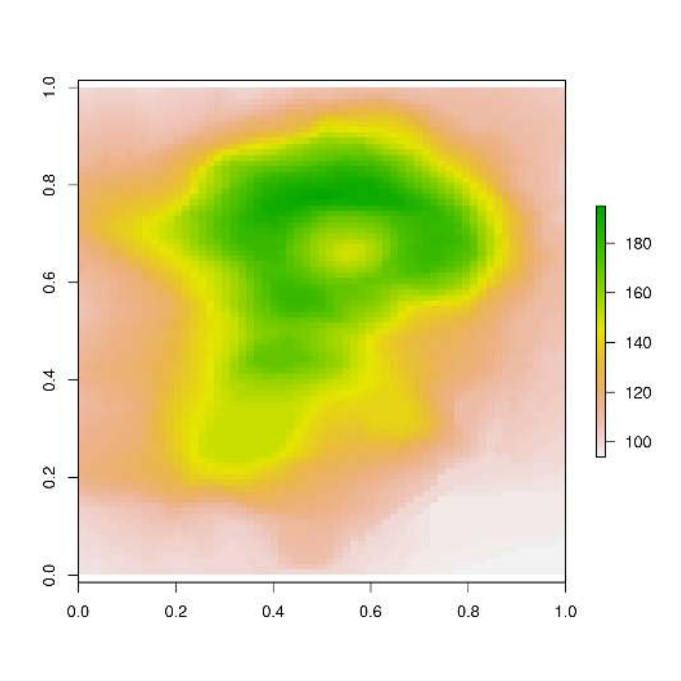
```
## class      : RasterLayer
## dimensions : 87, 61, 5307  (nrow, ncol, ncell)
## resolution : 0.01639344, 0.01149425  (x, y)
## extent     : 0, 1, 0, 1  (xmin, xmax, ymin, ymax)
## crs        : NA
## source     : memory
## names      : layer
## values     : 94, 195  (min, max)
```

2. Classes raster

RasterLayer

A classe **RasterLayer** representa apenas uma camada raster

```
# plot
plot(ra_layer)
```

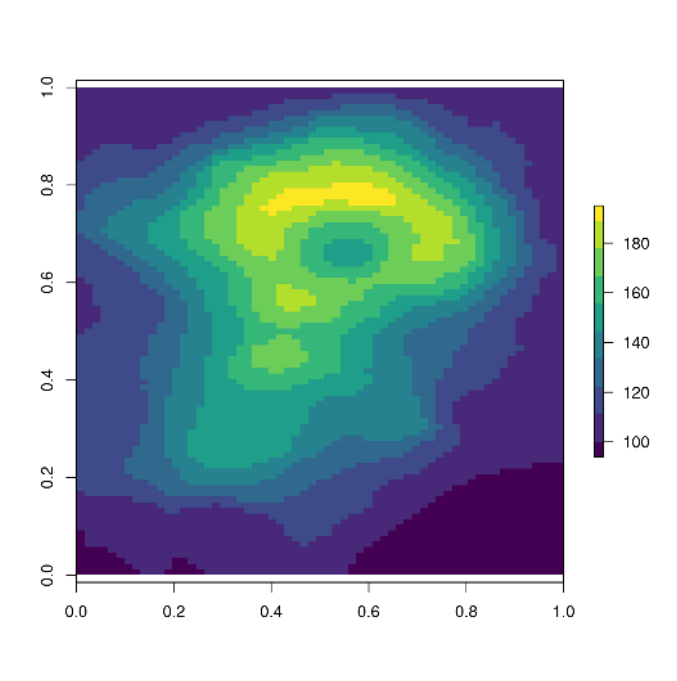


2. Classes raster

RasterLayer

A classe **RasterLayer** representa apenas uma camada raster

```
# plot
plot(ra_layer, col = viridis::viridis(10))
```

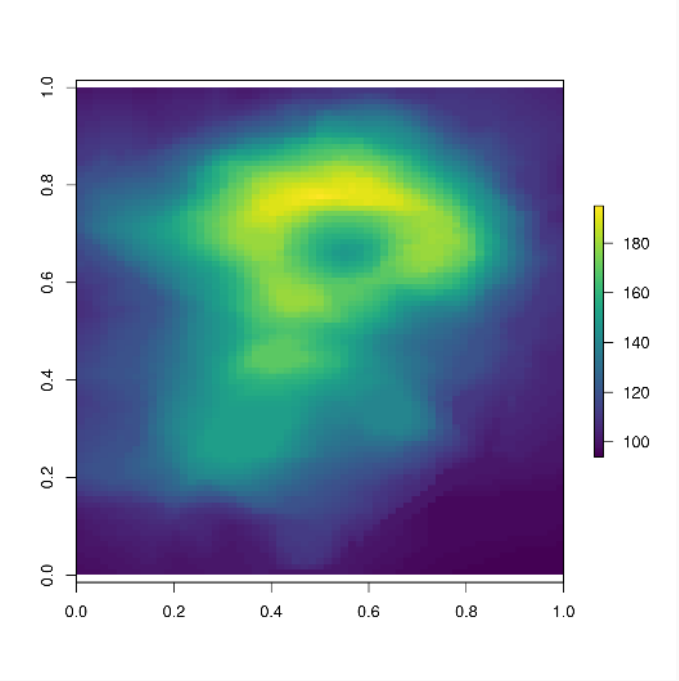


2. Classes raster

RasterLayer

A classe **RasterLayer** representa apenas uma camada raster

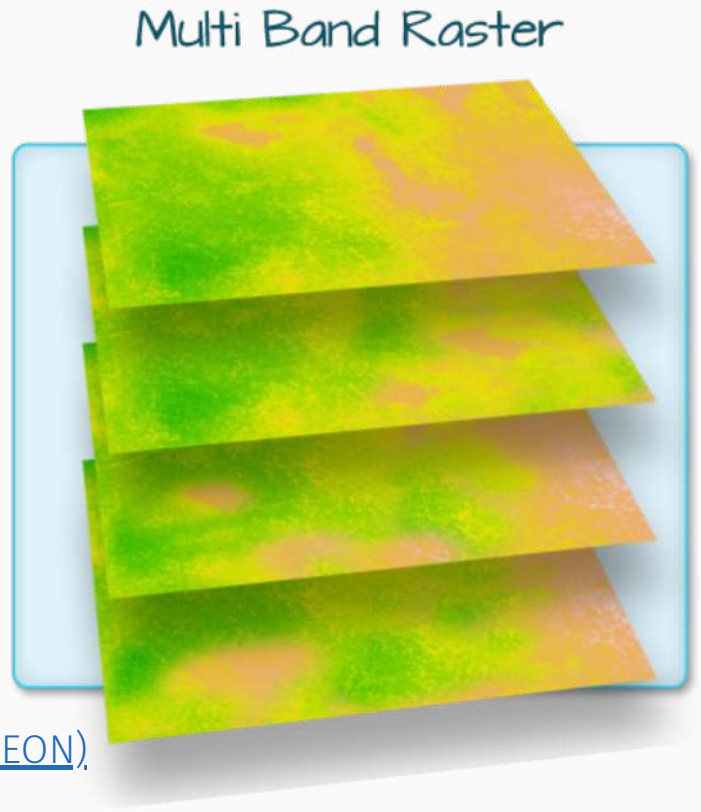
```
# plot
plot(ra_layer, col = viridis::viridis(100))
```



2. Classes raster

RasterStack

A classe **RasterStack** é uma lista de **RasterLayer** com extensão e resolução iguais



National Ecological Observatory Network (NEON)

2. Classes raster

RasterStack

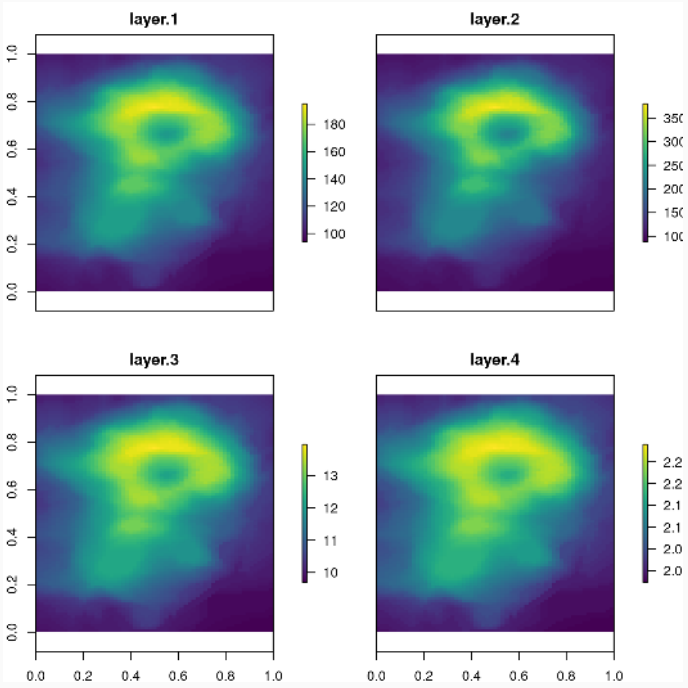
```
# raster stack
ra_layer1 ← ra_layer
ra_layer2 ← ra_layer * ra_layer
ra_layer3 ← sqrt(ra_layer)
ra_layer4 ← log10(ra_layer)

ra_stack ← raster::stack(ra_layer1,
                         ra_layer2,
                         ra_layer3,
                         ra_layer4)
ra_stack
```

```
## class      : RasterStack
## dimensions : 87, 61, 5307, 4  (nrow, ncol, ncell, nlayers)
## resolution : 0.01639344, 0.01149425  (x, y)
## extent     : 0, 1, 0, 1  (xmin, xmax, ymin, ymax)
## crs        : NA
## names      :      layer.1,      layer.2,      layer.3,      layer.4
## min values :    94.000000,  8836.000000,     9.695360,     1.973128
## max values :   195.000000, 38025.000000,    13.964240,     2.290035
```

2. Classes raster

RasterStack

```
# plot
plot(ra_stack, col = viridis::viridis(100))
```



2. Classes raster

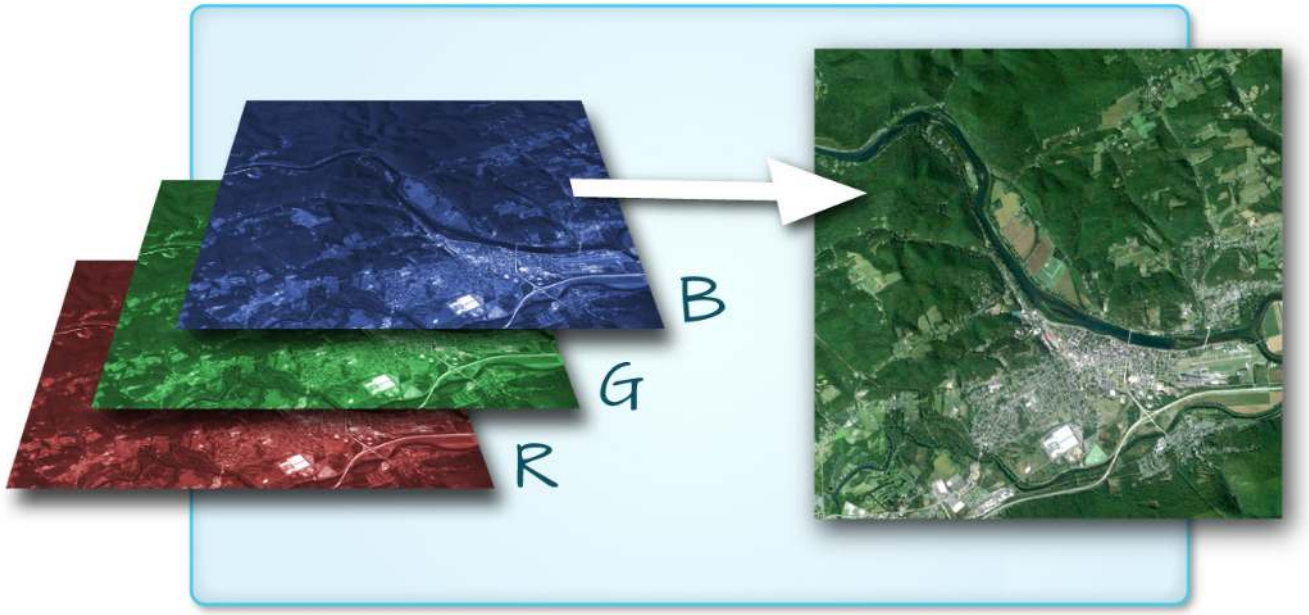
RasterBrick

A classe **RasterStack** é uma lista de **RasterLayer** com extensão e resolução iguais

2. Classes raster

RasterBrick

A principal diferença entre **RasterStack** e **RasterBrick** é que o segundo é vinculado a um **único arquivo (multicamadas)**



National Ecological Observatory Network (NEON)

2. Classes raster

RasterBrick

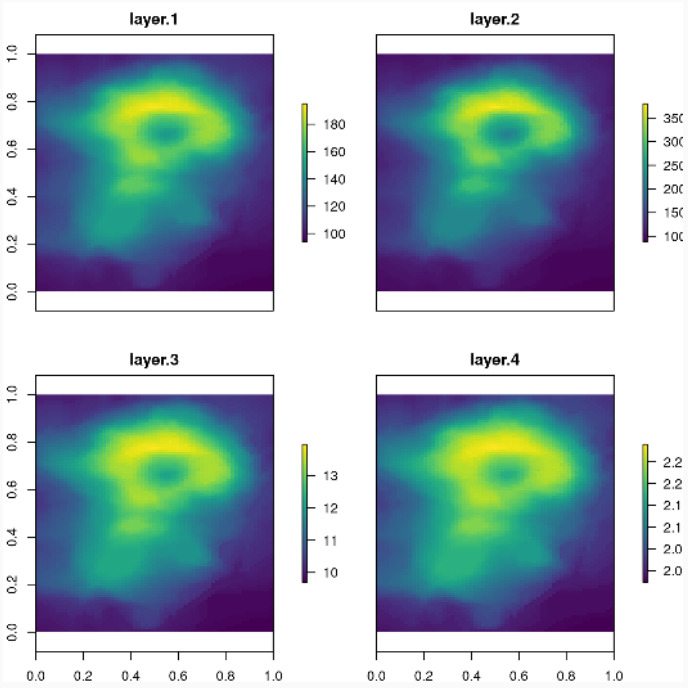
```
# raster brick
ra_layer1 ← ra_layer
ra_layer2 ← ra_layer * ra_layer
ra_layer3 ← sqrt(ra_layer)
ra_layer4 ← log10(ra_layer)

ra_brick ← raster::brick(ra_layer1,
                         ra_layer2,
                         ra_layer3,
                         ra_layer4)
ra_brick
```

```
## class      : RasterBrick
## dimensions : 87, 61, 5307, 4  (nrow, ncol, ncell, nlayers)
## resolution : 0.01639344, 0.01149425  (x, y)
## extent     : 0, 1, 0, 1  (xmin, xmax, ymin, ymax)
## crs        : NA
## source     : memory
## names      :      layer.1,      layer.2,      layer.3,      layer.4
## min values :    94.000000,  8836.000000,     9.695360,     1.973128
## max values :   195.000000, 38025.000000,    13.964240,     2.290035
```

2. Classes raster

RasterBrick

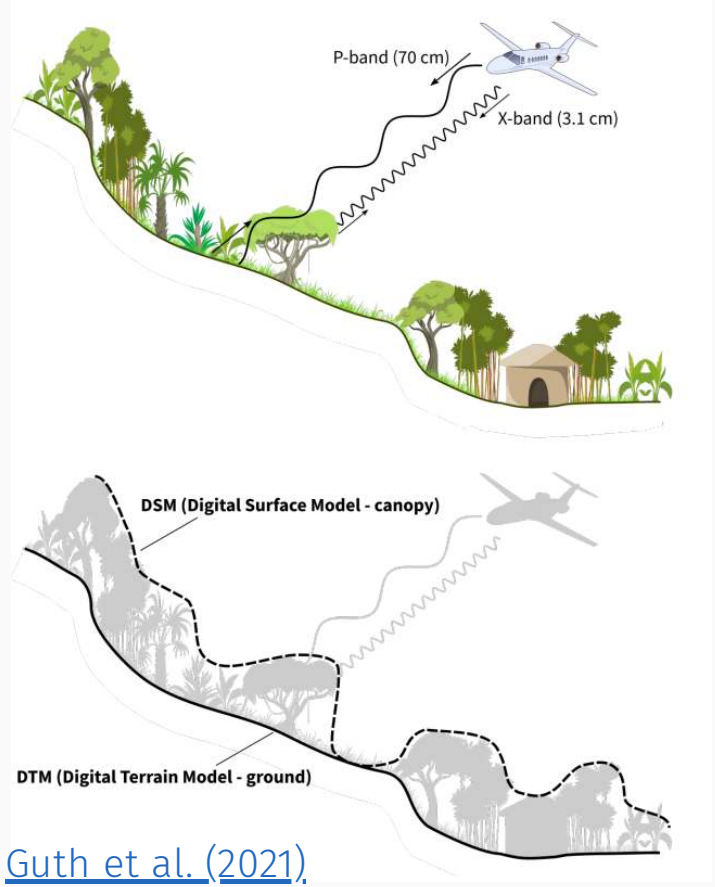
```
# plot
plot(ra_brick, col = viridis::viridis(100))
```

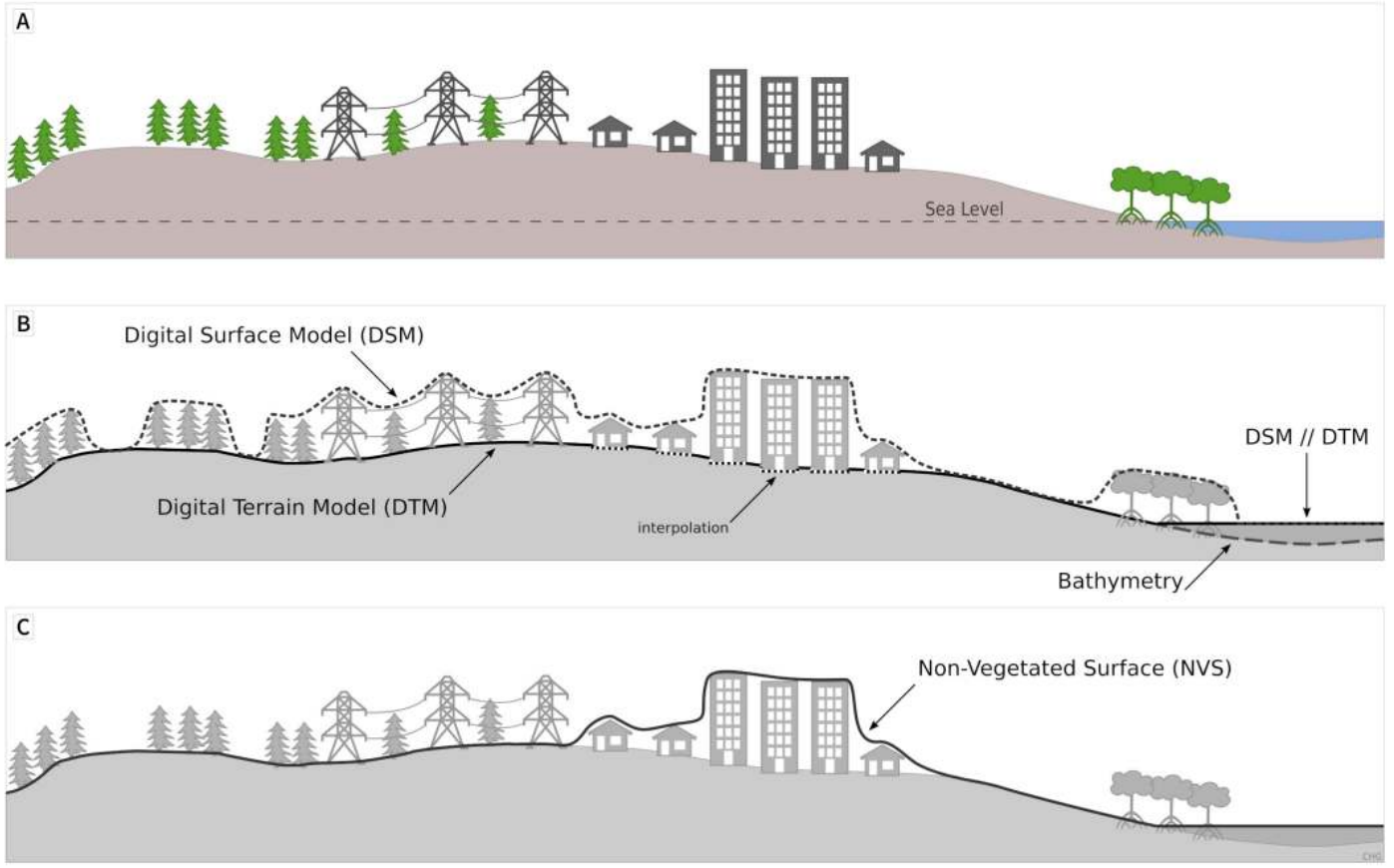


3. Importar dados matriciais

MDE (Modelo Digital de Elevação)

> Infere a **elevação** por sensores ativos - MDT (Modelo Digital de Terreno) e MDS (Modelo Digital de Superfície)

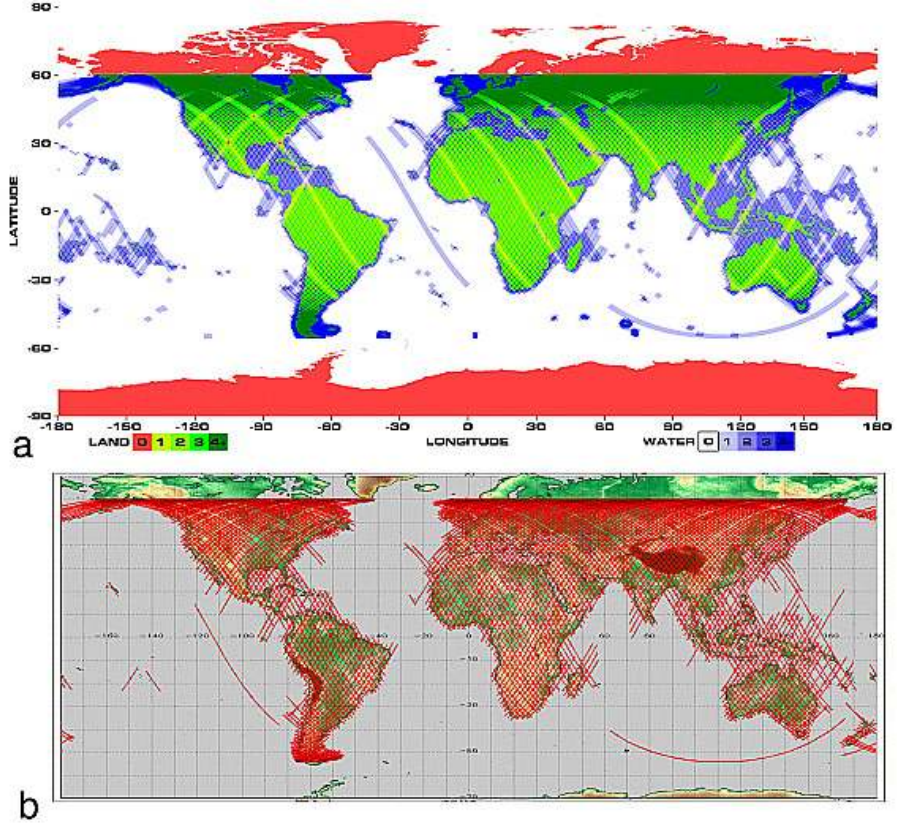


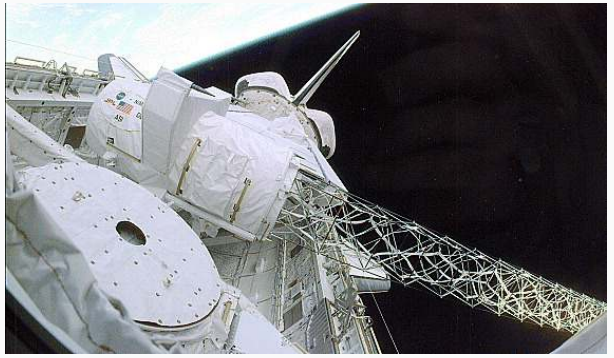


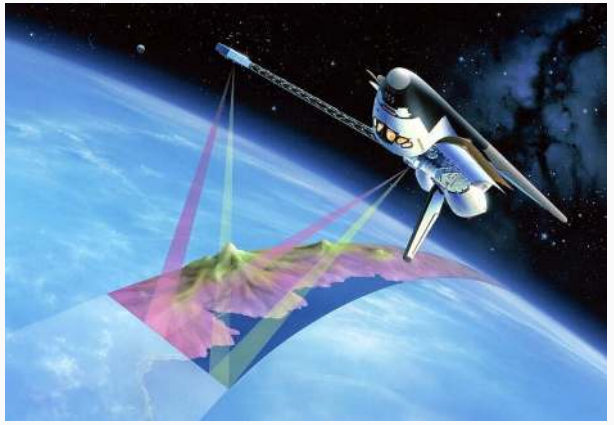
Guth et al. (2021)

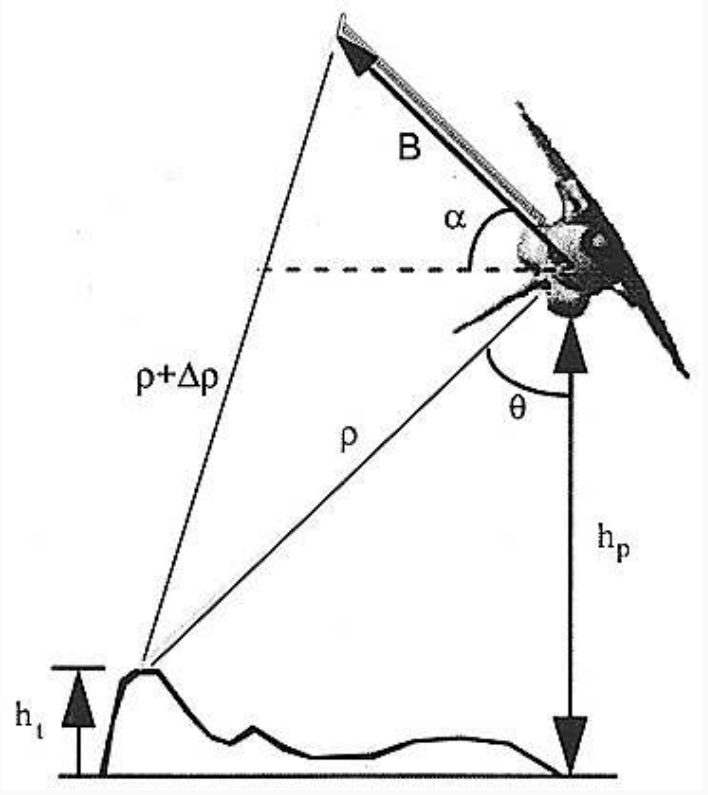
3. Importar dados matriciais

SRTM (*Shuttle Radar Topography Mission*)









Farr et al. (2007), Guth et al. (2021)

3. Importar dados matriciais

Importar uma camada

RasterLayer

```
# importar raster
dem ← raster::raster(here::here("03_dados", "raster", "srtm_27_17.tif"))
dem
```

```
## class      : RasterLayer
## dimensions : 6000, 6000, 3.6e+07  (nrow, ncol, ncell)
## resolution : 0.0008333333, 0.0008333333  (x, y)
## extent     : -50, -45, -25, -20  (xmin, xmax, ymin, ymax)
## crs        : +proj=longlat +datum=WGS84 +no_defs
## source     : srtm_27_17.tif
## names      : srtm_27_17
## values     : -32768, 32767  (min, max)
```

3. Importar dados matriciais

Importar uma camada

Rio Claro/SP

```
# rio claro
rc_2020 ← geobr::read_municipality(code_muni = 3543907, showProgress = FALSE) %>%
  sf::st_transform(crs = 4326)
rc_2020
```

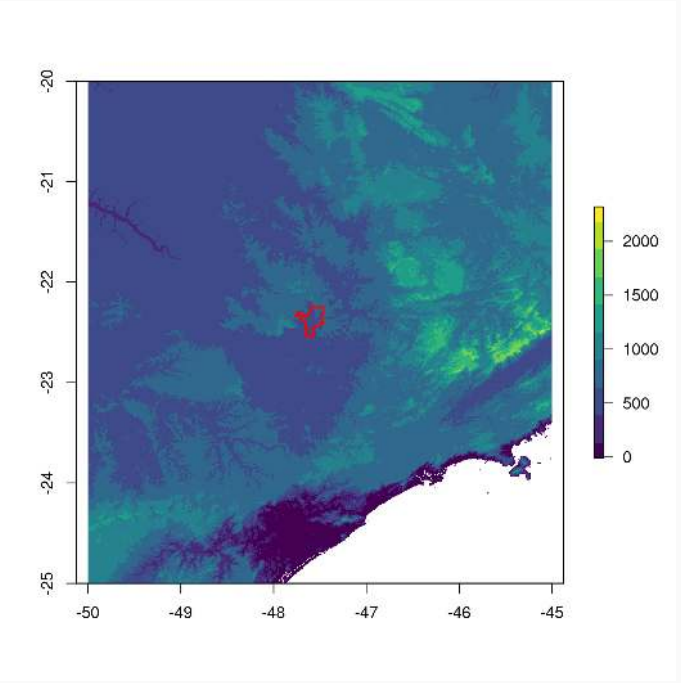
```
## Simple feature collection with 1 feature and 4 fields
## Geometry type: MULTIPOLYGON
## Dimension:     XY
## Bounding box:  xmin: -47.76536 ymin: -22.55207 xmax: -47.46165 ymax: -22.24368
## Geodetic CRS:  WGS 84
##     code_muni name_muni code_state abbrev_state                           geom
## 493   3543907 Rio Claro         35           SP MULTIPOLYGON (((-47.59464 - ...
```

3. Importar dados matriciais

Importar uma camada

RasterLayer

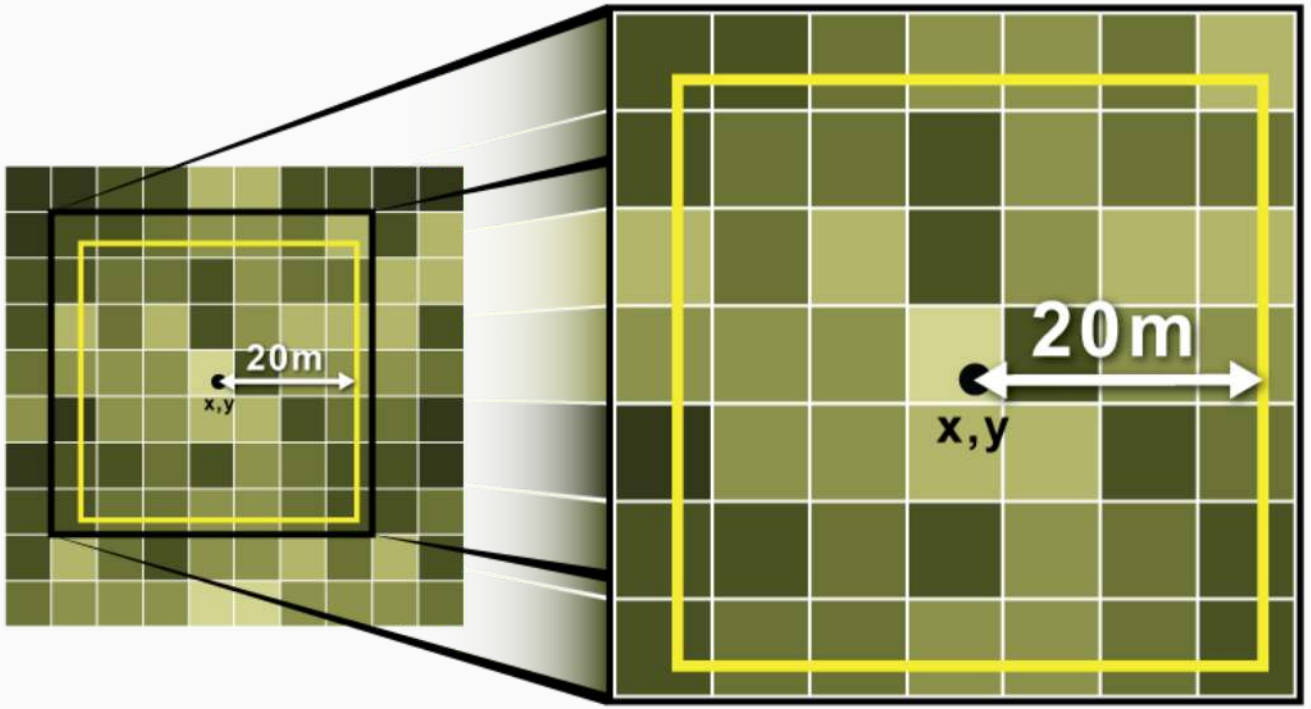
```
# plot
plot(dem, col = viridis::viridis(10))
plot(rc_2020$geom, col = NA, border = "red", lwd = 2, add = TRUE)
```



3. Importar dados matriciais

Importar uma camada

Ajuste do limite do RasterLayer



National Ecological Observatory Network (NEON)

3. Importar dados matriciais

Importar uma camada

Ajuste do limite do RasterLayer

```
# ajuste do limite
dem_rc ← raster::crop(dem, rc_2020)
dem_rc
```

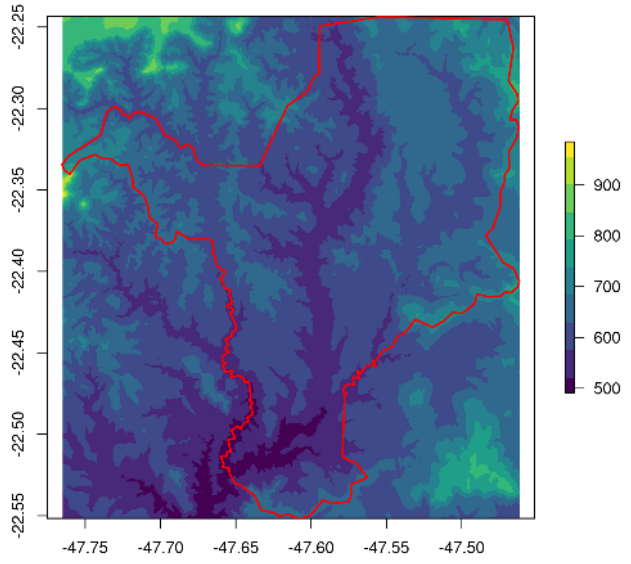
```
## class      : RasterLayer
## dimensions : 370, 364, 134680  (nrow, ncol, ncell)
## resolution : 0.0008333333, 0.0008333333  (x, y)
## extent     : -47.765, -47.46167, -22.55167, -22.24333  (xmin, xmax, ymin, ymax)
## crs        : +proj=longlat +datum=WGS84 +no_defs
## source     : memory
## names      : srtm_27_17
## values     : 491, 985  (min, max)
```

3. Importar dados matriciais

Importar uma camada

Ajuste do limite do RasterLayer

```
# plot
plot(dem_rc, col = viridis::viridis(10))
plot(rc_2020$geom, col = NA, border = "red", lwd = 2, add = TRUE)
```



3. Importar dados matriciais

O que é o WorldClim?

