# Introdução à análise geoespacial com R

## 6 Estrutura e manipulação de dados vetoriais

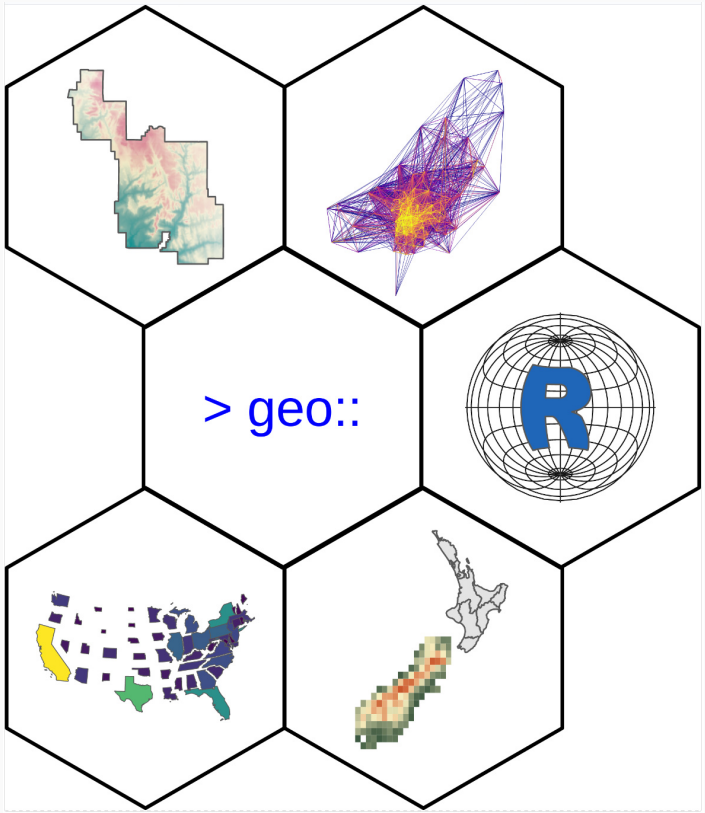
Maurício H. Vancine
Milton C. Ribeiro
22/10/2020


Fonte: @allison_horst

# 6 Estrutura e manipulação de vetores

## Tópicos

1. Pacotes
2. Geometrias sf
3. Classes sf
4. Importar dados vetoriais
5. Descrição de objetos sf
6. Converter dados para sf
7. Converter CRS
8. Operações de atributos
9. Operações espaciais
10. Operações geométricas
11. Exportar dados vetoriais



# 6 Estrutura e manipulação de vetores

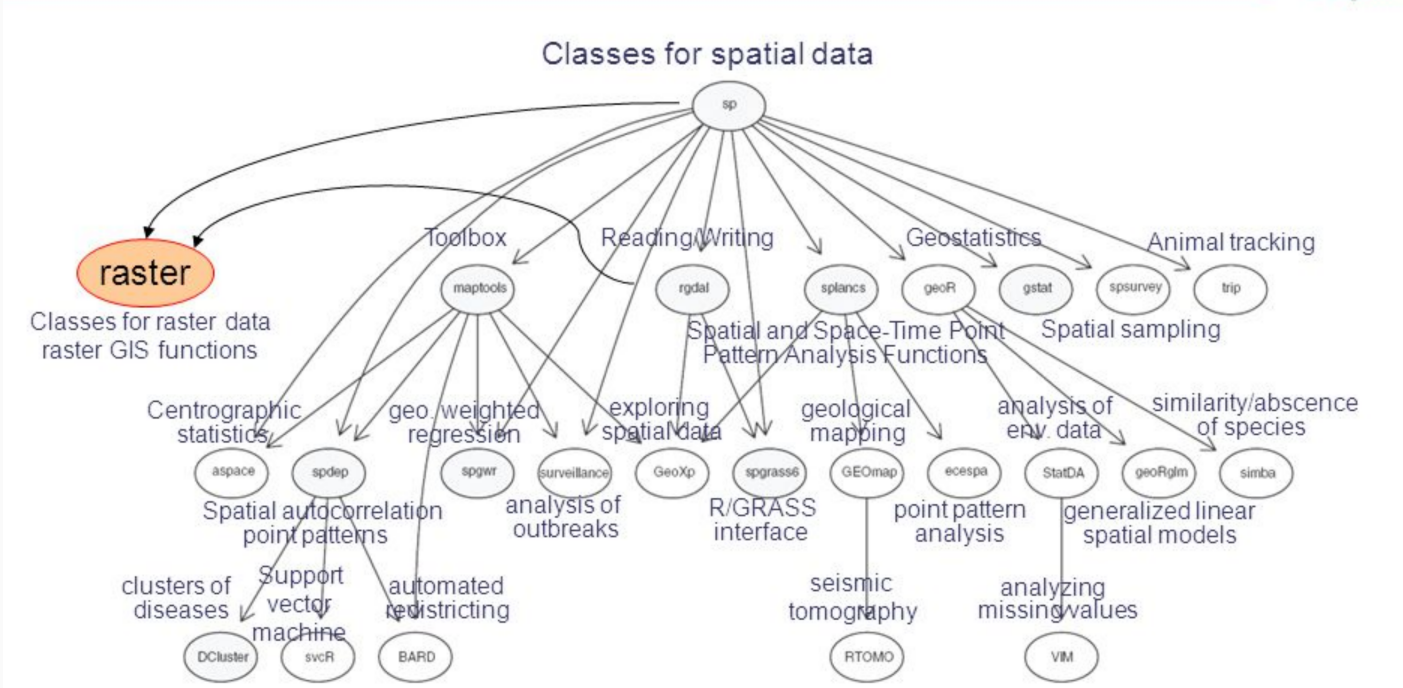
## Script

`06_script_intro_geocomp_r.R`

# 6.1 Pacotes

## Pacote sp

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

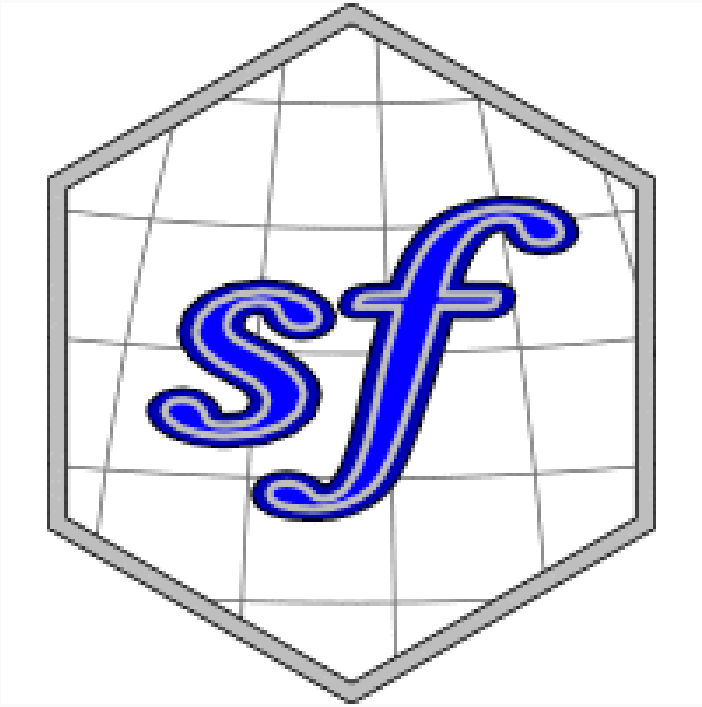



[*] https://github.com/edzer/sp/

# 6.1 Pacotes

## Pacote sf

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

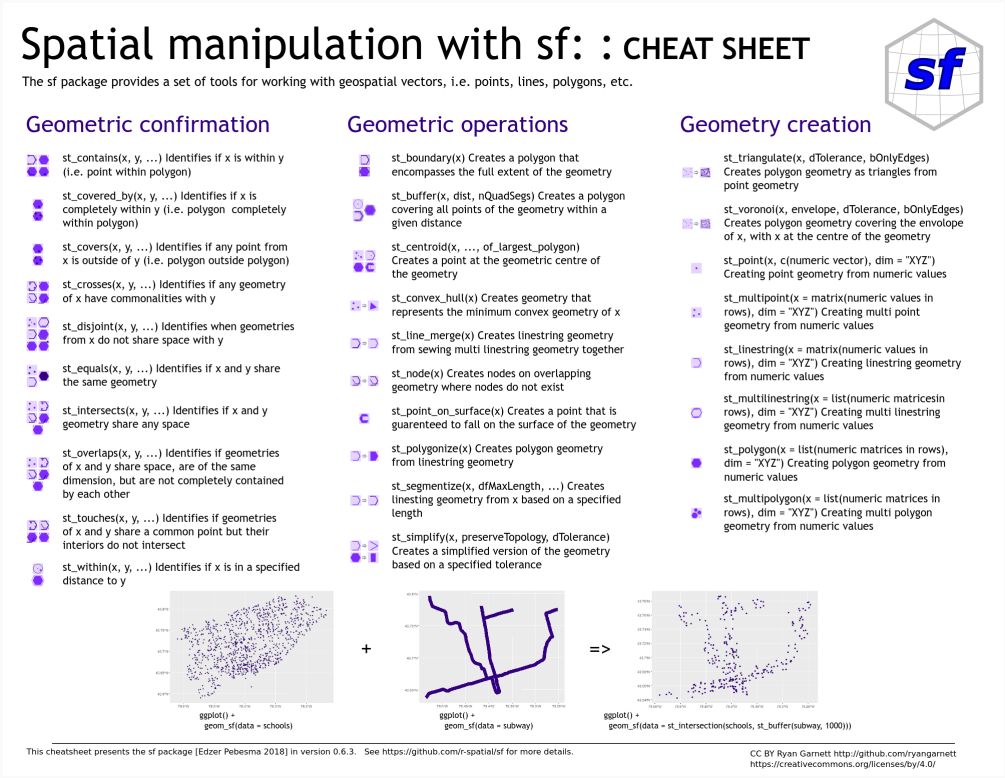



[*] https://r-spatial.github.io/sf/

# 6.1 Pacotes

## Pacote sf

### Cheatsheets



Spatial manipulation with sf: : CHEAT SHEET

The sf package provides a set of tools for working with geospatial vectors, i.e. points, lines, polygons, etc.

**Geometric confirmation**

st_contains(x, y, ...) Identifies if x is within y (i.e. point within polygon)

st_covered_by(x, y, ...) Identifies if x is completely within y (i.e. polygon completely within polygon)

st_covers(x, y, ...) Identifies if any point from x is outside of y (i.e. polygon outside polygon)

st_crosses(x, y, ...) Identifies if any geometry of x have commonalities with y

st_disjoint(x, y, ...) Identifies when geometries from x do not share space with y

st_equals(x, y, ...) Identifies if x and y share the same geometry

st_intersects(x, y, ...) Identifies if x and y geometry share any space

st_overlaps(x, y, ...) Identifies if geometries of x and y share space, are of the same dimension, but are not completely contained by each other

st_touches(x, y, ...) Identifies if geometries of x and y share a common point but their interiors do not intersect

st_within(x, y, ...) Identifies if x is in a specified distance to y

**Geometric operations**

st_boundary(x) Creates a polygon that encompasses the full extent of the geometry

st_buffer(x, dist, nQuadSegs) Creates a polygon covering all points of the geometry within a given distance

st_centroid(x, ..., of_largest_polygon) Creates a point at the geometric centre of the geometry

st_convex_hull(x) Creates geometry that represents the minimum convex geometry of x

st_line_merge(x) Creates linestring geometry from sewing multi linestring geometry together

st_node(x) Creates nodes on overlapping geometry where nodes do not exist

st_point_on_surface(x) Creates a point that is guarenteed to fall on the surface of the geometry

st_polygonize(x) Creates polygon geometry from linestring geometry

st_segmentize(x, dfMaxLength, ...) Creates linestring geometry from x based on a specified length

st_simplify(x, preserveTopology, dTolerance) Creates a simplified version of the geometry based on a specified tolerance

**Geometry creation**

st_triangulate(x, dTolerance, bOnlyEdges) Creates polygon geometry as triangles from point geometry

st_voronoi(x, envelope, dTolerance, bOnlyEdges) Creates polygon geometry covering the envelope of x, with x at the centre of the geometry

st_point(x, c(numeric vector), dim = "XYZ") Creating point geometry from numeric values

st_multipoint(x = matrix(numeric values in rows), dim = "XYZ") Creating multi point geometry from numeric values

st_linestring(x = matrix(numeric values in rows), dim = "XYZ") Creating linestring geometry from numeric values

st_multilinestring(x = list(numeric matricesin rows), dim = "XYZ") Creating multi linestring geometry from numeric values

st_polygon(x = list(numeric matrices in rows), dim = "XYZ") Creating polygon geometry from numeric values

st_multipolygon(x = list(numeric matrices in rows), dim = "XYZ") Creating multi polygon geometry from numeric values



This cheatsheet presents the sf package [Edzer Pebesma 2018] in version 0.6.3. See https://github.com/r-spatial/sf for more details.



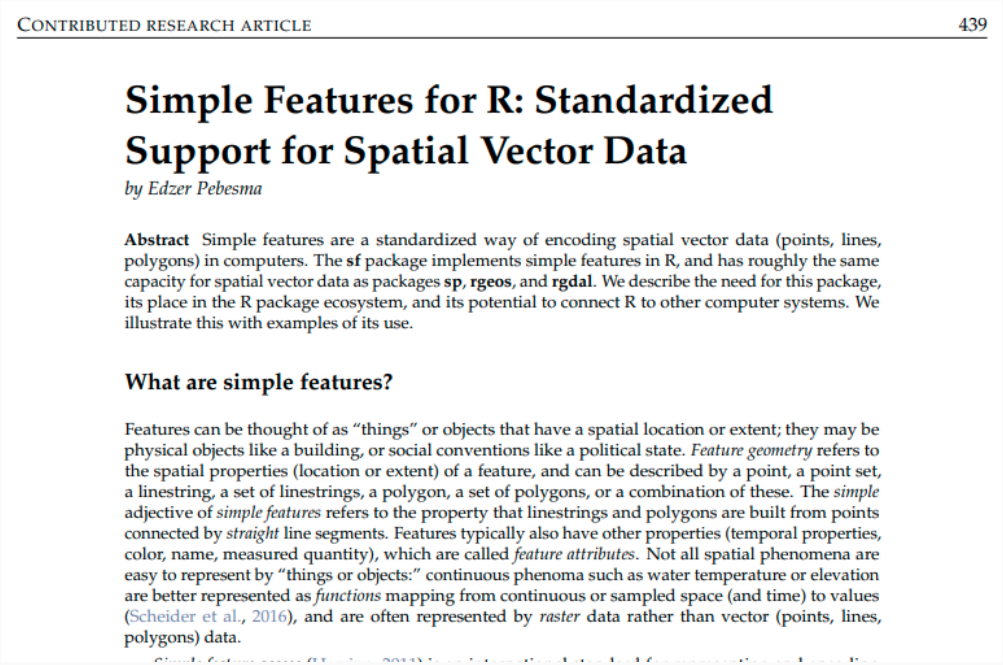
[*] https://raw.githubusercontent.com/rstudio/cheatsheets/master/sf.pdf

# 6.1 Pacotes

## Pacote sf

### Artigo

- Pebesma, Edzer. "Simple Features for R: Standardized Support for Spatial Vector Data." The R Journal 10.01 (2018): 439-446.



CONTRIBUTED RESEARCH ARTICLE 439

**Simple Features for R: Standardized Support for Spatial Vector Data**

*by Edzer Pebesma*

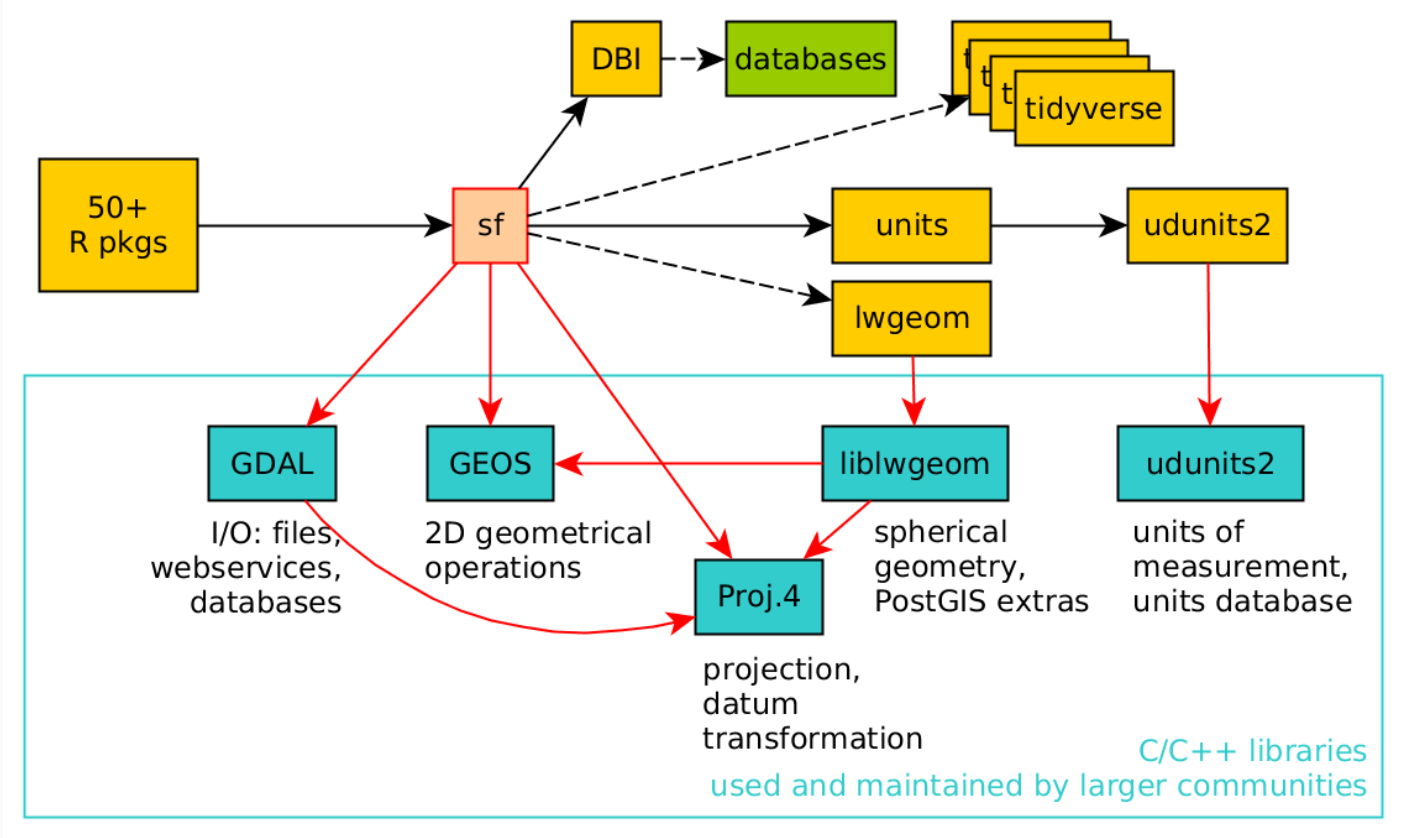
**Abstract** Simple features are a standardized way of encoding spatial vector data (points, lines, polygons) in computers. The **sf** package implements simple features in R, and has roughly the same capacity for spatial vector data as packages **sp**, **rgeos**, and **rgdal**. We describe the need for this package, its place in the R package ecosystem, and its potential to connect R to other computer systems. We illustrate this with examples of its use.

**What are simple features?**

Features can be thought of as "things" or objects that have a spatial location or extent; they may be physical objects like a building, or social conventions like a political state. *Feature geometry* refers to the spatial properties (location or extent) of a feature, and can be described by a point, a point set, a linestring, a set of linestrings, a polygon, a set of polygons, or a combination of these. The *simple* adjective of *simple features* refers to the property that linestrings and polygons are built from points connected by *straight* line segments. Features typically also have other properties (temporal properties, color, name, measured quantity), which are called *feature attributes*. Not all spatial phenomena are easy to represent by "things or objects:" continuous phenoma such as water temperature or elevation are better represented as *functions* mapping from continuous or sampled space (and time) to values (Scheider et al., 2016), and are often represented by *raster* data rather than vector (points, lines, polygons) data.

## Pacote sf

### Dependências de outros pacotes



# 6.1 Pacotes

## Pacote sf

### Métodos

| class | methods |
|---|---|
| sfg | as.matrix, c, coerce, format, head, Ops, plot, print, st_as_binary, st_as_grob, st_as_text, st_transform, st_coordinates, st_geometry, st_boundary, st_buffer, st_centroid, st_convex_hull, st_difference, st_intersection, st_line_merge, st_make_valid, st_node, st_point_on_surface, st_polygonize, st_segmentize, st_simplify, st_split, st_sym_difference, st_triangulate, st_union, st_voronoi, st_cast, st_collection_extract, st_is, st_zm |
| sfc | [, [<-, as.data.frame, c, coerce, format, Ops, print, rep, st_as_binary, st_as_text, st_bbox, st_coordinates, st_crs, st_crs<-, st_geometry, st_precision, st_set_precision, str, summary, st_boundary, st_buffer, st_centroid, st_convex_hull, st_difference, st_intersection, st_line_merge, st_make_valid, st_node, st_point_on_surface, st_polygonize, st_segmentize, st_simplify, st_split, st_sym_difference, st_transform, st_triangulate, st_union, st_voronoi, st_cast, st_collection_extract, st_is, st_zm, obj_sum, type_sum |
| sf | [, [[<-, $<-, aggregate, cbind, coerce, merge, plot, print, rbind, st_agr, st_agr<-, st_bbox, st_coordinates, st_crs, st_crs<-, st_geometry, st_geometry<-, st_precision, st_set_precision, st_transform, st_boundary, st_buffer, st_centroid, st_convex_hull, st_difference, st_intersection, st_line_merge, st_make_valid, st_node, st_point_on_surface, st_polygonize, st_segmentize, st_simplify, st_split, st_sym_difference, st_triangulate, st_union, st_voronoi, st_cast, st_collection_extract, st_is, st_zm, anti_join, arrange, distinct, filter, full_join, gather, group_by, inner_join, left_join, nest, mutate, rename, right_join, sample_frac, sample_n, select, semi_join, separate, slice, spread, summarise, transmute, ungroup, unite |
| crs | $, is.na, Ops, print, st_as_text, st_crs |

## Pacote sf

### Funções

| category | functions |
|---|---|
| binary predicates | st_contains, st_contains_properly, st_covered_by, st_covers, st_crosses, st_disjoint, st_equals, st_equals_exact st_intersects, st_is_within_distance, st_within, st_touches, st_overlaps |
| binary operations | st_relate, st_distance |
| unary operations | st_dimension, st_area, st_length, st_is_longlat, st_is_simple, st_is_valid, st_jitter, st_geohash, st_geometry_type |
| miscellaneous | st_sample, st_line_sample, st_join, st_interpolate_aw, st_make_grid, st_graticule, sf_extSoftVersion, rawToHex, st_proj_info |
| setters | st_set_agr, st_set_crs |
| constructors | st_sfc, st_sf, st_as_sf, st_as_sfc, st_point, st_multipoint, st_linestring, st_multilinestring, st_polygon, st_multipolygon, st_geometrycollection, st_combine, st_bind_cols |
| in- & output | st_read, st_read_db, st_write, st_write_db, read_sf, write_sf, st_drivers, st_layers |
| plotting | st_viewport, st_wrap_dateline, sf.colors |

# 6.1 Pacotes

## Pacote sf

**Rápida importação e exportação** de dados

**Desempenho aprimorado** de mapas e gráficos

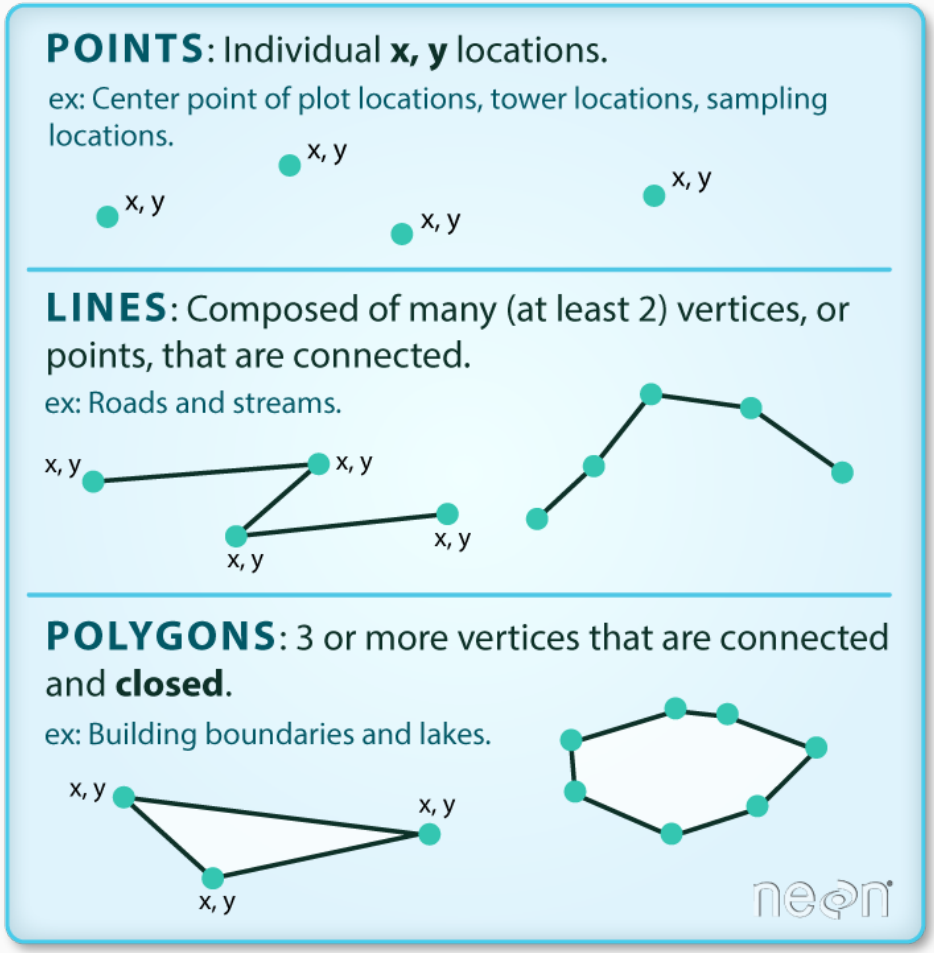
Objetos `sf` podem ser tratados como `data frames` (`tibbles`) na maioria das **operações de manipulação**

As funções `sf` **podem ser combinadas** usando o operador `%>%` e funcionam bem com `tidyverse`

Os nomes das funções sf são relativamente **consistentes e intuitivos** (todos começam com `sf::st_`)

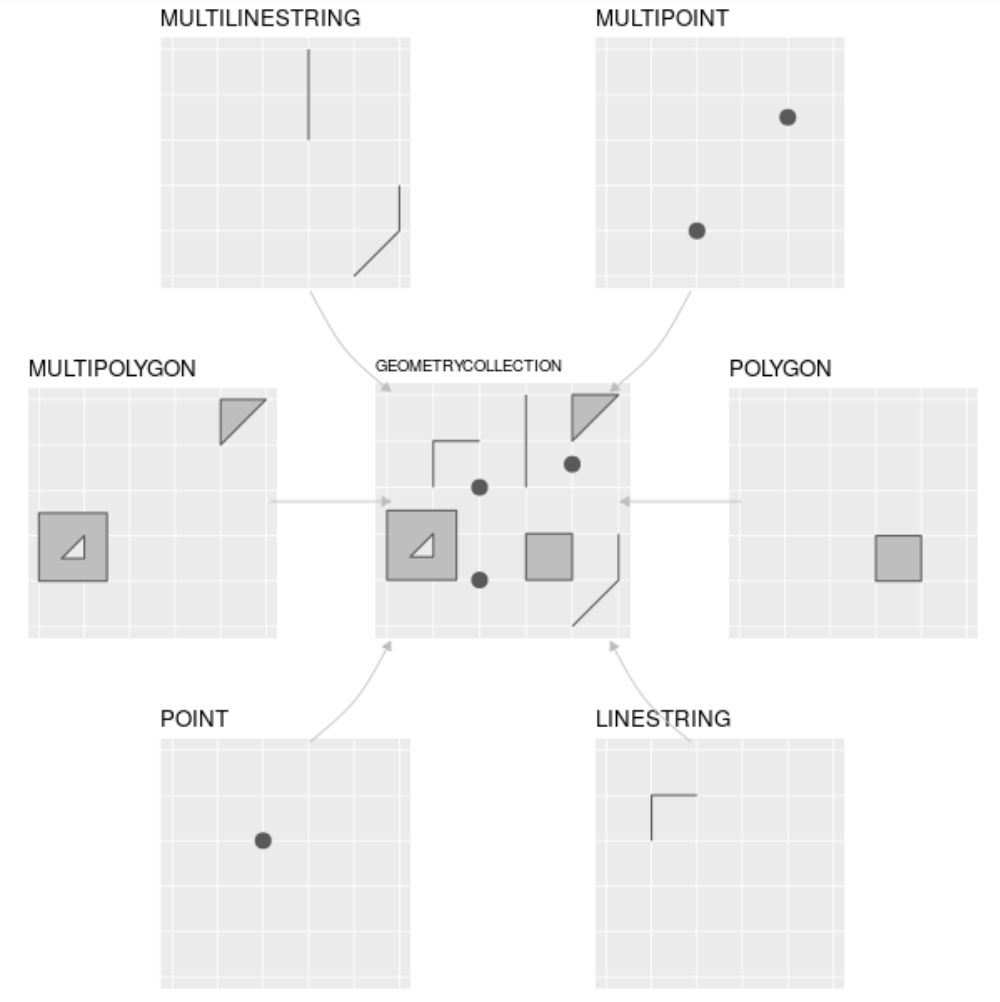
# 6.2 Tipos de geometrias sf

## Tipos de dados geoespaciais vetoriais



# 6.2 Tipos de geometrias sf

## Tipos de geometrias



# 6.3 Classes sf

## Simple feature geometries (sfg)

A classe `sfg` representa os **diferentes tipos de geometrias** no R: ponto, linha, polígono (e seus equivalentes 'multi') ou coleção de geometrias

Funções para criar geometrias da classe `sfg`:

```
# simple
sf::st_point()
sf::st_linestring()
sf::st_polygon()

# multi
sf::st_multipoint()
sf::st_multilinestring()
sf::st_multipolygon()

# collections
sf::st_geometrycollection()
```

# 6.3 Classes sf

## Simple feature geometries (sfg)

Os objetos `sfg` podem ser **criados** a partir de **três tipos de dados R base**:

1. **vetor numérico**: um único ponto

2. **matriz**: um conjunto de pontos, onde cada linha representa um ponto, um multiponto ou uma linha

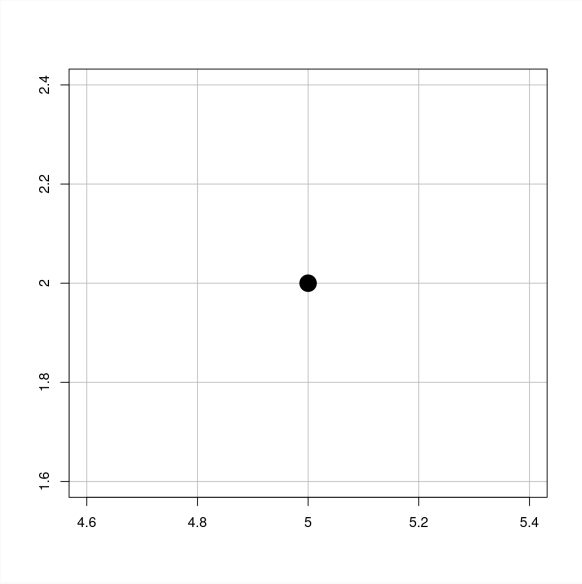
3. **lista**: uma coleção de objetos como matrizes, cadeias multilinha ou coleções de geometrias

# 6.3 Classes sf

## Simple feature geometries (sfg)

```
# vector - point
vec ← c(5, 2)
po ← sf::st_point(vec)
po
```

```
# plot
plot(po, pch = 20, cex = 4, axes = TRUE, graticule = TRUE)
```

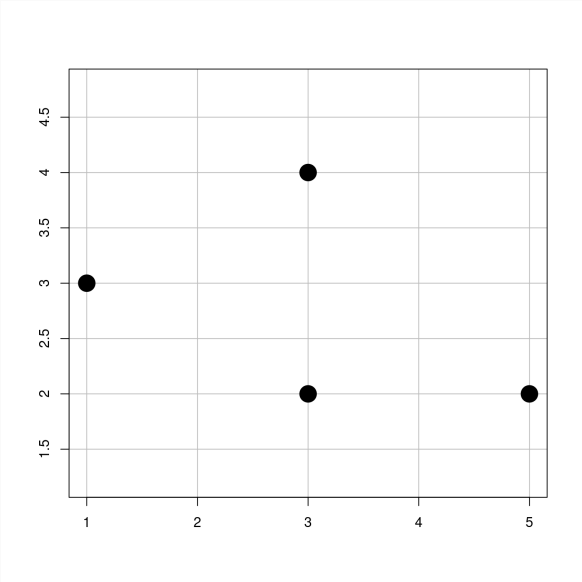


# 6.3 Classes sf

## Simple feature geometries (sfg)

```
# matrix - multipoint
multipoint_matrix ← rbind(c(5, 2), c(1, 3), c(3, 4), c(3, 2))
po_mul ← sf::st_multipoint(multipoint_matrix)
po_mul
```

```
# plot
plot(po_mul, pch = 20, cex = 4, axes = TRUE, graticule = TRUE)
```

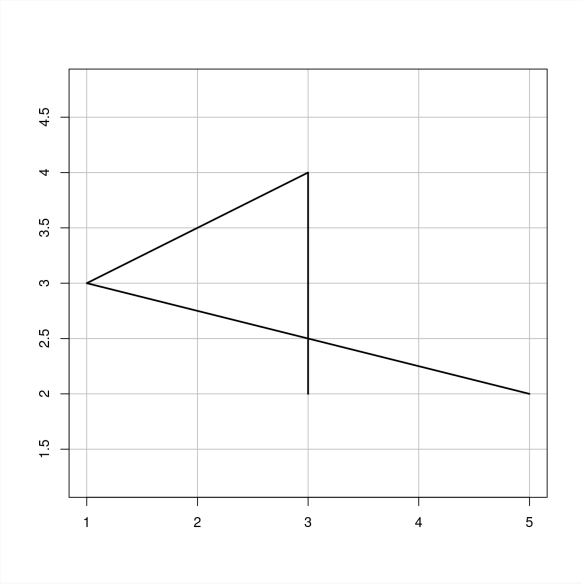


# 6.3 Classes sf

## Simple feature geometries (sfg)

```
# matrix - multipoint
multipoint_matrix ← rbind(c(5, 2), c(1, 3), c(3, 4), c(3, 2))
lin ← sf::st_linestring(multipoint_matrix)
lin
```

```
# plot
plot(lin, lwd = 2, axes = TRUE, graticule = TRUE)
```

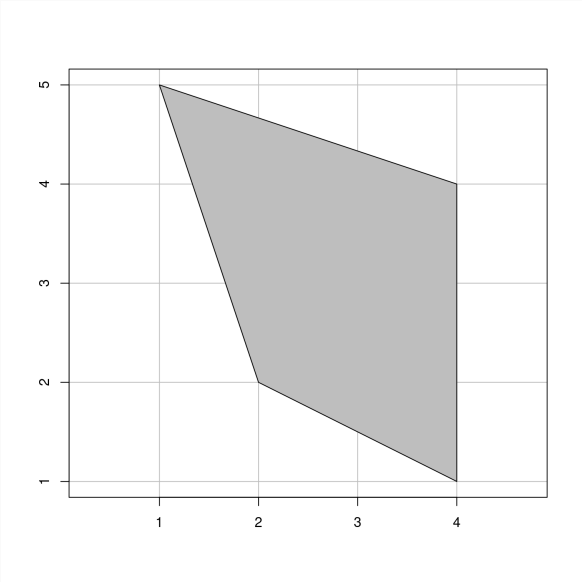


# 6.3 Classes sf

## Simple feature geometries (sfg)

```
# list - polygon
polygon_list ← list(rbind(c(1, 5), c(2, 2), c(4, 1), c(4, 4), c(1, 5)))
pol ← sf::st_polygon(polygon_list)
pol
```

```
# plot
plot(pol, col = "gray", axes = TRUE, graticule = TRUE)
```



# 6.3 Classes sf

## Simple feature columns (sfc)

Uma **lista de objetos** `sfg` que possui o **mesmo CSR** (*Coordinate Reference System*)

Pode conter objetos da **mesma geometria** ou de **geometrias diferentes**

Funções para **criar** a classe `sfc` e verificar o **tipo da geometria e CRS**:

```
sf::st_sfc()
sf::st_geometry_type()
sf::st_crs()
```