- Principal bases de **dados climáticos** para o mundo
- Dados temporais de **temperatura e precipitação**
- Construído a partir de dados de **estações meteorológicas, interpolação e topografia**
- Principal forma de uso: **variáveis bioclimáticas**

INTERNATIONAL JOURNAL OF CLIMATOLOGY
Int. J. Climatol. (2017)
Published online in Wiley Online Library
(wileyonlinelibrary.com) DOI: 10.1002/joc.5086

RMetS
Royal Meteorological Society

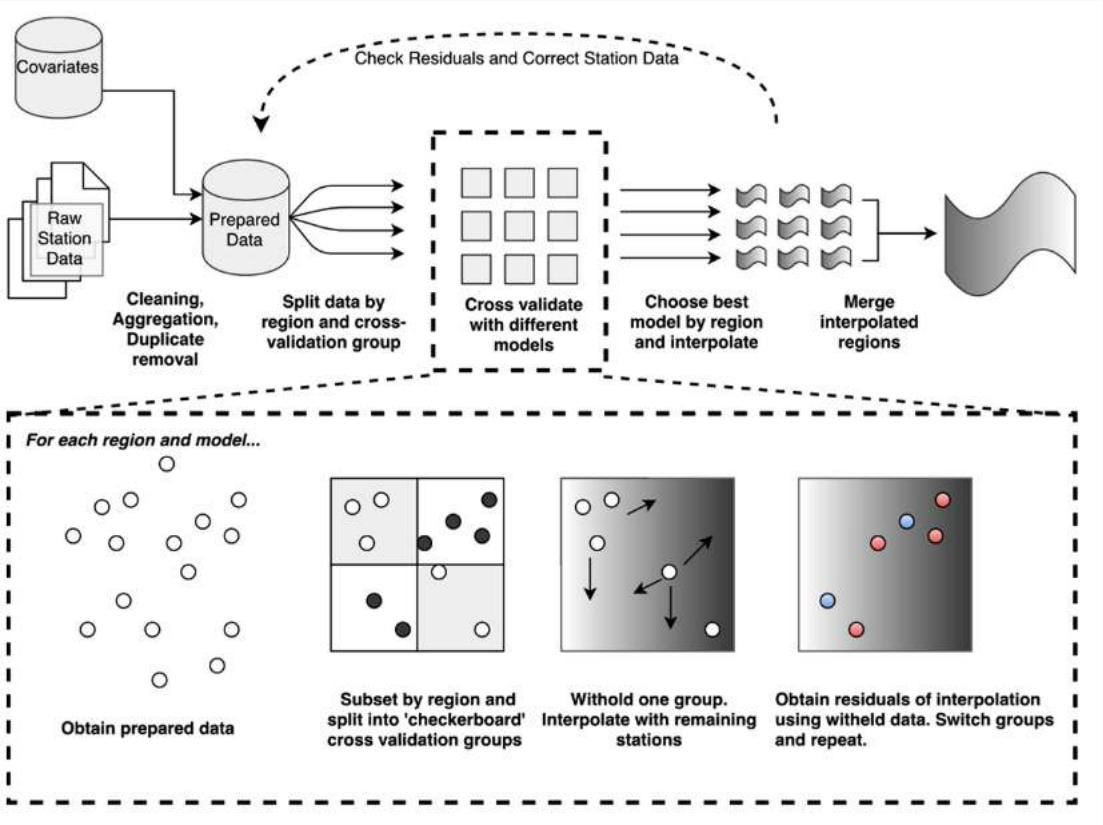
WorldClim 2: new 1-km spatial resolution climate surfaces for global land areas

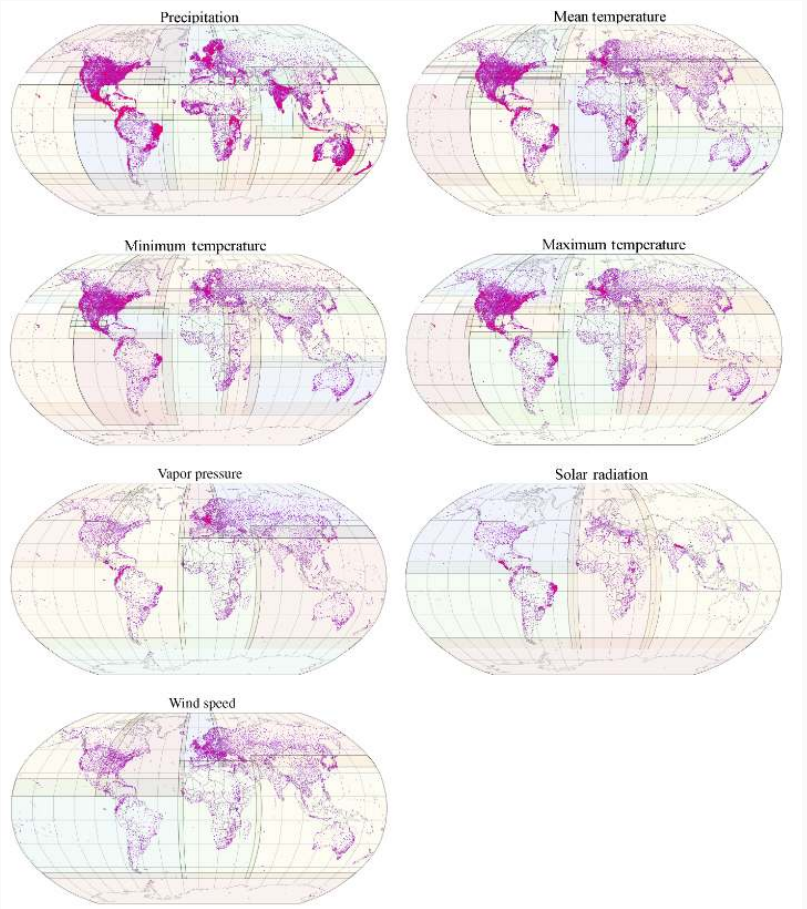
Stephen E. Fick[a]* and Robert J. Hijmans[b]
[a] *Department of Plant Sciences, University of California, Davis, CA, USA*
[b] *Department of Environmental Science and Policy, University of California, Davis, CA, USA*

Fick & Hijmans (2017)

3. Importar dados matriciais

Estações meteorológicas e interpolação





Fick & Hijmans (2017)

3. Importar dados matriciais

Variáveis bioclimáticas

- São **19 variáveis** relacionadas com a **biologia das organismos**
- Combinações temporais de **temperatura (BIO01-BIO11)** e **precipitação (BIO12-BIO19)**

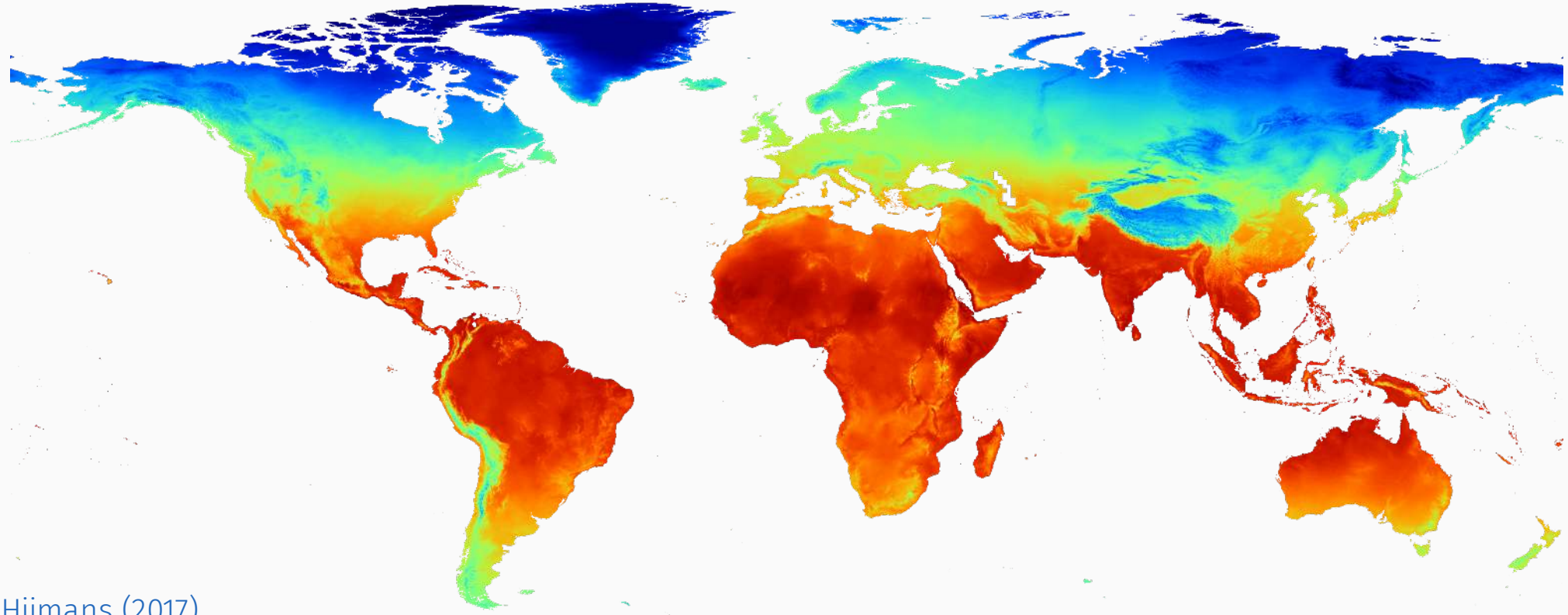
Variable	Description	Temporal Scale
Bioclim 1	Annual Mean Temperature	Annual
Bioclim 2	Mean Diurnal Range	Variation
Bioclim 3	Isothermality	Variation
Bioclim 4	Temperature Seasonality	Variation
Bioclim 5	Maximum Temperature of the Warmest Month	Month
Bioclim 6	Minimum Temperature of the Coldest Month	Month
Bioclim 7	Temperature Annual Range	Annual
Bioclim 8	Mean Temperature of Wettest Quarter	Quarter
Bioclim 9	Mean Temperature of Driest Quarter	Quarter
Bioclim 10	Mean Temperature of Warmest Quarter	Quarter
Bioclim 11	Mean Temperature of Coldest Quarter	Quarter
Bioclim 12	Annual Precipitation	Annual
Bioclim 13	Precipitation of Wettest Month	Month
Bioclim 14	Precipitation of Driest Month	Month
Bioclim 15	Precipitation Seasonality	Variation
Bioclim 16	Precipitation of Wettest Quarter	Quarter
Bioclim 17	Precipitation of Driest Quarter	Quarter
Bioclim 18	Precipitation of Warmest Quarter	Quarter
Bioclim 19	Precipitation of Coldest Quarter	Quarter

Fick & Hijmans (2017)

3. Importar dados matriciais

Variáveis bioclimáticas

BIO01: Temperatura média anual (º C)

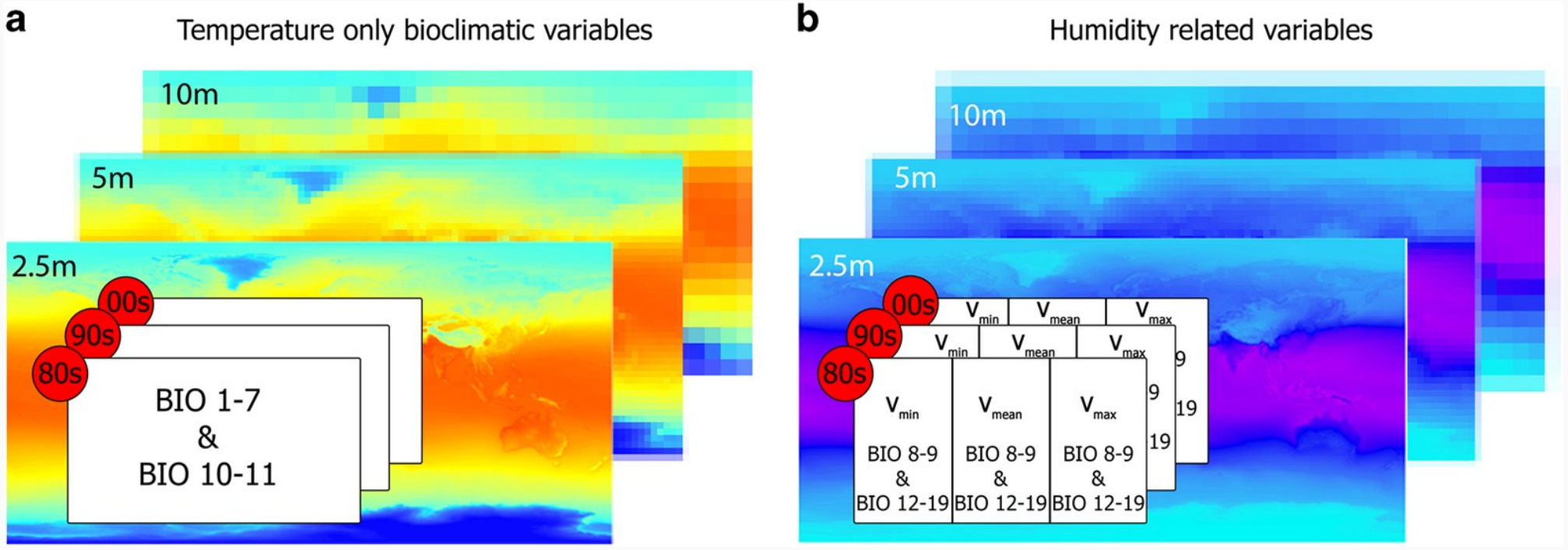


Fick & Hijmans (2017)

3. Importar dados matriciais

Variáveis Bioclimáticas

Disponível em **várias resoluções** e **períodos (passado, presente e futuro)**



Vega (2017)

3. Importar dados matriciais

Site

WorldClim

3. Importar dados matriciais

Importar várias camadas

Listar arquivos

```
# listar arquivos
files ← dir(path = here::here("03_dados", "raster"), pattern = "wc", full.names = TRUE) %>%
  grep(".tif", ., value = TRUE)
files
```

```
##  [1] "/home/mude/data/github/course-geospatial-data-r/03_dados/raster/wc2.1_10m_bio_1.tif"
##  [2] "/home/mude/data/github/course-geospatial-data-r/03_dados/raster/wc2.1_10m_bio_10.tif"
##  [3] "/home/mude/data/github/course-geospatial-data-r/03_dados/raster/wc2.1_10m_bio_11.tif"
##  [4] "/home/mude/data/github/course-geospatial-data-r/03_dados/raster/wc2.1_10m_bio_12.tif"
##  [5] "/home/mude/data/github/course-geospatial-data-r/03_dados/raster/wc2.1_10m_bio_13.tif"
##  [6] "/home/mude/data/github/course-geospatial-data-r/03_dados/raster/wc2.1_10m_bio_14.tif"
##  [7] "/home/mude/data/github/course-geospatial-data-r/03_dados/raster/wc2.1_10m_bio_15.tif"
##  [8] "/home/mude/data/github/course-geospatial-data-r/03_dados/raster/wc2.1_10m_bio_16.tif"
##  [9] "/home/mude/data/github/course-geospatial-data-r/03_dados/raster/wc2.1_10m_bio_17.tif"
## [10] "/home/mude/data/github/course-geospatial-data-r/03_dados/raster/wc2.1_10m_bio_18.tif"
## [11] "/home/mude/data/github/course-geospatial-data-r/03_dados/raster/wc2.1_10m_bio_19.tif"
## [12] "/home/mude/data/github/course-geospatial-data-r/03_dados/raster/wc2.1_10m_bio_2.tif"
## [13] "/home/mude/data/github/course-geospatial-data-r/03_dados/raster/wc2.1_10m_bio_3.tif"
```

3. Importar dados matriciais

Importar várias camadas

RasterStack

```
# importar
bioclim ← raster::stack(files)
bioclim
```

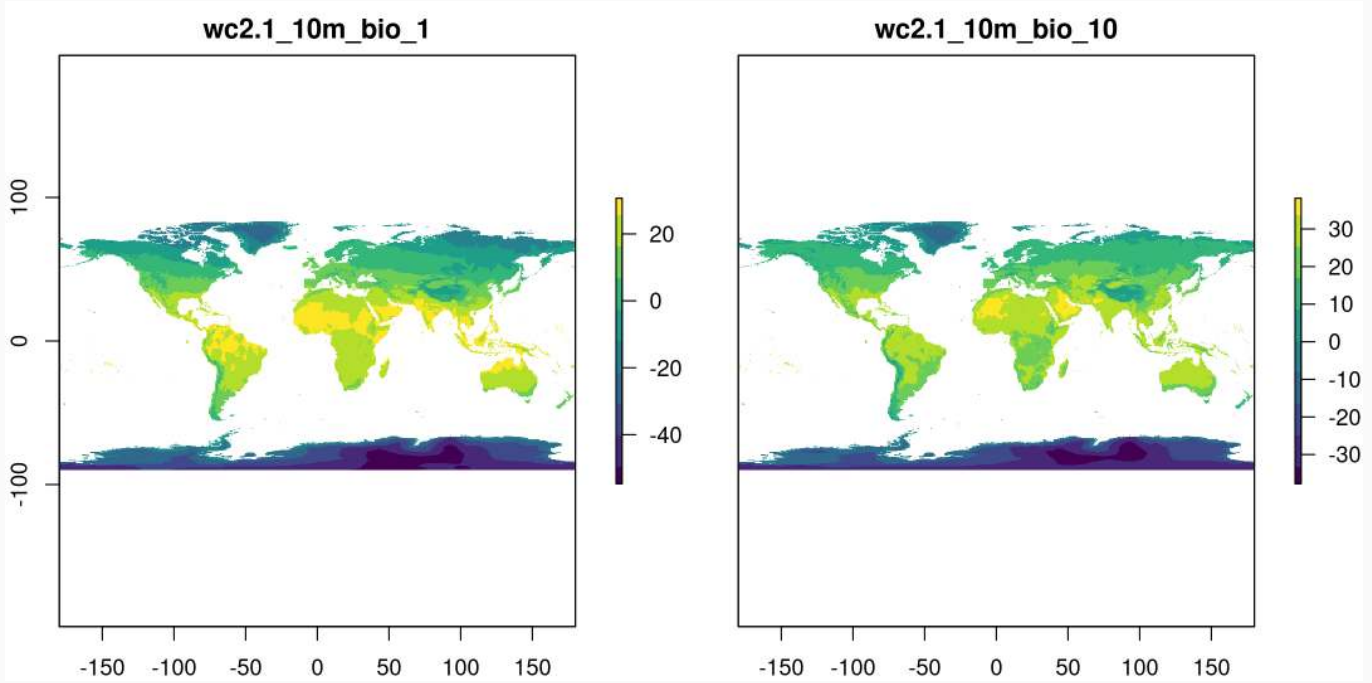
```
## class      : RasterStack
## dimensions : 1080, 2160, 2332800, 19  (nrow, ncol, ncell, nlayers)
## resolution : 0.1666667, 0.1666667  (x, y)
## extent     : -180, 180, -90, 90  (xmin, xmax, ymin, ymax)
## crs        : +proj=longlat +datum=WGS84 +no_defs
## names      : wc2.1_10m_bio_1, wc2.1_10m_bio_10, wc2.1_10m_bio_11, wc2.1_10m_bio_12, wc2.1_10m_bio_13, wc2.1_10m_bi
## min values :      -54.724354,       -37.781418,       -66.311249,         0.000000,         0.000000,         0.00
## max values :        30.98764,         38.21617,         29.15299,      11191.00000,       2381.00000,        484.0
```

3. Importar dados matriciais

Importar várias camadas

RasterStack

```
# plot
plot(bioclim[[1:2]], col = viridis::viridis(10))
```



3. Importar dados matriciais

Imagens de satélite

Landsat

- Série de satélites dedicado exclusivamente à observação dos **recursos naturais terrestres**, desenvolvido pela NASA

3. Importar dados matriciais

Imagens de satélite

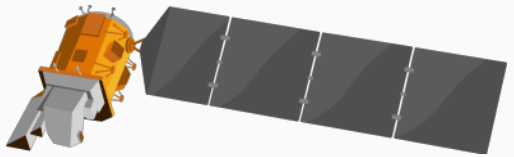
Landsat - 9 Missões

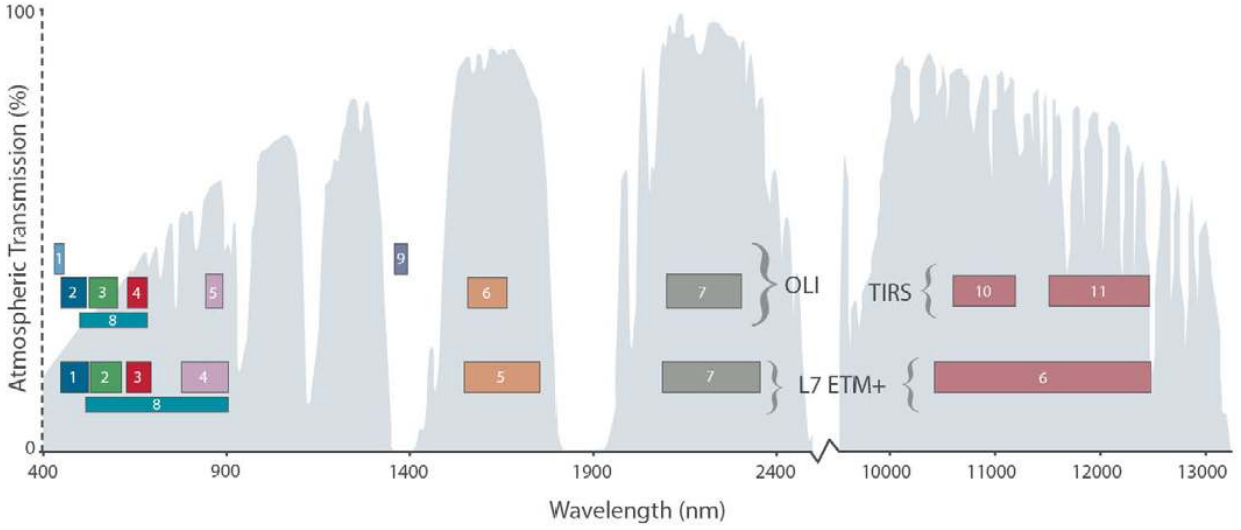
landsat-9

3. Importar dados matriciais

Imagens de satélite

Landsat 8 - Bandas



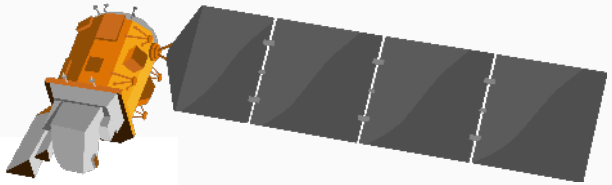


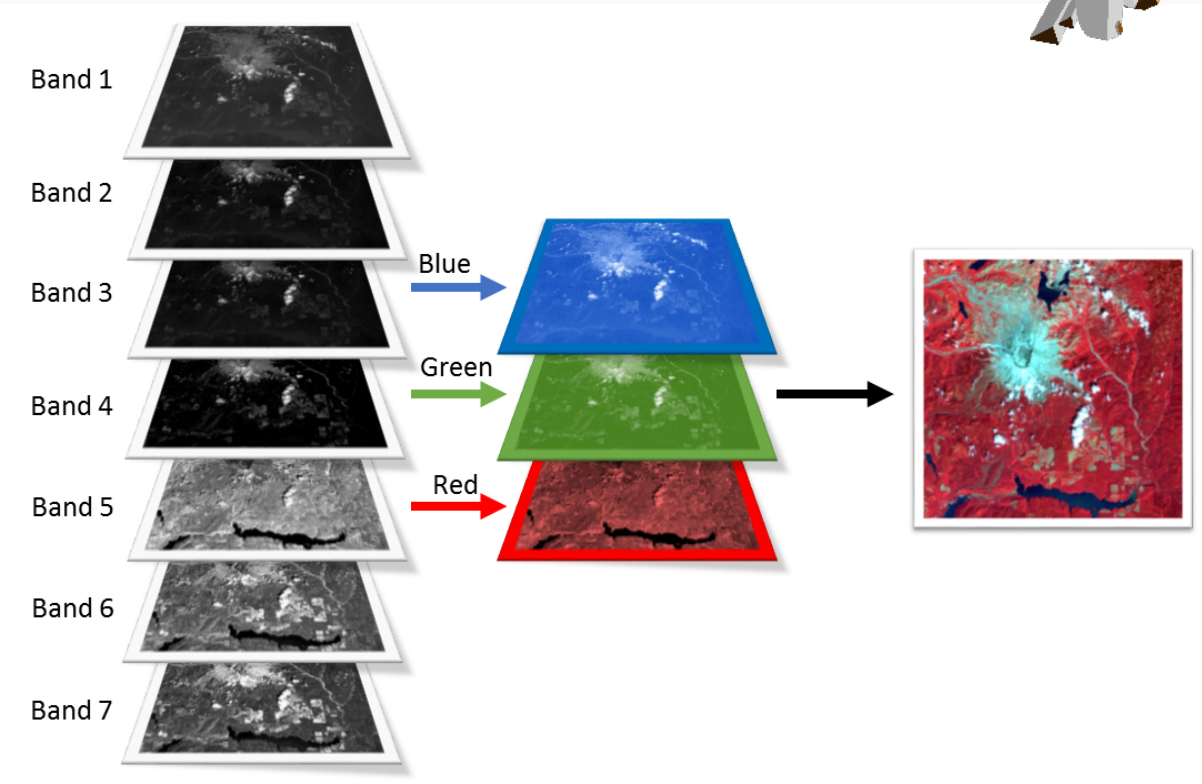
	No Landsat 5 & 7	No Landsat 8
Falsas cores Infra Vermelho:	4, 3, 2	5,4,3
Cores Naturais:	3, 2, 1	4,3,2
Cores Naturais Simuladas:	5,4,3	6,5,4
Cores Naturais Simuladas:	7,5,3	7,6,4
Cores Naturais Simuladas:	7,4,2	7,5,3

3. Importar dados matriciais

Imagens de satélite

Landsat 8 - Bandas como camadas





3. Importar dados matriciais

Importar várias camadas

RasterBrick

```
# importar
landsat_rc ← raster::brick(here::here("03_dados", "imagem", "landsat_rc.tif"))
landsat_rc
```

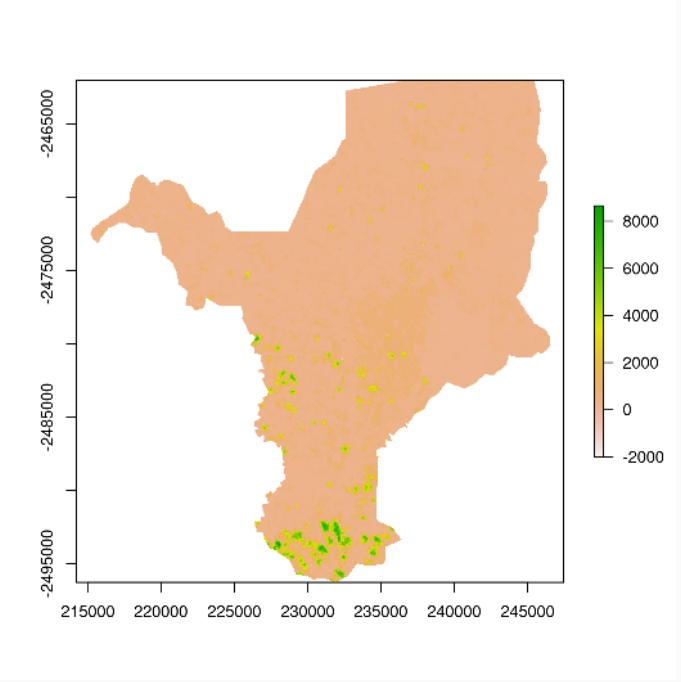
```
## class      : RasterBrick
## dimensions : 1142, 1047, 1195674, 7  (nrow, ncol, ncell, nlayers)
## resolution : 30, 30  (x, y)
## extent     : 215145, 246555, -2496285, -2462025  (xmin, xmax, ymin, ymax)
## crs        : +proj=utm +zone=23 +datum=WGS84 +units=m +no_defs
## source     : landsat_rc.tif
## names      : landsat_rc.1, landsat_rc.2, landsat_rc.3, landsat_rc.4, landsat_rc.5, landsat_rc.6, landsat_rc.7
## min values :        -2000,        -2000,         -135,         -303,           -8,          -50,           11
## max values :         8199,         8701,         9440,         9991,        11246,        12168,        11846
```

3. Importar dados matriciais

Importar várias camadas

RasterBrick

```
# plot
plot(landsat_rc$landsat_rc.2)
```

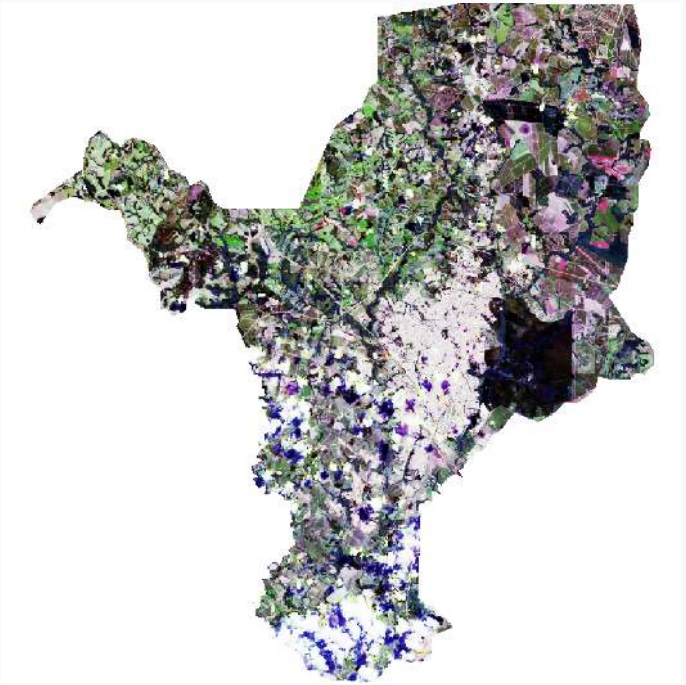


3. Importar dados matriciais

Importar várias camadas

RasterBrick

```
# plot - cores naturais
plotRGB(landsat_rc, r = 4, g = 3, b = 2, stretch = "hist")
```

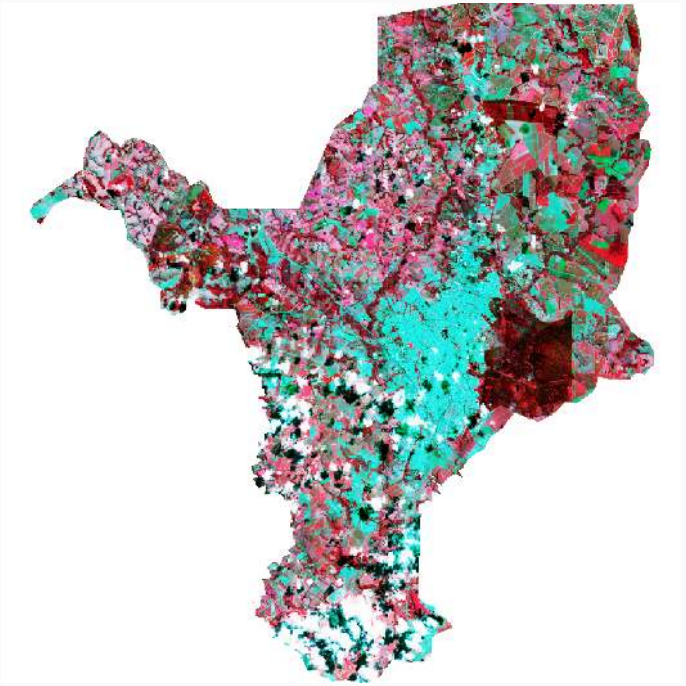


3. Importar dados matriciais

Importar várias camadas

RasterBrick

```
# plot - falsa cor infravermelho
plotRGB(landsat_rc, r = 5, g = 4, b = 3, stretch = "hist")
```

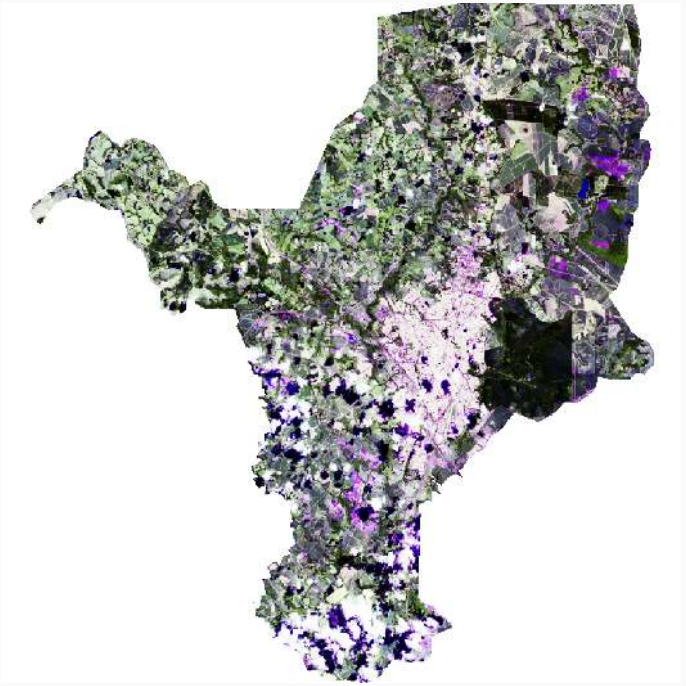


3. Importar dados matriciais

Importar várias camadas

RasterBrick

```
# plot - cores naturais simuladas
plotRGB(landsat_rc, r = 7, g = 6, b = 4, stretch = "hist")
```

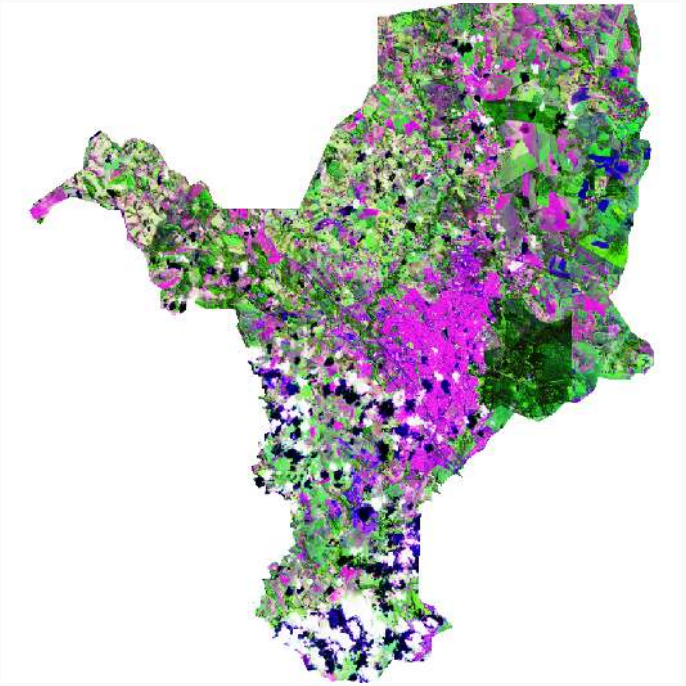


3. Importar dados matriciais

Importar várias camadas

RasterBrick

```
# plot - cores naturais simuladas
plotRGB(landsat_rc, r = 6, g = 5, b = 4, stretch = "hist")
```

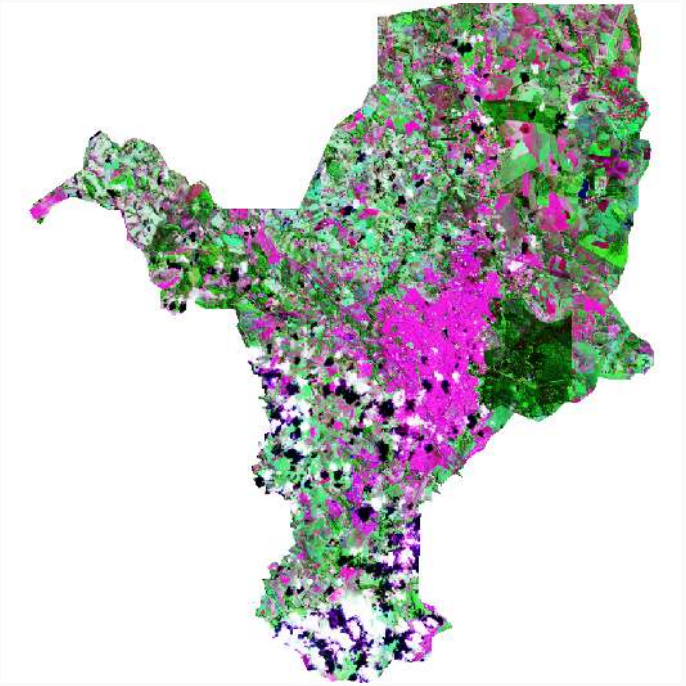


3. Importar dados matriciais

Importar várias camadas

RasterBrick

```
# plot - cores naturais simuladas
plotRGB(landsat_rc, r = 7, g = 5, b = 3, stretch = "hist")
```



4. Descrição de objetos raster

Informações geográficas

```
# rio claro
dem_rc
```

```
## class      : RasterLayer
## dimensions : 370, 364, 134680  (nrow, ncol, ncell)
## resolution : 0.0008333333, 0.0008333333  (x, y)
## extent     : -47.765, -47.46167, -22.55167, -22.24333  (xmin, xmax, ymin, ymax)
## crs        : +proj=longlat +datum=WGS84 +no_defs
## source     : memory
## names      : srtm_27_17
## values     : 491, 985  (min, max)
```

- *class*: classe raster do objeto
- *dimensions*: número de linhas, colunas, células e camadas
- *resolution*: largura e altura da célula
- *extent*: coordenadas mínimas e máximas da longitude e latitude
- *crs*: Sistema de Referência de Coordenadas
- *source*: fonte dos dados (memória ou disco)
- *names*: nome das camadas

4. Descrição de objetos raster

Informações geográficas

Classe

```
# classe
class(dem_rc)
```

```
## [1] "RasterLayer"
## attr(,"package")
## [1] "raster"
```

Dimensões

```
# dimensoes
dim(dem_rc)
```

```
## [1] 370 364   1
```

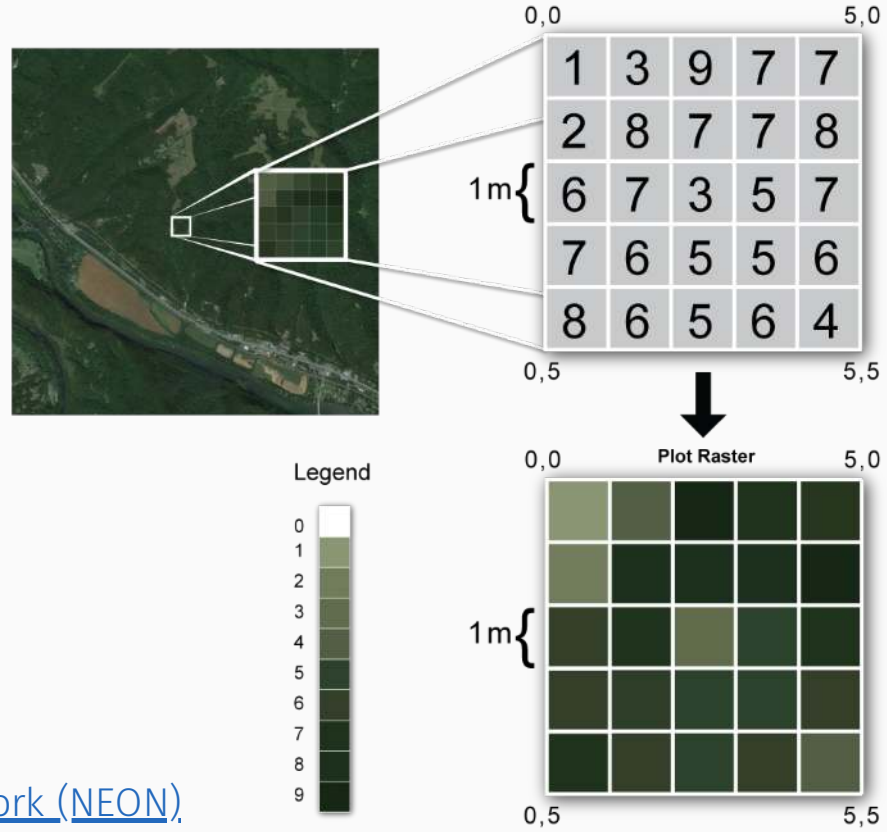
Número de camadas

```
# numero de camadas
nlayers(dem_rc)
```

4. Descrição de objetos raster

Informações geográficas

Número de linhas, colunas e células



National Ecological Observatory Network (NEON)

4. Descrição de objetos raster

Informações geográficas

Número de linhas

```
# numero de linhas
nrow(dem_rc)
```

```
## [1] 370
```

Número de colunas

```
# numero de colunas
ncol(dem_rc)
```

```
## [1] 364
```

Número de células

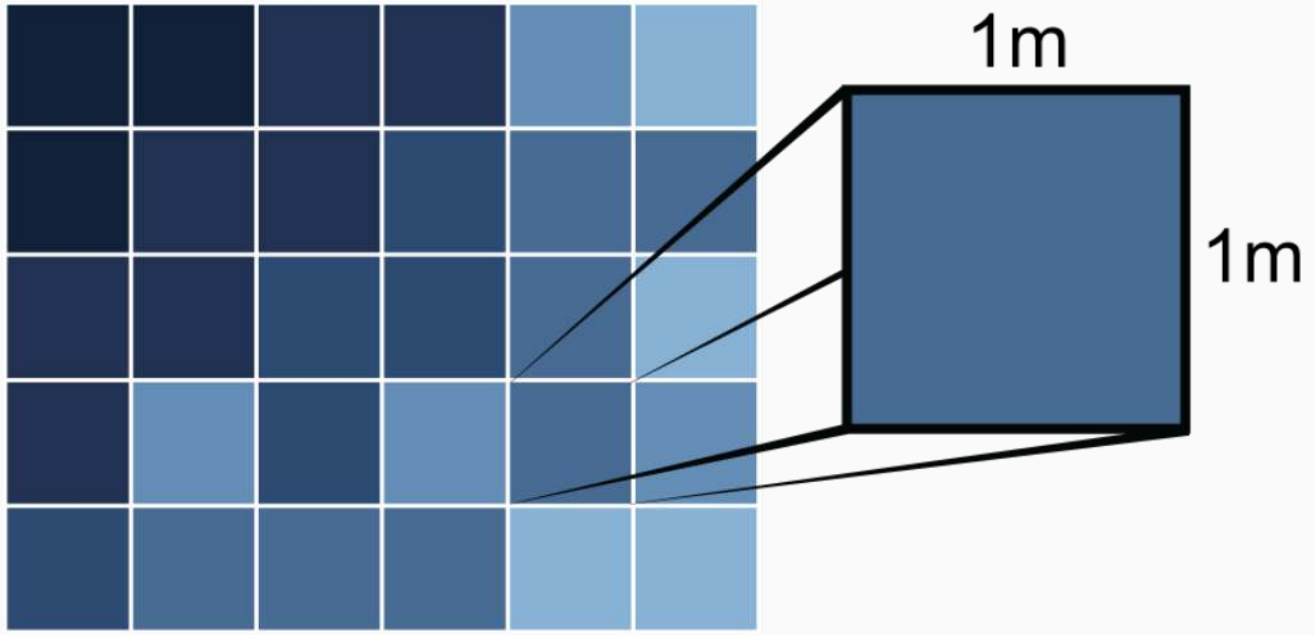
```
# numero de celulas
ncell(dem_rc)
```

```
## [1] 134680
```

4. Descrição de objetos raster

Informações geográficas

Resolução



National Ecological Observatory Network (NEON)

4. Descrição de objetos raster

Informações geográficas

Resolução

```
# resolucao do raster
res(dem_rc)
```

```
## [1] 0.0008333333 0.0008333333
```

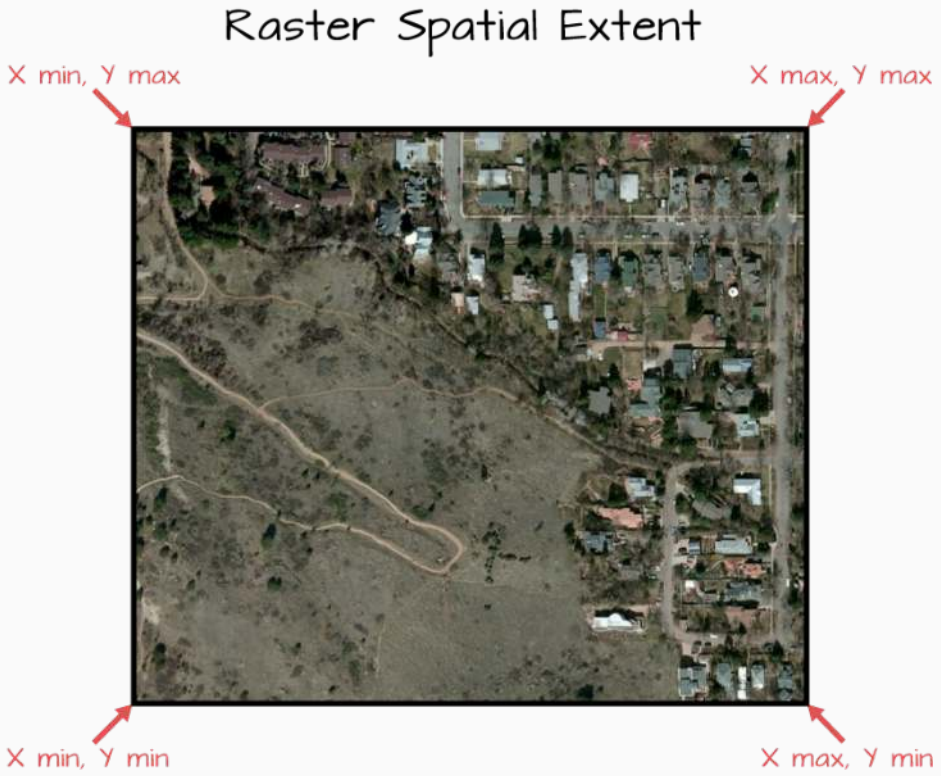
```
# resolucao do stack
res(bioclim)
```

```
## [1] 0.1666667 0.1666667
```

4. Descrição de objetos raster

Informações geográficas

Extensão



National Ecological Observatory Network (NEON)

4. Descrição de objetos raster

Informações geográficas

Extensão

```
# extensao do raster
extent(dem_rc)
```

```
## class      : Extent
## xmin       : -47.765
## xmax       : -47.46167
## ymin       : -22.55167
## ymax       : -22.24333
```

```
# extensao do stack
extent(bioclim)
```

```
## class      : Extent
## xmin       : -180
## xmax       : 180
## ymin       : -90
## ymax       : 90
```

4. Descrição de objetos raster

Informações geográficas

Projeção

```
# projecao
projection(dem_rc)
```

```
## [1] "+proj=longlat +datum=WGS84 +no_defs"
```

```
# projecao
projection(bioclim)
```

```
## [1] "+proj=longlat +datum=WGS84 +no_defs"
```

4. Descrição de objetos raster

Informações geográficas

Nomes

```
# nomes da camada do raster
names(dem_rc)
```

```
## [1] "srtm_27_17"
```

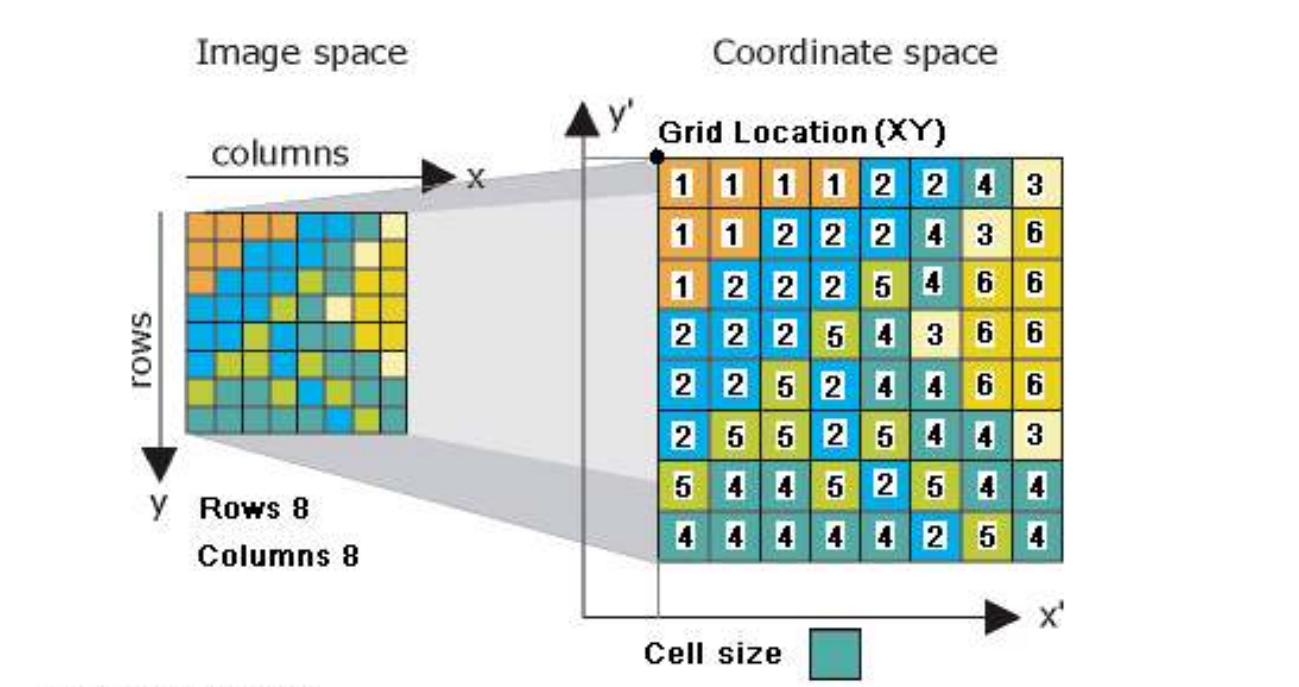
```
# nomes das camadas do stack
names(bioclim)
```

```
##  [1] "wc2.1_10m_bio_1"  "wc2.1_10m_bio_10" "wc2.1_10m_bio_11" "wc2.1_10m_bio_12" "wc2.1_10m_bio_13" "wc2.1_10m_bio
##  [8] "wc2.1_10m_bio_16" "wc2.1_10m_bio_17" "wc2.1_10m_bio_18" "wc2.1_10m_bio_19" "wc2.1_10m_bio_2"  "wc2.1_10m_bio
## [15] "wc2.1_10m_bio_5"  "wc2.1_10m_bio_6"  "wc2.1_10m_bio_7"  "wc2.1_10m_bio_8"  "wc2.1_10m_bio_9"
```

4. Descrição de objetos raster

Informações geográficas

Valores



[Raster basics - ESRI]

4. Descrição de objetos raster

Informações geográficas

Valores

```
# valores do raster
getValues(dem_rc) %>% head()
```

```
## [1] 859 856 856 856 853 852
```

```
# valores do raster
values(dem_rc) %>% head()
```

```
## [1] 859 856 856 856 853 852
```

```
# valores do raster
dem_rc[] %>% head()
```

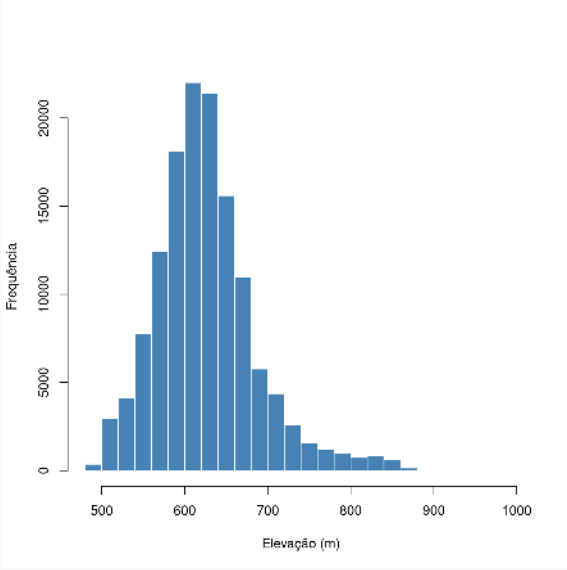
```
## [1] 859 856 856 856 853 852
```

4. Descrição de objetos raster

Informações geográficas

Valores

```
# valores do raster - histograma
dem_rc %>%
  raster::values() %>%
  hist(col = "steelblue", border = "white", main = NA, xlab = "Elevação (m)", ylab = "Frequência")
```



4. Descrição de objetos raster

Informações geográficas

Valores

```
# valores do stack
values(bioclim[[1:3]]) %>% head()
```

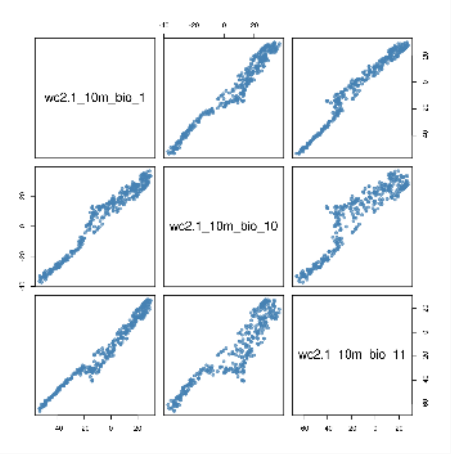
```
##      wc2.1_10m_bio_1 wc2.1_10m_bio_10 wc2.1_10m_bio_11
## [1,]              NA               NA               NA
## [2,]              NA               NA               NA
## [3,]              NA               NA               NA
## [4,]              NA               NA               NA
## [5,]              NA               NA               NA
## [6,]              NA               NA               NA
```

4. Descrição de objetos raster

Informações geográficas

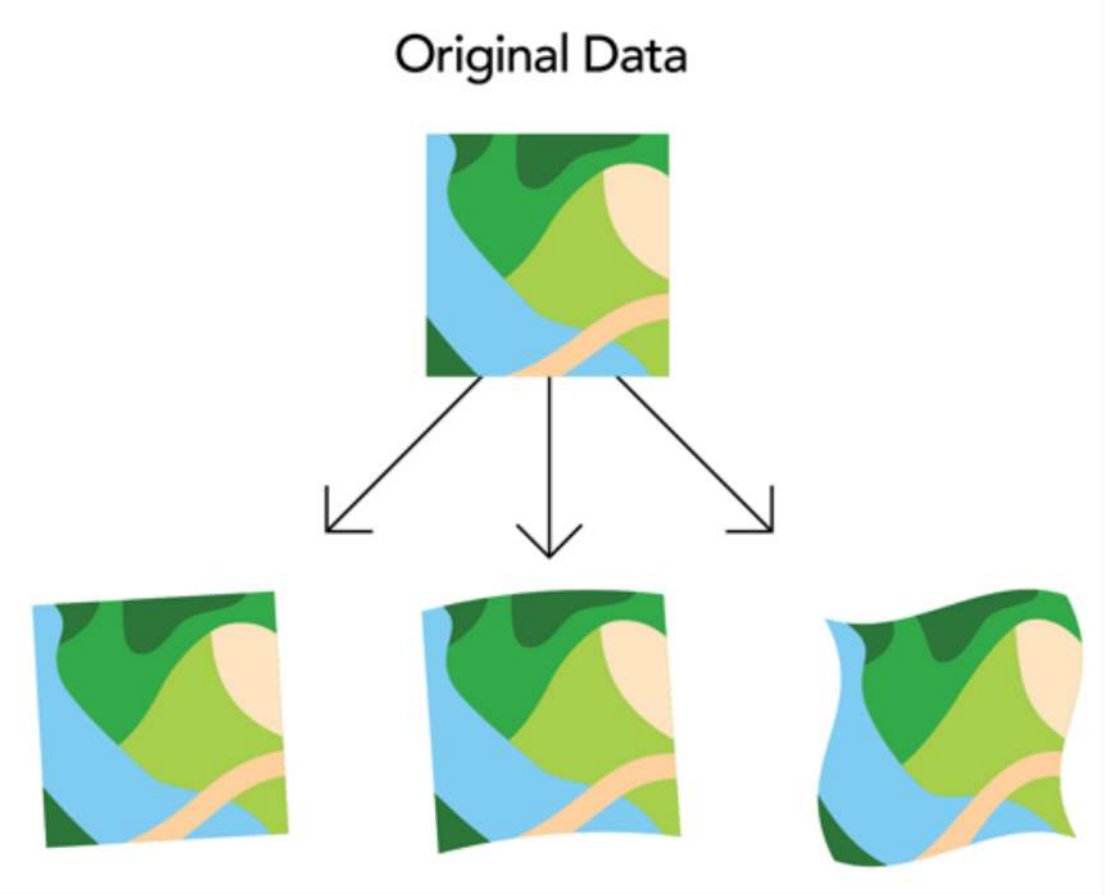
Valores

```
# valores do stack - grafico pareado
bioclim[[1:3]] %>%
  raster::values() %>%
  tibble::as_tibble() %>%
  dplyr::sample_n(1e3) %>%
  pairs(cex = 1.4, pch = 20, col = adjustcolor("steelblue", .7))
```



5. Converter CRS de objetos raster

Reprojeção

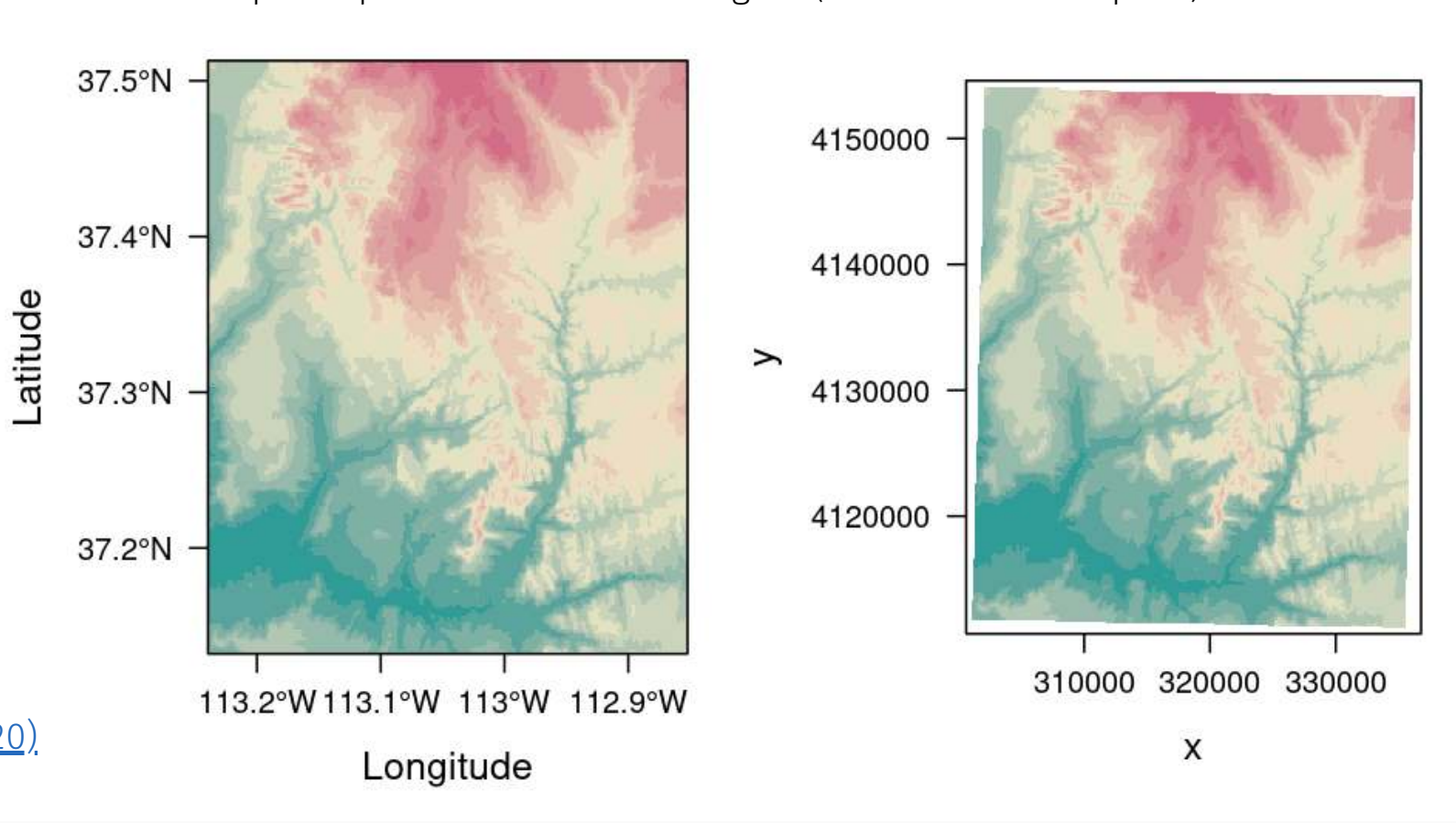


Hardware Implementation of the Video Polynomial Transformation + LPF

5. Converter CRS de objetos raster

Reprojeção

1. Reprojeção vetorial de centroides celulares (muda a **posição e tamanho** do pixel)
2. Cálculo de novos valores dos pixels por meio de reamostragem (muda o **valor** do pixel)



Lovelace et al. (2020)

5. Converter CRS de objetos raster

Converter CRS local

WGS84/GCS -> SIRGAS2000/UTM23s (proj4string)

```
# projecao do raster
raster::projection(dem_rc)
```

```
## [1] "+proj=longlat +datum=WGS84 +no_defs"
```

```
raster::crs(dem_rc)
```

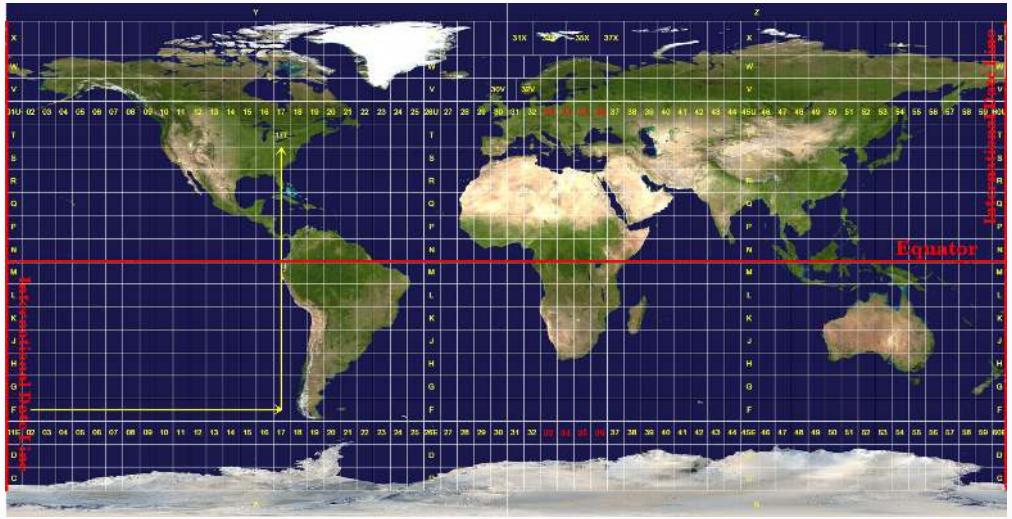
```
## CRS arguments: +proj=longlat +datum=WGS84 +no_defs
```

5. Converter CRS de objetos raster

Converter CRS local

WGS84/GCS -> SIRGAS2000/UTM23s (proj4string)

```
# proj4string utm 23s
utm23s ← "+init=epsg:31983"
```



EPSG: 32723 ESRI (2021)

5. Converter CRS de objetos raster

Converter CRS local

WGS84/GCS -> SIRGAS2000/UTM23s (proj4string)

```
# reprojecao
dem_rc_utm23s ← raster::projectRaster(dem_rc, crs = utm23s)
dem_rc_utm23s
```

```
## class      : RasterLayer
## dimensions : 386, 381, 147066  (nrow, ncol, ncell)
## resolution : 85.8, 92.3  (x, y)
## extent     : 214575.4, 247265.2, 7503009, 7538637  (xmin, xmax, ymin, ymax)
## crs        : +proj=utm +zone=23 +south +ellps=GRS80 +units=m +no_defs
## source     : memory
## names      : srtm_27_17
## values     : 491.6033, 980.4151  (min, max)
```

5. Converter CRS de objetos raster

Converter CRS local - especificando parâmetros

WGS84/GCS -> SIRGAS2000/UTM23s (proj4string)

```
# reprojecao
dem_rc_utm23s ← raster::projectRaster(dem_rc, crs = utm23s, res = 90, method = "bilinear")
dem_rc_utm23s
```

```
## class      : RasterLayer
## dimensions : 396, 364, 144144  (nrow, ncol, ncell)
## resolution : 90, 90  (x, y)
## extent     : 214554.4, 247314.4, 7502985, 7538625  (xmin, xmax, ymin, ymax)
## crs        : +proj=utm +zone=23 +south +ellps=GRS80 +units=m +no_defs
## source     : memory
## names      : srtm_27_17
## values     : 493.2395, 986.686  (min, max)
```

5. Converter CRS de objetos raster

Converter CRS local - especificando parâmetros

WGS84/GCS -> SIRGAS2000/UTM23s (proj4string)

```
# reprojecao de vector
rc_2020_utm23s ← sf::st_transform(rc_2020, crs = utm23s)
rc_2020_utm23s
```

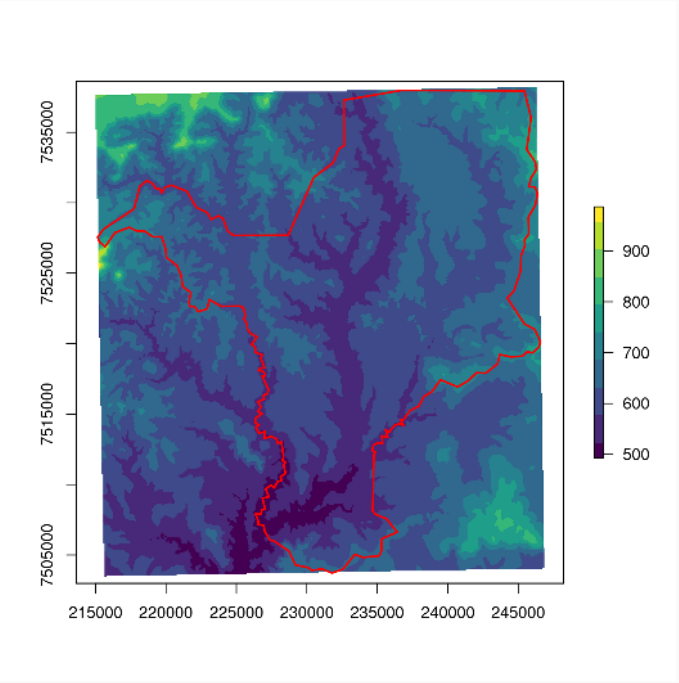
```
## Simple feature collection with 1 feature and 4 fields
## Geometry type: MULTIPOLYGON
## Dimension:     XY
## Bounding box:  xmin: 215151.7 ymin: 7503723 xmax: 246580.7 ymax: 7537978
## Projected CRS: SIRGAS 2000 / UTM zone 23S
##     code_muni name_muni code_state abbrev_state                           geom
## 493   3543907 Rio Claro         35           SP MULTIPOLYGON (((232640.1 75 ...
```

5. Converter CRS de objetos raster

Converter CRS local

WGS84/GCS -> SIRGAS2000/UTM23s (proj4string)

```
# plot
plot(dem_rc_utm23s, col = viridis::viridis(10))
plot(rc_2020_utm23s$geom, col = NA, border = "red", lwd = 2, add = TRUE)
```



5. Converter CRS de objetos raster

Converter CRS global

Datum WGS84 e coordenadas geográficas

```
# WGS84/GCS
bioclim$wc2.1_10m_bio_1
```

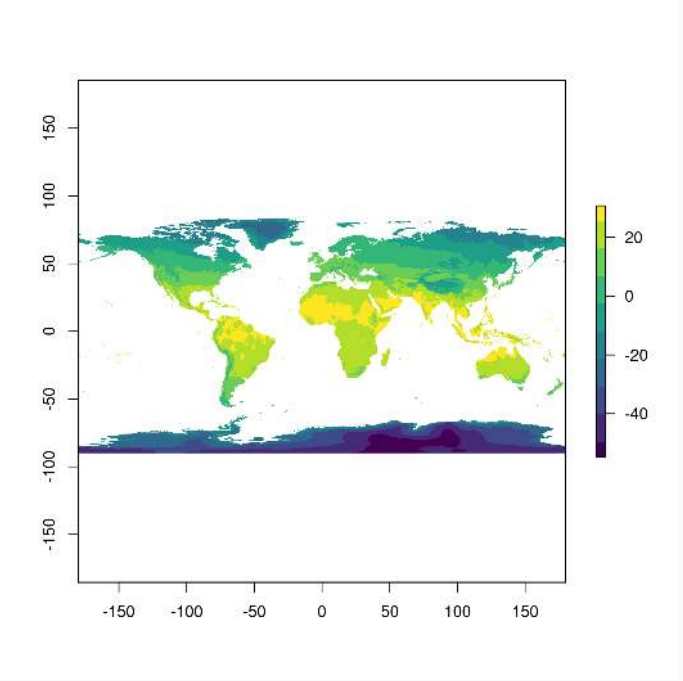
```
## class      : RasterLayer
## dimensions : 1080, 2160, 2332800  (nrow, ncol, ncell)
## resolution : 0.1666667, 0.1666667  (x, y)
## extent     : -180, 180, -90, 90  (xmin, xmax, ymin, ymax)
## crs        : +proj=longlat +datum=WGS84 +no_defs
## source     : wc2.1_10m_bio_1.tif
## names      : wc2.1_10m_bio_1
## values     : -54.72435, 30.98764  (min, max)
```

5. Converter CRS de objetos raster

Converter CRS global

Datum WGS84 e coordenadas geográficas

```
# plot
plot(bioclim$wc2.1_10m_bio_1, col = viridis::viridis(10))
```