# 6.3 Classes sf

## Simple feature columns (sfc)

```
# sfc point
point1 ← sf::st_point(c(5, 2))
point2 ← sf::st_point(c(1, 3))
points_sfc ← sf::st_sfc(point1, point2)
points_sfc
```

```
## Geometry set for 2 features
## geometry type:  POINT
## dimension:      XY
## bbox:           xmin: 1 ymin: 2 xmax: 5 ymax: 3
## CRS:            NA
```

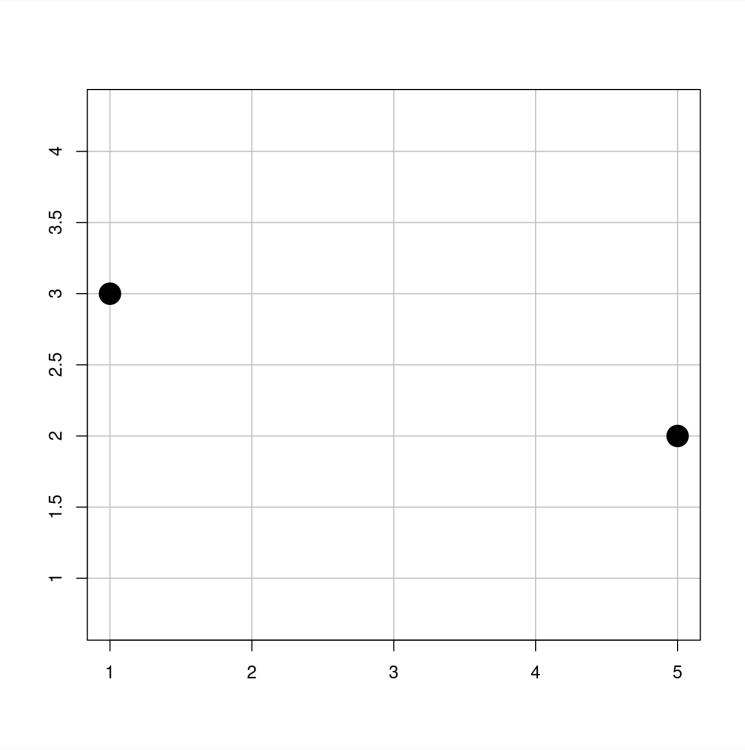
```
sf::st_geometry_type(points_sfc)
```

```
## [1] POINT POINT
## 18 Levels: GEOMETRY POINT LINESTRING POLYGON MULTIPOINT MULTILINESTRING MULTIPOLYGON
```

# 6.3 Classes sf

## Simple feature columns (sfc)

```
# plot
plot(points_sfc, pch = 20, cex = 4, axes = TRUE, graticule = TRUE)
```



# 6.3 Classes sf

## Simple feature columns (sfc)

```
# sfc geometry
po_pol_sfc ← sf::st_sfc(po, pol)
po_pol_sfc
```

```
## Geometry set for 2 features
## geometry type:  GEOMETRY
## dimension:      XY
## bbox:           xmin: 1 ymin: 1 xmax: 5 ymax: 5
## CRS:            NA
```

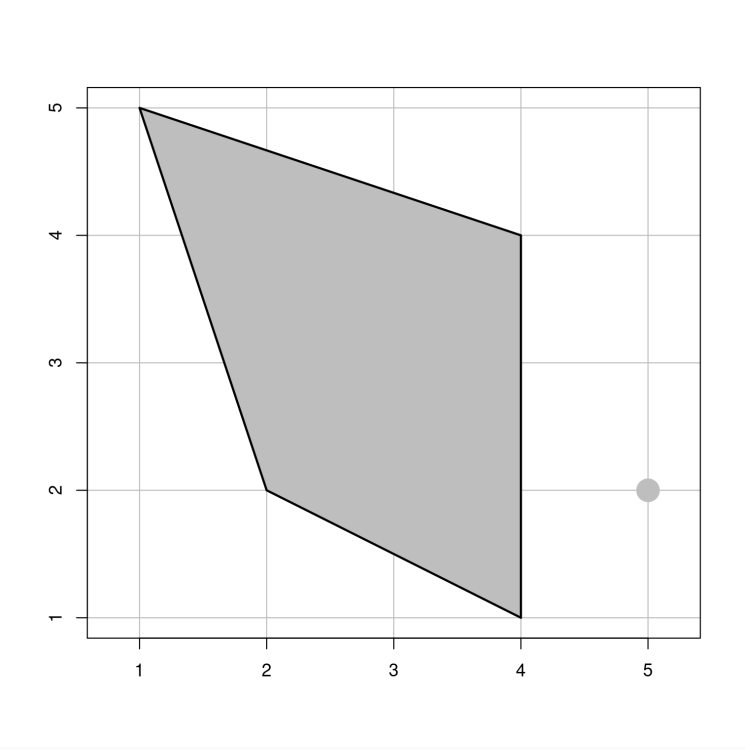
```
sf::st_geometry_type(po_pol_sfc)
```

```
## [1] POINT   POLYGON
## 18 Levels: GEOMETRY POINT LINESTRING POLYGON MULTIPOINT MULTILINESTRING MULTIPOLYGON
```

# 6.3 Classes sf

## Simple feature columns (sfc)

```
# plot
plot(po_pol_sfc, pch = 20, cex = 4, lwd = 2, col = "gray", axes = TRUE, graticule
```



# 6.3 Classes sf

## Simple feature columns (sfc)

### Coordinate Reference Systems (CRS) - EPSG

```
# epsg definition
points_sfc_wgs ← sf::st_sfc(point1, point2, crs = 4326)
sf::st_crs(points_sfc_wgs)
```

```
## Coordinate Reference System:
##   User input: EPSG:4326
##   wkt:
## GEOGCRS["WGS 84",
##     DATUM["World Geodetic System 1984",
##         ELLIPSOID["WGS 84",6378137,298.257223563,
##             LENGTHUNIT["metre",1]]],
##     PRIMEM["Greenwich",0,
##         ANGLEUNIT["degree",0.0174532925199433]],
##     CS[ellipsoidal,2],
##         AXIS["geodetic latitude (Lat)",north,
##             ORDER[1],
##             ANGLEUNIT["degree",0.0174532925199433]],
##         AXIS["geodetic longitude (Lon)",east,
```

# 6.3 Classes sf

## Simple feature columns (sfc)

### Coordinate Reference Systems (CRS) - proj4string

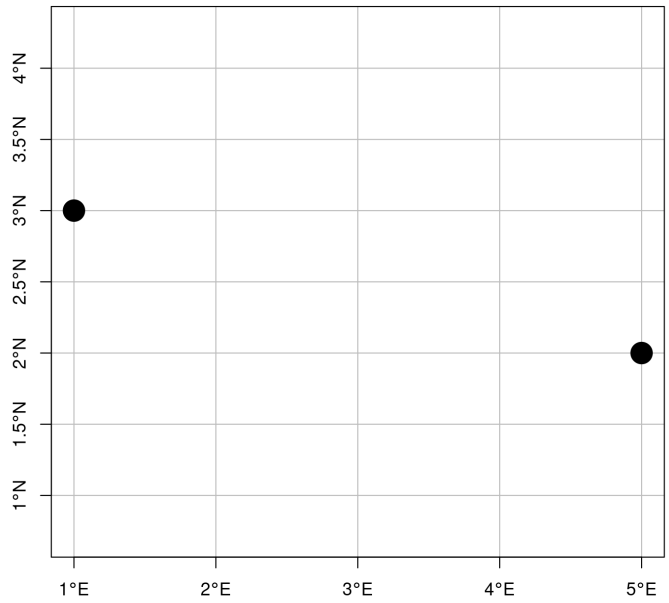
```
# proj4string definition
points_sfc_wgs ← sf::st_sfc(point1, point2, crs = "+proj=longlat +datum=WGS84 +no
sf::st_crs(points_sfc_wgs)
```

```
## Coordinate Reference System:
##   User input: +proj=longlat +datum=WGS84 +no_defs
##   wkt:
## GEOGCRS["unknown",
##     DATUM["World Geodetic System 1984",
##         ELLIPSOID["WGS 84",6378137,298.257223563,
##             LENGTHUNIT["metre",1]],
##         ID["EPSG",6326]],
##     PRIMEM["Greenwich",0,
##         ANGLEUNIT["degree",0.0174532925199433],
##         ID["EPSG",8901]],
##     CS[ellipsoidal,2],
##         AXIS["longitude",east,
##             ORDER[1],
```

# 6.3 Classes sf

## Simple feature columns (sfc)

```
# plot
plot(points_sfc_wgs, pch = 20, cex = 4, axes = TRUE, graticule = TRUE)
```



# 6.3 Classes sf

## A classe sf

A classe sf (simple features) são *data.frames* com **atributos espaciais** armazenados em uma coluna, geralmente chamada de **geometry**

Essa **dualidade** é central para o conceito de **simple features**: na maioria das vezes, um sf pode **ser tratado e se comporta** como um *data.frame*

**Simple features** são, em essência, *data.frames* com uma **extensão espacial**

# 6.3 Classes sf

## A classe sf

### Criar um objeto sf

```
rc_point ← sf::st_point(c(-47.57,-22.39))          # sfg object
rc_geom ← sf::st_sfc(rc_point, crs = 4326)         # sfc object
rc_attrib ← data.frame(                           # data.frame object
  name = "Rio Claro",
  temperature = 19,
  date = as.Date("2020-10-13")
)
rc_sf ← sf::st_sf(rc_attrib, geometry = rc_geom)  # sf object
```

# 6.3 Classes sf

## A classe sf

### A estrutura de um objeto sf

```
rc_sf
```

```
## Simple feature collection with 1 feature and 3 fields
## geometry type:  POINT
## dimension:      XY
## bbox:           xmin: -47.57 ymin: -22.39 xmax: -47.57 ymax: -22.39
## geographic CRS: WGS 84
##        name temperature       date              geometry
## 1 Rio Claro          19 2020-10-13 POINT (-47.57 -22.39)
```

### As duas classes do objeto sf

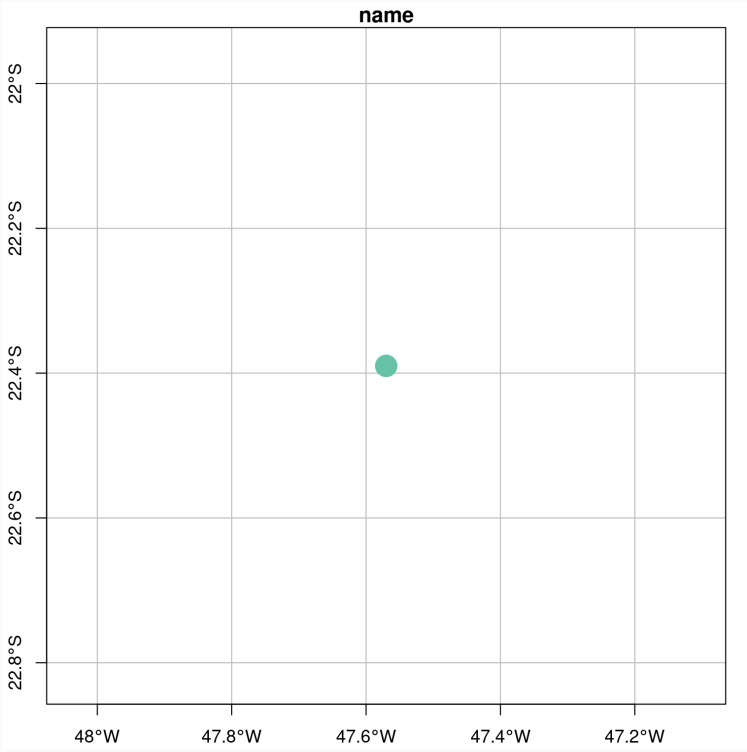
```
class(rc_sf)
```

```
## [1] "sf"         "data.frame"
```

# 6.3 Classes sf

## A classe sf

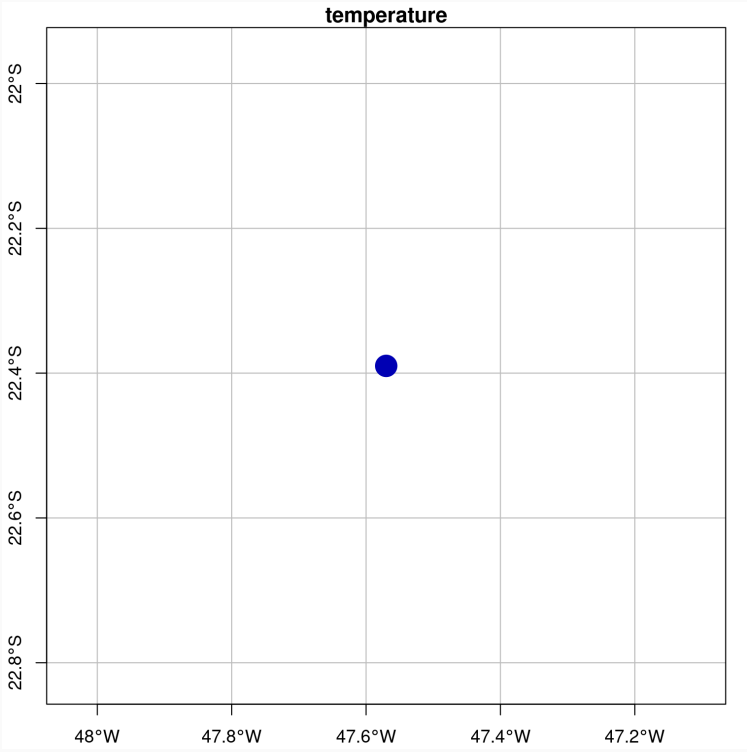
```
# plot
plot(rc_sf[1], pch = 20, cex = 4, axes = TRUE, graticule = TRUE) # rc_sf[1] - plot
```



# 6.3 Classes sf

## A classe sf

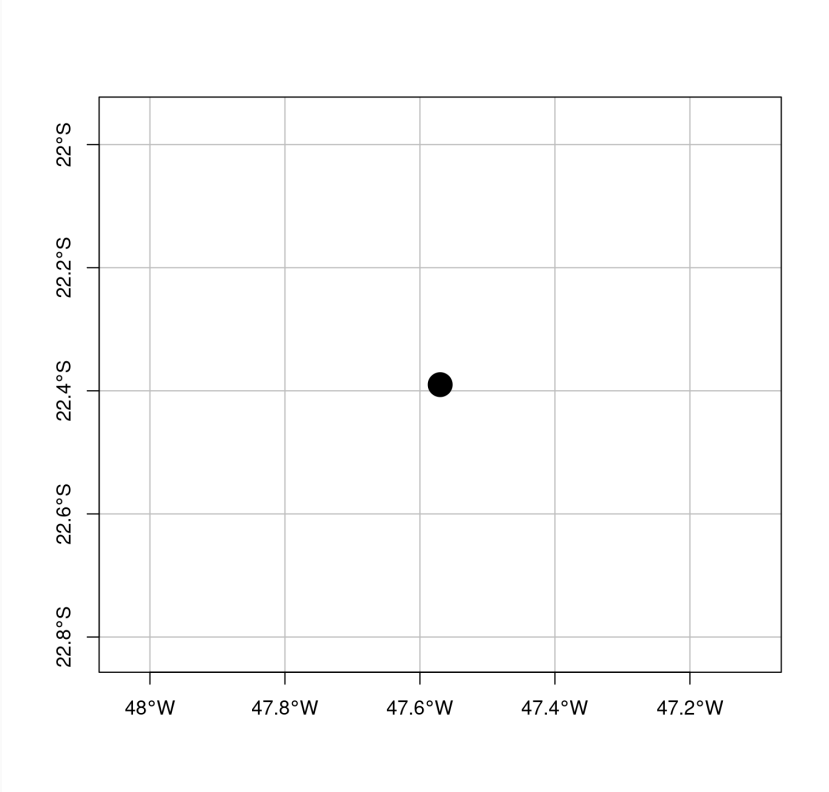
```
# plot
plot(rc_sf[2], pch = 20, cex = 4, axes = TRUE, graticule = TRUE) # rc_sf[2] - plot
```



# 6.3 Classes sf

## A classe sf

```
# plot
plot(rc_sf$geometry, pch = 20, cex = 4, axes = TRUE, graticule = TRUE) # rc_sf$geo
```



# Dúvidas?

Vamos começar fazendo o download de vetores pelo R

# 6.4 Importar dados

## Fundação Brasileira Desenvolvimento Sustentável (FBDS)

Em 2015, a FBDS deu início ao *Projeto de Mapeamento em Alta Resolução dos Biomas Brasileiros*:

- Cobertura da terra
- Hidrografia (nascentes, rios e lagos)
- Áreas de Preservação Permanente (APP)

O mapeamento foi concluído para os municípios dos biomas *Mata Atlântica e Cerrado*

Site: https://www.fbds.org.br/
Repositório: http://geo.fbds.org.br/

# 6.4 Importar dados

## Download

### Criar um diretório

```
# create directory
dir.create(here::here("03_dados", "vetor"))
```

### Download de **pontos de nascentes** para Rio Claro/SP

```
# increase time to download
options(timeout = 600)

# download lines
for(i in c(".dbf", ".prj", ".shp", ".shx")){
  download.file(url = paste0("http://geo.fbds.org.br/SP/RIO_CLARO/HIDROGRAFIA/SP_3
                destfile = here::here("03_dados", "vetor", paste0("SP_3543907_NASC
}
```

# 6.4 Importar dados

## Download

### Download de **linhas da hidrografia** para Rio Claro/SP

```
# download lines
for(i in c(".dbf", ".prj", ".shp", ".shx")){
  download.file(url = paste0("http://geo.fbds.org.br/SP/RIO_CLARO/HIDROGRAFIA/SP_3
                destfile = here::here("03_dados", "vetor", paste0("SP_3543907_RIOS
}
```

### Download de **polígonos de cobertura da terra** para Rio Claro/SP

```
# download polygon
for(i in c(".dbf", ".prj", ".shp", ".shx")){
  download.file(url = paste0("http://geo.fbds.org.br/SP/RIO_CLARO/USO/SP_3543907_U
                destfile = here::here("03_dados", "vetor", paste0("SP_3543907_USO"
}
```