5. Converter CRS de objetos raster

Converter CRS global

Projeção de Mollweide: preserva as relações de área

```
# proj4string mollweide
moll ← "+proj=moll"
```

EPSG: 54009 ESRI (2021)

5. Converter CRS de objetos raster

Converter CRS global

Projeção de Mollweide

```
# reprojecao
bio01_moll ← raster::projectRaster(bioclim$wc2.1_10m_bio_1, crs = moll, res = 25e3, method = "bilinear")
bio01_moll
```

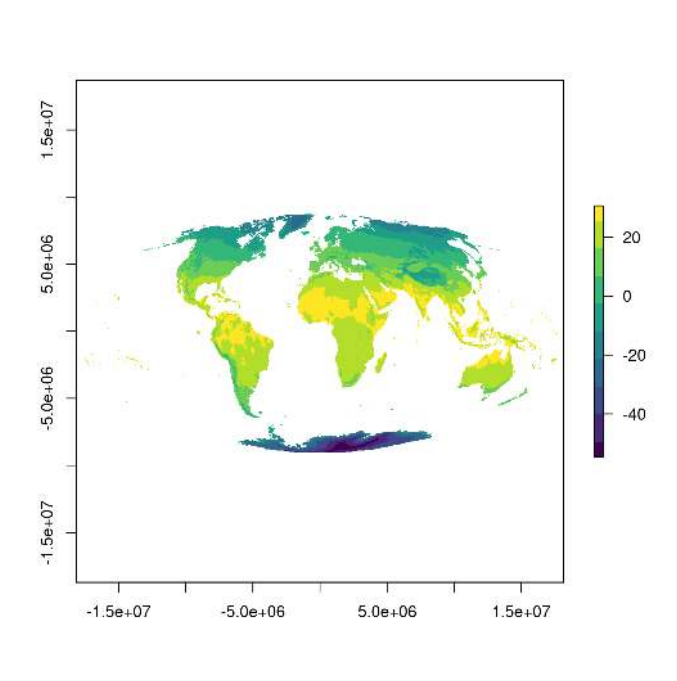
```
## class      : RasterLayer
## dimensions : 732, 1453, 1063596  (nrow, ncol, ncell)
## resolution : 25000, 25000  (x, y)
## extent     : -18159905, 18165095, -9154952, 9145048  (xmin, xmax, ymin, ymax)
## crs        : +proj=moll +lon_0=0 +x_0=0 +y_0=0 +datum=WGS84 +units=m +no_defs
## source     : memory
## names      : wc2.1_10m_bio_1
## values     : -54.66752, 30.71805  (min, max)
```

5. Converter CRS de objetos raster

Converter CRS global

Projeção de Mollweide

```
# plot
plot(bio01_moll, col = viridis::viridis(10))
```

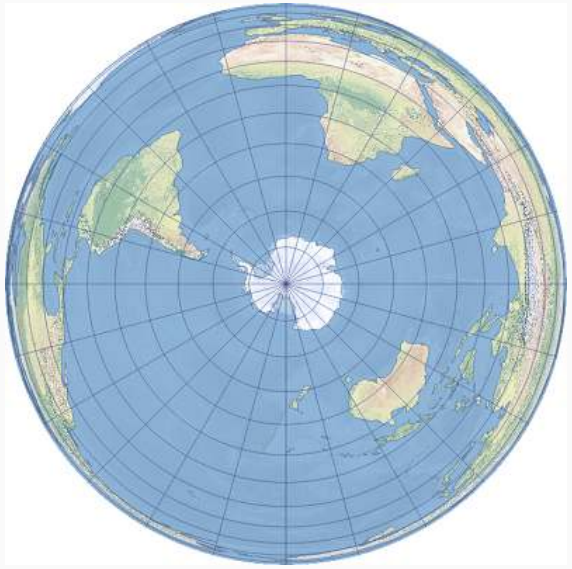


5. Converter CRS de objetos raster

Converter CRS global

Projeção Azimutal de Lambert: preserva os tamanhos e senso de direção (centro)

```
# proj4string lambert
laea ← "+proj=laea +x_0=0 +y_0=0 +lon_0=0 +lat_0=0"
```



ESRI(2021)

5. Converter CRS de objetos raster

Converter CRS global

Projeção Azimutal de Lambert: preserva os tamanhos e senso de direção (centro)

```
# reprojecao
bio01_laea ← raster::projectRaster(bioclim$wc2.1_10m_bio_1, crs = laea, res = 25e3, method = "bilinear")
bio01_laea
```

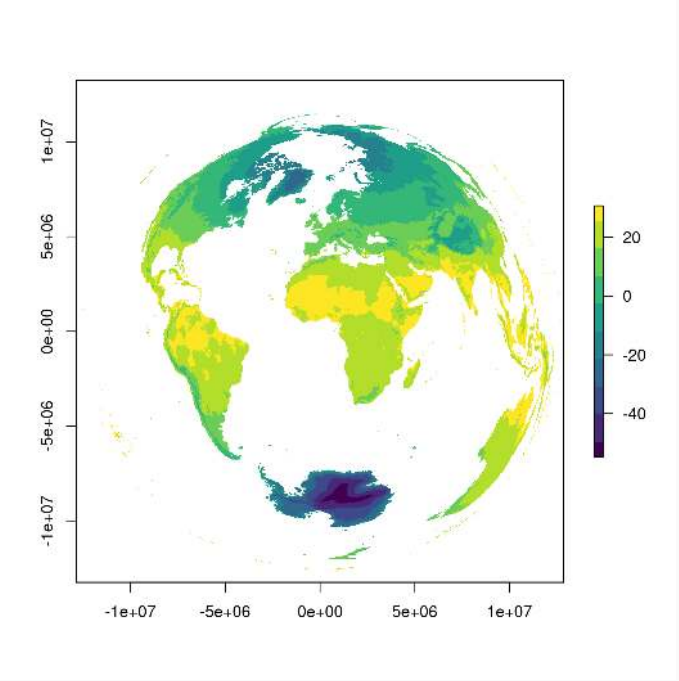
```
## class      : RasterLayer
## dimensions : 1028, 1029, 1057812  (nrow, ncol, ncell)
## resolution : 25000, 25000  (x, y)
## extent     : -12863885, 12861115, -12848994, 12851006  (xmin, xmax, ymin, ymax)
## crs        : +proj=laea +lat_0=0 +lon_0=0 +x_0=0 +y_0=0 +datum=WGS84 +units=m +no_defs
## source     : memory
## names      : wc2.1_10m_bio_1
## values     : -54.70442, 30.71119  (min, max)
```

5. Converter CRS de objetos raster

Converter CRS global

Projeção Azimutal de Lambert: preserva os tamanhos e senso de direção (centro)

```
# plot
plot(bio01_laea, col = viridis::viridis(10))
```



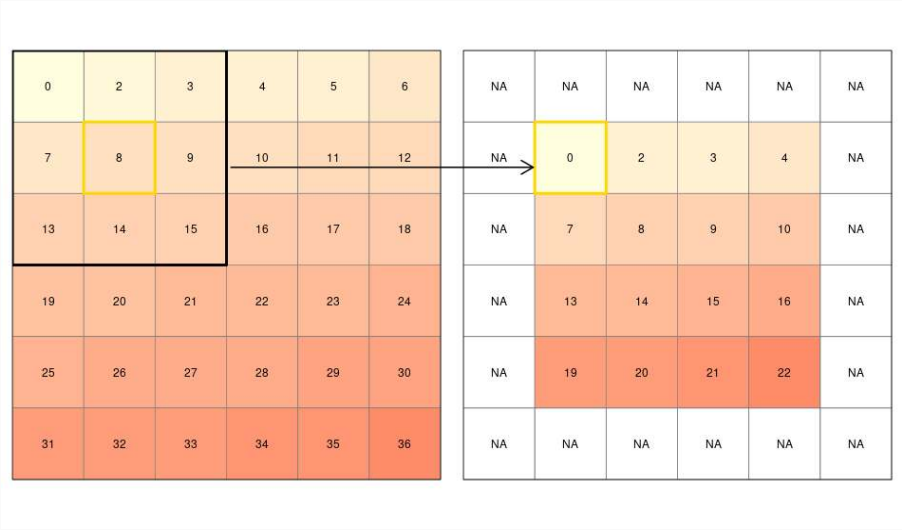
6. Operações de objetos raster

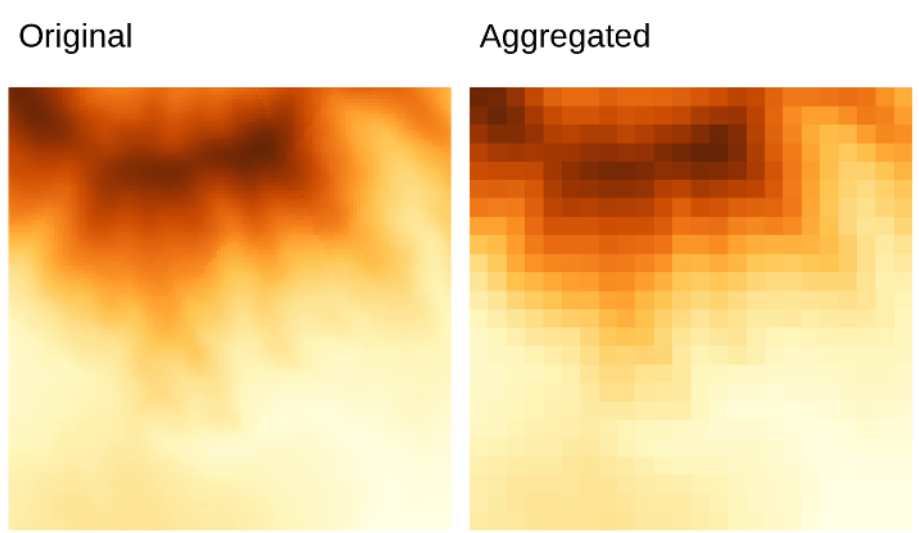
As operações podem ser separadas em três tipos

1. **Operações de atributos**: acessar valores, camadas e nome das camadas

2. **Operações espaciais**: aplicar regras espaciais para mudar valores dos pixels

3. **Operações geométricas**: mudar a posição, tamanho e número dos pixels





Lovelace et al. (2020)

6. Operações de objetos raster

1. Operações de atributos

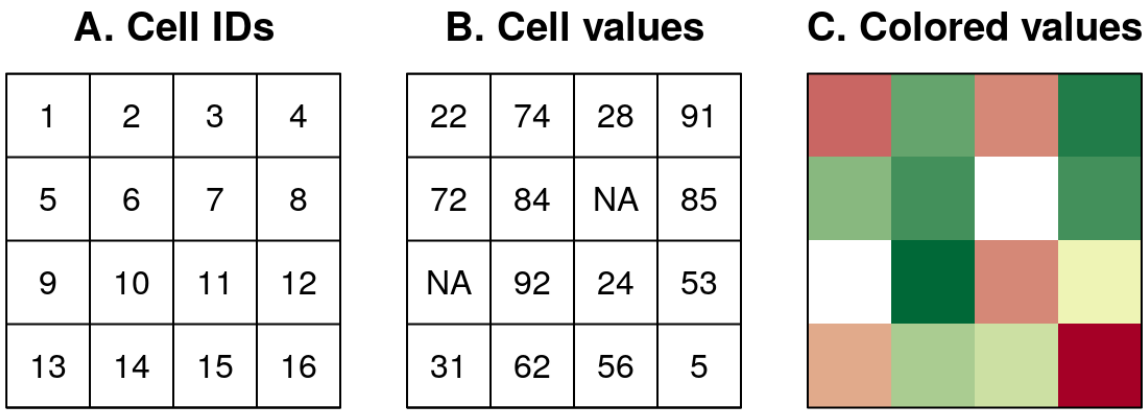
Modificação de objetos *raster* baseado em **informações não espaciais**, como **valores das células e nome das camadas**

Operações

1.1. Subconjunto de células ou camadas

1.2. Renomear camadas

1.3. Estatística das células



Lovelace et al. (2020)

1. Operações de atributos

1.1. Subconjunto - Seleção de valores

```
# raster - linha 1 e coluna 1
dem_rc[1, 1]
```

```
## [1] 859
```

```
# celula 1
dem_rc[1]
```

```
## [1] 859
```

6. Operações de objetos raster

1. Operações de atributos

1.1. Subconjunto - Seleção de valores

```
# stack - linha 1 e coluna 1
bioclim[1, 1]
```

```
##      wc2.1_10m_bio_1 wc2.1_10m_bio_10 wc2.1_10m_bio_11 wc2.1_10m_bio_12 wc2.1_10m_bio_13 wc2.1_10m_bio_14 wc2.1_10
## [1,]              NA                NA                NA                NA                NA                NA
##      wc2.1_10m_bio_17 wc2.1_10m_bio_18 wc2.1_10m_bio_19 wc2.1_10m_bio_2 wc2.1_10m_bio_3 wc2.1_10m_bio_4 wc2.1_10m_
## [1,]               NA               NA               NA              NA              NA              NA
##      wc2.1_10m_bio_7 wc2.1_10m_bio_8 wc2.1_10m_bio_9
## [1,]              NA              NA              NA
```

6. Operações de objetos raster

1. Operações de atributos

1.1. Subconjunto - Seleção de valores

```
# celula 1
bioclim[1]
```

```
##      wc2.1_10m_bio_1 wc2.1_10m_bio_10 wc2.1_10m_bio_11 wc2.1_10m_bio_12 wc2.1_10m_bio_13 wc2.1_10m_bio_14 wc2.1_10
## [1,]              NA                NA                NA                NA                NA                NA
##      wc2.1_10m_bio_17 wc2.1_10m_bio_18 wc2.1_10m_bio_19 wc2.1_10m_bio_2 wc2.1_10m_bio_3 wc2.1_10m_bio_4 wc2.1_10m_
## [1,]               NA                NA                NA               NA               NA               NA
##      wc2.1_10m_bio_7 wc2.1_10m_bio_8 wc2.1_10m_bio_9
## [1,]              NA               NA               NA
```

6. Operações de objetos raster

1. Operações de atributos

1.1. Subconjunto - Seleção de camadas

```
# selecao de camada num objeto stack utilizando a funcao subset
bioclim_bio01 ← raster::subset(bioclim, "wc2.1_10m_bio_1")
bioclim_bio01
```

```
## class      : RasterLayer
## dimensions : 1080, 2160, 2332800  (nrow, ncol, ncell)
## resolution : 0.1666667, 0.1666667  (x, y)
## extent     : -180, 180, -90, 90  (xmin, xmax, ymin, ymax)
## crs        : +proj=longlat +datum=WGS84 +no_defs
## source     : wc2.1_10m_bio_1.tif
## names      : wc2.1_10m_bio_1
## values     : -54.72435, 30.98764  (min, max)
```

6. Operações de objetos raster

1. Operações de atributos

1.1. Subconjunto - Seleção de camadas

```
# selecao de camada num objeto stack utilizando a funcao raster
bioclim_bio01 ← raster::raster(bioclim, layer = 1)
bioclim_bio01
```

```
## class      : RasterLayer
## dimensions : 1080, 2160, 2332800  (nrow, ncol, ncell)
## resolution : 0.1666667, 0.1666667  (x, y)
## extent     : -180, 180, -90, 90  (xmin, xmax, ymin, ymax)
## crs        : +proj=longlat +datum=WGS84 +no_defs
## source     : wc2.1_10m_bio_1.tif
## names      : wc2.1_10m_bio_1
## values     : -54.72435, 30.98764  (min, max)
```

6. Operações de objetos raster

1. Operações de atributos

1.1. Subconjunto - Seleção de camadas

```
# selecao de camada num objeto stack utilizando os operadores [[]] e o nome
bioclim_bio01 ← bioclim[["wc2.1_10m_bio_1"]]
bioclim_bio01
```

```
## class      : RasterLayer
## dimensions : 1080, 2160, 2332800  (nrow, ncol, ncell)
## resolution : 0.1666667, 0.1666667  (x, y)
## extent     : -180, 180, -90, 90  (xmin, xmax, ymin, ymax)
## crs        : +proj=longlat +datum=WGS84 +no_defs
## source     : wc2.1_10m_bio_1.tif
## names      : wc2.1_10m_bio_1
## values     : -54.72435, 30.98764  (min, max)
```

6. Operações de objetos raster

1. Operações de atributos

1.1. Subconjunto - Seleção de camadas

```
# selecao de camada num objeto stack utilizando os operadores [[]] e a posicao
bioclim_bio01 ← bioclim[[1]]
bioclim_bio01
```

```
## class      : RasterLayer
## dimensions : 1080, 2160, 2332800  (nrow, ncol, ncell)
## resolution : 0.1666667, 0.1666667  (x, y)
## extent     : -180, 180, -90, 90  (xmin, xmax, ymin, ymax)
## crs        : +proj=longlat +datum=WGS84 +no_defs
## source     : wc2.1_10m_bio_1.tif
## names      : wc2.1_10m_bio_1
## values     : -54.72435, 30.98764  (min, max)
```

6. Operações de objetos raster

1. Operações de atributos

1.1. Subconjunto - Seleção de camadas

```
# selecao de camada num objeto stack utilizando o operador $
bioclim_bio01 ← bioclim$wc2.1_10m_bio_1
bioclim_bio01
```

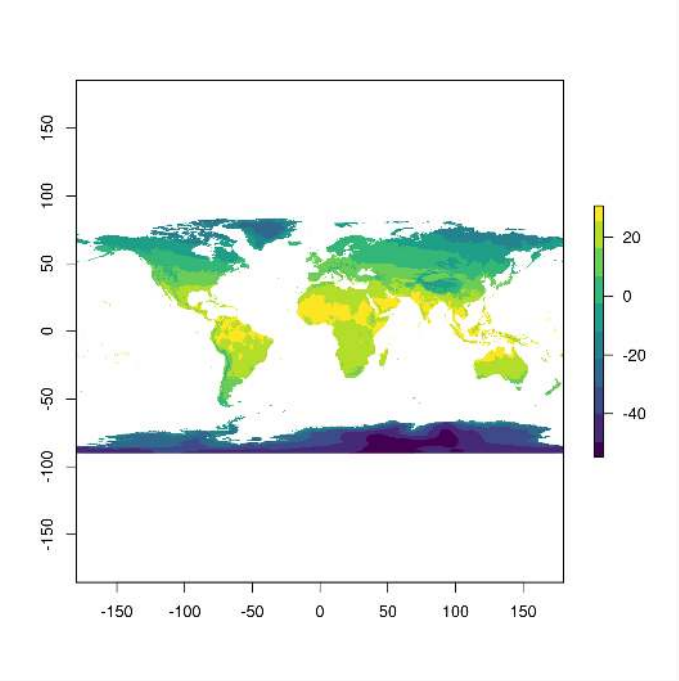
```
## class      : RasterLayer
## dimensions : 1080, 2160, 2332800  (nrow, ncol, ncell)
## resolution : 0.1666667, 0.1666667  (x, y)
## extent     : -180, 180, -90, 90  (xmin, xmax, ymin, ymax)
## crs        : +proj=longlat +datum=WGS84 +no_defs
## source     : wc2.1_10m_bio_1.tif
## names      : wc2.1_10m_bio_1
## values     : -54.72435, 30.98764  (min, max)
```

6. Operações de objetos raster

1. Operações de atributos

1.1. Subconjunto - Seleção de camadas

```
# plot
plot(bioclim_bio01, col = viridis::viridis(10))
```



1. Operações de atributos

1.2. Renomear camadas

```
# nomes
names(dem_rc)
```

```
## [1] "srtm_27_17"
```

```
# renomear
names(dem_rc) ← "elevation"
```

```
# nomes
names(dem_rc)
```

```
## [1] "elevation"
```

6. Operações de objetos raster

1. Operações de atributos

1.2. Renomear camadas

```
# nomes
names(bioclim)
```

```
##  [1] "wc2.1_10m_bio_1"  "wc2.1_10m_bio_10" "wc2.1_10m_bio_11" "wc2.1_10m_bio_12" "wc2.1_10m_bio_13" "wc2.1_10m_bio
##  [8] "wc2.1_10m_bio_16" "wc2.1_10m_bio_17" "wc2.1_10m_bio_18" "wc2.1_10m_bio_19" "wc2.1_10m_bio_2"  "wc2.1_10m_bio
## [15] "wc2.1_10m_bio_5"  "wc2.1_10m_bio_6"  "wc2.1_10m_bio_7"  "wc2.1_10m_bio_8"  "wc2.1_10m_bio_9"
```

```
# renomear
names(bioclim) ← c("bio01", paste0("bio", 10:19), paste0("bio0", 2:9))
```

```
# nomes
names(bioclim)
```

```
##  [1] "bio01" "bio10" "bio11" "bio12" "bio13" "bio14" "bio15" "bio16" "bio17" "bio18" "bio19" "bio02" "bio03" "bio0
## [18] "bio08" "bio09"
```

6. Operações de objetos raster

1. Operações de atributos

1.3. Estatística das células - Média

```
# media de todas as celulas de altitude
raster::cellStats(x = dem_rc, stat = mean)
```

```
## [1] 625.8273
```

```
# media de todas as celulas de cada camada bioclimatica
raster::cellStats(x = bioclim, stat = mean)
```

```
##       bio01       bio10       bio11       bio12       bio13       bio14       bio15       bio16       bio17
##  -4.0378283   7.2035545 -13.8963286 550.0569022  93.4633916  15.3689993  74.7084151 241.6525005  55.4149542 156.42
##       bio03       bio04       bio05       bio06       bio07       bio08       bio09
##  34.5221528 880.1215546  13.9386423 -19.7938943  33.7325366  -0.9226276  -5.3774489
```

1. Operações de atributos

1.3. Estatística das células - Frequência

```
# frequencia das celulas
raster::freq(x = dem_rc) %>% head()
```

```
##      value count
## [1,]   491     1
## [2,]   492     4
## [3,]   493     9
## [4,]   494    19
## [5,]   495    32
## [6,]   496    44
```

1. Operações de atributos

1.3. Estatística das células - Frequência

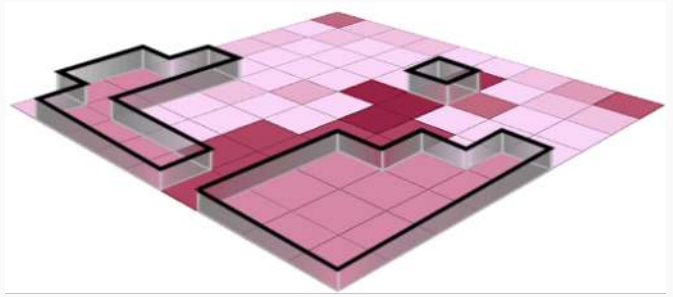
```
# frequencia das celulas
raster::freq(x = bioclim[[1]]) %>% head()
```

```
##      value count
## [1,]   -55   319
## [2,]   -54  4529
## [3,]   -53  5778
## [4,]   -52  6128
## [5,]   -51  6090
## [6,]   -50  7892
```

6. Operações de objetos raster

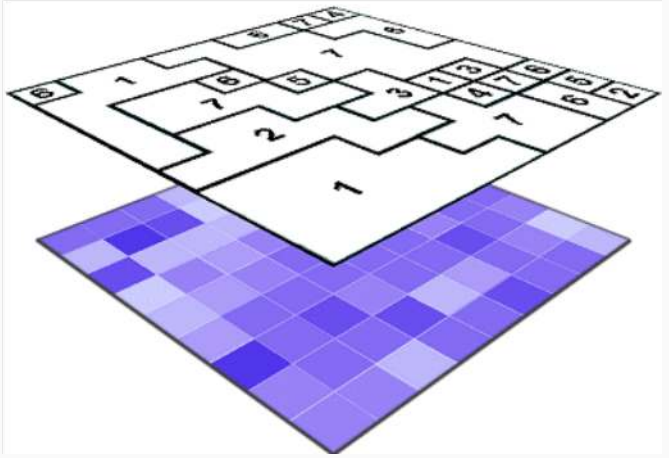
2. Operações espaciais

Aplicação de **regras espaciais** para **mudar valores dos pixels**, geralmente com o intuito de **resumir as informações das células**



Operações

2.1. Operações locais (por célula)
2.2. Operações focais (por bloco de múltiplas células regulares - e.g. 3x3)
2.3. Operações zonais (por bloco de múltiplas células irregulares)
2.4. Operações globais (por um ou vários rasters inteiros)



The Zonal Statistics function

2. Operações espaciais

2.1. Operações locais - Álgebra de raster

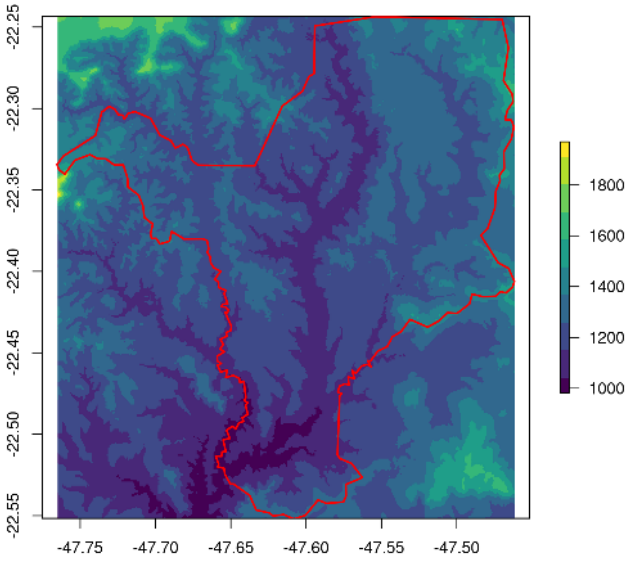
```
# soma
dem_rc2 ← dem_rc + dem_rc
dem_rc2
```

```
## class      : RasterLayer
## dimensions : 370, 364, 134680  (nrow, ncol, ncell)
## resolution : 0.0008333333, 0.0008333333  (x, y)
## extent     : -47.765, -47.46167, -22.55167, -22.24333  (xmin, xmax, ymin, ymax)
## crs        : +proj=longlat +datum=WGS84 +no_defs
## source     : memory
## names      : layer
## values     : 982, 1970  (min, max)
```

2. Operações espaciais

2.1. Operações locais - Álgebra de raster

```
# plot
plot(dem_rc2, col = viridis::viridis(10))
plot(rc_2020$geom, col = NA, border = "red", lwd = 2, add = TRUE)
```



2. Operações espaciais

2.1. Operações locais - Álgebra de raster

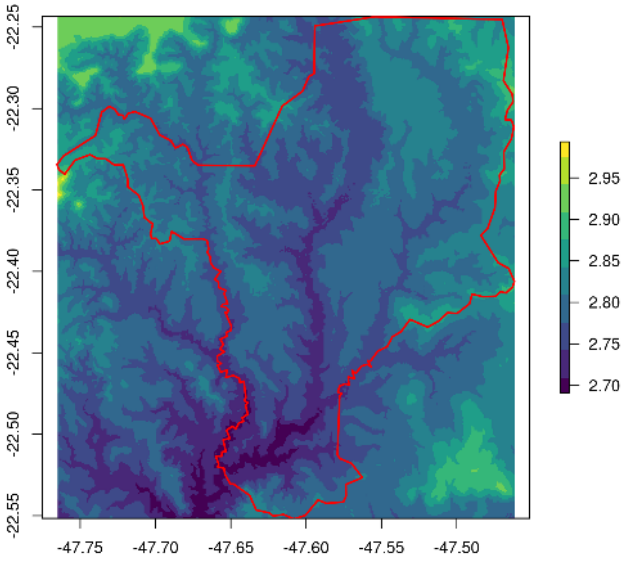
```
# log10
dem_rc_log10 ← log10(dem_rc)
dem_rc_log10
```

```
## class      : RasterLayer
## dimensions : 370, 364, 134680  (nrow, ncol, ncell)
## resolution : 0.0008333333, 0.0008333333  (x, y)
## extent     : -47.765, -47.46167, -22.55167, -22.24333  (xmin, xmax, ymin, ymax)
## crs        : +proj=longlat +datum=WGS84 +no_defs
## source     : memory
## names      : layer
## values     : 2.691081, 2.993436  (min, max)
```

2. Operações espaciais

2.1. Operações locais - Álgebra de raster

```
# plot
plot(dem_rc_log10, col = viridis::viridis(10))
plot(rc_2020$geom, col = NA, border = "red", lwd = 2, add = TRUE)
```



2. Operações espaciais

2.1. Operações locais - Álgebra de raster - Operador lógico

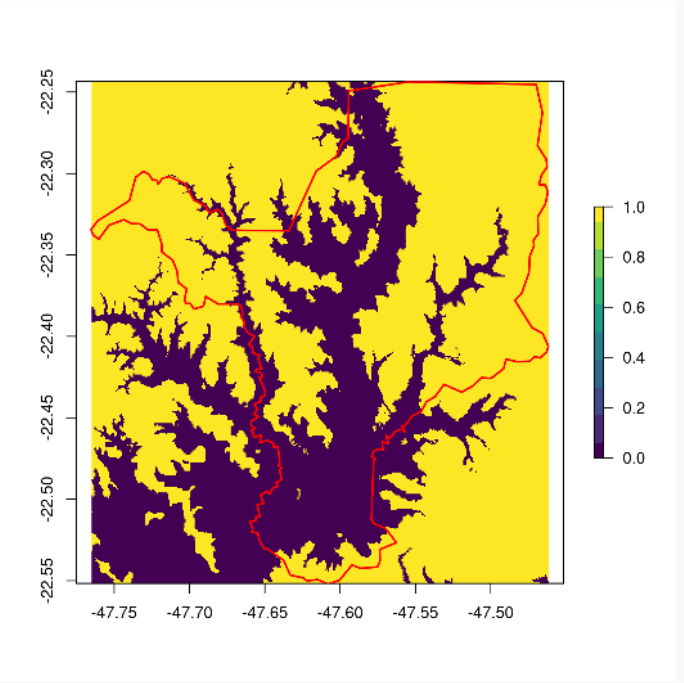
```
# acima de 600
dem_rc_acima_600 ← dem_rc > 600
dem_rc_acima_600
```

```
## class      : RasterLayer
## dimensions : 370, 364, 134680  (nrow, ncol, ncell)
## resolution : 0.0008333333, 0.0008333333  (x, y)
## extent     : -47.765, -47.46167, -22.55167, -22.24333  (xmin, xmax, ymin, ymax)
## crs        : +proj=longlat +datum=WGS84 +no_defs
## source     : memory
## names      : layer
## values     : 0, 1  (min, max)
```

2. Operações espaciais

2.1. Operações locais - Álgebra de raster - Operador lógico

```
# plot
plot(dem_rc_acima_600, col = viridis::viridis(10))
plot(rc_2020$geom, col = NA, border = "red", lwd = 2, add = TRUE)
```



2. Operações espaciais

2.1. Operações locais - Álgebra de raster - Função `calc()`

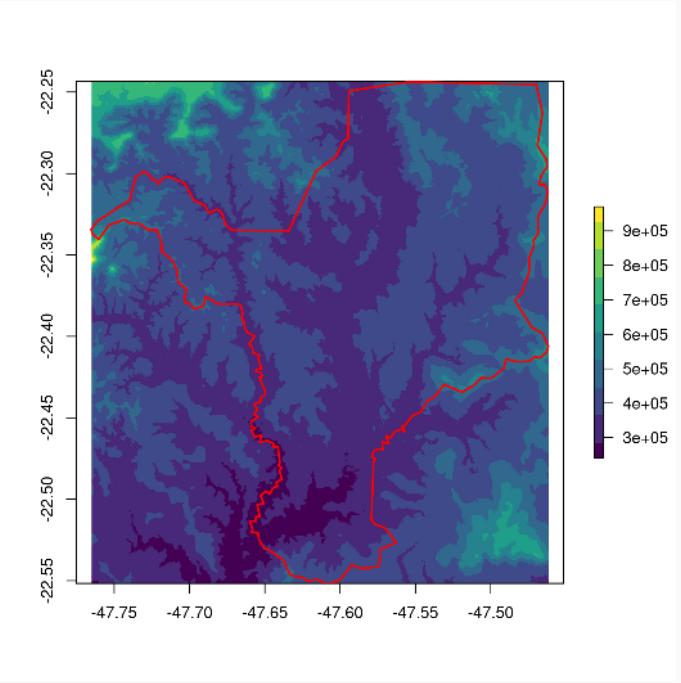
```
# produto dos pixel - calc
dem_rc_prod ← raster::calc(x = dem_rc, fun = function(x){x * x})
dem_rc_prod
```

```
## class      : RasterLayer
## dimensions : 370, 364, 134680  (nrow, ncol, ncell)
## resolution : 0.0008333333, 0.0008333333  (x, y)
## extent     : -47.765, -47.46167, -22.55167, -22.24333  (xmin, xmax, ymin, ymax)
## crs        : +proj=longlat +datum=WGS84 +no_defs
## source     : memory
## names      : layer
## values     : 241081, 970225  (min, max)
```

2. Operações espaciais

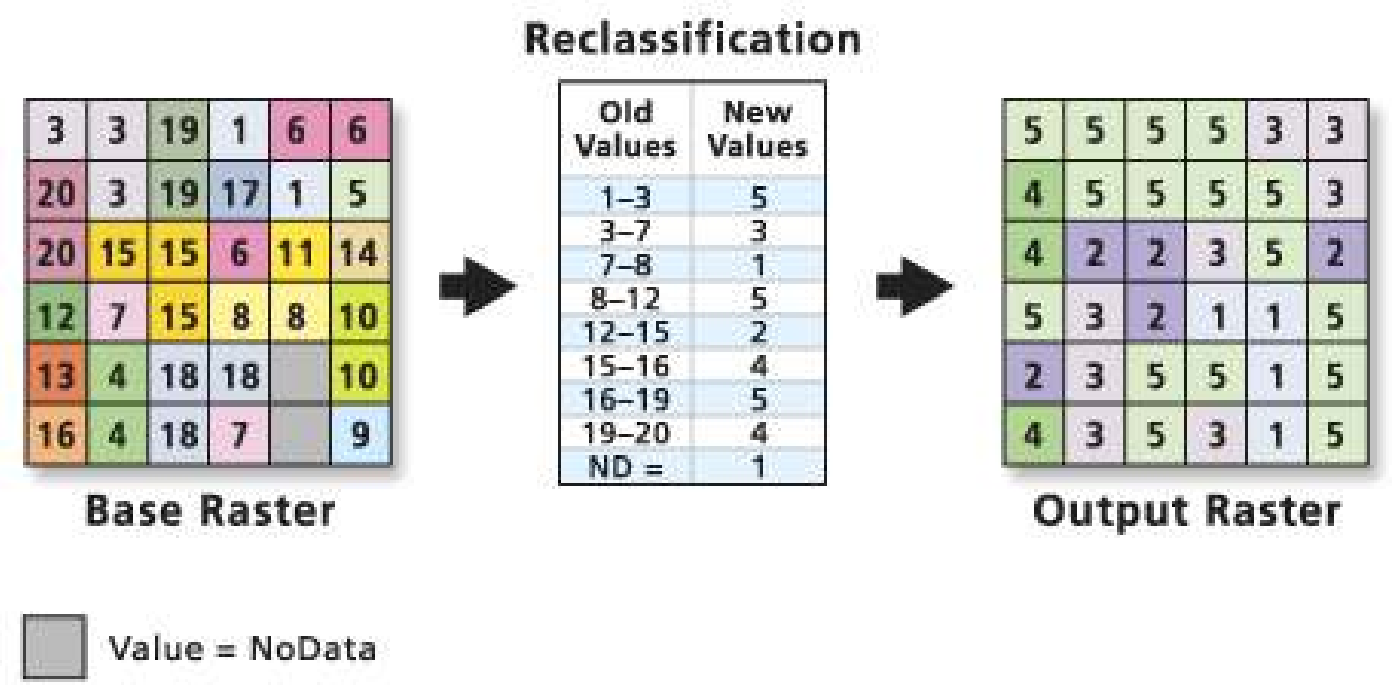
2.1. Operações locais - Álgebra de raster - Função `calc()`

```
# plot
plot(dem_rc_prod, col = viridis::viridis(10))
plot(rc_2020$geom, col = NA, border = "red", lwd = 2, add = TRUE)
```



2. Operações espaciais

2.1. Operações locais - Reclassificação



[Reclass by individual values - ESRI]

2. Operações espaciais

- Reclassificação

```
# matriz de reclassificacao
rcl  ← matrix(
  c(400, 600, 1,
    600, 800, 2,
    800, 1000, 3),
  ncol = 3, byrow = TRUE)
rcl
```

```
##      [,1] [,2] [,3]
## [1,]  400  600    1
## [2,]  600  800    2
## [3,]  800 1000    3
```

2. Operações espaciais

2.1. Operações locais - Reclassificação

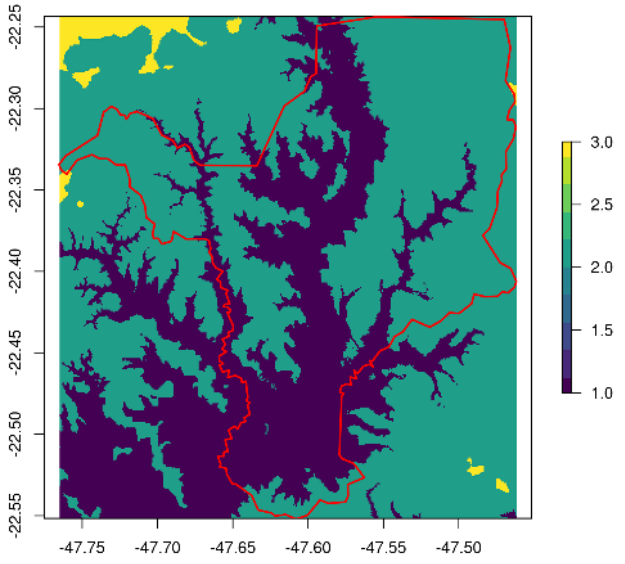
```
# reclassificao
dem_rc_rcl ← raster::reclassify(x = dem_rc, rcl = rcl)
dem_rc_rcl
```

```
## class      : RasterLayer
## dimensions : 370, 364, 134680  (nrow, ncol, ncell)
## resolution : 0.0008333333, 0.0008333333  (x, y)
## extent     : -47.765, -47.46167, -22.55167, -22.24333  (xmin, xmax, ymin, ymax)
## crs        : +proj=longlat +datum=WGS84 +no_defs
## source     : memory
## names      : elevation
## values     : 1, 3  (min, max)
```

2. Operações espaciais

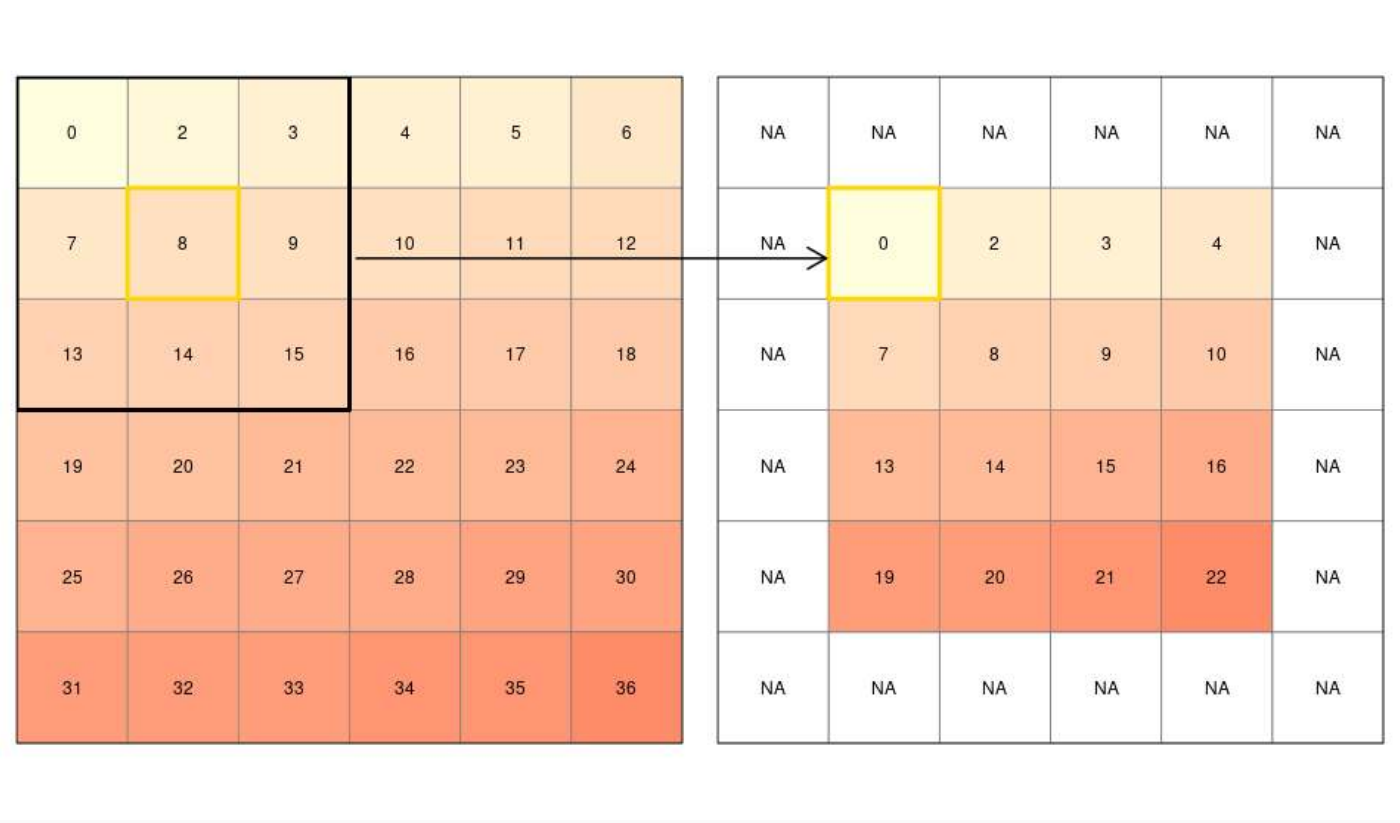
2.1. Operações locais - Reclassificação

```
# plot
plot(dem_rc_rcl, col = viridis::viridis(10))
plot(rc_2020$geom, col = NA, border = "red", lwd = 2, add = TRUE)
```



2. Operações espaciais

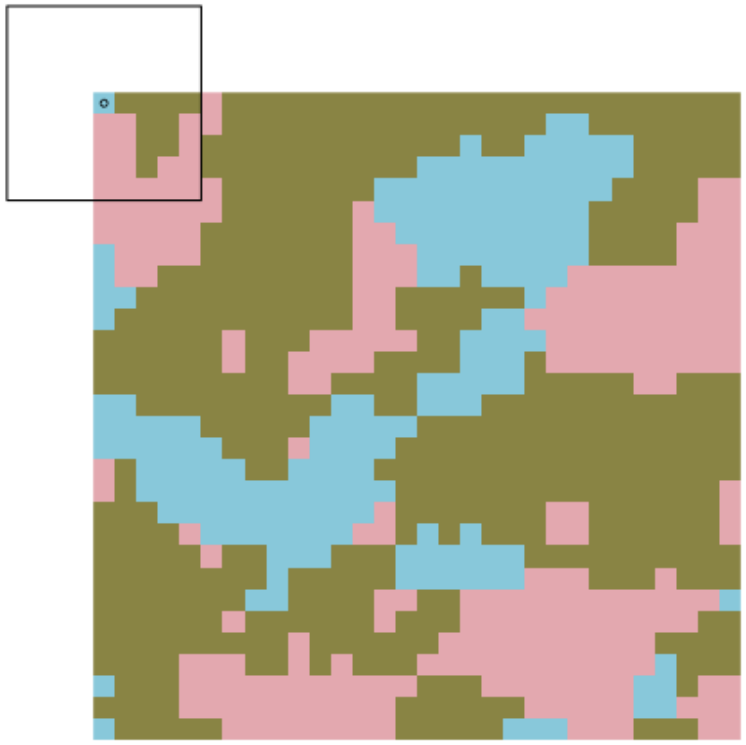
2.2 Operação focal - Janela móvel (Moving window)



Lovelace et al. (2020)

2. Operações espaciais

2.2 Operação focal - Janela móvel (Moving window)



Nowosad & Hesselbarth (2020)

2. Operações espaciais

2.2 Operação focal - Janela móvel (Moving window)

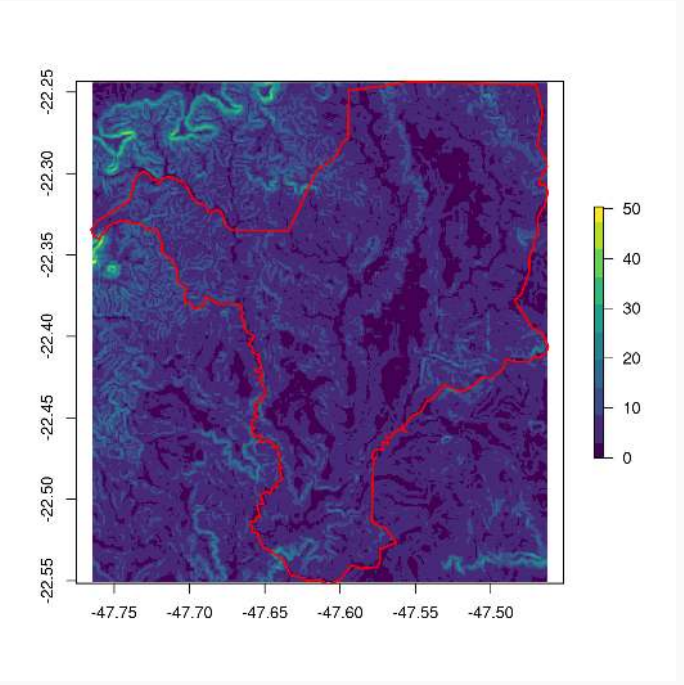
```
# janela movel
dem_rc_focal_sd ← raster::focal(x = dem_rc, w = matrix(data = 1, nrow = 3, ncol = 3), fun = sd)
dem_rc_focal_sd
```

```
## class      : RasterLayer
## dimensions : 370, 364, 134680  (nrow, ncol, ncell)
## resolution : 0.0008333333, 0.0008333333  (x, y)
## extent     : -47.765, -47.46167, -22.55167, -22.24333  (xmin, xmax, ymin, ymax)
## crs        : +proj=longlat +datum=WGS84 +no_defs
## source     : memory
## names      : layer
## values     : 0, 50.38959  (min, max)
```

2. Operações espaciais

2.2 Operação focal - Janela móvel (Moving window)

```
# plot
plot(dem_rc_focal_sd, col = viridis::viridis(10))
plot(rc_2020$geom, col = NA, border = "red", lwd = 2, add = TRUE)
```



2. Operações espaciais

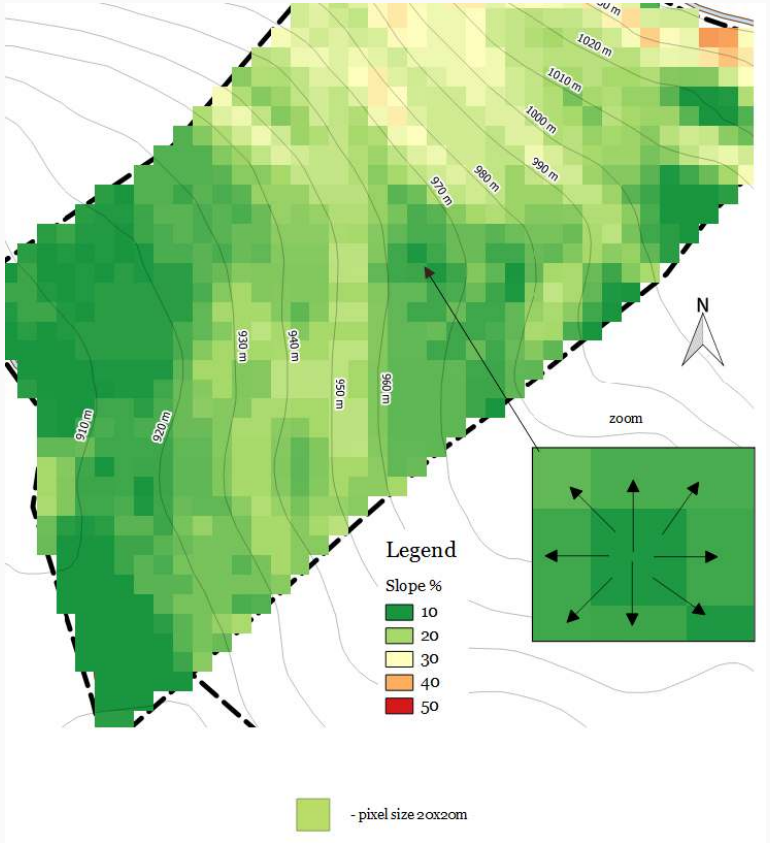
Declividade (Slope)

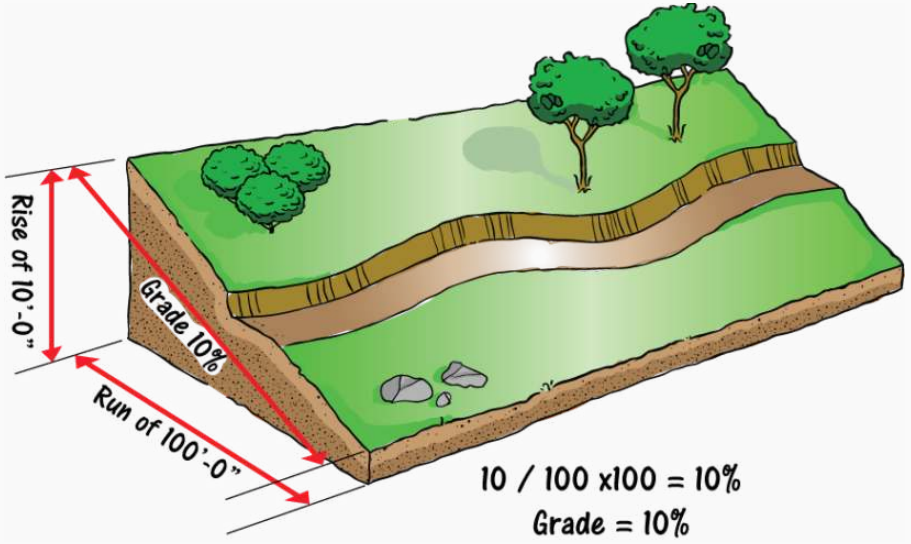
```
# declividade
dem_rc_dec ← raster::terrain(x = dem_rc, opt = "slope", unit = "degrees")
dem_rc_dec
```

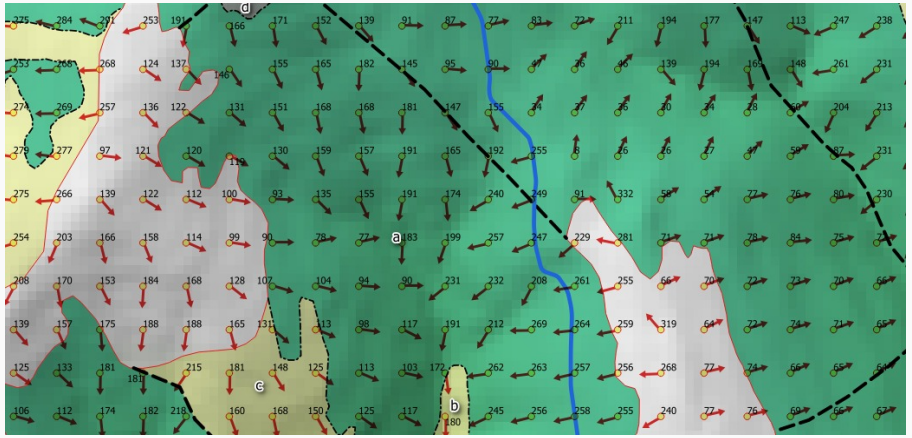
```
## class      : RasterLayer
## dimensions : 370, 364, 134680  (nrow, ncol, ncell)
## resolution : 0.0008333333, 0.0008333333  (x, y)
## extent     : -47.765, -47.46167, -22.55167, -22.24333  (xmin, xmax, ymin, ymax)
## crs        : +proj=longlat +datum=WGS84 +no_defs
## source     : memory
## names      : slope
## values     : 0, 33.18868  (min, max)
```

2. Operações espaciais

Declividade (Slope)



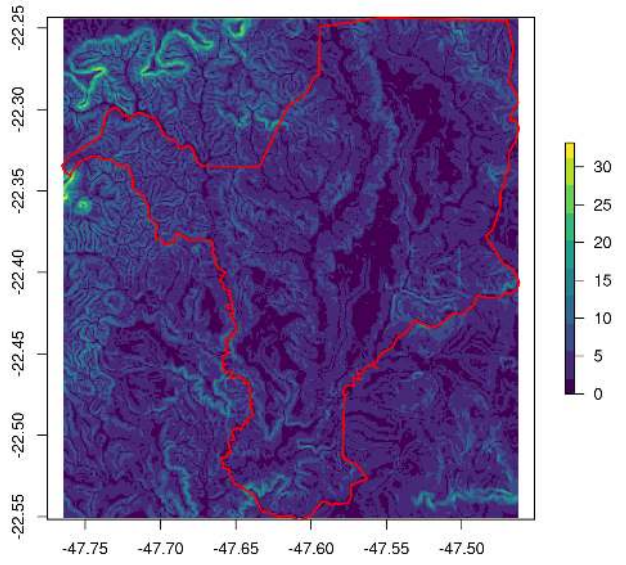




2. Operações espaciais

Declividade (Slope)

```
# plot
plot(dem_rc_dec, col = viridis::viridis(10))
plot(rc_2020$geom, col = NA, border = "red", lwd = 2, add = TRUE)
```



2. Operações espaciais

2.3. Operações zonais - Estatística por zonas

```
# estatistica zonal
dem_rc_zonal ← data.frame(raster::zonal(dem_rc, dem_rc_rcl, fun = "summary"))
dem_rc_zonal
```

```
##   zone value_1 value_2 value_3  value_4 value_5 value_6
## 1    1     491     552   574.0 567.5995     589     600
## 2    2     601     620   640.0 650.6829     670     800
## 3    3     801     817   832.5 834.2732     846     985
```

2. Operações espaciais

2.3. Operações zonais - Estatística por zonas

```
# estatistica zonal
colnames(dem_rc_zonal) ← c("zona", "min", "1qt", "mediana", "media", "3qt", "max")
dem_rc_zonal
```

```
##   zona min 1qt mediana    media 3qt max
## 1    1 491 552   574.0 567.5995 589 600
## 2    2 601 620   640.0 650.6829 670 800
## 3    3 801 817   832.5 834.2732 846 985
```

2. Operações espaciais

2.4. Operações globais - Distância Euclideana

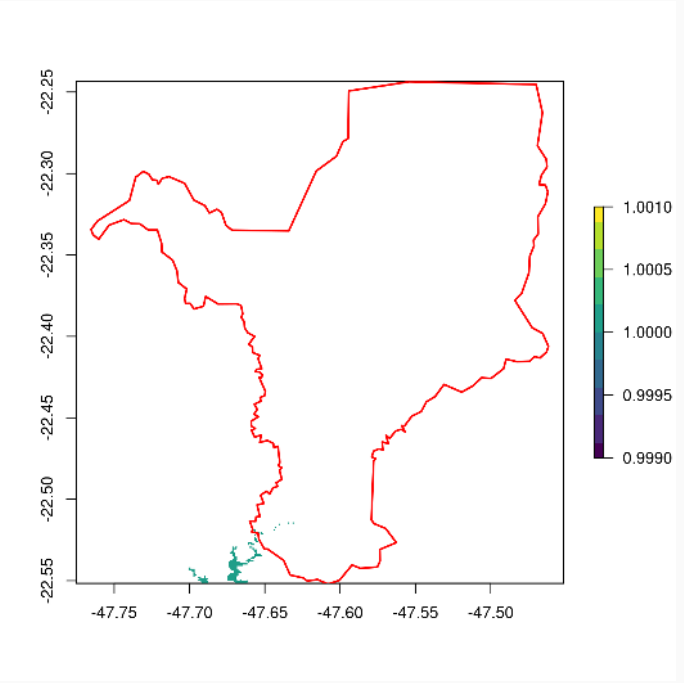
```
# reclassificacao
dem_rc_abaixo_500 ← raster::calc(x = dem_rc, fun = function(x) ifelse(x < 500, 1, NA))
dem_rc_abaixo_500
```

```
## class      : RasterLayer
## dimensions : 370, 364, 134680  (nrow, ncol, ncell)
## resolution : 0.0008333333, 0.0008333333  (x, y)
## extent     : -47.765, -47.46167, -22.55167, -22.24333  (xmin, xmax, ymin, ymax)
## crs        : +proj=longlat +datum=WGS84 +no_defs
## source     : memory
## names      : layer
## values     : 1, 1  (min, max)
```

2. Operações espaciais

2.4. Operações globais - Distância Euclideana

```
# plot
plot(dem_rc_abaixo_500, col = viridis::viridis(10))
plot(rc_2020$geom, col = NA, border = "red", lwd = 2, add = TRUE)
```



2. Operações espaciais

2.4. Operações globais - Distância Euclideana

```
# distancia euclideana
dem_rc_global_dist ← raster::distance(dem_rc_abaixo_500)
dem_rc_global_dist
```

```
## class      : RasterLayer
## dimensions : 370, 364, 134680  (nrow, ncol, ncell)
## resolution : 0.0008333333, 0.0008333333  (x, y)
## extent     : -47.765, -47.46167, -22.55167, -22.24333  (xmin, xmax, ymin, ymax)
## crs        : +proj=longlat +datum=WGS84 +no_defs
## source     : memory
## names      : layer
## values     : 0, 34683.79  (min, max)
```

2. Operações espaciais

2.4. Operações globais - Distância Euclideana

```
# plot
plot(dem_rc_global_dist, col = viridis::viridis(10))
plot(dem_rc_abaixo_500, add = TRUE, col = "white", legend = FALSE)
plot(rc_2020$geom, col = NA, border = "red", lwd = 2, add = TRUE)
```

3. Operações geométricas

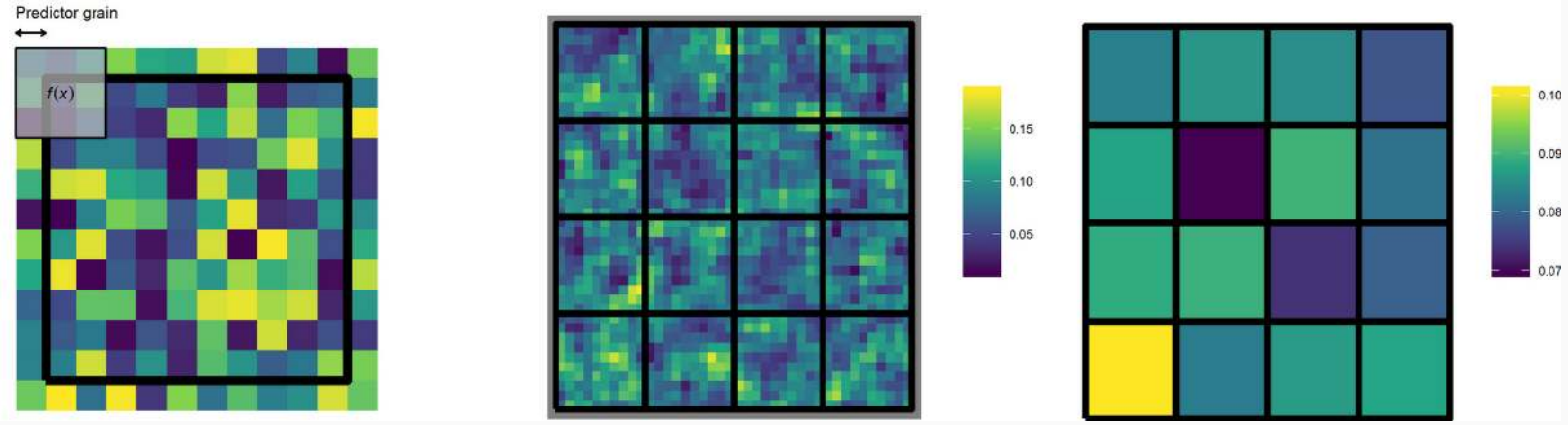
Envolve mudar a **posição, tamanho e número dos pixels** e **atribuir novos valores**, geralmente aumentando ou diminuindo o tamanho desses pixels

Operações

3.1. Agregação

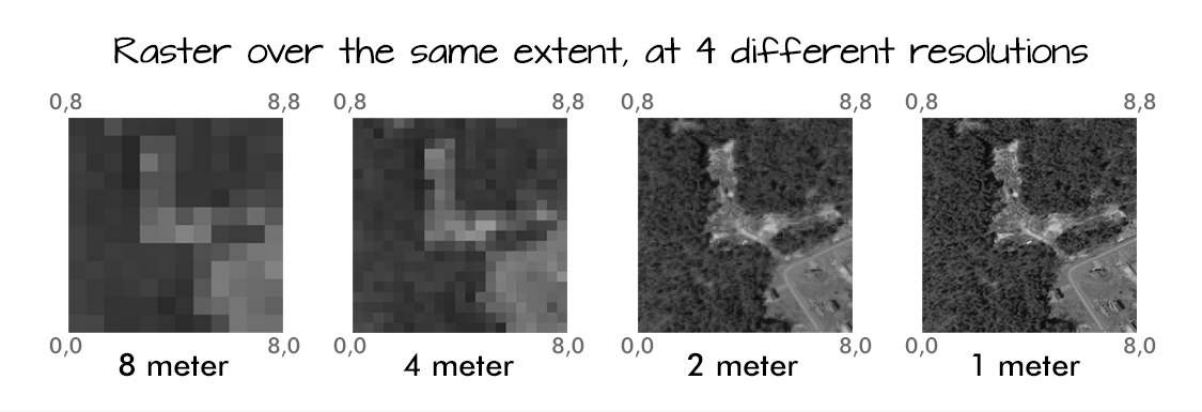
3.2. Desagregação

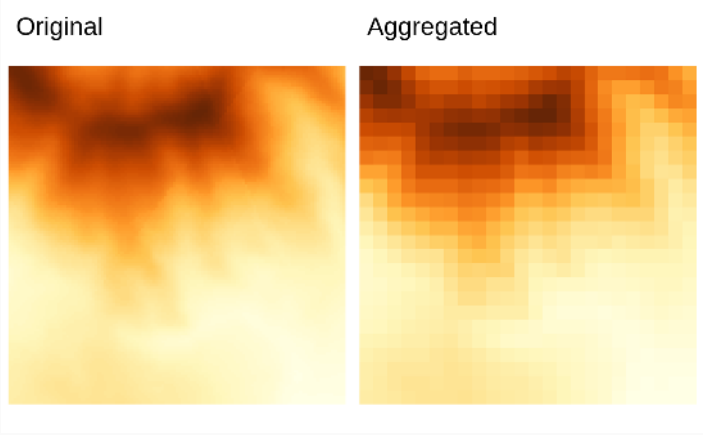
3.3. Alinhamento



3. Operações geométricas

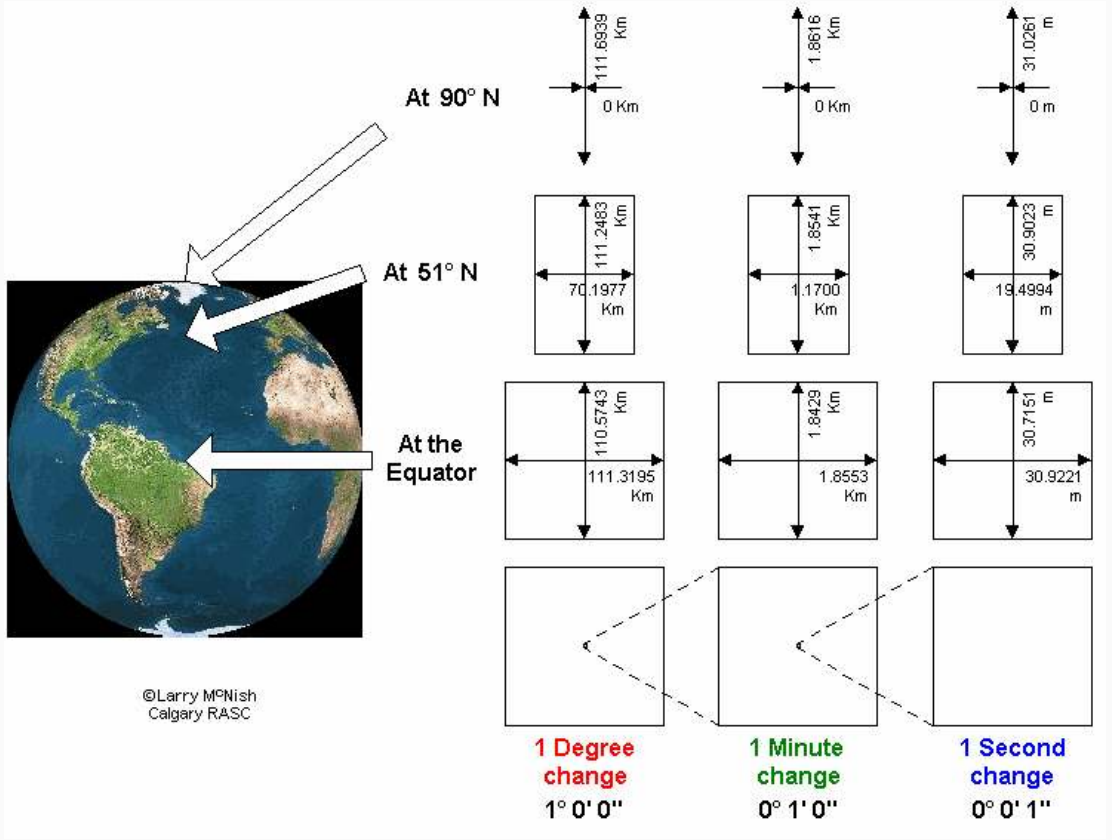
Agregação e desagregação





3. Operações geométricas

Resolução - Relação de graus e distância em metros



3. Operações geométricas

3.1 Agregação - Aumentar o tamanho do pixel

```
# resolucao
res(dem_rc_utm23s)[1]
```

```
## [1] 90
```

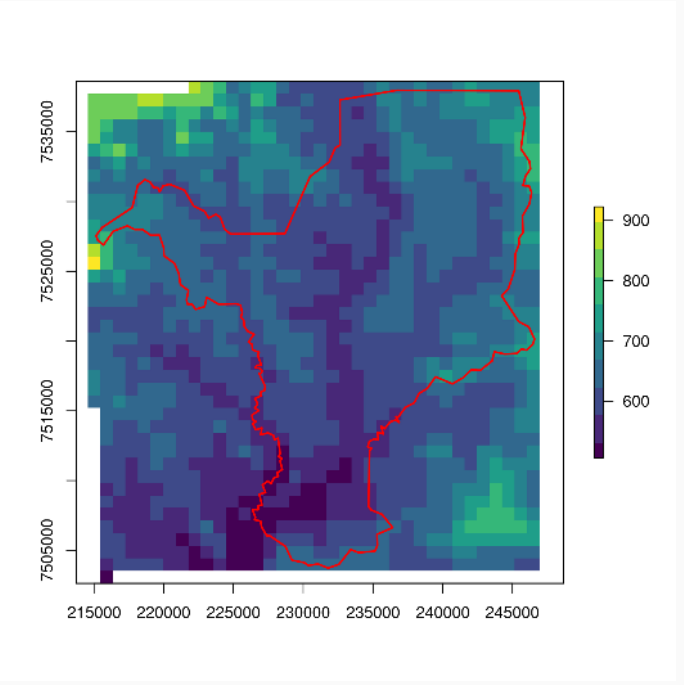
```
# agregacao - aumentar o tamanho do pixel
dem_rc_utm23s_agre_media ← raster::aggregate(x = dem_rc_utm23s, fact = 10, fun = "mean")
dem_rc_utm23s_agre_media
```

```
## class      : RasterLayer
## dimensions : 40, 37, 1480  (nrow, ncol, ncell)
## resolution : 900, 900  (x, y)
## extent     : 214554.4, 247854.4, 7502625, 7538625  (xmin, xmax, ymin, ymax)
## crs        : +proj=utm +zone=23 +south +ellps=GRS80 +units=m +no_defs
## source     : memory
## names      : srtm_27_17
## values     : 506.0024, 922.8709  (min, max)
```

3. Operações geométricas

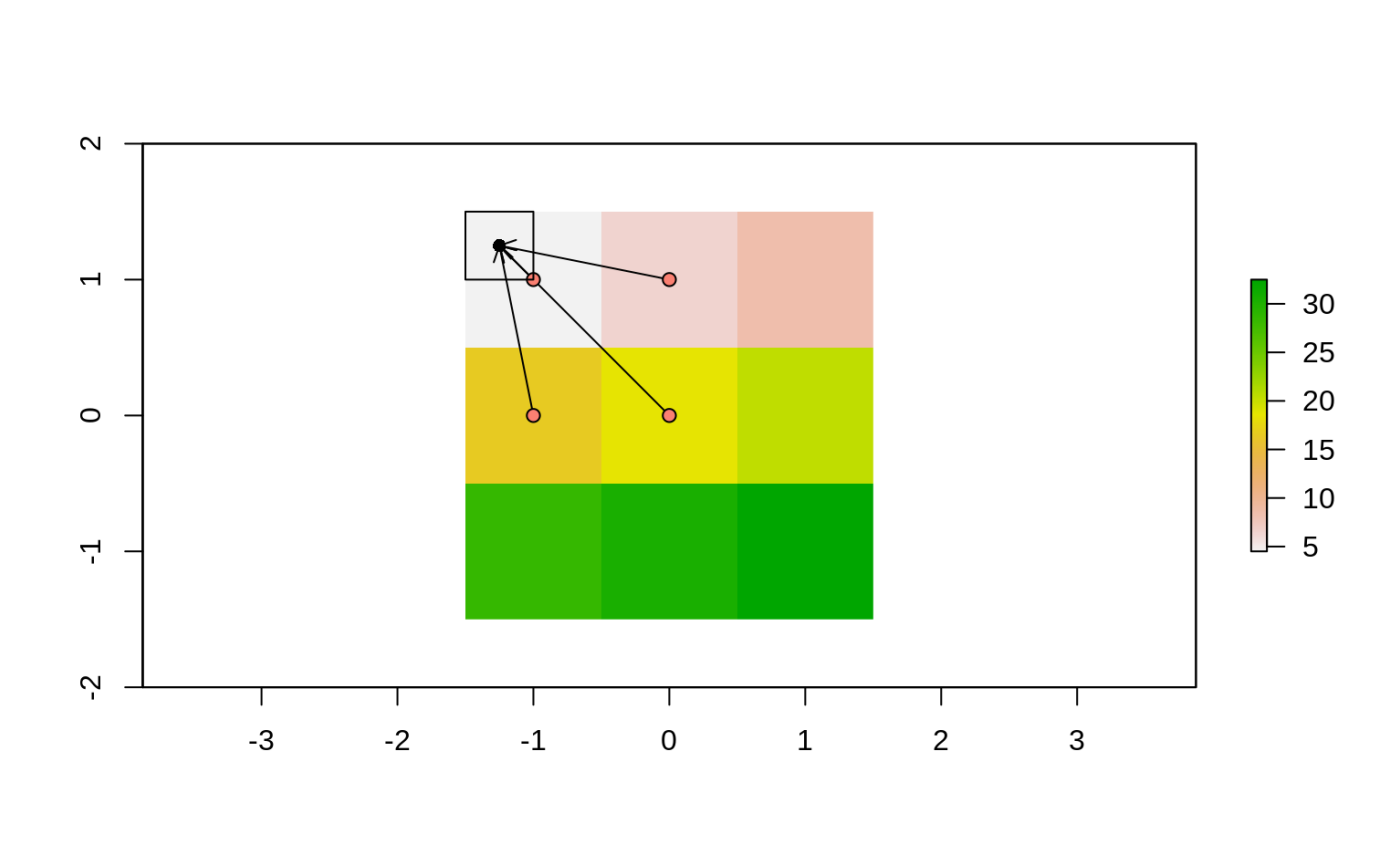
3.1. Agregação - Aumentar o tamanho do pixel

```
# plot
plot(dem_rc_utm23s_agre_media, col = viridis::viridis(10))
plot(rc_2020_utm23s$geom, col = NA, border = "red", lwd = 2, add = TRUE)
```