# Importar dados preexistentes

# 6.4 Importar dados

## Dados preexistentes

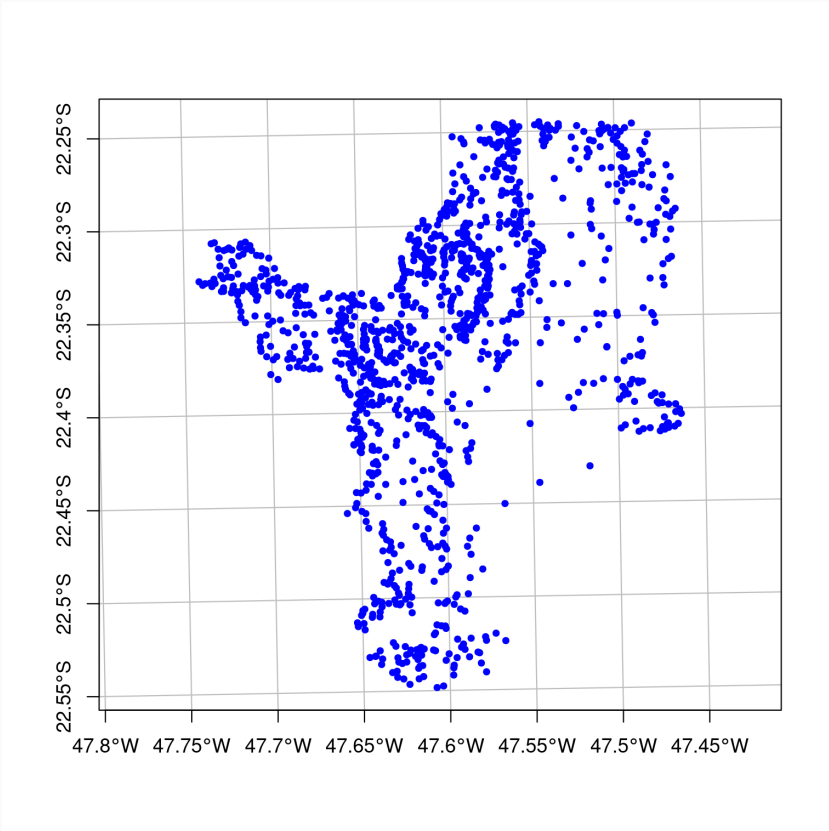
```
# import points
rc_spr ← sf::st_read(here::here("03_dados", "vetor", "SP_3543907_NASCENTES.shp"),
rc_spr
```

```
## Simple feature collection with 1220 features and 5 fields
## geometry type:  POINT
## dimension:      XY
## bbox:           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    3543907 RIO CLARO SP    35 nascente POINT (218535.4 7530642)
## 9    3543907 RIO CLARO SP    35 nascente POINT (218583.5 7528215)
## 10   3543907 RIO CLARO SP    35 nascente POINT (218760.1 7530258)
```

# 6.4 Importar dados

## Dados preexistentes

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



# 6.4 Importar dados

## Dados preexistentes

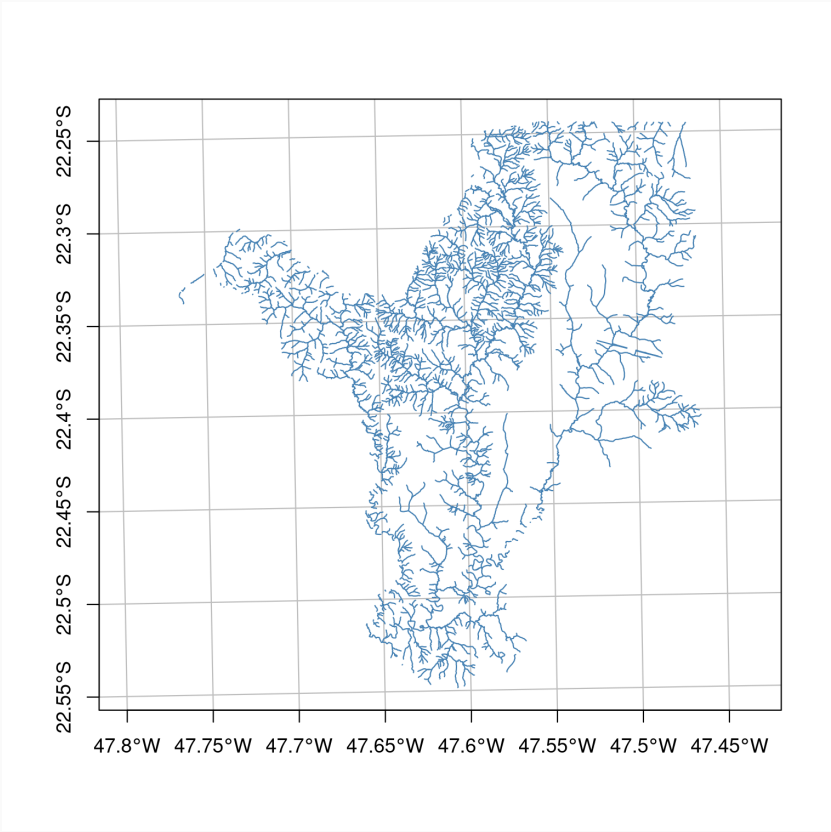
```
# import lines
rc_riv ← sf::st_read(here::here("03_dados", "vetor", "SP_3543907_RIOS_SIMPLES.shp
rc_riv
```

```
## Simple feature collection with 1 feature and 6 fields
## geometry type:  MULTILINESTRING
## dimension:      XY
## bbox:           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
## 1   3543907 RIO CLARO SP    35 curso d'água (0 - 10m) 1142.98 MULTILINESTRING ((2318
```

# 6.4 Importar dados

## Dados preexistentes

```
# plot
plot(rc_riv$geometry, col = "steelblue", main = NA, axes = TRUE, graticule = TRUE)
```



# 6.4 Importar dados

## Dados preexistentes

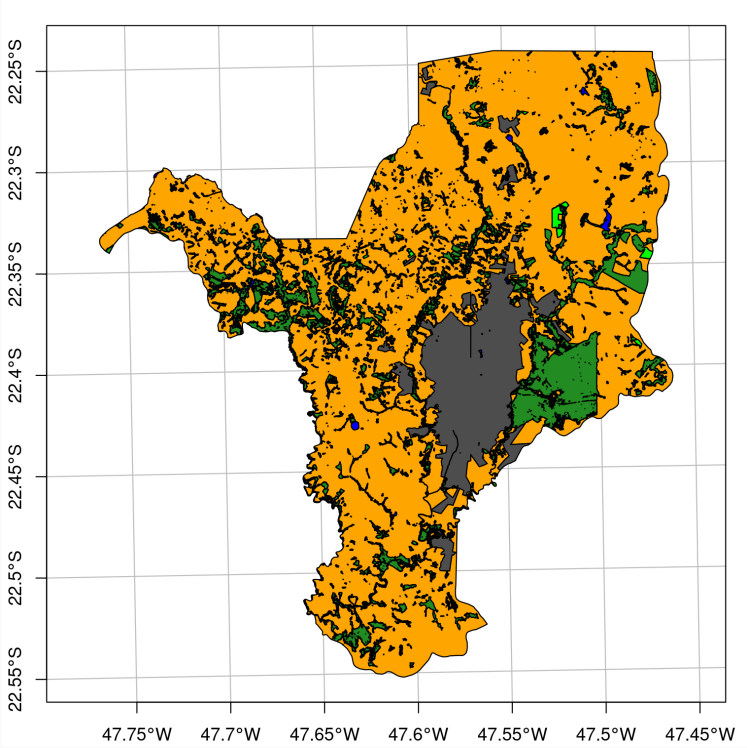
```
# import polygons
rc_use ← sf::st_read(here::here("03_dados", "vetor", "SP_3543907_USO.shp"), quiet
rc_use
```

```
## Simple feature collection with 5 features and 6 fields
## geometry type:  MULTIPOLYGON
## dimension:      XY
## bbox:           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                       ge
## 1   3543907 RIO CLARO SP    35              água   357.027 MULTIPOLYGON (((235487.6
## 2   3543907 RIO CLARO SP    35  área antropizada 37297.800 MULTIPOLYGON (((232275 7
## 3   3543907 RIO CLARO SP    35    área edificada  5078.330 MULTIPOLYGON (((233123.6
## 4   3543907 RIO CLARO SP    35 formação florestal  7017.990 MULTIPOLYGON (((232355 7
## 5   3543907 RIO CLARO SP    35       silvicultura   138.173 MULTIPOLYGON (((243052.1
```

# 6.4 Importar dados

## Dados preexistentes

```
# plot
plot(rc_use[5], col = c("blue", "orange", "gray30", "forestgreen", "green"), main
```



# Importar dados de GPS

Um agradecimento ao Rafael (Urucum), que gentilmente cedeu os dados 😅

# 6.4 Importar dados

## Dados de GPS (.gpx)

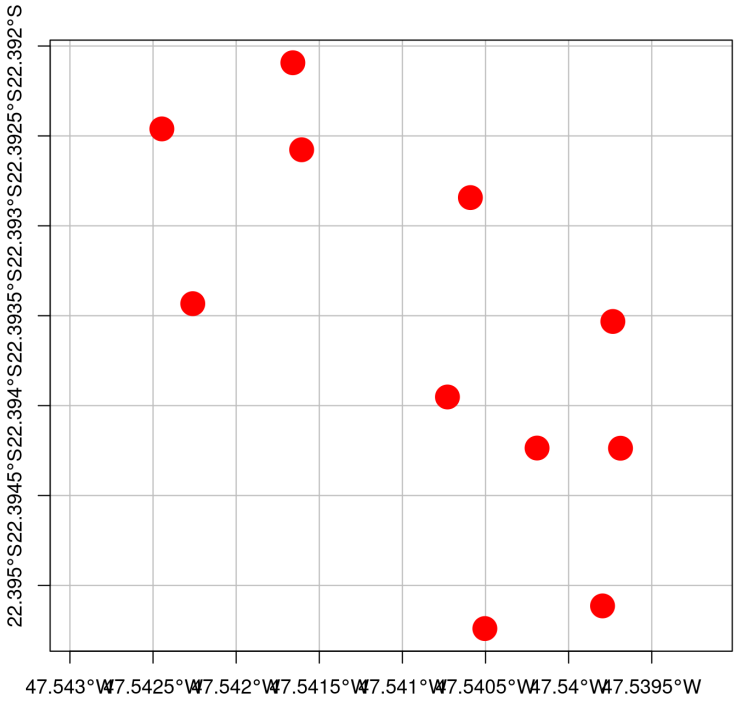
```
# import gps data
gps_gpx ← sf::read_sf(here::here("03_dados", "vetor", "waypoints.gpx"), layer = "
gps_gpx
```

```
## Simple feature collection with 11 features and 23 fields
## geometry type:  POINT
## dimension:      XY
## bbox:           xmin: -47.54245 ymin: -22.39524 xmax: -47.53969 ymax: -22.39209
## geographic CRS: WGS 84
## # A tibble: 11 x 24
##       ele time                magvar geoidheight name  cmt   desc  src   link1_href l
##     <dbl> <dttm>                <dbl>       <dbl> <chr> <chr> <chr> <chr> <chr>      <
##  1  603. 2020-08-20 10:13:35     NA          NA 102   33    <NA>  <NA>  <NA>       <
##  2  595. 2020-08-20 10:22:52     NA          NA 103   71    <NA>  <NA>  <NA>       <
##  3  582. 2020-08-20 10:32:30     NA          NA 104   11    <NA>  <NA>  <NA>       <
##  4  587. 2020-08-20 10:42:05     NA          NA 105   72    <NA>  <NA>  <NA>       <
##  5  581. 2020-08-20 10:49:32     NA          NA 106   68    <NA>  <NA>  <NA>       <
##  6  594. 2020-08-20 11:06:53     NA          NA 107   121   <NA>  <NA>  <NA>       <
##  7  591. 2020-08-20 11:16:36     NA          NA 108   83    <NA>  <NA>  <NA>       <
##  8  594. 2020-08-20 11:27:59     NA          NA 109   162   <NA>  <NA>  <NA>       <
##  9  594. 2020-08-20 11:43:04     NA          NA 110   103   <NA>  <NA>  <NA>       <
```

# 6.4 Importar dados

## Dados de GPS (.gpx)

```
# plot
plot(gps_gpx$geometry, cex = 4, pch = 20, col = "red", main = NA, axes = TRUE, gra
```



# 6.4 Importar dados

## Dados de GPS (.kml)

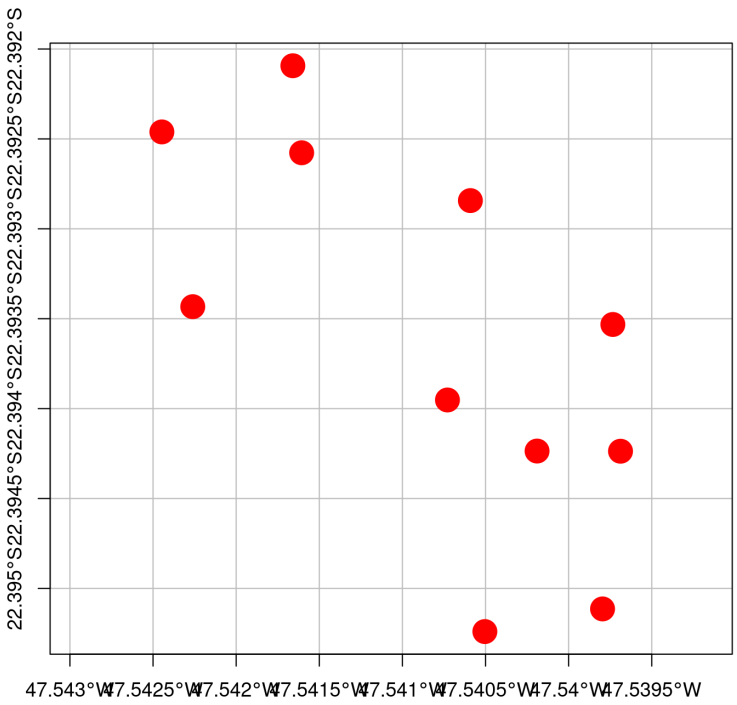
```
# import gps data
gps_kml ← sf::read_sf(here::here("03_dados", "vetor", "waypoints.kml"))
gps_kml
```

```
## Simple feature collection with 11 features and 16 fields
## geometry type:  POINT
## dimension:      XY
## bbox:           xmin: -47.54245 ymin: -22.39524 xmax: -47.53969 ymax: -22.39209
## geographic CRS: WGS 84
## # A tibble: 11 x 17
##    Name  description timestamp           begin               end                 alt
##    <chr> <chr>       <dttm>              <dttm>              <dttm>              <ch
##  1 102   <NA>        NA                  NA                  NA                  <NA
##  2 103   <NA>        NA                  NA                  NA                  <NA
##  3 104   <NA>        NA                  NA                  NA                  <NA
##  4 105   <NA>        NA                  NA                  NA                  <NA
##  5 106   <NA>        NA                  NA                  NA                  <NA
##  6 107   <NA>        NA                  NA                  NA                  <NA
##  7 108   <NA>        NA                  NA                  NA                  <NA
##  8 109   <NA>        NA                  NA                  NA                  <NA
##  9 110   <NA>        NA                  NA                  NA                  <NA
```

# 6.4 Importar dados

## Dados de GPS (.kml)

```
# plot
plot(gps_kml$geometry, cex = 4, pch = 20, col = "red", main = NA, axes = TRUE, gra
```



# Importar dados de pacotes

# 6.4 Importar dadosf

## Dados de pacotes: geobr

```
install.packages("geobr")
library(geobr)
```


[*] https://github.com/ipeaGIT/geobr

# 6.4 Importar dados

## Dados de pacotes: geobr

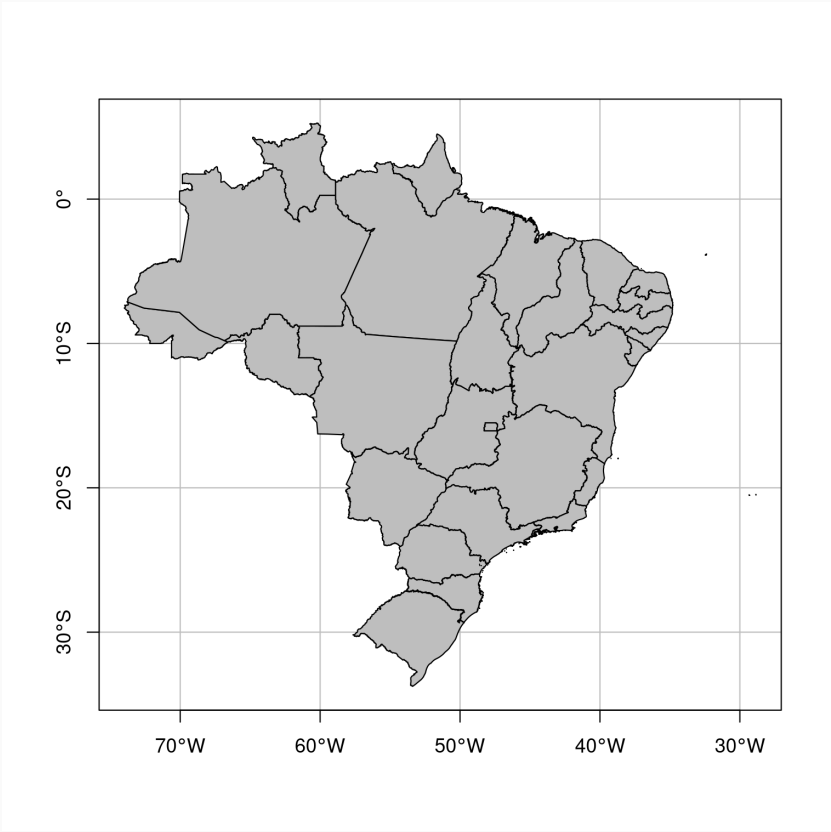
```
# brazil 2019
br_2019 <- geobr::read_country(year = 2019, showProgress = FALSE)
br_2019
```

```
## Simple feature collection with 27 features and 5 fields
## geometry type:  MULTIPOLYGON
## dimension:      XY
## bbox:           xmin: -73.99045 ymin: -33.75118 xmax: -28.84784 ymax: 5.271841
## geographic CRS: SIRGAS 2000
## First 10 features:
##    code_state abbrev_state name_state code_region name_region
## 1          11           RO   Rondônia           1       Norte MULTIPOLYGON (((-65.38
## 2          12           AC       Acre           1       Norte MULTIPOLYGON (((-71.07
## 3          13           AM   Amazônas           1       Norte MULTIPOLYGON (((-69.83
## 4          14           RR    Roraima           1       Norte MULTIPOLYGON (((-63.96
## 5          15           PA       Pará           1       Norte MULTIPOLYGON (((-51.43
## 6          16           AP      Amapá           1       Norte MULTIPOLYGON (((-53.27
## 7          17           TO  Tocantins           1       Norte MULTIPOLYGON (((-48.23
## 8          21           MA   Maranhão           2    Nordeste MULTIPOLYGON (((-46.16
## 9          22           PI      Piauí           2    Nordeste MULTIPOLYGON (((-42.91
## 10         23           CE      Ceará           2    Nordeste MULTIPOLYGON (((-41.18
```