3. Operações geométricas

3.2 Desagregação - Diminuir o tamanho do pixel



3. Operações geométricas

3.2. Desagregação - Diminuir o tamanho do pixel

```
# resolucao
res(dem_rc_utm23s)[1]
```

```
## [1] 90
```

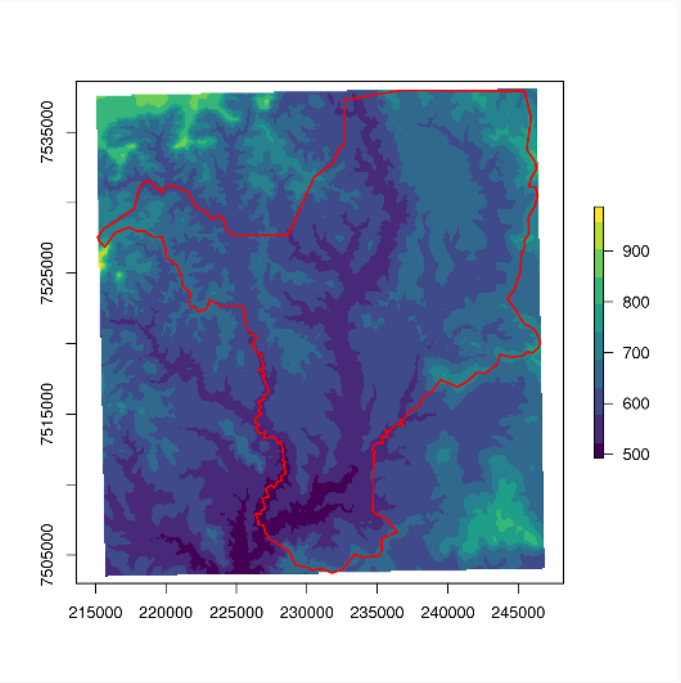
```
# desagregacao - diminuir o tamanho do pixel
dem_rc_utm23s_dis_bil ← raster::disaggregate(dem_rc_utm23s, fact = 2, fun = "bilinear")
dem_rc_utm23s_dis_bil
```

```
## class      : RasterLayer
## dimensions : 792, 728, 576576  (nrow, ncol, ncell)
## resolution : 45, 45  (x, y)
## extent     : 214554.4, 247314.4, 7502985, 7538625  (xmin, xmax, ymin, ymax)
## crs        : +proj=utm +zone=23 +south +ellps=GRS80 +units=m +no_defs
## source     : memory
## names      : srtm_27_17
## values     : 493.2395, 986.686  (min, max)
```

3. Operações geométricas

3.2. Desagregação - Diminuir o tamanho do pixel

```
# plot
plot(dem_rc_utm23s_dis_bil, col = viridis::viridis(10))
plot(rc_2020_utm23s$geom, col = NA, border = "red", lwd = 2, add = TRUE)
```



3. Operações geométricas

3.3. Alinhamento - Ajustar extensão, número e origem dos pixels

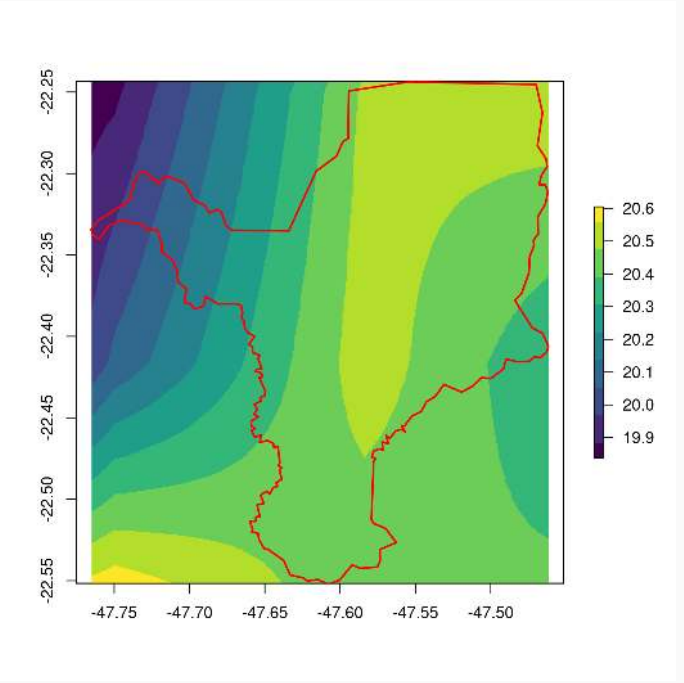
```
# reamostragem
st_rc ← raster::resample(x = bioclim$bio01, y = dem_rc, method = "bilinear")
st_rc
```

```
## class      : RasterLayer
## dimensions : 370, 364, 134680  (nrow, ncol, ncell)
## resolution : 0.0008333333, 0.0008333333  (x, y)
## extent     : -47.765, -47.46167, -22.55167, -22.24333  (xmin, xmax, ymin, ymax)
## crs        : +proj=longlat +datum=WGS84 +no_defs
## source     : memory
## names      : bio01
## values     : 19.8383, 20.60492  (min, max)
```

3. Operações geométricas

3.3. Alinhamento - Ajustar extensão, número e origem dos pixels

```
# plot
plot(st_rc, col = viridis::viridis(10))
plot(rc_2020$geom, col = NA, border = "red", lwd = 2, add = TRUE)
```



7. Interação raster-vetor

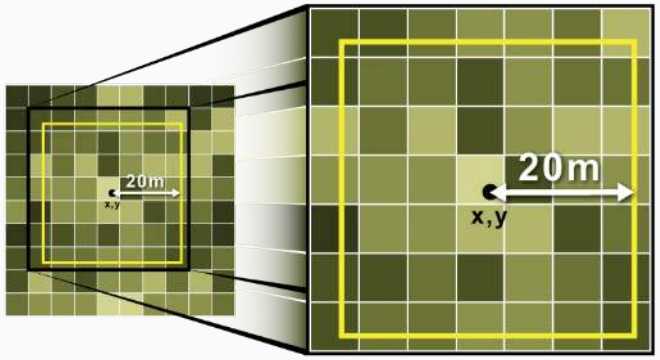
Podemos fazer operações da **interação entre objetos vetoriais e raster**, com diferentes finalidades, que podem ser diferentes tipos de vetores

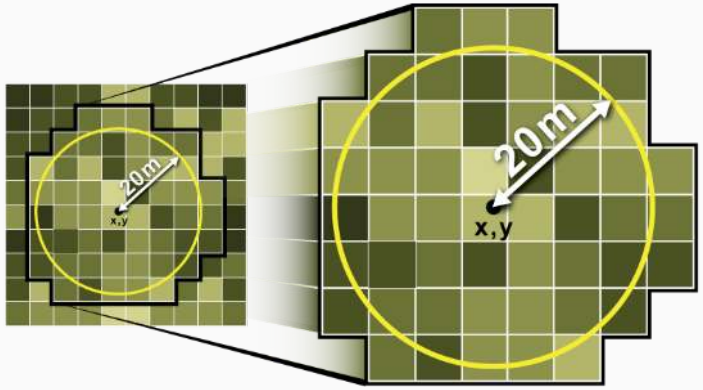
Interações

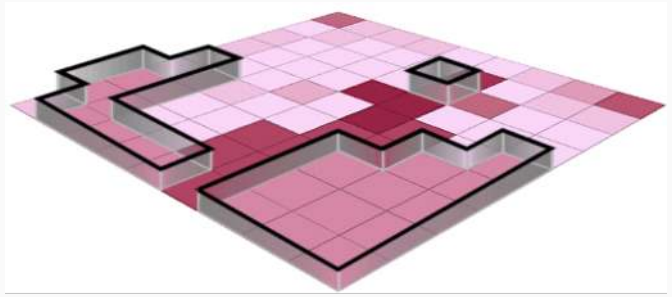
1. Corte e máscara: ajuste da extensão e limite

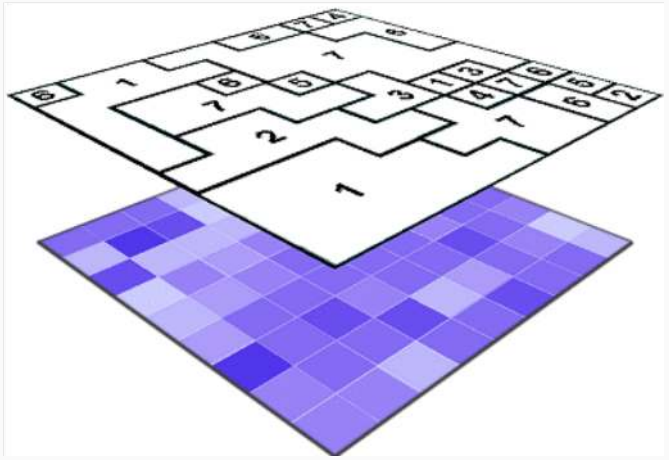
2. Extração dos valores dos pixels para vetores

3. Estatísticas zonais para um vetor







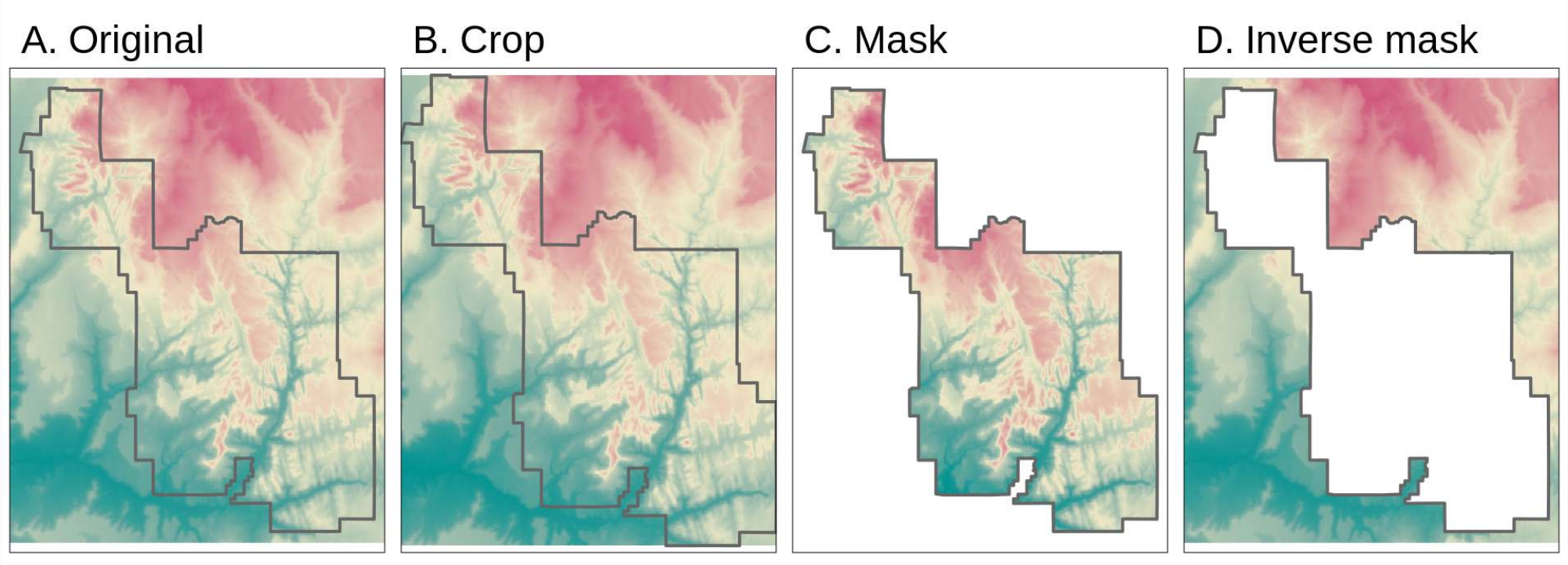


National Ecological Observatory Network (NEON), The Zonal Statistics function

7. Interação raster-vetor

1. Cortes e máscaras

Ajustar o **tamanho de um objeto raster** a uma **área menor de interesse**, geralmente definido por um **objeto vetorial**



Lovelace et al. (2020)

7. Interação raster-vetor

1. Cortes e máscaras

Crop - Ajusta a extensão

```
# crop - ajustar da extensao
dem_rc_crop ← raster::crop(dem, rc_2020)
dem_rc_crop
```

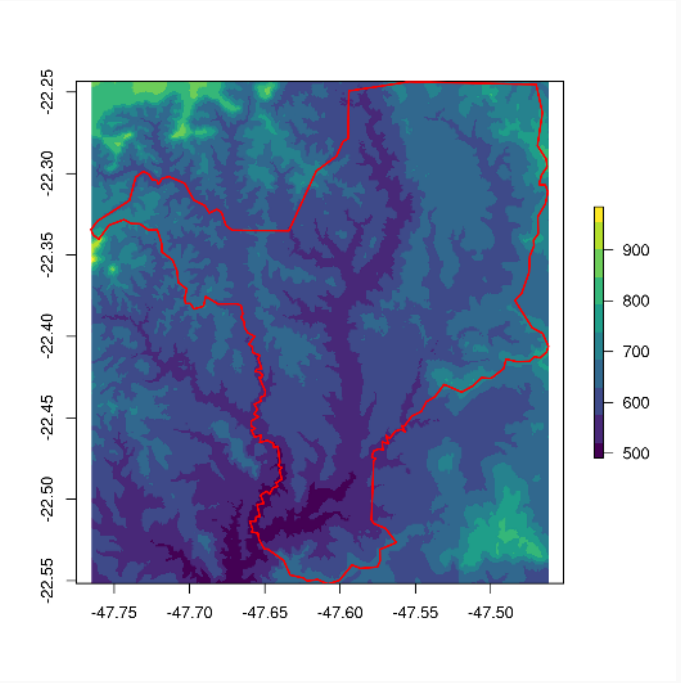
```
## class      : RasterLayer
## dimensions : 370, 364, 134680  (nrow, ncol, ncell)
## resolution : 0.0008333333, 0.0008333333  (x, y)
## extent     : -47.765, -47.46167, -22.55167, -22.24333  (xmin, xmax, ymin, ymax)
## crs        : +proj=longlat +datum=WGS84 +no_defs
## source     : memory
## names      : srtm_27_17
## values     : 491, 985  (min, max)
```

7. Interação raster-vetor

1. Cortes e máscaras

Crop - Ajusta a extensão

```
# plot
plot(dem_rc_crop, col = viridis::viridis(10))
plot(rc_2020$geom, col = NA, border = "red", lwd = 2, add = TRUE)
```



7. Interação raster-vetor

1. Cortes e máscaras

Mask - Ajusta o limite

```
# mask - ajuste o limite
dem_rc_mask ← raster::mask(dem, rc_2020)
dem_rc_mask
```

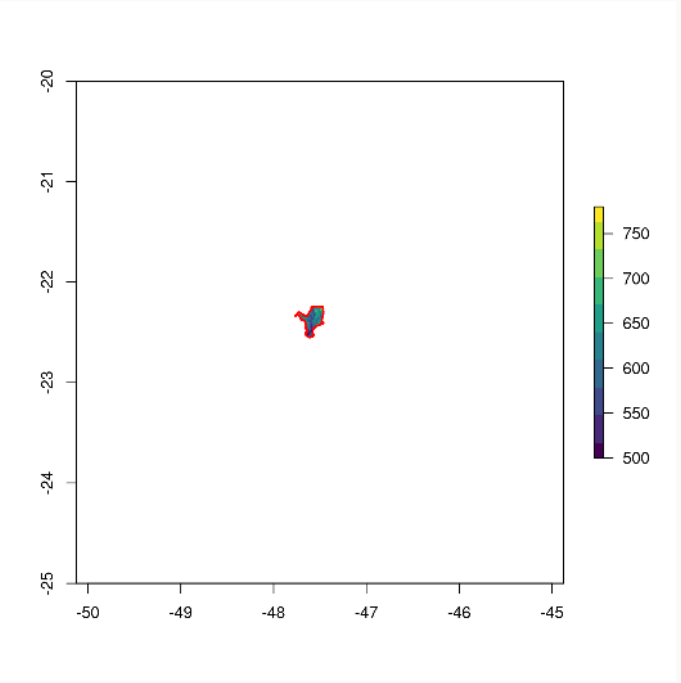
```
## class      : RasterLayer
## dimensions : 6000, 6000, 3.6e+07  (nrow, ncol, ncell)
## resolution : 0.0008333333, 0.0008333333  (x, y)
## extent     : -50, -45, -25, -20  (xmin, xmax, ymin, ymax)
## crs        : +proj=longlat +datum=WGS84 +no_defs
## source     : memory
## names      : srtm_27_17
## values     : 497, 860  (min, max)
```

7. Interação raster-vetor

1. Cortes e máscaras

Mask

```
# plot
plot(dem_rc_mask, col = viridis::viridis(10))
plot(rc_2020$geom, col = NA, border = "red", lwd = 2, add = TRUE)
```



7. Interação raster-vetor

1. Cortes e máscaras

Crop e Mask - Ajuste da extensão e do limite

```
# crop e mask - ajuste da extensao e do limite
dem_rc_crop_mask ← dem %>%
  raster::crop(rc_2020) %>%
  raster::mask(rc_2020)
dem_rc_crop_mask
```

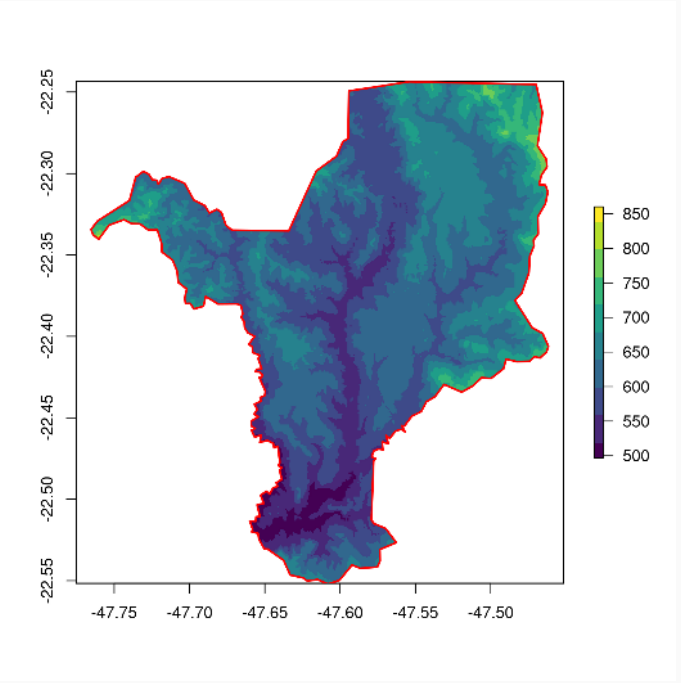
```
## class      : RasterLayer
## dimensions : 370, 364, 134680  (nrow, ncol, ncell)
## resolution : 0.0008333333, 0.0008333333  (x, y)
## extent     : -47.765, -47.46167, -22.55167, -22.24333  (xmin, xmax, ymin, ymax)
## crs        : +proj=longlat +datum=WGS84 +no_defs
## source     : memory
## names      : srtm_27_17
## values     : 497, 860  (min, max)
```

7. Interação raster-vetor

1. Cortes e máscaras

Crop e Mask - Ajuste da extensão e do limite

```
# plot
plot(dem_rc_crop_mask, col = viridis::viridis(10))
plot(rc_2020$geom, col = NA, border = "red", lwd = 2, add = TRUE)
```



7. Interação raster-vetor

1. Cortes e máscaras

Crop e Mask inverse - Ajuste da extensão e do limite inverso

```
# crop e mask inversa - ajuste da extensão e do limite inverso
dem_rc_crop_mask_inv ← dem %>%
  raster::crop(rc_2020) %>%
  raster::mask(rc_2020, inverse = TRUE)
dem_rc_crop_mask_inv
```

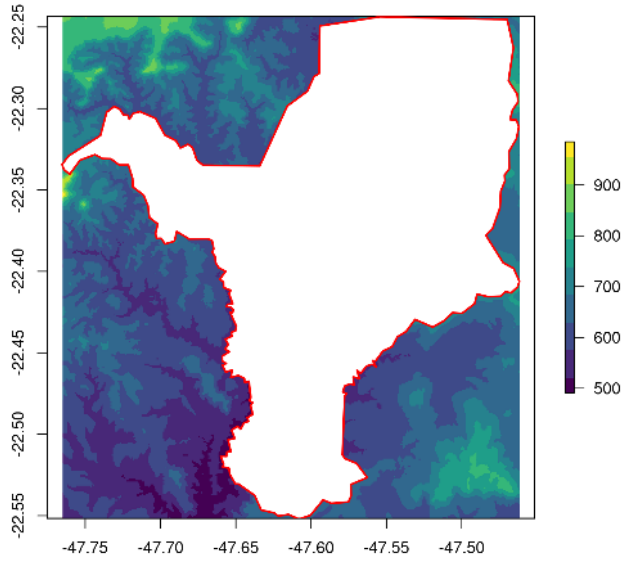
```
## class      : RasterLayer
## dimensions : 370, 364, 134680  (nrow, ncol, ncell)
## resolution : 0.0008333333, 0.0008333333  (x, y)
## extent     : -47.765, -47.46167, -22.55167, -22.24333  (xmin, xmax, ymin, ymax)
## crs        : +proj=longlat +datum=WGS84 +no_defs
## source     : memory
## names      : srtm_27_17
## values     : 491, 985  (min, max)
```

7. Interação raster-vetor

1. Cortes e máscaras

Crop e Mask inverse - Ajuste da extensão e do limite inverso

```
# plot
plot(dem_rc_crop_mask_inv, col = viridis::viridis(10))
plot(rc_2020$geom, col = NA, border = "red", lwd = 2, add = TRUE)
```



7. Interação raster-vetor

2. Extração

Importar pontos das nascentes

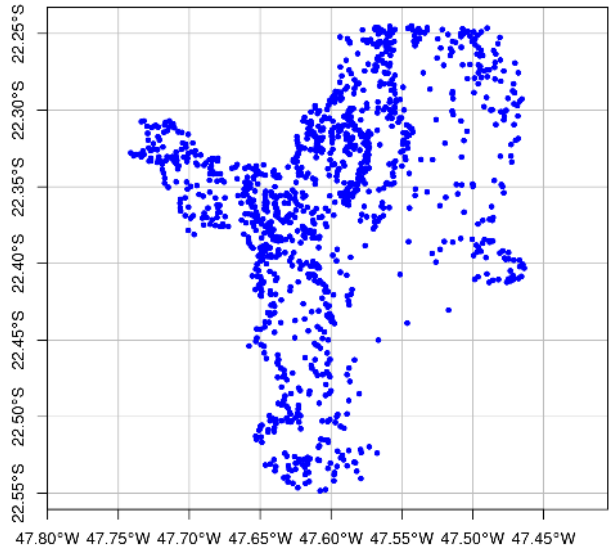
```
# importar pontos
rc_nas ← sf::st_read(here::here("03_dados", "vetor", "SP_3543907_NASCENTES.shp"), quiet = TRUE) %>%
  sf::st_transform(crs = 4326)
rc_nas
```

```
## Simple feature collection with 1220 features and 5 fields
## Geometry type: POINT
## Dimension:     XY
## Bounding box:  xmin: -47.74126 ymin: -22.54837 xmax: -47.46367 ymax: -22.24496
## Geodetic CRS:  WGS 84
## First 10 features:
##    GEOCODIGO MUNICIPIO UF CD_UF    HIDRO                    geometry
## 1    3543907 RIO CLARO SP    35 nascente POINT (-47.74126 -22.32786)
## 2    3543907 RIO CLARO SP    35 nascente POINT (-47.73922 -22.32981)
## 3    3543907 RIO CLARO SP    35 nascente POINT (-47.73773 -22.32894)
## 4    3543907 RIO CLARO SP    35 nascente POINT (-47.73481 -22.32868)
## 5    3543907 RIO CLARO SP    35 nascente POINT (-47.73384 -22.30751)
## 6    3543907 RIO CLARO SP    35 nascente POINT (-47.73384 -22.33055)
```

2. Extração

Importar pontos das nascentes

```
# plot
plot(rc_nas$geometry, pch = 20, col = "blue", main = NA, axes = TRUE, graticule = TRUE)
```



2. Extração

Extração da altitude para os pontos das nascentes

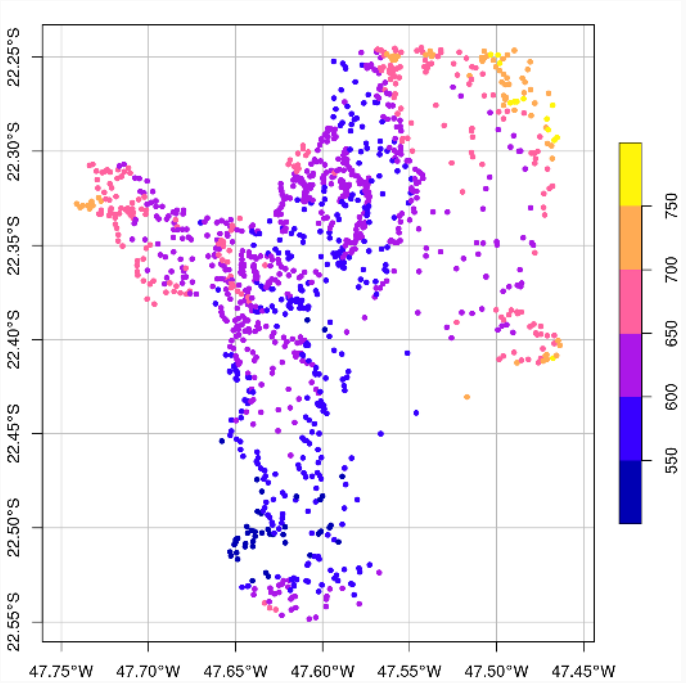
```
# extracao
rc_nas_ele ← rc_nas %>%
  dplyr::mutate(elev = raster::extract(x = dem_rc_utm23s, y = .))
rc_nas_ele
```

```
## Simple feature collection with 1220 features and 6 fields
## Geometry type: POINT
## Dimension:     XY
## Bounding box:  xmin: -47.74126 ymin: -22.54837 xmax: -47.46367 ymax: -22.24496
## Geodetic CRS:  WGS 84
## First 10 features:
##    GEOCODIGO MUNICIPIO UF CD_UF    HIDRO                    geometry     elev
## 1    3543907 RIO CLARO SP    35 nascente POINT (-47.74126 -22.32786) 728.2511
## 2    3543907 RIO CLARO SP    35 nascente POINT (-47.73922 -22.32981) 717.5408
## 3    3543907 RIO CLARO SP    35 nascente POINT (-47.73773 -22.32894) 717.3808
## 4    3543907 RIO CLARO SP    35 nascente POINT (-47.73481 -22.32868) 706.6163
## 5    3543907 RIO CLARO SP    35 nascente POINT (-47.73384 -22.30751) 668.4878
## 6    3543907 RIO CLARO SP    35 nascente POINT (-47.73384 -22.33055) 715.0871
```

2. Extração

Extração da altitude para os pontos das nascentes

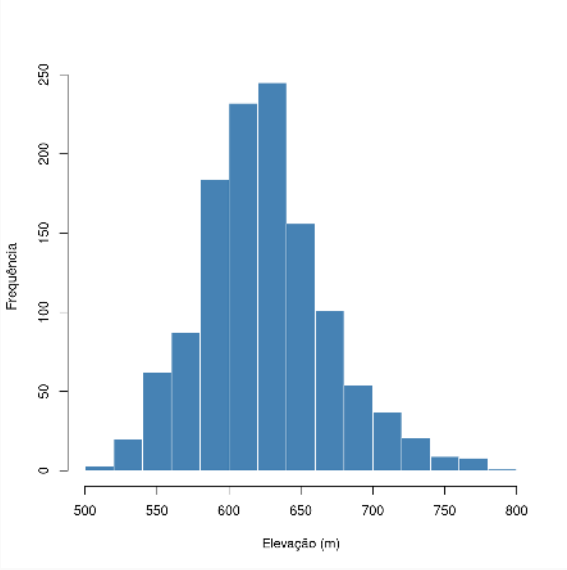
```
# plot
plot(rc_nas_ele["elev"], pch = 20, main = NA, axes = TRUE, graticule = TRUE)
```



2. Extração

Histograma dos valores

```
# histograma
rc_nas_ele %>%
  dplyr::pull(elev) %>%
  hist(col = "steelblue", border = "white", main = NA, xlab = "Elevação (m)", ylab = "Frequência")
```



3. Estatística zonal

Buffers

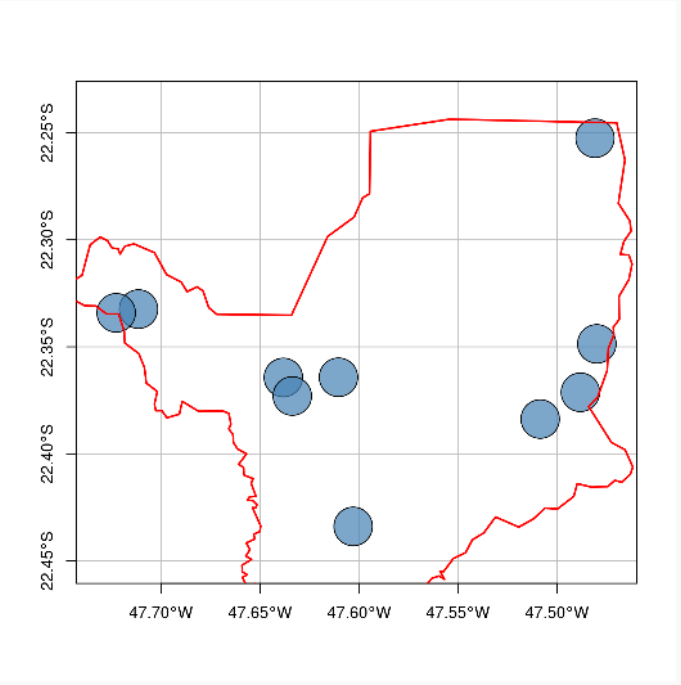
```
# buffers
set.seed(42)
rc_nas_buf ← rc_nas %>%
  dplyr::sample_n(10) %>%
  sf::as_Spatial() %>%
  raster::buffer(width = 1000, dissolve = FALSE) %>%
  sf::st_as_sf()
rc_nas_buf
```

```
## Simple feature collection with 10 features and 5 fields
## Geometry type: POLYGON
## Dimension:     XY
## Bounding box:  xmin: -47.73232 ymin: -22.44296 xmax: -47.47012 ymax: -22.2435
## Geodetic CRS:  WGS 84
##    GEOCODIGO MUNICIPIO UF CD_UF    HIDRO                          geometry
## 1    3543907 RIO CLARO SP    35 nascente POLYGON ((-47.61015 -22.355 ...
## 2    3543907 RIO CLARO SP    35 nascente POLYGON ((-47.638 -22.3553, ...
## 3    3543907 RIO CLARO SP    35 nascente POLYGON ((-47.47966 -22.339 ...
```

3. Estatística zonal

Estatística zonal - Buffers

```
# plot
plot(rc_nas_buf$geometry, col = adjustcolor("steelblue", .7), pch = 20, main = NA, axes = TRUE, graticule = TRUE
plot(rc_2020$geom, col = NA, border = "red", lwd = 2, add = TRUE)
```



3. Estatística zonal

Estatística zonal

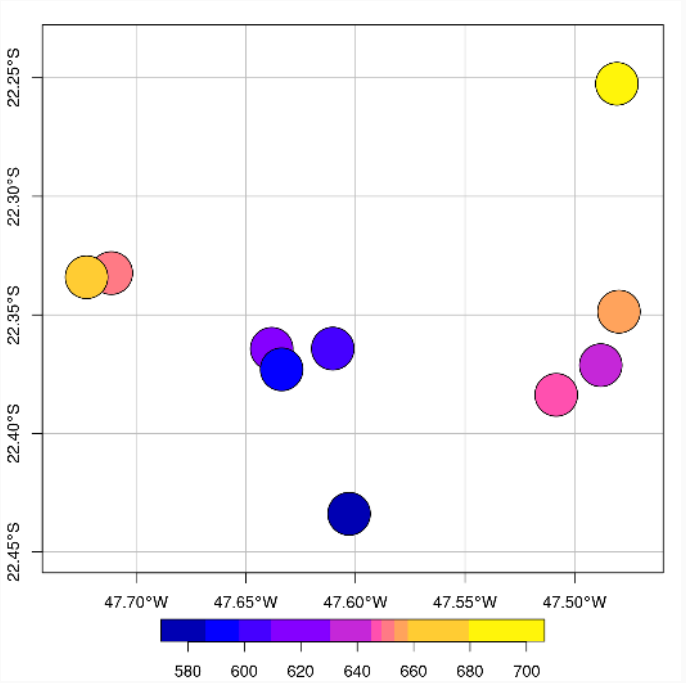
```
# estatistica zonal
rc_nas_buf ← rc_nas_buf %>%
  dplyr::mutate(elev_mean = raster::extract(x = dem_rc, y = rc_nas_buf, fun = mean, na.rm = TRUE))
rc_nas_buf
```

```
## Simple feature collection with 10 features and 6 fields
## Geometry type: POLYGON
## Dimension:     XY
## Bounding box:  xmin: -47.73232 ymin: -22.44296 xmax: -47.47012 ymax: -22.2435
## Geodetic CRS:  WGS 84
##    GEOCODIGO MUNICIPIO UF CD_UF    HIDRO                       geometry elev_mean
## 1    3543907 RIO CLARO SP    35 nascente POLYGON ((-47.61015 -22.355 ...   601.4461
## 2    3543907 RIO CLARO SP    35 nascente POLYGON ((-47.638 -22.3553, ...   617.4470
## 3    3543907 RIO CLARO SP    35 nascente POLYGON ((-47.47966 -22.339 ...   656.2843
## 4    3543907 RIO CLARO SP    35 nascente POLYGON ((-47.50825 -22.374 ...   646.9322
## 5    3543907 RIO CLARO SP    35 nascente POLYGON ((-47.48058 -22.243 ...   699.5970
## 6    3543907 RIO CLARO SP    35 nascente POLYGON ((-47.60273 -22.424 ...   577.2166
## 7    3543907 RIO CLARO SP    35 nascente POLYGON ((-47.71117 -22.323 ...   650.4125
```