# 6.4 Importar dados

## Dados de pacotes: geobr

```
# plot
plot(br_2019$geom, col = "gray", main = NA, axes = TRUE, graticule = TRUE)
```



# 6.4 Importar dados

## Dados de pacotes: geobr

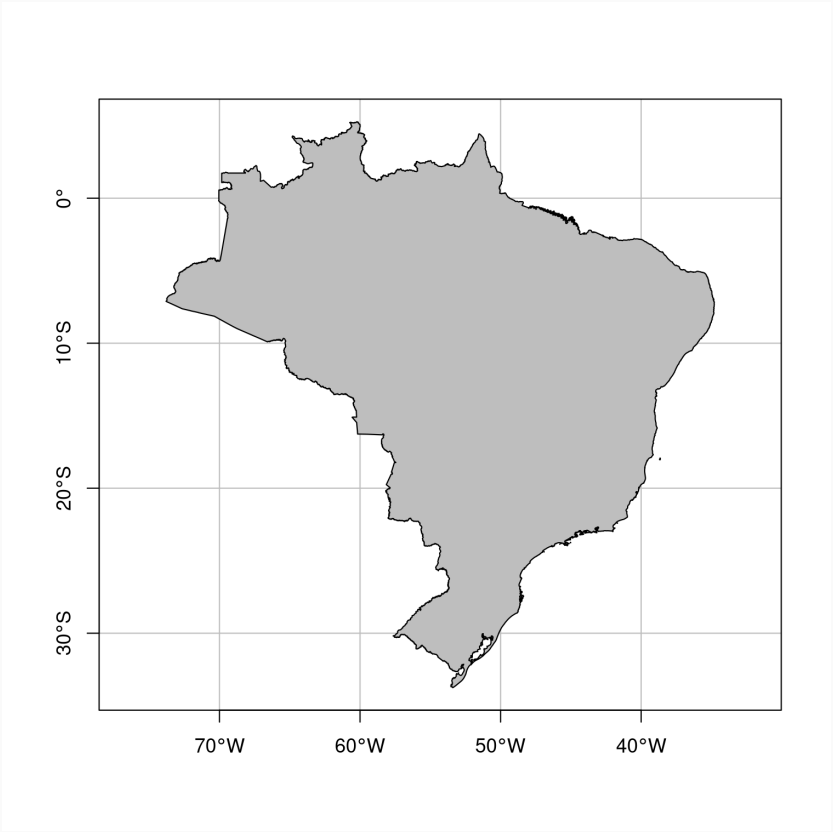
```
# brazil 1872
br_1872 ← geobr::read_country(year = 1872, showProgress = FALSE)
br_1872
```

```
## Simple feature collection with 1 feature and 0 fields
## geometry type:  MULTIPOLYGON
## dimension:      XY
## bbox:           xmin: -73.80132 ymin: -33.74912 xmax: -34.79313 ymax: 5.271891
## geographic CRS: SIRGAS 2000
##                                geom
## 1 MULTIPOLYGON (((-48.40975 - ...
```

# 6.4 Importar dados

## Dados de pacotes: geobr

```
# plot
plot(br_1872$geom, col = "gray", main = NA, axes = TRUE, graticule = TRUE)
```



# 6.4 Importar dados

## Dados de pacotes: geobr

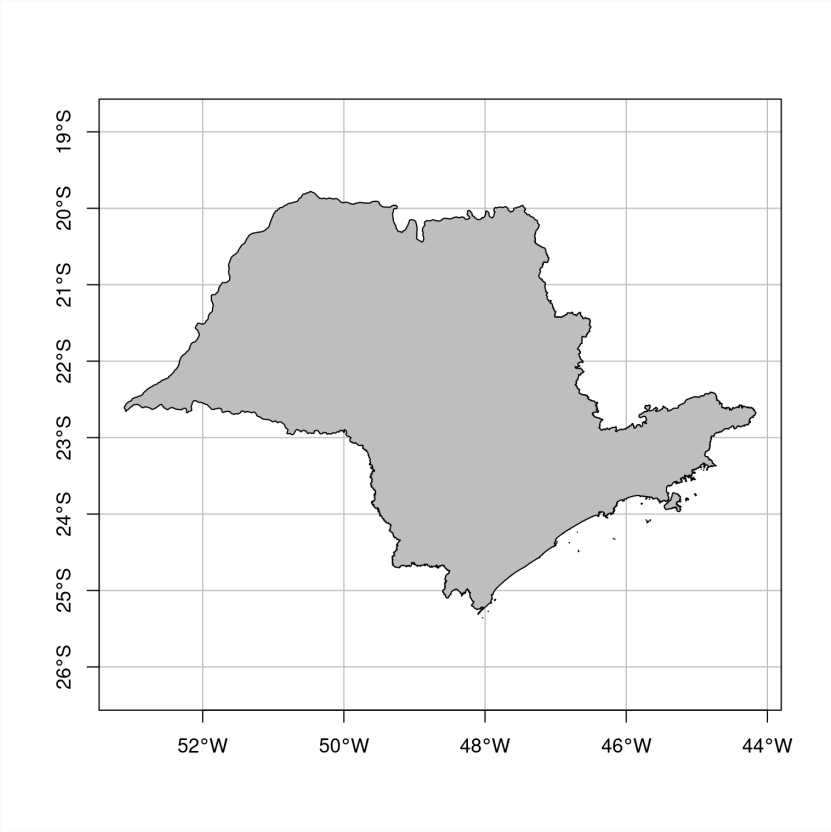
```
# sao paulo state
sp_2019 ← geobr::read_state(code_state = "SP", year = 2019, showProgress = FALSE)
sp_2019
```

```
## Simple feature collection with 1 feature and 5 fields
## geometry type:  MULTIPOLYGON
## dimension:      XY
## bbox:           xmin: -53.10986 ymin: -25.35794 xmax: -44.16137 ymax: -19.77989
## geographic CRS: SIRGAS 2000
##   code_state abbrev_state name_state code_region name_region
## 1         35           SP  São Paulo           3     Sudeste MULTIPOLYGON (((-48.037
```

# 6.4 Importar dados

## Dados de pacotes: geobr

```
# plot
plot(sp_2019$geom, col = "gray", main = NA, axes = TRUE, graticule = TRUE)
```



# 6.4 Importar dados

## Dados de pacotes: geobr

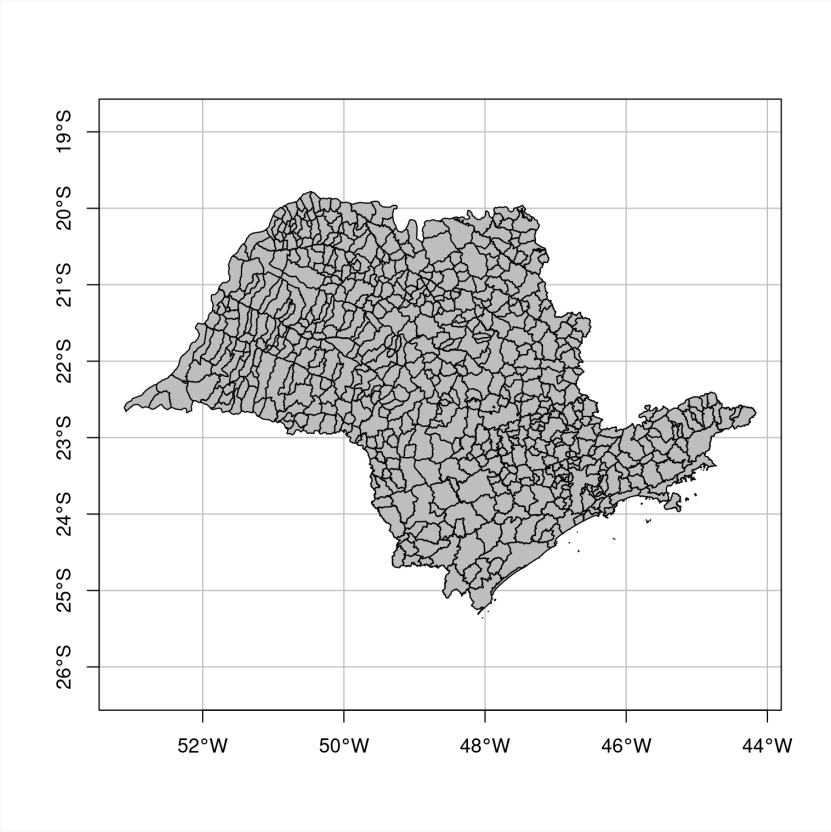
```
# sao paulo municipalities
sp_mun_2019 ← geobr::read_municipality(code_muni = "SP", year = 2019, showProgres
sp_mun_2019
```

```
## Simple feature collection with 645 features and 7 fields
## geometry type:  MULTIPOLYGON
## dimension:      XY
## bbox:           xmin: -53.10986 ymin: -25.35794 xmax: -44.16137 ymax: -19.77966
## geographic CRS: SIRGAS 2000
## First 10 features:
##    code_muni               name_muni code_state abbrev_state name_state code_region n
## 1    3500105             Adamantina         35           SP  São Paulo           3
## 2    3500204                 Adolfo         35           SP  São Paulo           3
## 3    3500303                  Aguaí         35           SP  São Paulo           3
## 4    3500402          Águas Da Prata         35           SP  São Paulo           3
## 5    3500501        Águas De Lindóia         35           SP  São Paulo           3
## 6    3500550 Águas De Santa Bárbara         35           SP  São Paulo           3
## 7    3500600     Águas De São Pedro         35           SP  São Paulo           3
## 8    3500709                 Agudos         35           SP  São Paulo           3
## 9    3500758               Alambari         35           SP  São Paulo           3
## 10   3500808      Alfredo Marcondes         35           SP  São Paulo           3
```

# 6.4 Importar dados

## Dados de pacotes: geobr

```
# plot
plot(sp_mun_2019$geom, col = "gray", main = NA, axes = TRUE, graticule = TRUE)
```



# 6.4 Importar dados

## Dados de pacotes: geobr

```
# rio claro
rc_2019 ← geobr::read_municipality(code_muni = 3543907, year = 2019, showProgress
rc_2019
```

```
## Simple feature collection with 1 feature and 7 fields
## geometry type:  MULTIPOLYGON
## dimension:      XY
## bbox:           xmin: -47.76536 ymin: -22.55203 xmax: -47.46188 ymax: -22.24368
## geographic CRS: SIRGAS 2000
##     code_muni name_muni code_state abbrev_state name_state code_region name_region
## 493   3543907 Rio Claro         35           SP  São Paulo           3     Sudeste M
```

# 6.4 Importar dados

## Dados de pacotes: geobr

```
# plot
plot(rc_2019$geom, col = "gray", main = NA, axes = TRUE, graticule = TRUE)
```

# 6.4 Importar dados

## Dados de pacotes: geobr

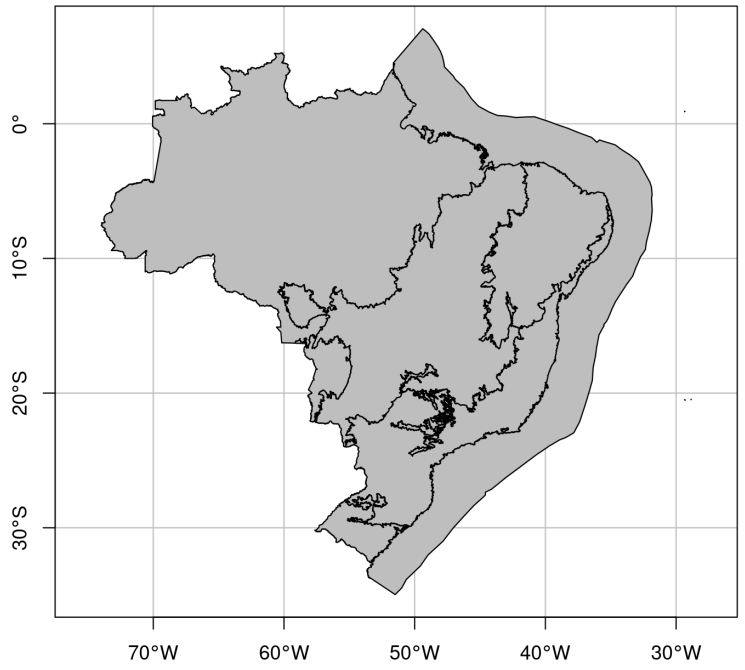
```
# biomes
bi_2019 ← geobr::read_biomes(year = 2019, showProgress = FALSE)
bi_2019
```

```
## Simple feature collection with 7 features and 3 fields
## geometry type:  MULTIPOLYGON
## dimension:      XY
## bbox:           xmin: -73.98304 ymin: -34.95942 xmax: -28.84785 ymax: 7.053767
## geographic CRS: SIRGAS 2000
##         name_biome code_biome year                               geom
## 1         Amazônia          1 2019 MULTIPOLYGON (((-44.08515 - ...
## 2         Caatinga          2 2019 MULTIPOLYGON (((-41.7408 -2 ...
## 3          Cerrado          3 2019 MULTIPOLYGON (((-43.39009 - ...
## 4   Mata Atlântica          4 2019 MULTIPOLYGON (((-48.70814 - ...
## 5            Pampa          5 2019 MULTIPOLYGON (((-52.82472 - ...
## 6         Pantanal          6 2019 MULTIPOLYGON (((-57.75946 - ...
## 7 Sistema Costeiro         NA 2019 MULTIPOLYGON (((-44.64799 - ...
```

# 6.4 Importar dados

## Dados de pacotes: geobr

```
# plot
plot(bi_2019$geom, col = "gray", main = NA, axes = TRUE, graticule = TRUE)
```



# 6.4 Importar dados

## Dados de pacotes: geobr

| Function | Geographies available | Years available | Source |
|---|---|---|---|
| `read_country()` | Country | 1872, 1900, 1911, 1920, 1933, 1940, 1950, 1960, 1970, 1980, 1991, 2000, 2001, 2010, 2013, 2014, 2015, 2016, 2017, 2018, 2019 | IBGE |
| `read_state()` | States | 1872, 1900, 1911, 1920, 1933, 1940, 1950, 1960, 1970, 1980, 1991, 2000, 2001, 2010, 2013, 2014, 2015, 2016, 2017, 2018, 2019 | IBGE |
| `read_municipality()` | Municipality | 1872, 1900, 1911, 1920, 1933, 1940, 1950, 1960, 1970, 1980, 1991, 2000, 2001, 2005, 2007, 2010, 2013, 2014, 2015, 2016, 2017, 2018, 2019 | IBGE |
| `read_biomes()` | Biomes | 2004, 2019 | IBGE |

# 6.4 Importar dados

## Dados de pacotes: geobr

```
# list all datasets available in the geobr package
geobr::list_geobr()
```

```
## # A tibble: 23 x 4
##    `function`           geography                      years
##    <chr>                <chr>                          <chr>
##  1 `read_country`       Country                        1872, 1900, 1911, 1920, 1933,
##  2 `read_region`        Region                         2000, 2001, 2010, 2013, 2014,
##  3 `read_state`         States                         1872, 1900, 1911, 1920, 1933,
##  4 `read_meso_region`   Meso region                    2000, 2001, 2010, 2013, 2014,
##  5 `read_micro_region`  Micro region                   2000, 2001, 2010, 2013, 2014,
##  6 `read_intermediate_… Intermediate region            2017, 2019
##  7 `read_immediate_reg… Immediate region               2017, 2019
##  8 `read_weighting_are… Census weighting area (área … 2010
##  9 `read_census_tract`  Census tract (setor censitár… 2000, 2010, 2017
## 10 `read_municipal_sea… Municipality seats (sedes mu… 1872, 1900, 1911, 1920, 1933,
## # … with 13 more rows
```

# 6.4 Importar dados

## Dados de pacotes: rnaturalearth

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


[*] http://www.naturalearthdata.com/

[*] https://docs.ropensci.org/rnaturalearth/

# 6.4 Importar dados

## Dados de pacotes: rnaturalearth

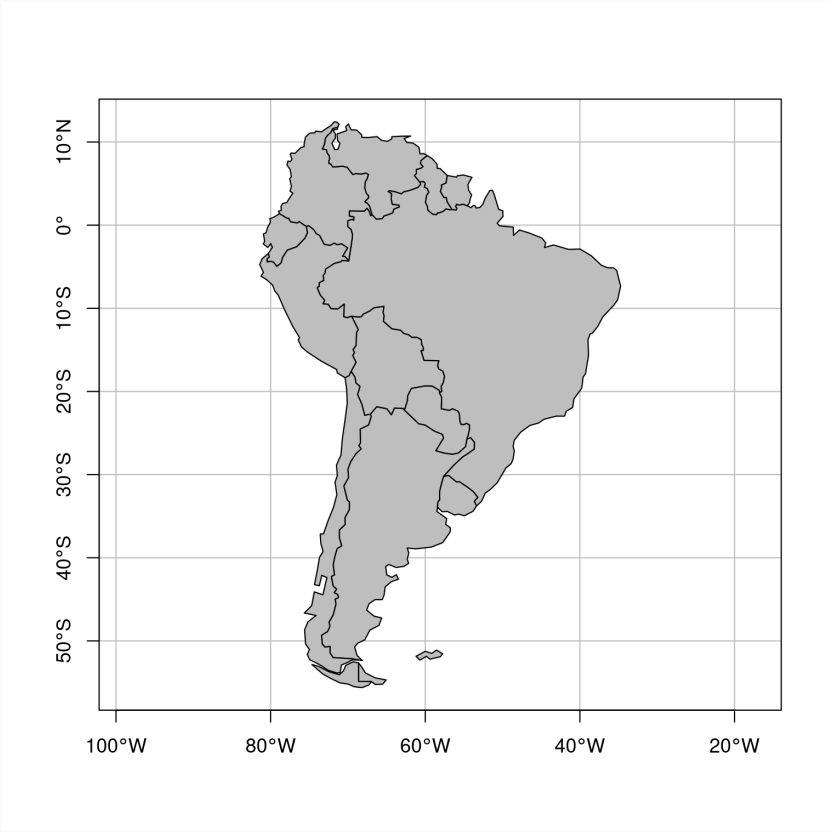
```
# south america
sa ← rnaturalearth::ne_countries(scale = "small", continent = "South America", re
sa
```

```
## Simple feature collection with 13 features and 63 fields
## geometry type:  MULTIPOLYGON
## dimension:      XY
## bbox:           xmin: -81.41094 ymin: -55.61183 xmax: -34.72999 ymax: 12.4373
## CRS:            +proj=longlat +datum=WGS84 +no_defs +ellps=WGS84 +towgs84=0,0,0
## First 10 features:
##     scalerank      featurecla labelrank     sovereignt sov_a3 adm0_dif level
## 4           1 Admin-0 country         2      Argentina    ARG        0     2 Soverei
## 21          1 Admin-0 country         3        Bolivia    BOL        0     2 Soverei
## 22          1 Admin-0 country         2         Brazil    BRA        0     2 Soverei
## 29          1 Admin-0 country         2          Chile    CHL        0     2 Soverei
## 35          1 Admin-0 country         2       Colombia    COL        0     2 Soverei
## 46          1 Admin-0 country         3        Ecuador    ECU        0     2 Soverei
## 54          1 Admin-0 country         5 United Kingdom    GB1        1     2
## 67          1 Admin-0 country         4         Guyana    GUY        0     2 Soverei
## 124         1 Admin-0 country         2           Peru    PER        0     2 Soverei
## 131         1 Admin-0 country         4       Paraguay    PRY        0     2 Soverei
```

# 6.4 Importar dados

## Dados de pacotes: rnaturalearth

```
# plot
plot(sa$geometry, col = "gray", main = NA, axes = TRUE, graticule = TRUE)
```



# 6.4 Importar dados

## Dados de pacotes: rnaturalearth

```
# coast lines
coastline ← rnaturalearth::ne_coastline(scale = "small", returnclass = "sf")
coastline
```

```
## Simple feature collection with 134 features and 2 fields
## geometry type:  LINESTRING
## dimension:      XY
## bbox:           xmin: -180 ymin: -85.60904 xmax: 180 ymax: 83.64513
## CRS:            +proj=longlat +datum=WGS84 +no_defs +ellps=WGS84 +towgs84=0,0,0
## First 10 features:
##   scalerank featurecla                       geometry
## 0         1  Coastline LINESTRING (-163.7129 -78.5 ...
## 1         0  Coastline LINESTRING (-6.197885 53.86 ...
## 2         0  Coastline LINESTRING (141.0002 -2.600 ...
## 3         0  Coastline LINESTRING (114.204 4.52587 ...
## 4         1  Coastline LINESTRING (-93.61276 74.98 ...
## 5         1  Coastline LINESTRING (-93.84 77.52, - ...
## 6         1  Coastline LINESTRING (-96.7544 78.765 ...
## 7         0  Coastline LINESTRING (-88.15035 74.39 ...
## 8         1  Coastline LINESTRING (-111.2644 78.15 ...
## 9         1  Coastline LINESTRING (-110.9637 78.80 ...
```

# 6.4 Importar dados

## Dados de pacotes: rnaturalearth

```
# plot
plot(coastline$geometry, col = "blue", main = NA)
```

# Descrição de objetos sf

# 6.5 Descrição de objetos sf

## Informações geográficas e tabela de atributos

```
# south america
sa
```

```
## Simple feature collection with 13 features and 63 fields
## geometry type:  MULTIPOLYGON
## dimension:      XY
## bbox:           xmin: -81.41094 ymin: -55.61183 xmax: -34.72999 ymax: 12.4373
## CRS:            +proj=longlat +datum=WGS84 +no_defs +ellps=WGS84 +towgs84=0,0,0
## First 10 features:
##     scalerank      featurecla labelrank     sovereignt sov_a3 adm0_dif level
## 4           1 Admin-0 country         2      Argentina    ARG        0     2 Soverei
## 21          1 Admin-0 country         3        Bolivia    BOL        0     2 Soverei
## 22          1 Admin-0 country         2         Brazil    BRA        0     2 Soverei
## 29          1 Admin-0 country         2          Chile    CHL        0     2 Soverei
## 35          1 Admin-0 country         2       Colombia    COL        0     2 Soverei
## 46          1 Admin-0 country         3        Ecuador    ECU        0     2 Soverei
## 54          1 Admin-0 country         5 United Kingdom    GB1        1     2
## 67          1 Admin-0 country         4         Guyana    GUY        0     2 Soverei
## 124         1 Admin-0 country         2           Peru    PER        0     2 Soverei
## 131         1 Admin-0 country         4       Paraguay    PRY        0     2 Soverei
##               geounit gu_a3 su_dif          subunit su_a3 brk_diff        name
```

# 6.5 Descrição de objetos sf

## Informações geográficas

### Geometry type

```
# geometry
sf::st_geometry_type(sa)
```

```
##  [1] MULTIPOLYGON MULTIPOLYGON MULTIPOLYGON MULTIPOLYGON MULTIPOLYGON MULTIPOLYGON M
## [11] MULTIPOLYGON MULTIPOLYGON MULTIPOLYGON
## 18 Levels: GEOMETRY POINT LINESTRING POLYGON MULTIPOINT MULTILINESTRING MULTIPOLYGON
```

### Bounding box

```
# extention
sf::st_bbox(sa)
```

```
##      xmin      ymin      xmax      ymax
## -81.41094 -55.61183 -34.72999  12.43730
```

# 6.5 Descrição de objetos sf

## Informações geográficas

### Coordinate reference system

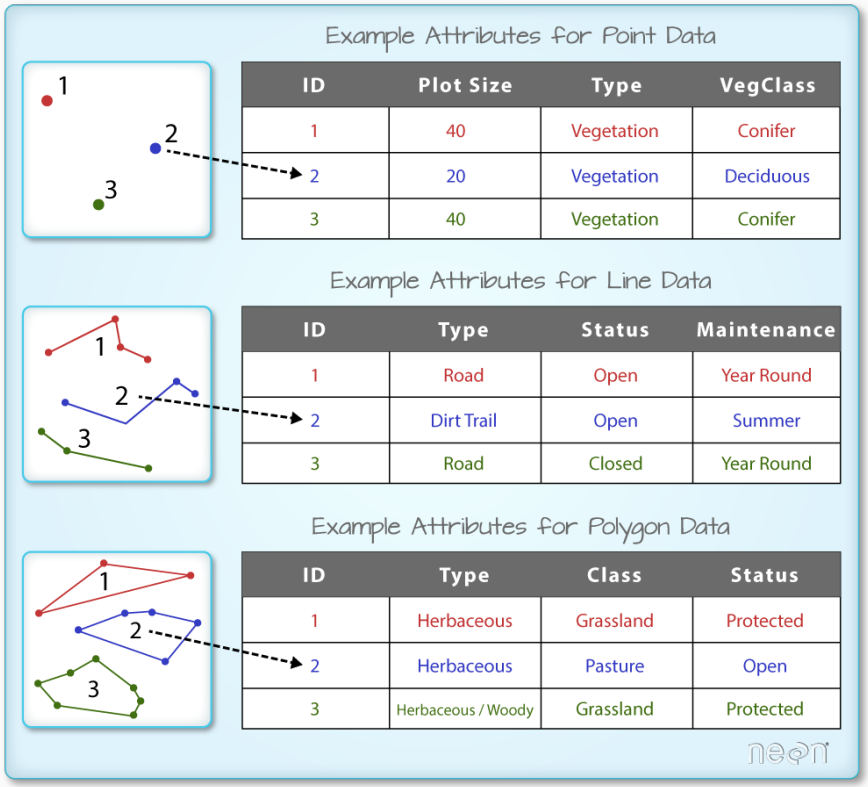
```
# coordinate reference system
sf::st_crs(sa)
```

```
## Coordinate Reference System:
##   User input: +proj=longlat +datum=WGS84 +no_defs +ellps=WGS84 +towgs84=0,0,0
##   wkt:
## BOUNDCRS[
##     SOURCECRS[
##         GEOGCRS["unknown",
##             DATUM["World Geodetic System 1984",
##                 ELLIPSOID["WGS 84",6378137,298.257223563,
##                     LENGTHUNIT["metre",1]],
##                 ID["EPSG",6326]],
##             PRIMEM["Greenwich",0,
##                 ANGLEUNIT["degree",0.0174532925199433],
##                 ID["EPSG",8901]],
##             CS[ellipsoidal,2],
##                 AXIS["longitude",east,
```

# 6.5 Descrição de objetos sf

## Tabela de atributos

### Relação entre a geometria e suas características



Example Attributes for Point Data

| ID | Plot Size | Type | VegClass |
|---|---|---|---|
| 1 | 40 | Vegetation | Conifer |
| 2 | 20 | Vegetation | Deciduous |
| 3 | 40 | Vegetation | Conifer |

Example Attributes for Line Data

| ID | Type | Status | Maintenance |
|---|---|---|---|
| 1 | Road | Open | Year Round |
| 2 | Dirt Trail | Open | Summer |
| 3 | Road | Closed | Year Round |

Example Attributes for Polygon Data

| ID | Type | Class | Status |
|---|---|---|---|
| 1 | Herbaceous | Grassland | Protected |
| 2 | Herbaceous | Pasture | Open |
| 3 | Herbaceous / Woody | Grassland | Protected |

# 6.5 Descrição de objetos sf

## Tabela de atributos

### Acessar a tabela de atributos de um `sf`

```
# acessar a tabela de atributos
sa_tab ← sf::st_drop_geometry(sa)
sa_tab
```

```
##     scalerank      featurecla labelrank     sovereignt sov_a3 adm0_dif level
## 4           1 Admin-0 country         2      Argentina    ARG         0     2 Soverei
## 21          1 Admin-0 country         3        Bolivia    BOL         0     2 Soverei
## 22          1 Admin-0 country         2         Brazil    BRA         0     2 Soverei
## 29          1 Admin-0 country         2          Chile    CHL         0     2 Soverei
## 35          1 Admin-0 country         2       Colombia    COL         0     2 Soverei
## 46          1 Admin-0 country         3        Ecuador    ECU         0     2 Soverei
## 54          1 Admin-0 country         5 United Kingdom    GB1         1     2
## 67          1 Admin-0 country         4         Guyana    GUY         0     2 Soverei
## 124         1 Admin-0 country         2           Peru    PER         0     2 Soverei
## 131         1 Admin-0 country         4       Paraguay    PRY         0     2 Soverei
## 148         1 Admin-0 country         4       Suriname    SUR         0     2 Soverei
## 167         1 Admin-0 country         4        Uruguay    URY         0     2 Soverei
## 170         1 Admin-0 country         3      Venezuela    VEN         0     2 Soverei
```

# Importar dados de planilhas eletrônicas e converter em sf

# 6.6 Converter dados sf

## Dados de uma planilha eletrônica

```
# import table
si ← readr::read_csv(here::here("03_dados", "tabelas", "ATLANTIC_AMPHIBIANS_sites
si
```

```
## # A tibble: 1,163 x 25
##    id    reference_number species_number record sampled_habitat active_methods passi
##    <chr>            <dbl>          <dbl> <chr>  <chr>           <chr>          <chr>
##  1 amp1…             1001             19 ab     fo,ll           as             pt
##  2 amp1…             1002             16 co     fo,la,ll        as             pt
##  3 amp1…             1002             14 co     fo,la,ll        as             pt
##  4 amp1…             1002             13 co     fo,la,ll        as             pt
##  5 amp1…             1003             30 co     fo,ll,br        as             <NA>
##  6 amp1…             1004             42 co     tp,pp,la,ll,is  <NA>           <NA>
##  7 amp1…             1005             23 co     sp              as             <NA>
##  8 amp1…             1005             19 co     sp,la,sw        as,sb,tr       <NA>
##  9 amp1…             1005             13 ab     fo              <NA>           pt
## 10 amp1…             1006              1 ab     fo              <NA>           pt
## # … with 1,153 more rows, and 15 more variables: year_start <dbl>, month_finish <dbl
## #   country <chr>, state <chr>, state_abbreviation <chr>, municipality <chr>, site <
## #   coordinate_precision <chr>, altitude <dbl>, temperature <dbl>, precipitation <db
```

# 6.6 Converter dados sf

## Add as duas colunas de `lon` e `lat`

```
# add column
si ← si %>%
  dplyr::mutate(lon = longitude, lat = latitude, .before = 1)
si
```

```
## # A tibble: 1,163 x 27
##      lon   lat id    reference_number species_number record sampled_habitat active_m
##    <dbl> <dbl> <chr>            <dbl>          <dbl> <chr>  <chr>            <chr>
##  1 -43.4 -8.68 amp1…             1001             19 ab     fo,ll            as
##  2 -38.9 -3.55 amp1…             1002             16 co     fo,la,ll         as
##  3 -38.9 -3.57 amp1…             1002             14 co     fo,la,ll         as
##  4 -38.9 -3.52 amp1…             1002             13 co     fo,la,ll         as
##  5 -38.9 -4.28 amp1…             1003             30 co     fo,ll,br         as
##  6 -36.4 -9.23 amp1…             1004             42 co     tp,pp,la,ll,is   <NA>
##  7 -40.9 -3.85 amp1…             1005             23 co     sp               as
##  8 -40.9 -3.83 amp1…             1005             19 co     sp,la,sw         as,sb,tr
##  9 -40.9 -3.84 amp1…             1005             13 ab     fo               <NA>
## 10 -35.2 -6.14 amp1…             1006              1 ab     fo               <NA>
## # … with 1,153 more rows, and 16 more variables: month_start <dbl>, year_start <dbl>
## #   effort_months <dbl>, country <chr>, state <chr>, state_abbreviation <chr>, munic
## #   longitude <dbl>, coordinate_precision <chr>, altitude <dbl>, temperature <dbl>,
```

# 6.6 Converter dados sf

## Converter dados de planilhas em sf

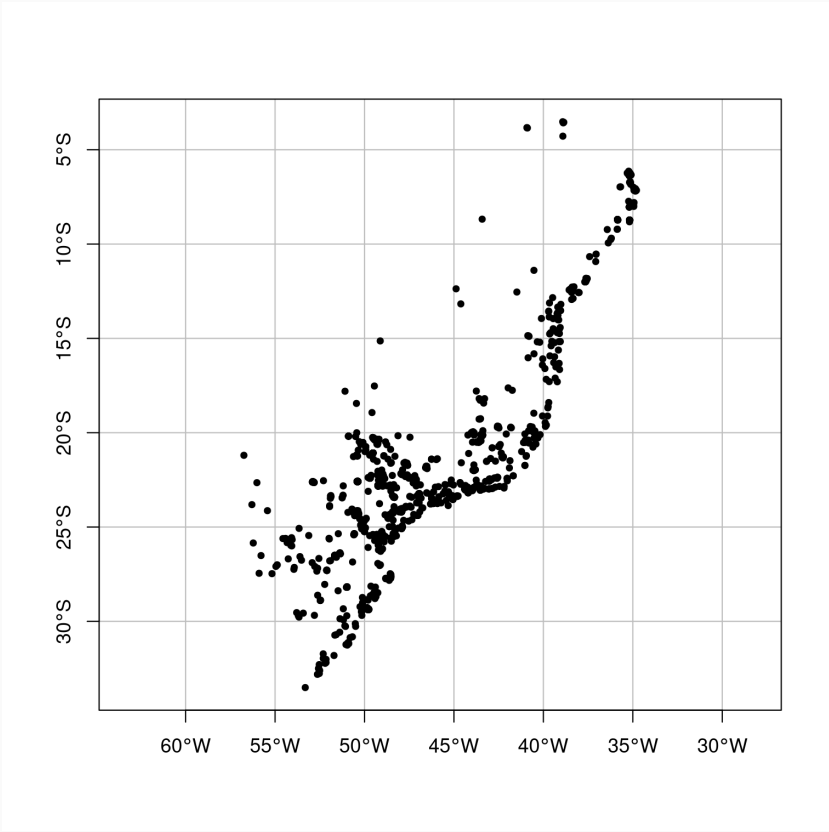
```
# convert to sf
si_ve ← si %>%
  sf::st_as_sf(coords = c("lon", "lat"), crs = 4326)
si_ve
```

```
## Simple feature collection with 1163 features and 25 fields
## geometry type:  POINT
## dimension:      XY
## bbox:           xmin: -56.74194 ymin: -33.51083 xmax: -34.79667 ymax: -3.51525
## geographic CRS: WGS 84
## # A tibble: 1,163 x 26
##    id    reference_number species_number record sampled_habitat active_methods passi
##  * <chr>            <dbl>          <dbl> <chr>  <chr>           <chr>          <chr>
##  1 amp1…             1001             19 ab     fo,ll           as             pt
##  2 amp1…             1002             16 co     fo,la,ll        as             pt
##  3 amp1…             1002             14 co     fo,la,ll        as             pt
##  4 amp1…             1002             13 co     fo,la,ll        as             pt
##  5 amp1…             1003             30 co     fo,ll,br        as             <NA>
##  6 amp1…             1004             42 co     tp,pp,la,ll,is  <NA>           <NA>
##  7 amp1…             1005             23 co     sp              as             <NA>
##  8 amp1…             1005             19 co     sp,la,sw        as,sb,tr       <NA>
```

# 6.6 Converter dados sf

## Converter dados de planilhas em sf

```
# plot
plot(si_ve$geometry, pch = 20, main = NA, axes = TRUE, graticule = TRUE)
```

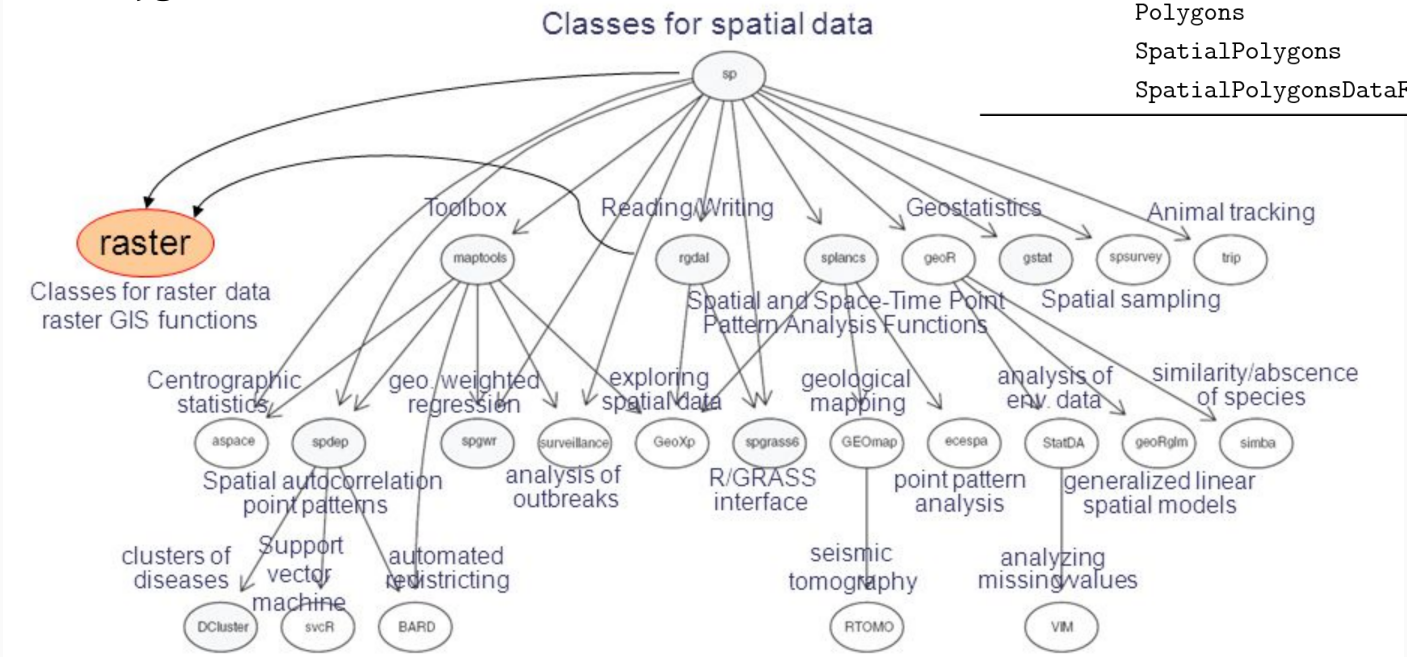


# 6.6 Converter dados sf

## Tipos de objetos `sp`

- SpatialPoints
- SpatialLines
- SpatialPolygons
- SpatialPointsDataFrame
- SpatialLinesDataFrame
- SpatialPolygonsDataFrame

| R package | Class |
|---|---|
| maps [8] | map |
| sp [10] | SpatialPoints |
| | SpatialPointsDataFrame |
| | Line |
| | Lines |
| | SpatialLines |
| | SpatialLinesDataFrame |
| | Polygon |
| | Polygons |
| | SpatialPolygons |
| | SpatialPolygonsDataFrame |