3. Estatística zonal

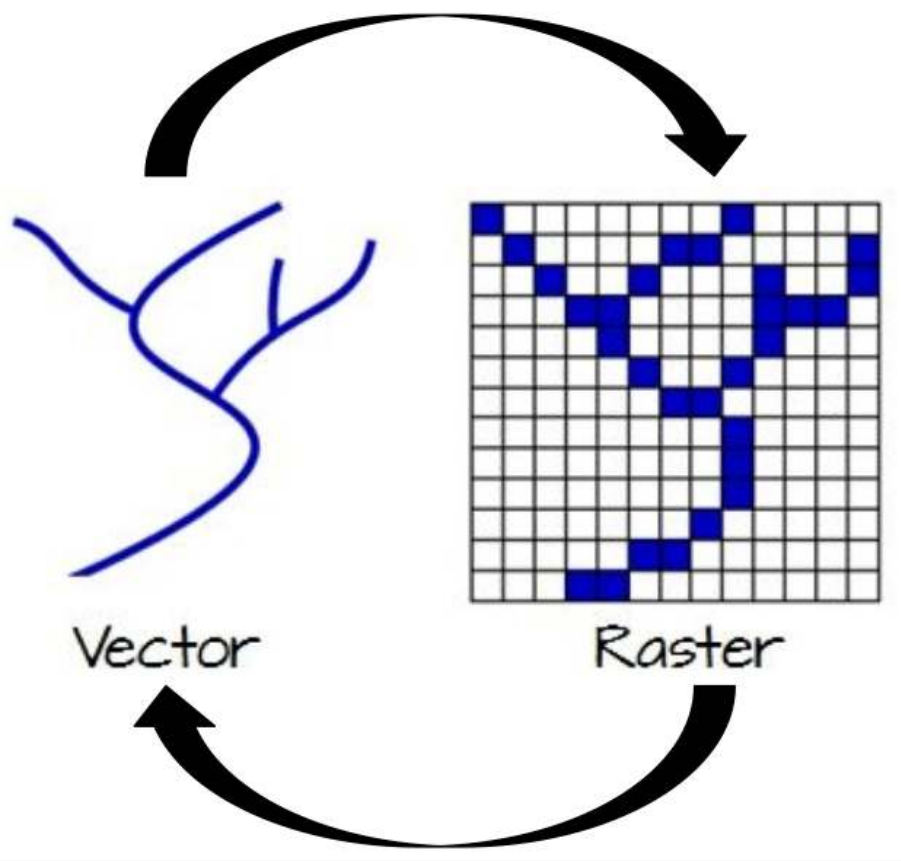
Estatística zonal

```
# plot
plot(rc_nas_buf["elev_mean"], pch = 20, main = NA, axes = TRUE, graticule = TRUE)
```



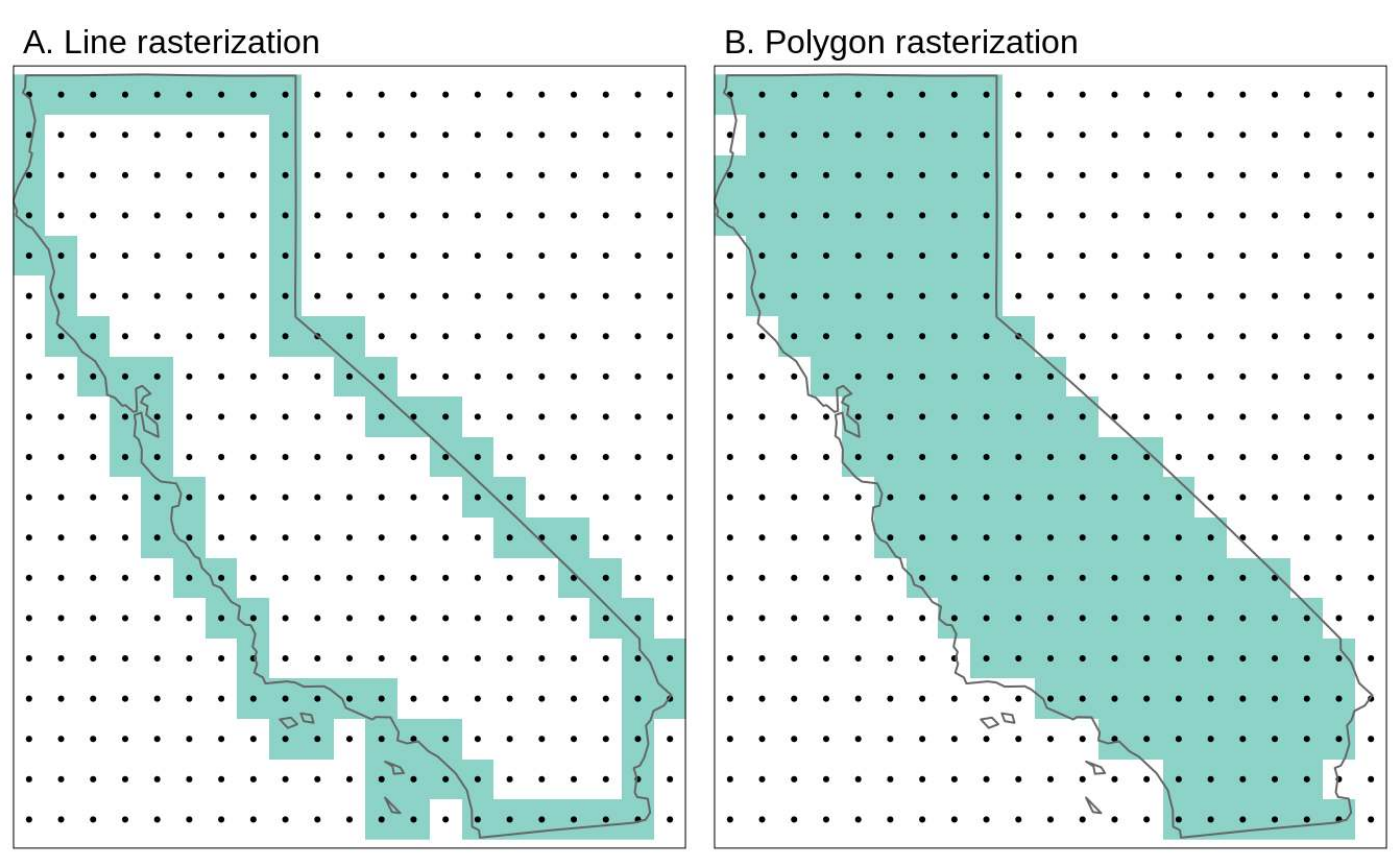
8. Conversão raster-vetor

Podemos fazer operações de conversão entre objetos vetoriais para raster e vice-versa



8. Conversão raster-vetor

Rasterização



Lovelace et al. (2020)

8. Conversão raster-vetor

Rasterização de pontos

Rasterize

```
# importar pontos
rc_nas ← sf::st_read(here::here("03_dados", "vetor", "SP_3543907_NASCENTES.shp"), quiet = TRUE)
rc_nas
```

```
## Simple feature collection with 1220 features and 5 fields
## Geometry type: POINT
## Dimension:     XY
## Bounding box:  xmin: 217622.9 ymin: 7504132 xmax: 246367.4 ymax: 7537855
## Projected CRS: SIRGAS 2000 / UTM zone 23S
## First 10 features:
##    GEOCODIGO MUNICIPIO UF CD_UF    HIDRO                 geometry
## 1    3543907 RIO CLARO SP    35 nascente POINT (217622.9 7528315)
## 2    3543907 RIO CLARO SP    35 nascente POINT (217836.5 7528103)
## 3    3543907 RIO CLARO SP    35 nascente POINT (217988.9 7528203)
## 4    3543907 RIO CLARO SP    35 nascente POINT (218288.9 7528237)
## 5    3543907 RIO CLARO SP    35 nascente POINT (218346.6 7530583)
## 6    3543907 RIO CLARO SP    35 nascente POINT (218393.1 7528031)
## 7    3543907 RIO CLARO SP    35 nascente POINT (218508.3 7528470)
```

8. Conversão raster-vetor

Rasterização de pontos

Rasterize

```
# rasterizar pontos
rc_nas_rasterizacao ← raster::rasterize(x = rc_nas, y = dem_rc_utm23s_agre_media, field = 1, fun = "count")
rc_nas_rasterizacao
```

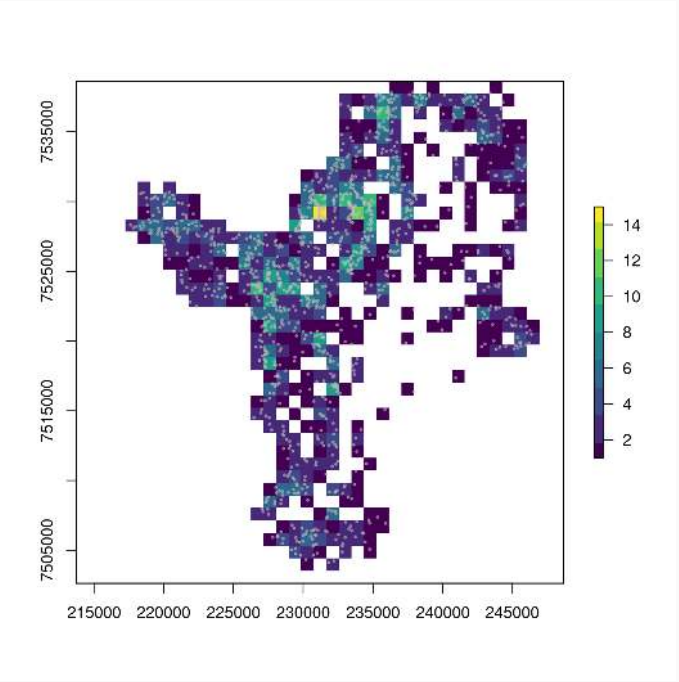
```
## class      : RasterLayer
## dimensions : 40, 37, 1480  (nrow, ncol, ncell)
## resolution : 900, 900  (x, y)
## extent     : 214554.4, 247854.4, 7502625, 7538625  (xmin, xmax, ymin, ymax)
## crs        : +proj=utm +zone=23 +south +ellps=GRS80 +units=m +no_defs
## source     : memory
## names      : layer
## values     : 1, 15  (min, max)
```

8. Conversão raster-vetor

Rasterização de pontos

Rasterize

```
# plot
plot(rc_nas_rasterizacao, col = viridis::viridis(10))
plot(rc_nas$geometry, pch = 20, cex = .5, col = adjustcolor("gray", .5), add = TRUE)
```



8. Conversão raster-vetor

Rasterização de linhas

Rasterize

```
# importar linhas
rc_hid_simp ← sf::st_read(here::here("03_dados", "vetor", "SP_3543907_RIOS_SIMPLES.shp"), quiet = TRUE) %>%
  sf::st_simplify(x = ., dTolerance = 1000)
rc_hid_simp
```

```
## Simple feature collection with 1 feature and 6 fields
## Geometry type: MULTILINESTRING
## Dimension:     XY
## Bounding box:  xmin: 215155.3 ymin: 7504132 xmax: 246367.4 ymax: 7537978
## Projected CRS: SIRGAS 2000 / UTM zone 23S
##   GEOCODIGO MUNICIPIO UF CD_UF                  HIDRO COMP_KM                          geometry
## 1   3543907 RIO CLARO SP    35 curso d'água (0 - 10m) 1142.98 MULTILINESTRING ((231815.7 ...
```

8. Conversão raster-vetor

Rasterização de linhas

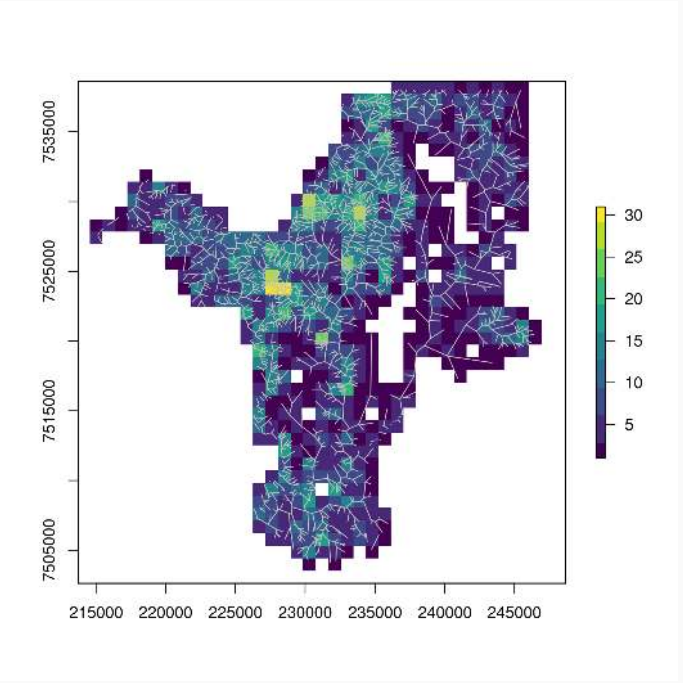
Rasterize

```
# rasterizar linhas
rc_hid_rasterizacao ← raster::rasterize(x = rc_hid_simp,
                                        y = dem_rc_utm23s_agre_media,
                                        field = 1, fun = "count")
```

Rasterização de linhas

Rasterize

```
# plot
plot(rc_hid_rasterizacao, col = viridis::viridis(10))
plot(rc_hid_simp$geom, col = "gray", add = TRUE)
```



8. Conversão raster-vetor

Rasterização de polígonos

Rasterize

```
# importar poligonos
rc_cob ← sf::st_read(here::here("03_dados", "vetor", "SP_3543907_USO.shp"), quiet = TRUE) %>%
  dplyr::mutate(classe = as.factor(CLASSE_USO))
rc_cob
```

```
## Simple feature collection with 5 features and 7 fields
## Geometry type: MULTIPOLYGON
## Dimension:     XY
## Bounding box:  xmin: 215151.7 ymin: 7503723 xmax: 246582.4 ymax: 7537978
## Projected CRS: SIRGAS 2000 / UTM zone 23S
##   GEOCODIGO MUNICIPIO UF CD_UF         CLASSE_USO   AREA_HA                       geometry             classe
## 1   3543907 RIO CLARO SP    35              água   357.027 MULTIPOLYGON (((235487.6 75 ...              água
## 2   3543907 RIO CLARO SP    35   área antropizada 37297.800 MULTIPOLYGON (((232275 7504 ...   área antropizada
## 3   3543907 RIO CLARO SP    35     área edificada  5078.330 MULTIPOLYGON (((233123.6 75 ...     área edificada
## 4   3543907 RIO CLARO SP    35 formação florestal  7017.990 MULTIPOLYGON (((232355 7504 ... formação florestal
## 5   3543907 RIO CLARO SP    35       silvicultura   138.173 MULTIPOLYGON (((243052.1 75 ...       silvicultura
```

8. Conversão raster-vetor

Rasterização de polígonos

Rasterize

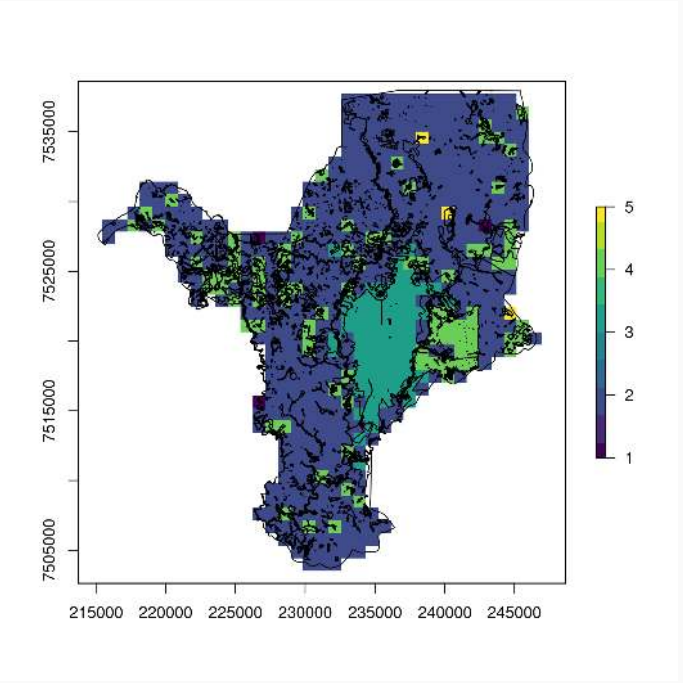
```
# rasterizar poligonos
rc_cob_rasterizacao ← raster::rasterize(x = rc_cob, y = dem_rc_utm23s_agre_media, field = "classe")
```

8. Conversão raster-vetor

Rasterização de polígonos

Rasterize

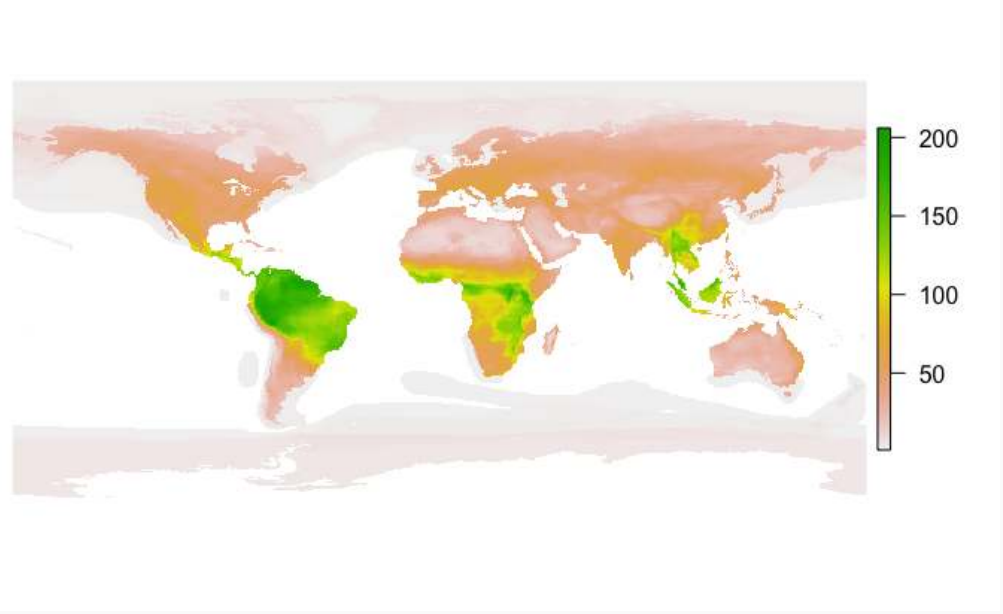
```
# plot
plot(rc_cob_rasterizacao, col = viridis::viridis(10))
plot(rc_cob$geom, add = TRUE)
```



Rasterização de polígonos

Fasterize

```
# package fasterize
# install.packages("fasterize")
library(fasterize)
```



8. Conversão raster-vetor

Rasterização de polígonos

Fasterize

```
# rasterizacao com fasterize
rc_cob_fast ← fasterize::fasterize(sf = rc_cob, raster = dem_rc_utm23s_agre_media, field = "classe")
rc_cob_fast
```

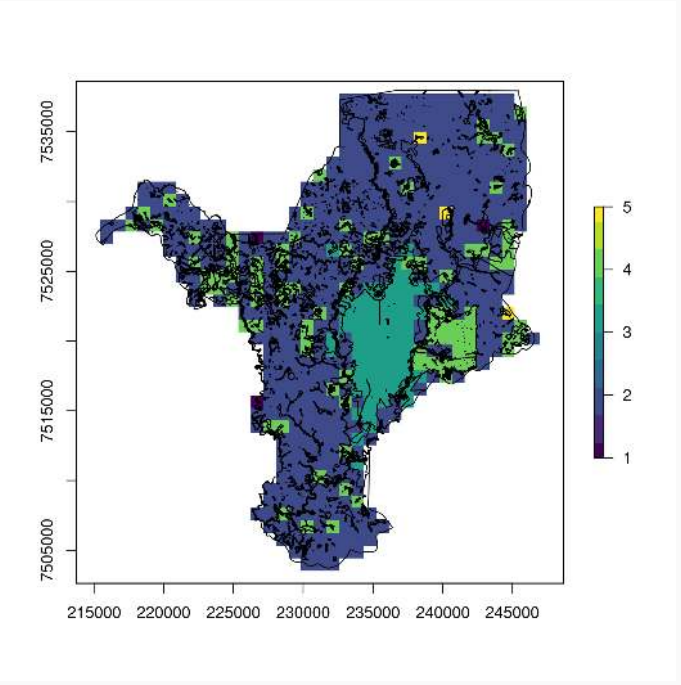
```
## class      : RasterLayer
## dimensions : 40, 37, 1480  (nrow, ncol, ncell)
## resolution : 900, 900  (x, y)
## extent     : 214554.4, 247854.4, 7502625, 7538625  (xmin, xmax, ymin, ymax)
## crs        : +proj=utm +zone=23 +south +ellps=GRS80 +units=m +no_defs
## source     : memory
## names      : layer
## values     : 1, 5  (min, max)
```

8. Conversão raster-vetor

Rasterização de polígonos

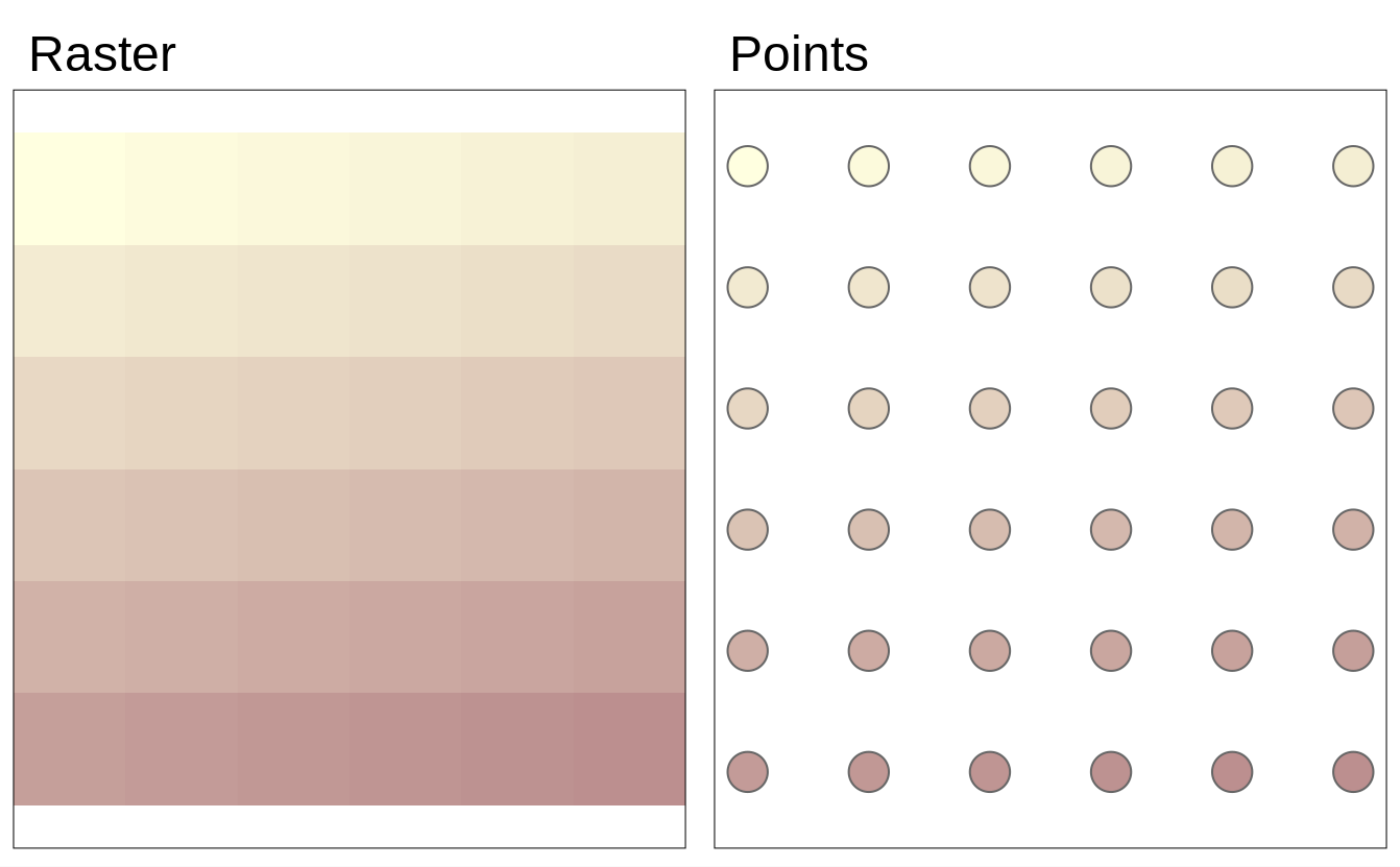
Fasterize

```
# plot
plot(rc_cob_fast, col = viridis::viridis(10))
plot(rc_cob$geom, add = TRUE)
```



8. Conversão raster-vetor

Vetorização - Raster para pontos



Lovelace et al. (2020)

8. Conversão raster-vetor

Vetorização - Raster para pontos

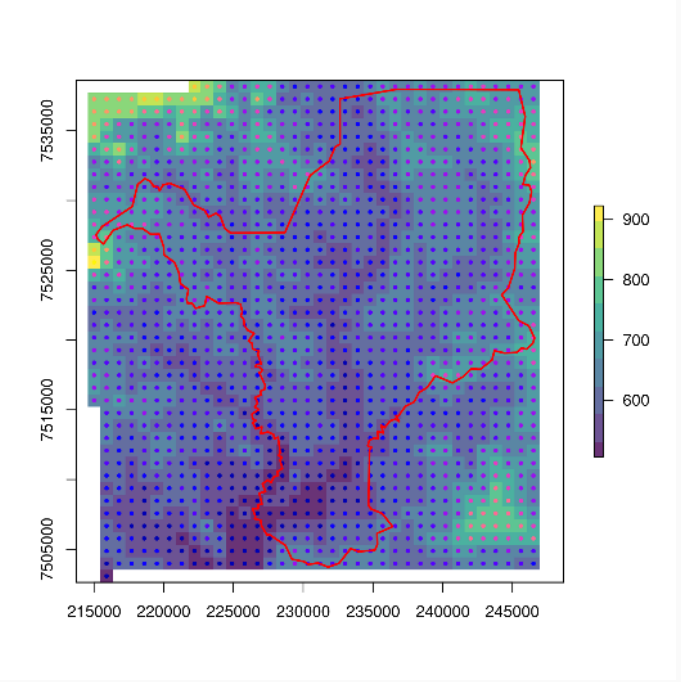
```
# vetorizacao de pontos
dem_rc_utm23s_agre_media_pontos ← raster::rasterToPoints(dem_rc_utm23s_agre_media, spatial = TRUE) %>%
  sf::st_as_sf()
dem_rc_utm23s_agre_media_pontos
```

```
## Simple feature collection with 1384 features and 1 field
## Geometry type: POINT
## Dimension:     XY
## Bounding box:  xmin: 215004.4 ymin: 7503075 xmax: 246504.4 ymax: 7538175
## Projected CRS: SIRGAS 2000 / UTM zone 23S
## First 10 features:
##    srtm_27_17                 geometry
## 1    861.5798 POINT (222204.4 7538175)
## 2    845.0291 POINT (223104.4 7538175)
## 3    767.5242 POINT (224004.4 7538175)
## 4    666.6461 POINT (224904.4 7538175)
## 5    679.7852 POINT (225804.4 7538175)
## 6    727.4907 POINT (226704.4 7538175)
## 7    716.2133 POINT (227604.4 7538175)
## 8    622.1591 POINT (228504.4 7538175)
```

8. Conversão raster-vetor

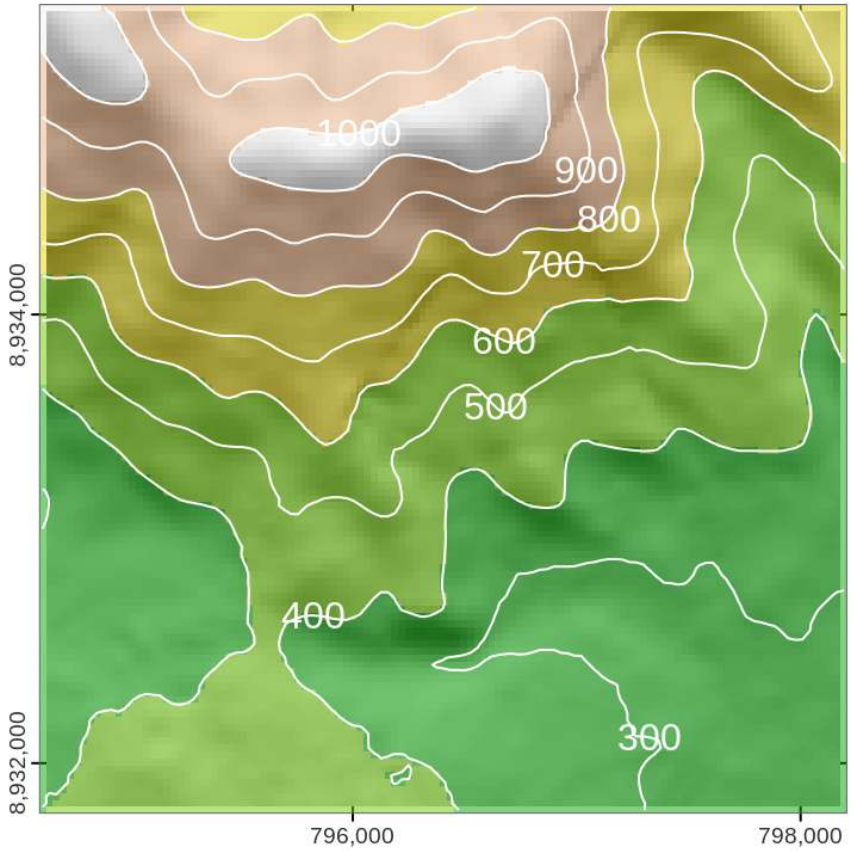
Vetorização - Raster para pontos

```
# plot
plot(dem_rc_utm23s_agre_media, col = viridis::viridis(10, alpha = .8))
plot(dem_rc_utm23s_agre_media_pontos, pch = 20, cex = .7, main = FALSE, add = TRUE)
plot(rc_2020_utm23s$geom, col = NA, border = "red", lwd = 2, add = TRUE)
```



8. Conversão raster-vetor

Vetorização - Raster para linhas



Lovelace et al. (2020)

8. Conversão raster-vetor

Vetorização - Raster para linhas

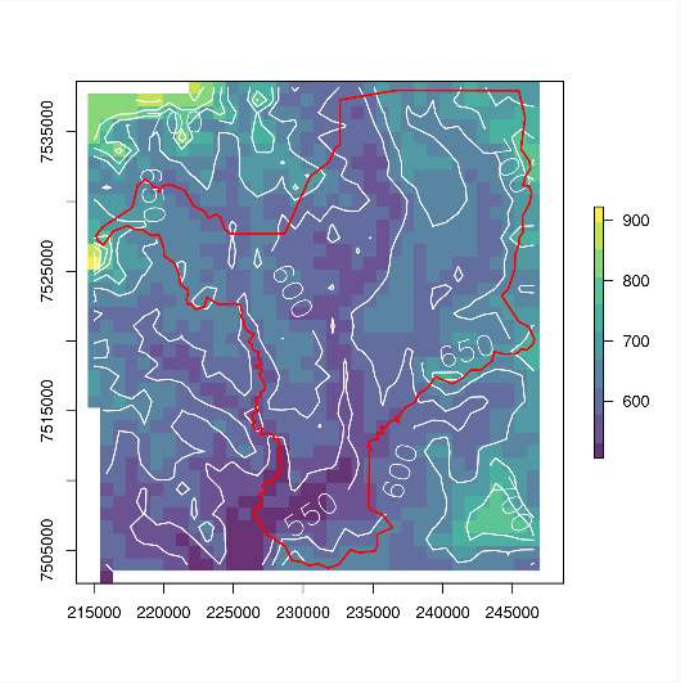
```
# vetorizacao de linhas
dem_rc_utm23s_agre_media_linhas ← raster::rasterToContour(x = dem_rc_utm23s_agre_media) %>%
  sf::st_as_sf()
dem_rc_utm23s_agre_media_linhas
```

```
## Simple feature collection with 8 features and 1 field
## Geometry type: MULTILINESTRING
## Dimension:     XY
## Bounding box:  xmin: 215004.4 ymin: 7503506 xmax: 246504.4 ymax: 7538175
## Projected CRS: SIRGAS 2000 / UTM zone 23S
##     level                       geometry
## C_1   550 MULTILINESTRING ((215904.4 ...
## C_2   600 MULTILINESTRING ((215004.4 ...
## C_3   650 MULTILINESTRING ((215004.4 ...
## C_4   700 MULTILINESTRING ((215004.4 ...
## C_5   750 MULTILINESTRING ((215004.4 ...
## C_6   800 MULTILINESTRING ((215004.4 ...
## C_7   850 MULTILINESTRING ((215004.4 ...
## C_8   900 MULTILINESTRING ((215004.4 ...
```