# 6.6 Converter dados sf

## Dados `sp`

```
# countries sp
co110_sp ← rnaturalearth::countries110
co110_sp
```

```
## class       : SpatialPolygonsDataFrame
## features    : 177
## extent      : -180, 180, -90, 83.64513  (xmin, xmax, ymin, ymax)
## crs         : +proj=longlat +datum=WGS84 +no_defs +ellps=WGS84 +towgs84=0,0,0
## variables   : 63
## names       : scalerank,      featurecla, labelrank,  sovereignt, sov_a3, adm0_dif,
## min values  :         1, Admin-0 country,         2, Afghanistan,    AFG,        0,
## max values  :         3, Admin-0 country,         7,    Zimbabwe,    ZWE,        1,
```

# 6.6 Converter dados sf

## Converter `sp` para `sf`

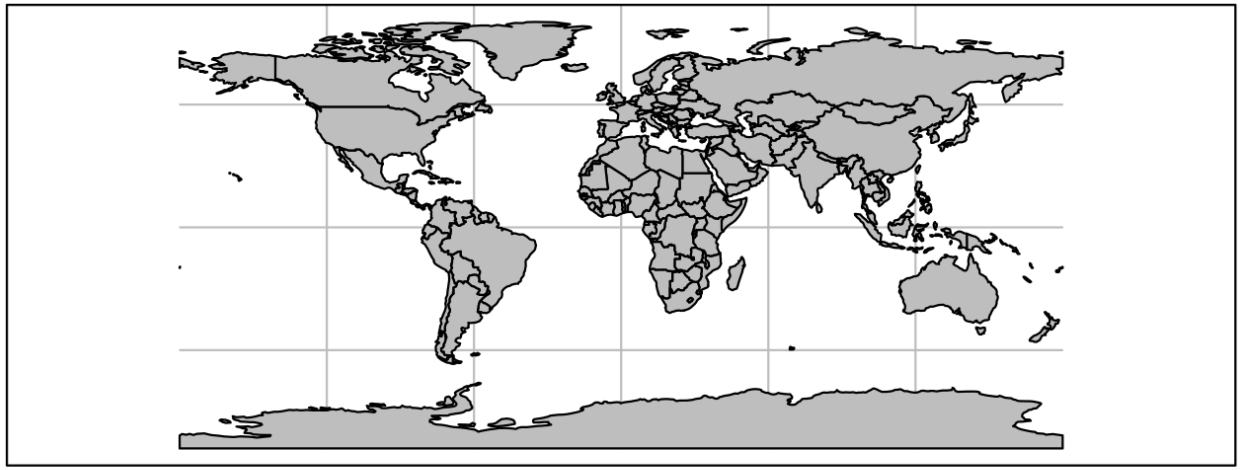
```
# countries sf
co110_sf ← sf::st_as_sf(co110_sp)
co110_sf
```

```
## Simple feature collection with 177 features and 63 fields
## geometry type:  MULTIPOLYGON
## dimension:      XY
## bbox:           xmin: -180 ymin: -90 xmax: 180 ymax: 83.64513
## CRS:            +proj=longlat +datum=WGS84 +no_defs +ellps=WGS84 +towgs84=0,0,0
## First 10 features:
##   scalerank      featurecla labelrank           sovereignt sov_a3 adm0_dif level
## 0         1 Admin-0 country         3          Afghanistan    AFG         0     2 Sov
## 1         1 Admin-0 country         3              Angola    AGO         0     2 Sov
## 2         1 Admin-0 country         6             Albania    ALB         0     2 Sov
## 3         1 Admin-0 country         4 United Arab Emirates    ARE         0     2 Sov
## 4         1 Admin-0 country         2           Argentina    ARG         0     2 Sov
## 5         1 Admin-0 country         6             Armenia    ARM         0     2 Sov
## 6         1 Admin-0 country         4           Antarctica    ATA         0     2
## 7         3 Admin-0 country         6              France    FR1         1     2
## 8         1 Admin-0 country         2           Australia    AU1         1     2
## 9         1 Admin-0 country         4             Austria    AUT         0     2 Sov
```

# 6.6 Converter dados sf

## Converter `sf` para `sp`

```
# plot
plot(co110_sf$geometry, col = "gray", main = NA, graticule = TRUE)
```



# 6.6 Converter dados sf

## Converter `sf` para `sp`

```
# countries sp
co110_sp ← sf::as_Spatial(co110_sf)
co110_sp
```

```
## class       : SpatialPolygonsDataFrame
## features    : 177
## extent      : -180, 180, -90, 83.64513  (xmin, xmax, ymin, ymax)
## crs         : +proj=longlat +ellps=WGS84 +no_defs
## variables   : 63
## names       : scalerank,      featurecla, labelrank,  sovereignt, sov_a3, adm0_dif,
## min values  :         1, Admin-0 country,         2, Afghanistan,    AFG,        0,
## max values  :         3, Admin-0 country,         7,    Zimbabwe,    ZWE,        1,
```

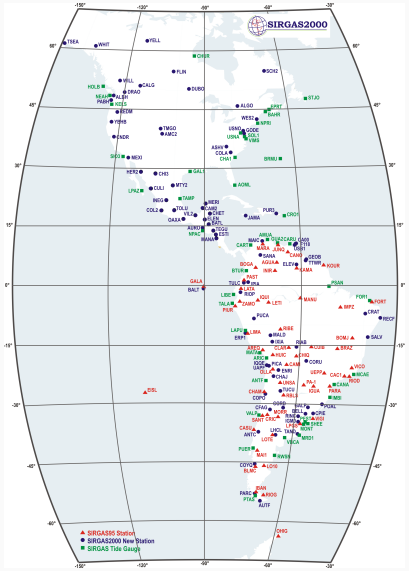
# Conversão do CRS

# 6.7 Conversão do CRS

## Converter CRS local

### SIRGAS2000/GCS -> SIRGAS2000/UTM23S

```
# convert coordinate system
rc_2019_sirgas2000_utm23s ← sf::st_transform(rc_2019, crs = 31983)
```



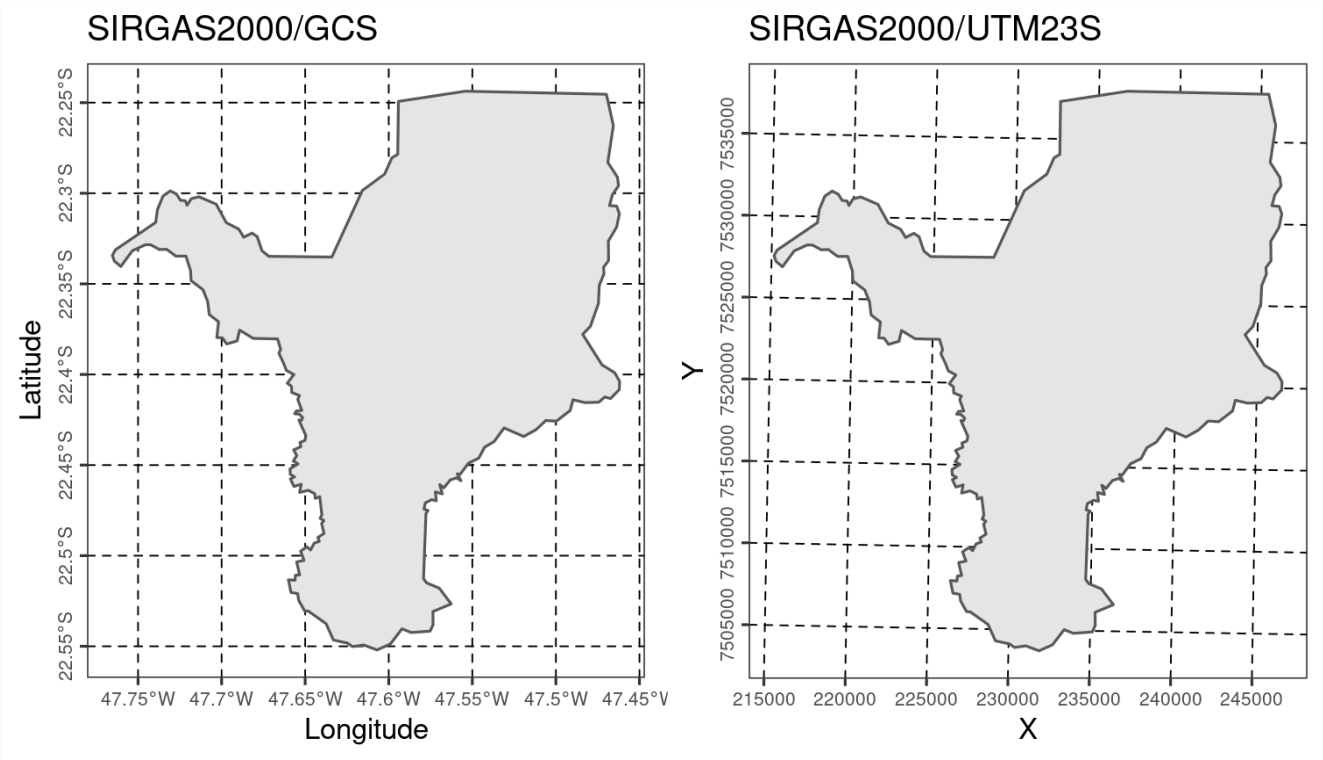

[*] https://epsg.io/31983

[*] http://www.sirgas.org/pt/sirgas-realizations/sirgas2000/

# 6.7 Conversão do CRS

## Converter CRS local

### plota

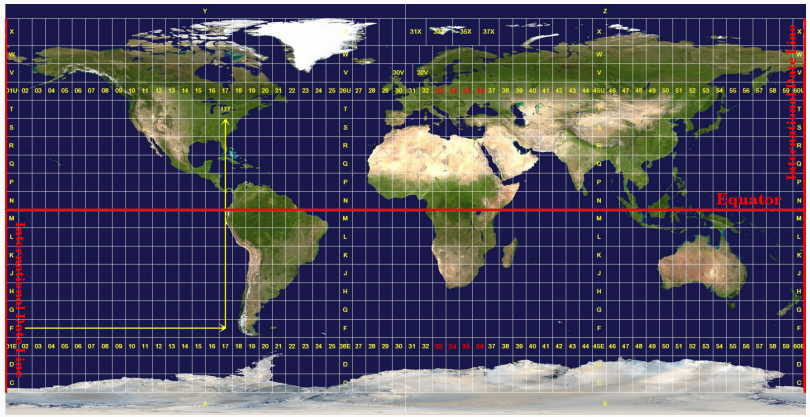


# 6.7 Conversão do CRS

## Converter CRS local

### SIRGAS2000/GCS -> WGS84/UTM23S

```
# convert datum and coordinate system
rc_2019_wgs84_utm23s ← sf::st_transform(rc_2019, crs = 32723)
```




[*] https://epsg.io/32723

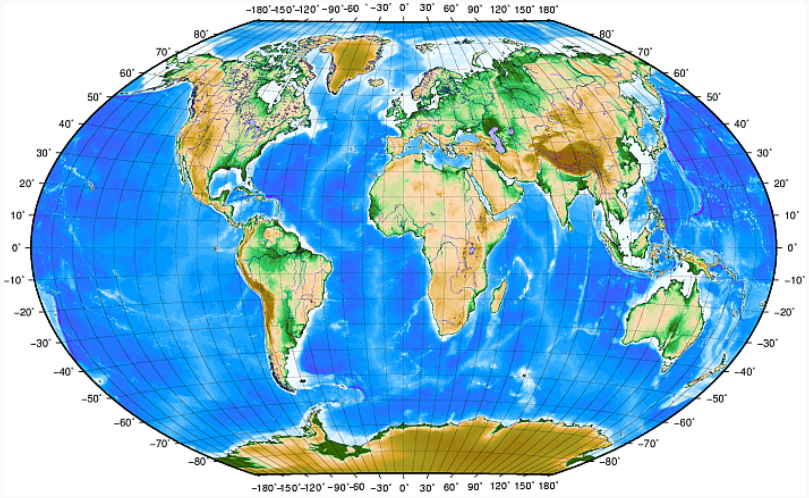
[*] https://pro.arcgis.com/en/pro-app/help/mapping/properties/transverse-mercator.htm

# 6.7 Conversão do CRS

## Converter CRS local

### SIRGAS2000/GCS -> WGS84/GCS

```
# convert datum
rc_2019_wgs84_gcs ← sf::st_transform(rc_2019, crs = 4326)
```



[*] https://epsg.io/4326

# 6.7 Conversão do CRS

## Converter CRS global

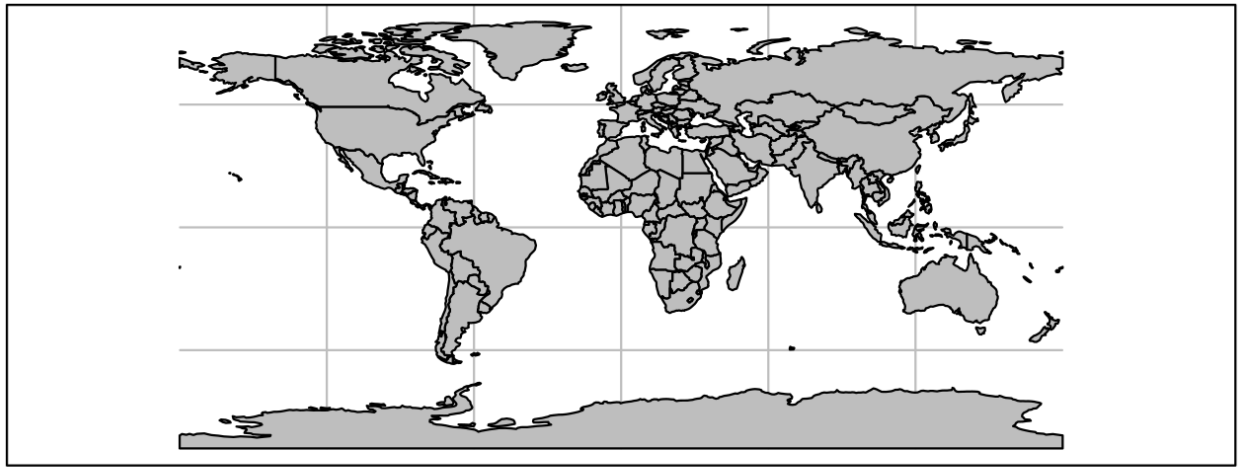
```
# countries - WGS84/GCS
co110_sf
```

```
## Simple feature collection with 177 features and 63 fields
## geometry type:  MULTIPOLYGON
## dimension:      XY
## bbox:           xmin: -180 ymin: -90 xmax: 180 ymax: 83.64513
## CRS:            +proj=longlat +datum=WGS84 +no_defs +ellps=WGS84 +towgs84=0,0,0
## First 10 features:
##   scalerank      featurecla labelrank           sovereignt sov_a3 adm0_dif level
## 0         1 Admin-0 country         3          Afghanistan    AFG         0     2 Sov
## 1         1 Admin-0 country         3               Angola    AGO         0     2 Sov
## 2         1 Admin-0 country         6              Albania    ALB         0     2 Sov
## 3         1 Admin-0 country         4 United Arab Emirates    ARE         0     2 Sov
## 4         1 Admin-0 country         2            Argentina    ARG         0     2 Sov
## 5         1 Admin-0 country         6              Armenia    ARM         0     2 Sov
## 6         1 Admin-0 country         4           Antarctica    ATA         0     2
## 7         3 Admin-0 country         6               France    FR1         1     2
## 8         1 Admin-0 country         2            Australia    AU1         1     2
## 9         1 Admin-0 country         4              Austria    AUT         0     2 Sov
##   adm0_a3 geou_dif                          geounit gu_a3 su_dif
```

# 6.7 Conversão do CRS

## Converter CRS global

```
# plot
plot(co110_sf$geometry, col = "gray", graticule = TRUE)
```

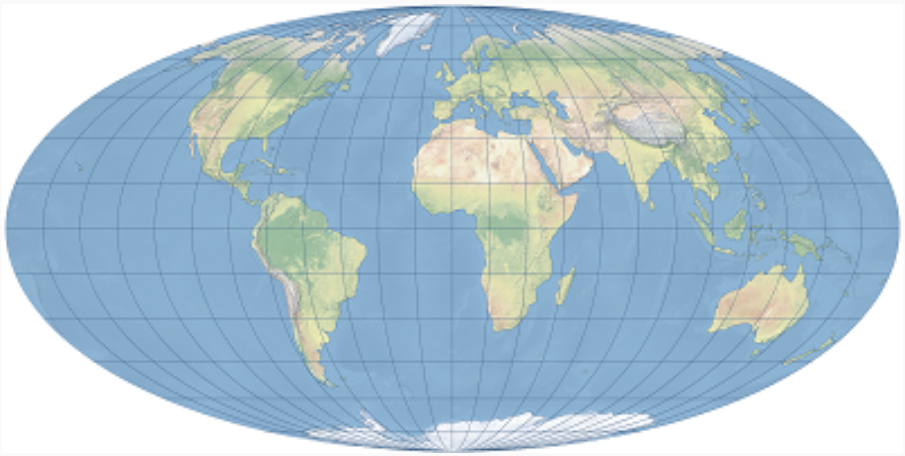


# 6.7 Conversão do CRS

## Converter CRS global

### Projeção de Mollweide: preserva as relações de área

```
# mollweide projection
co110_sf_moll ← sf::st_transform(co110_sf, crs = "+proj=moll")
```