8. Conversão raster-vetor

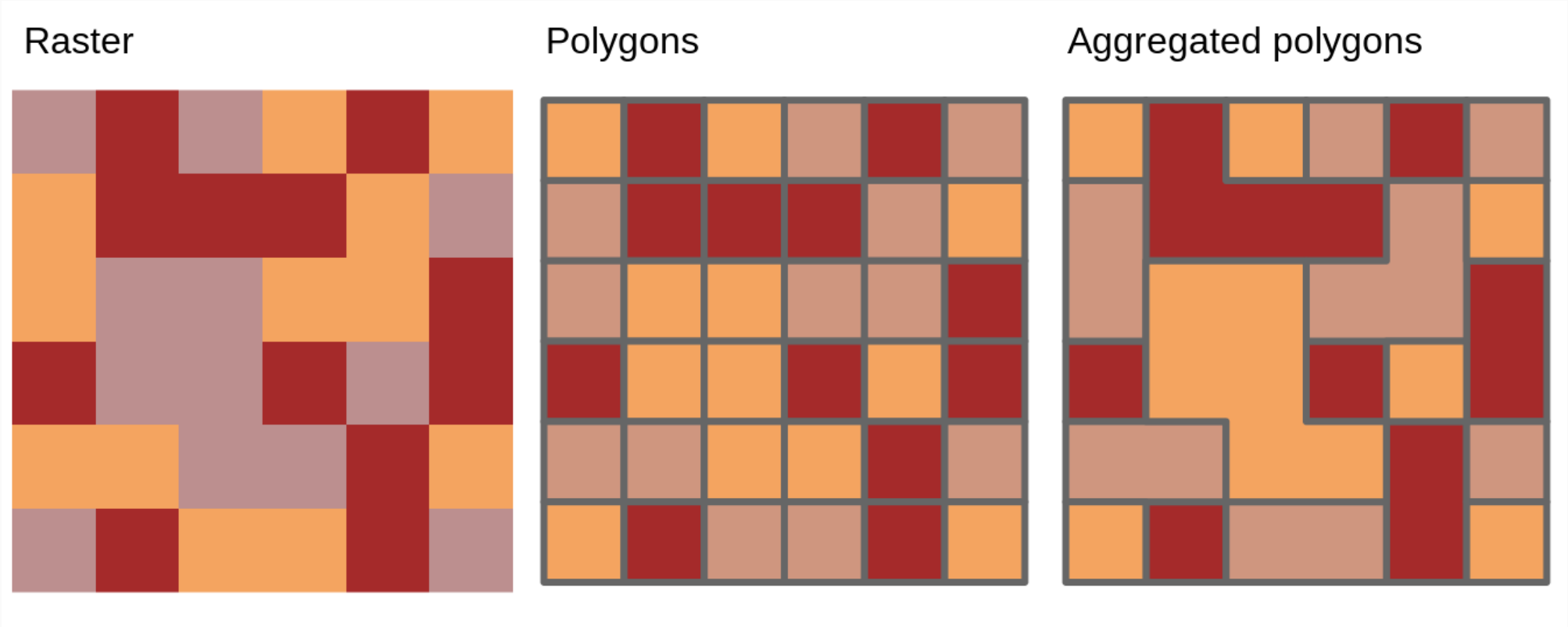
Vetorização - Raster para linhas

```
# plot
plot(dem_rc_utm23s_agre_media, col = viridis::viridis(10, alpha = .8))
contour(dem_rc_utm23s_agre_media, levels =  seq(500, 900, by = 50),
        col = "white", pch = 20, lwd = 1.5, labcex = 2, main = FALSE, add = TRUE)
plot(rc_2020_utm23s$geom, col = NA, border = "red", lwd = 2, add = TRUE)
```



8. Conversão raster-vetor

Vetorização - Raster para polígonos



Lovelace et al. (2020)

8. Conversão raster-vetor

Vetorização - Raster para polígonos

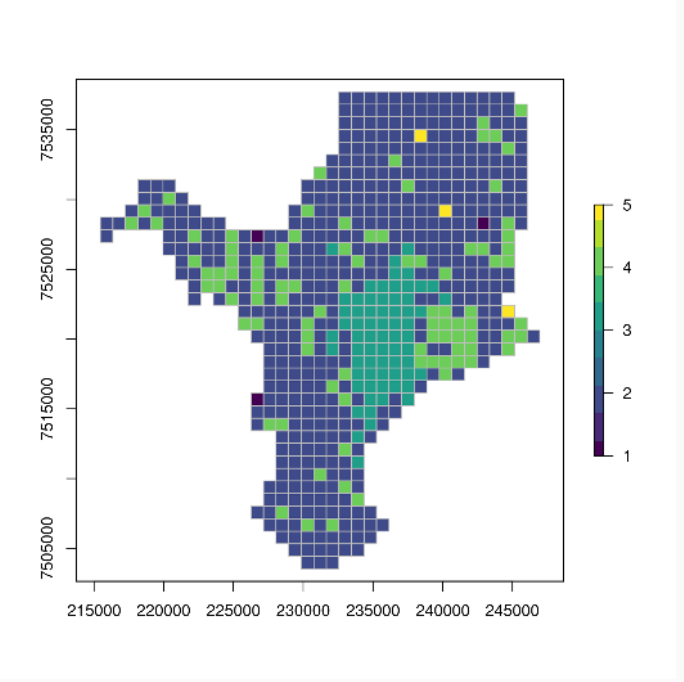
```
# vetorizacao de poligonos
rc_cob_rasterizacao_poligonos ← raster::rasterToPolygons(rc_cob_rasterizacao) %>%
  sf::st_as_sf()
rc_cob_rasterizacao_poligonos
```

```
## Simple feature collection with 618 features and 1 field
## Geometry type: POLYGON
## Dimension:     XY
## Bounding box:  xmin: 215454.4 ymin: 7503525 xmax: 246954.4 ymax: 7537725
## Projected CRS: SIRGAS 2000 / UTM zone 23S
## First 10 features:
##    layer                       geometry
## 1      2 POLYGON ((232554.4 7537725, ...
## 2      2 POLYGON ((233454.4 7537725, ...
## 3      2 POLYGON ((234354.4 7537725, ...
## 4      2 POLYGON ((235254.4 7537725, ...
## 5      2 POLYGON ((236154.4 7537725, ...
## 6      2 POLYGON ((237054.4 7537725, ...
## 7      2 POLYGON ((237954.4 7537725, ...
## 8      2 POLYGON ((238854.4 7537725, ...
```

8. Conversão raster-vetor

Vetorização - Raster para polígonos

```
# plot
plot(rc_cob_rasterizacao, col = viridis::viridis(10))
plot(rc_cob_rasterizacao_poligonos$geometry, col = NA, border = "gray", lwd = 1, main = FALSE, add = TRUE)
```



8. Conversão raster-vetor

Vetorização - Raster para polígonos dissolvidos

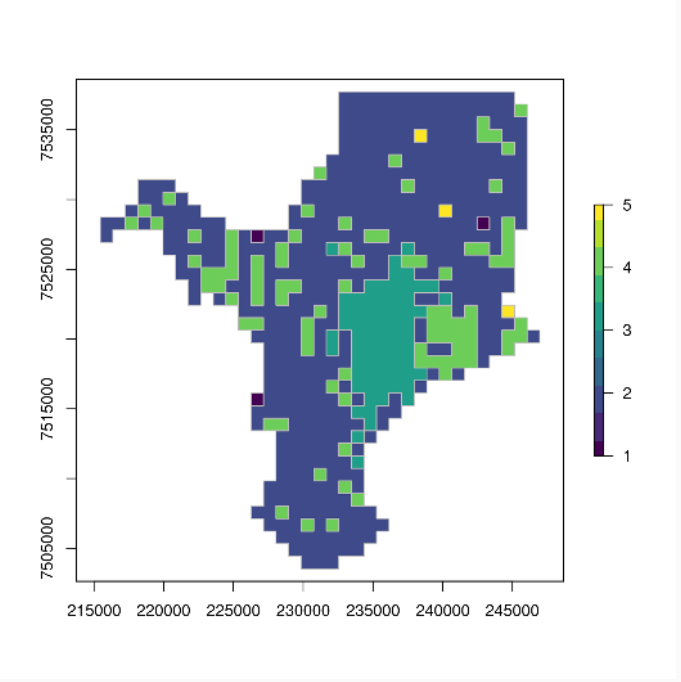
```
# vetorizacao de poligonos dissolvendo
rc_cob_rasterizacao_poligonos_dissolvidos ← raster::rasterToPolygons(rc_cob_rasterizacao, dissolve = TRUE) %>%
  sf::st_as_sf()
rc_cob_rasterizacao_poligonos_dissolvidos
```

```
## Simple feature collection with 5 features and 1 field
## Geometry type: MULTIPOLYGON
## Dimension:     XY
## Bounding box:  xmin: 215454.4 ymin: 7503525 xmax: 246954.4 ymax: 7537725
## Projected CRS: SIRGAS 2000 / UTM zone 23S
##   layer                          geometry
## 1     1 MULTIPOLYGON (((227154.4 75 ...
## 2     2 MULTIPOLYGON (((227154.4 75 ...
## 3     3 MULTIPOLYGON (((234354.4 75 ...
## 4     4 MULTIPOLYGON (((229854.4 75 ...
## 5     5 MULTIPOLYGON (((245154.4 75 ...
```

8. Conversão raster-vetor

Vetorização - Raster para polígonos dissolvidos

```
# plot
plot(rc_cob_rasterizacao, col = viridis::viridis(10))
plot(rc_cob_rasterizacao_poligonos_dissolvidos$geometry, col = NA, border = "gray", lwd = 1, main = FALSE, add =
```



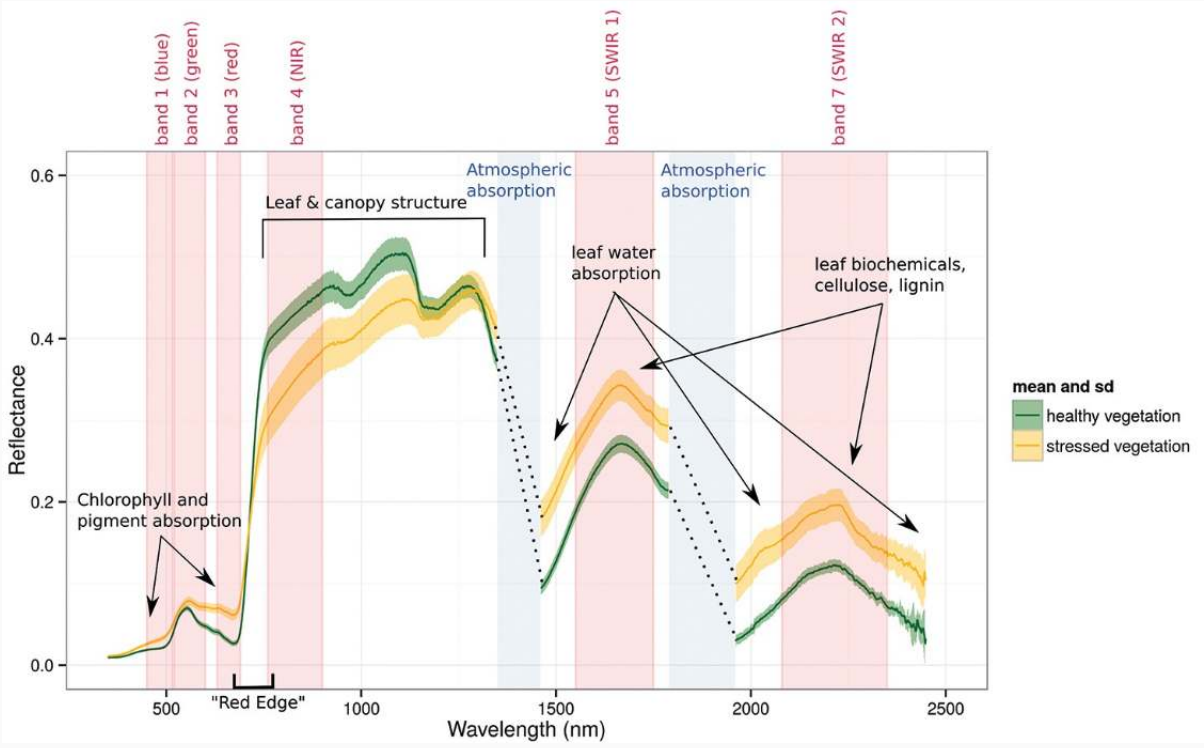
9. Índices espectrais

Definição

Equação matemática aplicada por pixel em **várias bandas espectrais** de uma imagem, cuja finalidade é **realçar alguma característica na imagem**

Aplicações

1. Vegetação
2. Solo
3. Incêndios
4. Água
5. ....



Wegmann et al. (2016)

9. Índices espectrais

Lista de índices espectrais (quase infinita...)

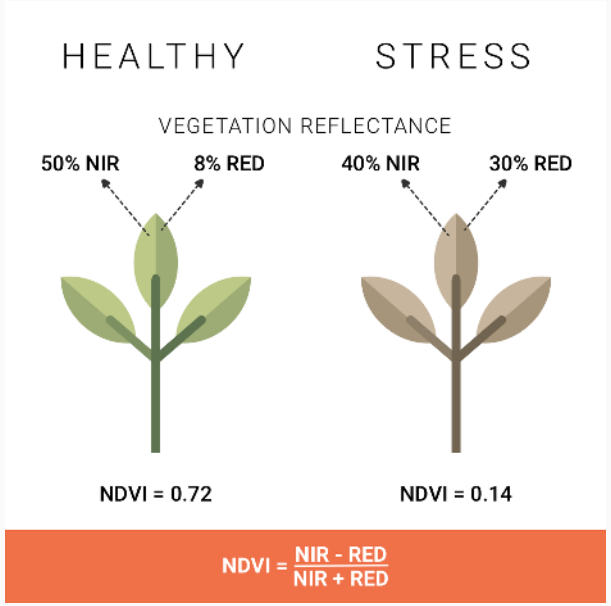
Index	Computation	Reference
Structural indices		
NDVI (Normalized Difference Vegetation Index)	$(\rho_n-\rho_r)/(\rho_n+\rho_r)$	Rouse et al. (1973)
SR (Simple Ratio)	ρ_n / ρ_r	Birth and McVey (1968)
SAVI (Soil Adjusted Vegetation Index)	$(\rho_n-\rho_r)(1+L)/(\rho_n+\rho_r+L)$ where L=0.5	Huete (1988)
MSAVI2 (Modified SAVI)	$\rho_n+0.5-((\rho_n+0.5)^2-2(\rho_n-\rho_r))^{0.5}$	Qi et al. (1994)
OSAVI (Optimized SAVI)	$(1+0.16)(\rho_{800}-\rho_{670})/(\rho_{800}+\rho_{670}+0.16)$	Rondeaux et al. (1996)
MSR (Modified SR)	$MSR=((R_{800}-R_{670})-1)/((R_{800}+R_{670})^{0.5}+1)$	Chen (1996)
RDVI (Renormalized Difference Vegetation Index)	$RDVI=(R_{800}-R_{670})/(R_{800}+R_{670})^{0.5}$	Roujean and Breon (1995)
Chlorophyll/pigment related indices		
MCARI (Modified CARI)	$MCARI=[(R_{700}-R_{670})-0.2(R_{700}-R_{550})](R_{700}/R_{670})$	Daughtry et al. (2000)
TCARI (Transformed CARI)	$TCARI=3[(R_{700}-R_{670})-0.2(R_{700}-R_{550})(R_{700}/R_{670})]$	Haboudane et al. (2002)
TVI (Triangular vegetation index)	$TVI=0.5[120(R_{750}-R_{550})-200(R_{670}-R_{550})]$	Broge and Leblanc (2000)
SIPI (Structural insensitive pigment index)	$SIPI=(R_{800}-R_{445})/(R_{800}+R_{680})$	Penuelas et al. (1995)
NPCI (Normalized Pigment Chlorophyll Index)	$NPCI=(R_{680}-R_{430})/(R_{680}+R_{430})$	Penuelas et al. (1995)
PRI (Photochemical Reflectance Index)	$(\rho_{531}-\rho_{570})/(\rho_{531}+\rho_{570})$	Gamon et al. (1992)
Red edge indices		
Red edge 750~700	$R_{750} - R_{700}$	Gitelson and Merzylak (1997)
Red edge 740~720	$R_{740} - R_{720}$	Vogelmann et al. (1993)
ZTM (Zarco-Tejada and Miller)	$ZTM=(R_{750} / R_{710})$	Zarco-Tejada et al. (2001)
Derivative Analysis Indices		
dg	Minimum of 1st derivative reflectance in the green (~570 nm)	Penuelas et al. 1994
dG	Maximum of 1st derivative reflectance of the green (~525 nm)	-do-
GGFN	Normalized Difference dG and dg	-do-
dRE	Maximum of 1st derivative of the reflectance in the red edge (~700–710 nm)	-do-
EGFN	Normalized difference of dRE and dG	-do-
ddRE	Maximum of 2nd derivative reflectance in the red edge (~690 nm)	-do-

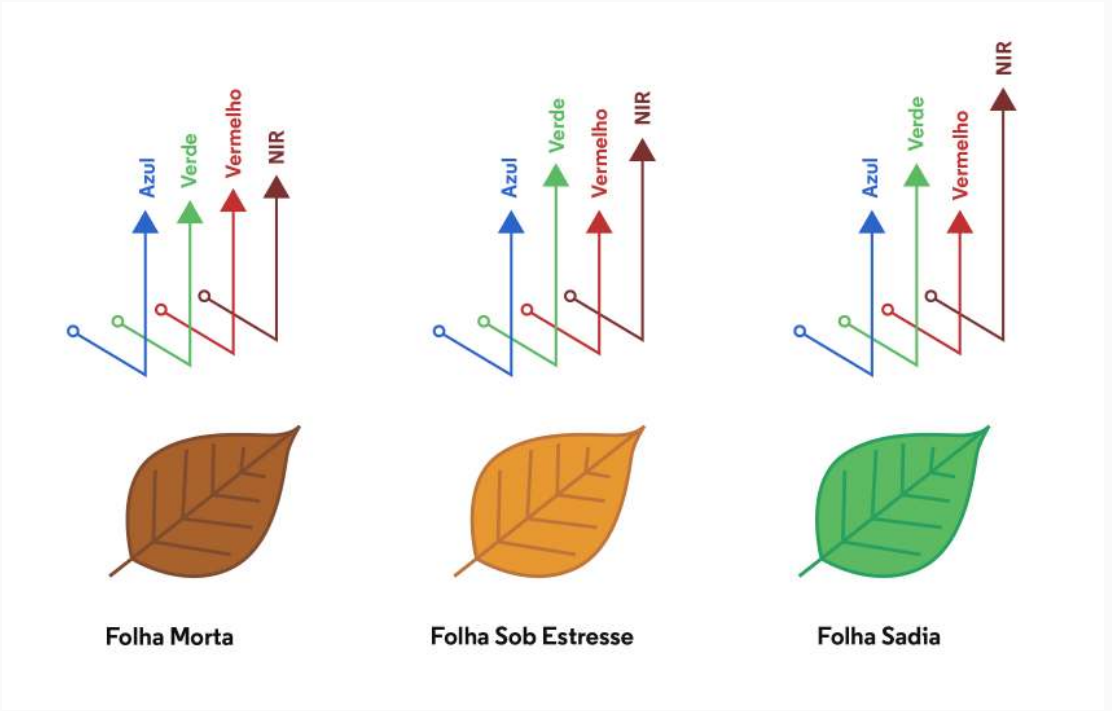
9. Índices espectrais

Normalized Difference Vegetation Index (NDVI)

- Os valores do NDVI variam de -1 a +1
- Valores **próximos de 1** indicam **maior atividade fotossintética** da vegetação
- Valores **próximos de 0** são geralmente **áreas sem vegetação**
- Valores **próximos de -1** geralmente indicam **água**

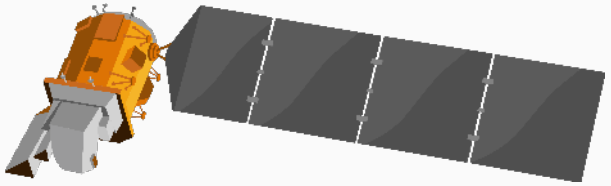
$$NDVI = \frac{\rho_{NIR} - \rho_{red}}{\rho_{NIR} + \rho_{red}}$$

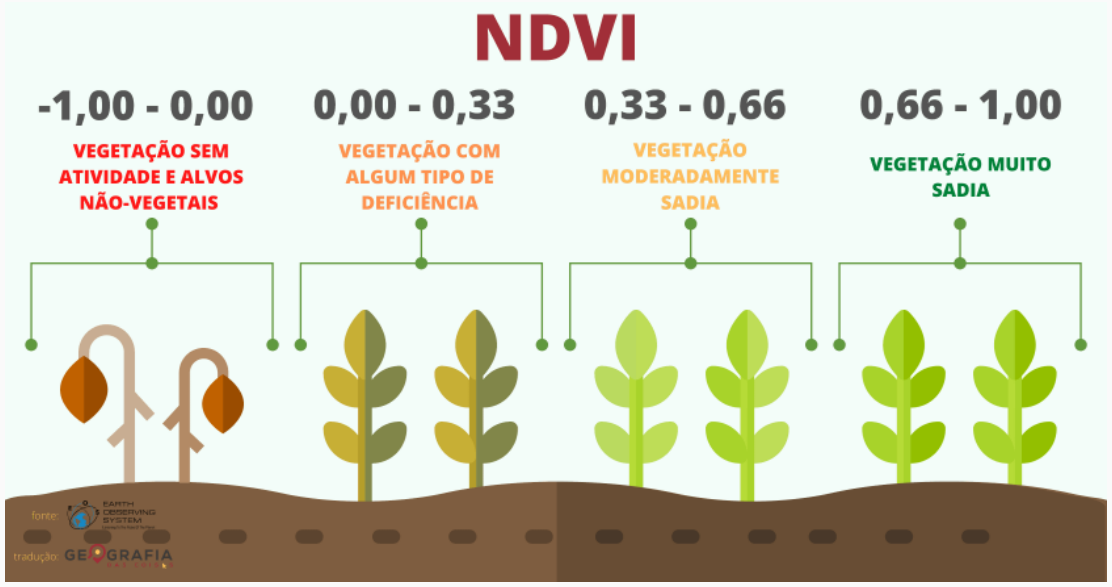


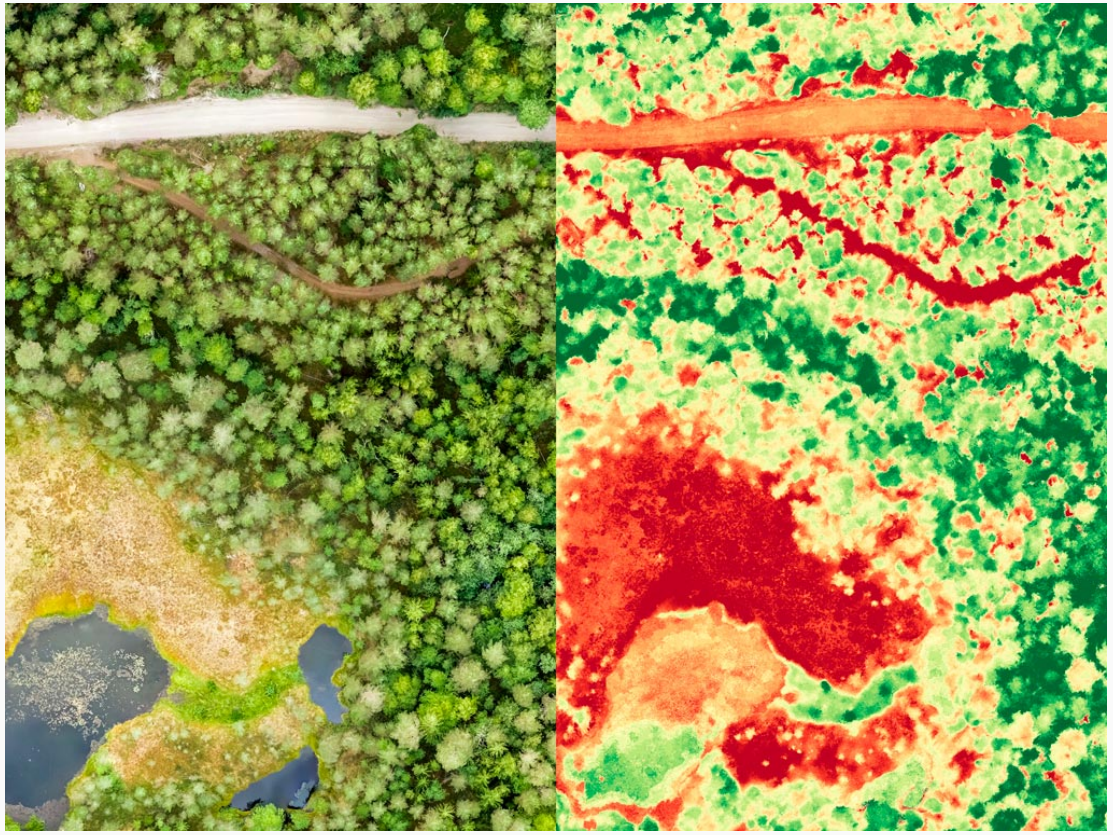


9. Índices espectrais

Normalized Difference Vegetation Index (NDVI)







9. Índices espectrais

Normalized Difference Vegetation Index (NDVI)

Landsat 8 - Rio Claro/SP

```
# landsat 8 - Rio Claro/SP
landsat_rc
```

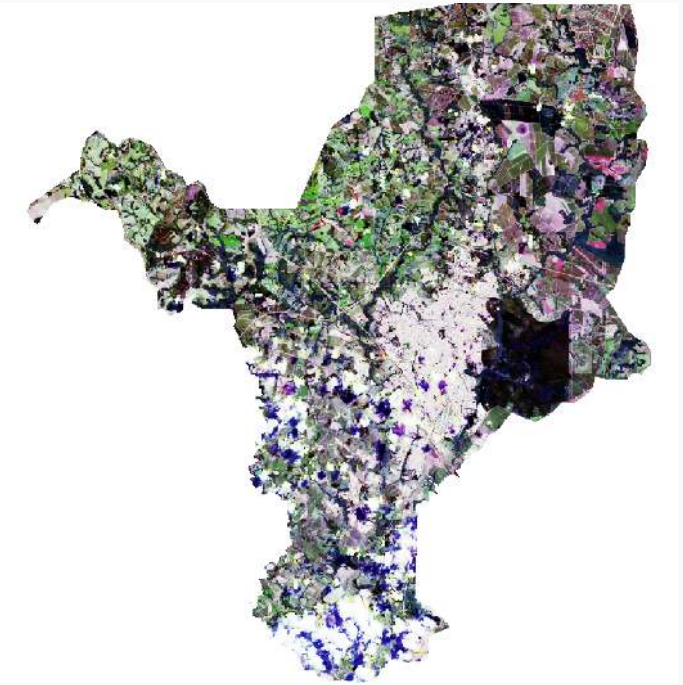
```
## class      : RasterBrick
## dimensions : 1142, 1047, 1195674, 7  (nrow, ncol, ncell, nlayers)
## resolution : 30, 30  (x, y)
## extent     : 215145, 246555, -2496285, -2462025  (xmin, xmax, ymin, ymax)
## crs        : +proj=utm +zone=23 +datum=WGS84 +units=m +no_defs
## source     : landsat_rc.tif
## names      : landsat_rc.1, landsat_rc.2, landsat_rc.3, landsat_rc.4, landsat_rc.5, landsat_rc.6, landsat_rc.7
## min values :        -2000,        -2000,         -135,         -303,           -8,          -50,           11
## max values :         8199,         8701,         9440,         9991,        11246,        12168,        11846
```

9. Índices espectrais

Normalized Difference Vegetation Index (NDVI)

Landsat 8 - Rio Claro/SP

```
# plot - cores naturais
plotRGB(landsat_rc, r = 4, g = 3, b = 2, stretch = "hist")
```



9. Índices espectrais

Normalized Difference Vegetation Index (NDVI)

Landsat 8 - Rio Claro/SP

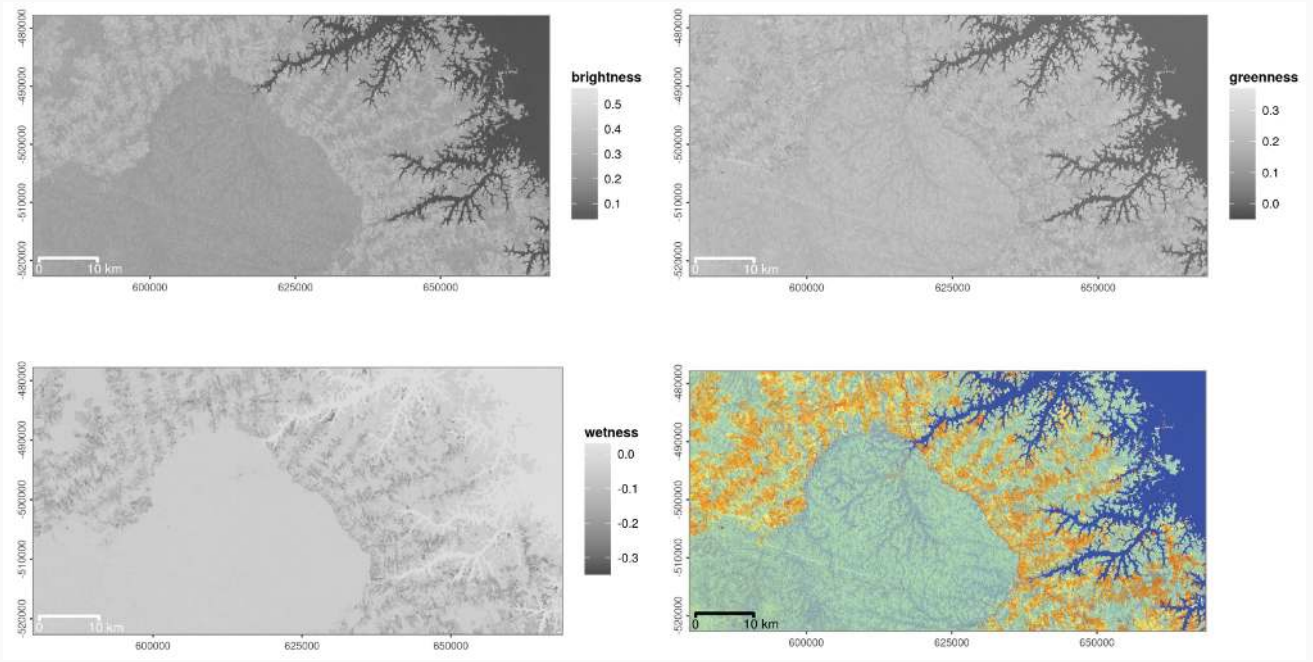
```
# plot - falsa cor infravermelho
plotRGB(landsat_rc, r = 5, g = 4, b = 3, stretch = "hist")
```

9. Índices espectrais

Normalized Difference Vegetation Index (NDVI)

RStoolbox

```
# package RStoolbox
# install.packages("RStoolbox")
library(RStoolbox)
```



[RStoolbox]

9. Índices espectrais

Normalized Difference Vegetation Index (NDVI)

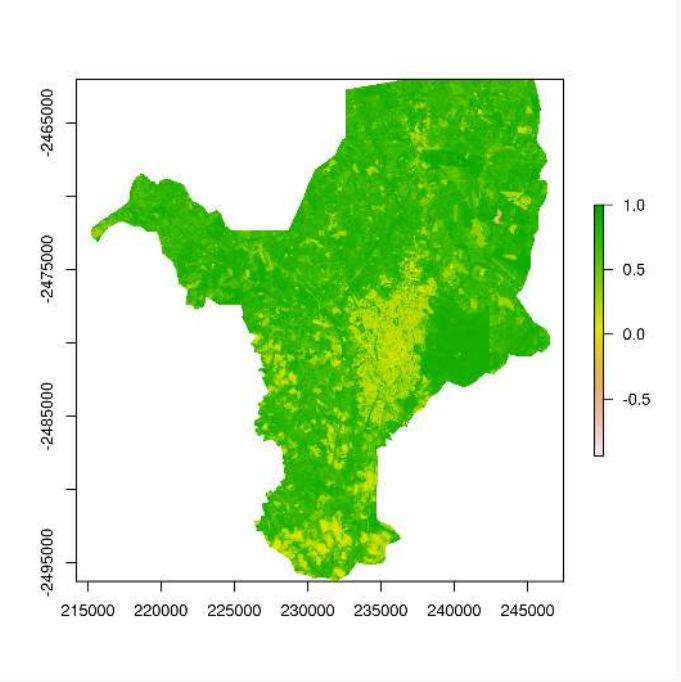
```
# ndvi
landsat_rc_ndvi ← RStoolbox::spectralIndices(img = landsat_rc, red = 4, nir = 5, indices = "NDVI")
landsat_rc_ndvi
```

```
## class      : RasterLayer
## dimensions : 1142, 1047, 1195674  (nrow, ncol, ncell)
## resolution : 30, 30  (x, y)
## extent     : 215145, 246555, -2496285, -2462025  (xmin, xmax, ymin, ymax)
## crs        : +proj=utm +zone=23 +datum=WGS84 +units=m +no_defs
## source     : memory
## names      : NDVI
## values     : -0.9387755, 1  (min, max)
```

9. Índices espectrais

Normalized Difference Vegetation Index (NDVI)

```
# plot
plot(landsat_rc_ndvi)
```

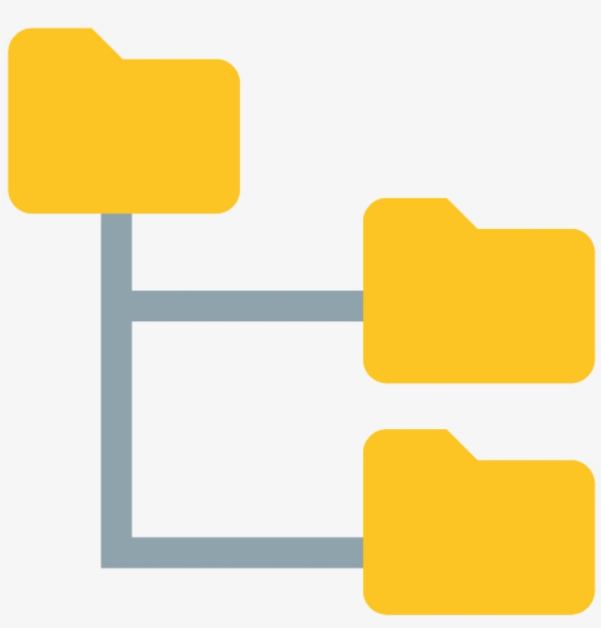


10. Exportar objetos raster

Exportar raster

Criar um diretório

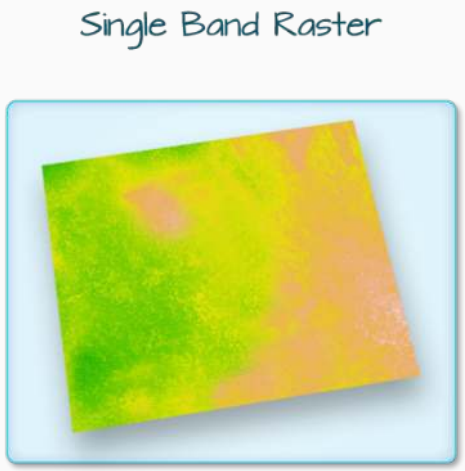
```
# diretorio
dir.create(here::here("03_dados", "raster", "exportados"))
```



10. Exportar objetos raster

Exportar raster

```
# exportar raster layer
raster::writeRaster(dem_rc,
                    filename = here::here("03_dados", "raster", "exportados", "elevation_rio_claro"),
                    format = "GTiff",
                    datatype = "INT2S",
                    options = c("COMPRESS=DEFLATE", "TFW=YES"),
                    progress = "text",
                    overwrite = TRUE)
```



10. Exportar objetos raster

Exportar stack ou brick

```
# exportar raster stack ou brick
raster::writeRaster(x = bioclim[[1:3]],
                    filename = here::here("03_dados", "raster", "exportados", names(bioclim[[1:3]])),
                    bylayer = TRUE,
                    format = "GTiff",
                    datatype = "INT2S",
                    options = c("COMPRESS=DEFLATE", "TFW=YES"),
                    progress = "text",
                    overwrite = TRUE)
```

Dúvidas?

Maurício Vancine

Contatos:

mauricio.vancine@gmail.com

@mauriciovancine

mauriciovancine

mauriciovancine.github.io

Slides criados via pacote xaringan e tema Metropolis. Animação dos sapos por @probzz.