[*] https://epsg.io/54009

[*] https://pro.arcgis.com/en/pro-app/help/mapping/properties/mollweide.htm

# 6.7 Conversão do CRS

## Converter CRS global

### Projeção de Mollweide

```
# plot
plot(co110_sf_moll$geometry, col = "gray", graticule = TRUE)
```

# 6.7 Conversão do CRS

## Converter CRS global

Projeção de Winkel Tripel: mínimo de distorção para área, direção e distância

```
# winkel tripel projection
co110_sf_wintri ← lwgeom::st_transform_proj(co110_sf, crs = "+proj=wintri")
```

[*] https://pro.arcgis.com/en/pro-app/help/mapping/properties/winkel-tripel.htm

# 6.7 Conversão do CRS

## Converter CRS global

### Projeção de Winkel Tripel

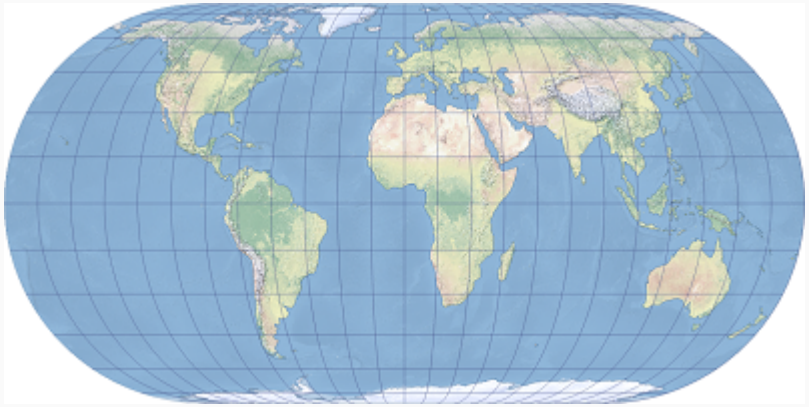
```
# plot
plot(co110_sf_wintri$geometry, col = "gray")
```

# 6.7 Conversão do CRS

## Converter CRS global

### Projeção de Eckert IV: presenva a área e com meridianos elípticos

```
# eckert iv projection
co110_sf_eck4 ← sf::st_transform(co110_sf, crs = "+proj=eck4")
```



[*] https://epsg.io/54012

[*] https://pro.arcgis.com/en/pro-app/help/mapping/properties/eckert-iv.htm

# 6.7 Conversão do CRS

## Converter CRS global

### Projeção de Eckert IV

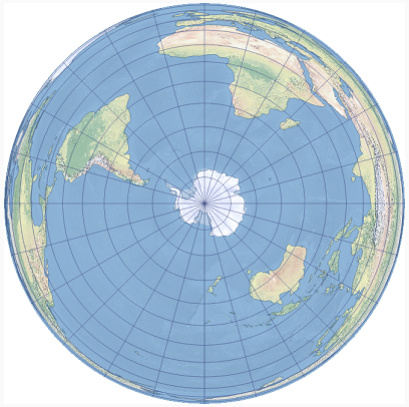
```
# plot
plot(co110_sf_eck4$geometry, col = "gray", graticule = TRUE)
```

# 6.7 Conversão do CRS

## Converter CRS global

### Projeção azimutal de Lambert: preserva os tamanhos relativos e senso de direção a partir do centro

```
# lambert projection
co110_sf_laea1 ← sf::st_transform(co110_sf, crs = "+proj=laea +x_0=0 +y_0=0 +lon_
```



[*] https://pro.arcgis.com/en/pro-app/help/mapping/properties/lambert-azimuthal-equal-area.htm

# 6.7 Conversão do CRS

## Converter CRS global

### Projeção azimutal de Lambert

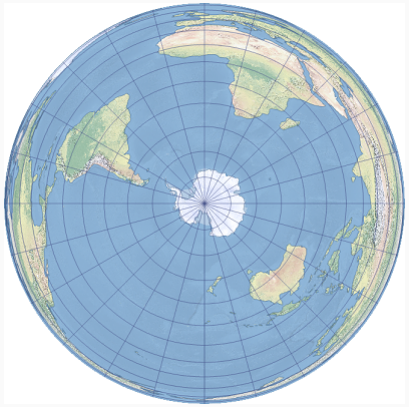
```
# plot
plot(co110_sf_laea1$geometry, col = "gray", graticule = TRUE)
```

# 6.7 Conversão do CRS

## Converter CRS global

### Projeção azimutal de Lambert: preserva os tamanhos relativos e senso de direção a partir do centro

```
# lambert projection
co110_sf_laea2 ← sf::st_transform(co110_sf, crs = "+proj=laea +x_0=0 +y_0=0 +lon_
```



[*] https://pro.arcgis.com/en/pro-app/help/mapping/properties/lambert-azimuthal-equal-area.htm

# 6.7 Conversão do CRS

## Converter CRS global

### Projeção azimutal de Lambert

```
# plot
plot(co110_sf_laea2$geometry, col = "gray", graticule = TRUE)
```

# Manipulação e análise de dados vetoriais

# 6.8 Operações de atributos

Modificação de objetos espaciais baseado em **informações não espaciais** associadas a dados geográficos

## Operações:

1. Subset de atributos
2. Join de atributos
3. Aggregation de atributos
4. Criação de atributos
5. Outros operações

# 6.8 Operações de atributos

## 1. Subset de atributos

```
# 1. attribute subsetting
rc_use_forest ← rc_use %>%
  dplyr::filter(CLASSE_USO == "formação florestal")
rc_use_forest
```

```
## Simple feature collection with 1 feature and 6 fields
## geometry type:  MULTIPOLYGON
## dimension:      XY
## bbox:           xmin: 215442.4 ymin: 7504235 xmax: 246282.3 ymax: 7537969
## projected CRS:  SIRGAS 2000 / UTM zone 23S
##   GEOCODIGO MUNICIPIO UF CD_UF         CLASSE_USO AREA_HA                          geom
## 1   3543907 RIO CLARO SP    35 formação florestal 7017.99 MULTIPOLYGON (((232355 750
```

# 6.8 Operações de atributos

## 1. Subset de atributos

```
# plot
plot(rc_2019_sirgas2000_utm23s$geom, col = "gray", main = NA, axes = TRUE, graticu
plot(rc_use_forest$geometry, col = "forestgreen", add = TRUE)
```

# 6.8 Operações de atributos

## 2. Join de atributos

```
# 2. attribute joining - create data
da_class ← tibble::tibble(CLASSE_USO = rc_use$CLASSE_USO,
                          classe = c("agua", "antropico", "edificado", "floresta"
da_class
```

```
## # A tibble: 5 x 2
##   CLASSE_USO         classe
##   <chr>              <chr>
## 1 água               agua
## 2 área antropizada   antropico
## 3 área edificada     edificado
## 4 formação florestal floresta
## 5 silvicultura       silvicultura
```

# 6.8 Operações de atributos

## 2. Join de atributos

```
# 2. attribute joining - join
rc_use_class ← dplyr::left_join(rc_use, da_class, by = "CLASSE_USO") %>%
  sf::st_drop_geometry()
rc_use_class
```

```
##   GEOCODIGO MUNICIPIO UF CD_UF         CLASSE_USO   AREA_HA       classe
## 1   3543907 RIO CLARO SP    35              água   357.027         agua
## 2   3543907 RIO CLARO SP    35  área antropizada 37297.800    antropico
## 3   3543907 RIO CLARO SP    35    área edificada  5078.330    edificado
## 4   3543907 RIO CLARO SP    35 formação florestal  7017.990     floresta
## 5   3543907 RIO CLARO SP    35       silvicultura   138.173 silvicultura
```

## 3. Aggregation de atributos

```
# 3. attribute aggregation
rc_spr_n ← rc_spr %>%
  dplyr::group_by(MUNICIPIO, HIDRO) %>%
  dplyr::summarise(n = n())
rc_spr_n
```

```
## Simple feature collection with 1 feature and 3 fields
## geometry type:  MULTIPOINT
## dimension:      XY
## bbox:           xmin: 217622.9 ymin: 7504132 xmax: 246367.4 ymax: 7537855
## projected CRS:  SIRGAS 2000 / UTM zone 23S
## # A tibble: 1 x 4
## # Groups:   MUNICIPIO [1]
##   MUNICIPIO HIDRO        n
##   <chr>     <chr>    <int>
## 1 RIO CLARO nascente  1220 ((217622.9 7528315), (217836.5 7528103), (217988.9 752820
```

# 6.8 Operações de atributos

## 3. Aggregation de atributos

```
# plot
plot(rc_2019_sirgas2000_utm23s$geom, col = "gray", main = NA, axes = TRUE, graticu
plot(rc_spr_n$geometry, pch = 20, col = "blue", add = TRUE)
```

# 6.8 Operações de atributos

## 4. Criação de atributos

```
# 4. attribute create - columns
rc_use_use_area ← rc_use %>%
  dplyr::mutate(classe_area = paste0(CLASSE_USO, " (", AREA_HA, " ha)")) %>%
  sf::st_drop_geometry()
rc_use_use_area
```

```
##   GEOCODIGO MUNICIPIO UF CD_UF         CLASSE_USO   AREA_HA                    clas
## 1   3543907 RIO CLARO SP    35              água   357.027                água (357.
## 2   3543907 RIO CLARO SP    35  área antropizada 37297.800  área antropizada (3729
## 3   3543907 RIO CLARO SP    35    área edificada  5078.330    área edificada (5078
## 4   3543907 RIO CLARO SP    35 formação florestal  7017.990 formação florestal (7017
## 5   3543907 RIO CLARO SP    35      silvicultura   138.173      silvicultura (138.
```

# 6.8 Operações de atributos

## 4. Criação de atributos

```
# 4. attribute create - area
rc_use_area ← rc_use %>%
  dplyr::mutate(area_m2 = sf::st_area(rc_use),
                area_ha = sf::st_area(rc_use)/1e4) %>%
  sf::st_drop_geometry()
rc_use_area
```

```
##   GEOCODIGO MUNICIPIO UF CD_UF         CLASSE_USO   AREA_HA         area_m2
## 1   3543907 RIO CLARO SP    35              água   357.027   3570267 [m^2]
## 2   3543907 RIO CLARO SP    35  área antropizada 37297.800 372978415 [m^2]
## 3   3543907 RIO CLARO SP    35    área edificada  5078.330  50783283 [m^2]
## 4   3543907 RIO CLARO SP    35 formação florestal  7017.990  70179895 [m^2]
## 5   3543907 RIO CLARO SP    35      silvicultura   138.173   1381726 [m^2]
##       area_ha
## 1   357.02
## 2 37297.84
## 3  5078.32
## 4  7017.98
## 5   138.17
```

# 6.8 Operações de atributos

## 5. Outros operações: funções do `dplyr` para manipulação de feições

```
dplyr::filter()
dplyr::distinc()
dplyr::slice()
dplyr::n_sample()
dplyr::group_by()
dplyr::summarise()
```

# 6.8 Operações de atributos

## 5. Outros operações: funções do `dplyr` para manipulação da tabela de atributos

```
dplyr::select()
dplyr::pull()
dplyr::rename()
dplyr::mutate()
dplyr::arrange()
dplyr::*_join()
```

# Operações espaciais

# 6.9 Operações espaciais

Modificação de objetos espaciais baseado na sua **localização e formato**

Operações:

1. Subset espacial
2. Join espacial
3. Pontos aleatórios
4. Quadrículas
5. Hexágonos

# 6.9 Operações espaciais

## 1. Subset espacial

```
# spatial subsetting
sf::st_intersects(x = rc_spr, y = rc_use_forest)
```

```
## Sparse geometry binary predicate list of length 1220, where the predicate was `inter
## first 10 elements:
##  1: 1
##  2: 1
##  3: (empty)
##  4: 1
##  5: (empty)
##  6: (empty)
##  7: (empty)
##  8: (empty)
##  9: 1
##  10: (empty)
```

# 6.9 Operações espaciais

## 1. Subset espacial

```
# spatial subsetting
sf::st_intersects(x = rc_spr, y = rc_use_forest, sparse = FALSE)
```

```
##          [,1]
##    [1,]  TRUE
##    [2,]  TRUE
##    [3,] FALSE
##    [4,]  TRUE
##    [5,] FALSE
##    [6,] FALSE
##    [7,] FALSE
##    [8,] FALSE
##    [9,]  TRUE
##   [10,] FALSE
##   [11,] FALSE
##   [12,] FALSE
##   [13,] FALSE
##   [14,] FALSE
##   [15,]  TRUE
##   [16,] FALSE
##   [17,] FALSE
```

# 6.9 Operações espaciais

## 1. Subset espacial - dentro

```
# spatial subsetting - inside
rc_spr_forest ← rc_spr %>%
  dplyr::filter(sf::st_intersects(x = ., y = rc_use_forest, sparse = FALSE))
rc_spr_forest
```

```
## Simple feature collection with 169 features and 5 fields
## geometry type:  POINT
## dimension:      XY
## bbox:           xmin: 217622.9 ymin: 7505568 xmax: 246181.4 ymax: 7537185
## 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 (218288.9 7528237)
## 4    3543907 RIO CLARO SP    35 nascente POINT (218583.5 7528215)
## 5    3543907 RIO CLARO SP    35 nascente POINT (219062.2 7528499)
## 6    3543907 RIO CLARO SP    35 nascente POINT (219246.2 7529011)
## 7    3543907 RIO CLARO SP    35 nascente POINT (219554.5 7528573)
## 8    3543907 RIO CLARO SP    35 nascente POINT (220534.8 7528014)
## 9    3543907 RIO CLARO SP    35 nascente POINT (220754.5 7530181)
```

# 6.9 Operações espaciais

## 1. Subset espacial [] - dentro

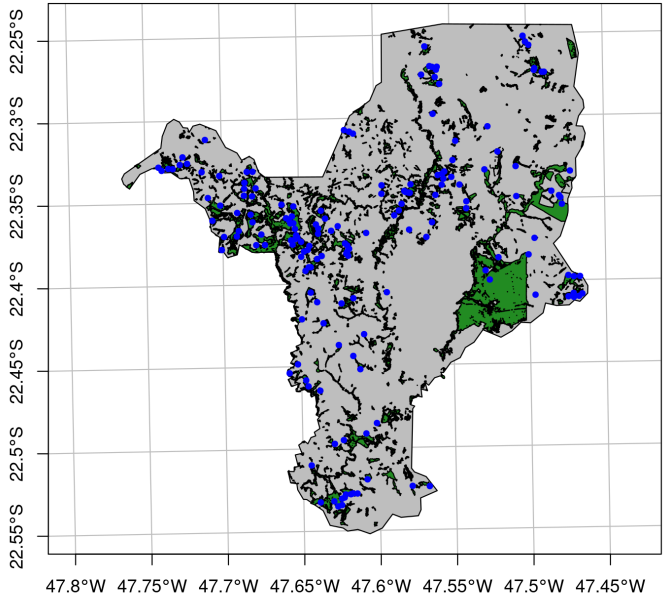
```
# spatial subsetting - inside
rc_spr_forest ← rc_spr[rc_use_forest, ]
rc_spr_forest
```

```
## Simple feature collection with 169 features and 5 fields
## geometry type:  POINT
## dimension:      XY
## bbox:           xmin: 217622.9 ymin: 7505568 xmax: 246181.4 ymax: 7537185
## 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)
## 4    3543907 RIO CLARO SP    35 nascente POINT (218288.9 7528237)
## 9    3543907 RIO CLARO SP    35 nascente POINT (218583.5 7528215)
## 15   3543907 RIO CLARO SP    35 nascente POINT (219062.2 7528499)
## 19   3543907 RIO CLARO SP    35 nascente POINT (219246.2 7529011)
## 23   3543907 RIO CLARO SP    35 nascente POINT (219554.5 7528573)
## 46   3543907 RIO CLARO SP    35 nascente POINT (220534.8 7528014)
## 50   3543907 RIO CLARO SP    35 nascente POINT (220754.5 7530181)
## 54   3543907 RIO CLARO SP    35 nascente POINT (220958.3 7526278)
```

# 6.9 Operações espaciais

## 1. Subset espacial [] - dentro

```
# plot
plot(rc_2019_sirgas2000_utm23s$geom, col = "gray", main = NA, axes = TRUE, graticu
plot(rc_use_forest$geometry, col = "forestgreen", add = TRUE)
plot(rc_spr_forest$geometry, col = "blue", pch = 20, cex = 1, add = TRUE)
```



# 6.9 Operações espaciais

## 1. Subset espacial - fora

```
# spatial subsetting - outside
rc_spr_forest_out ← rc_spr %>%
  dplyr::filter(!sf::st_intersects(x = ., y = rc_use_forest, sparse = FALSE))
rc_spr_forest_out
```

```
## Simple feature collection with 1051 features and 5 fields
## geometry type:  POINT
## dimension:      XY
## bbox:           xmin: 217988.9 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 (217988.9 7528203)
## 2    3543907 RIO CLARO SP    35 nascente POINT (218346.6 7530583)
## 3    3543907 RIO CLARO SP    35 nascente POINT (218393.1 7528031)
## 4    3543907 RIO CLARO SP    35 nascente POINT (218508.3 7528470)
## 5    3543907 RIO CLARO SP    35 nascente POINT (218535.4 7530642)
## 6    3543907 RIO CLARO SP    35 nascente POINT (218760.1 7530258)
## 7    3543907 RIO CLARO SP    35 nascente POINT (218821.1 7529750)
## 8    3543907 RIO CLARO SP    35 nascente POINT (218868.5 7529257)
## 9    3543907 RIO CLARO SP    35 nascente POINT (218919.2 7528697)
```

# 6.9 Operações espaciais

## 1. Subset espacial - fora

```
# plot
plot(rc_2019_sirgas2000_utm23s$geom, col = "gray", main = NA, axes = TRUE, graticu
plot(rc_use_forest$geometry, col = "forestgreen", add = TRUE)
plot(rc_spr_forest_out$geometry, col = "steelblue", pch = 20, cex = 1, add = TRUE)
plot(rc_spr_forest$geometry, col = "blue", pch = 20, cex = 1, add = TRUE)
```

# 6.9 Operações espaciais

## 2. Join espacial

```
# 2. spatial join
rc_spr_use ← rc_spr %>%
  sf::st_join(x = ., y = rc_use)
rc_spr_use
```

```
## Simple feature collection with 1220 features and 11 fields
## geometry type:  POINT
## dimension:      XY
## bbox:           xmin: 217622.9 ymin: 7504132 xmax: 246367.4 ymax: 7537855
## projected CRS:  SIRGAS 2000 / UTM zone 23S
## First 10 features:
##    GEOCODIGO.x MUNICIPIO.x UF.x CD_UF.x    HIDRO GEOCODIGO.y MUNICIPIO.y UF.y CD_UF.
## 1      3543907   RIO CLARO   SP      35 nascente     3543907   RIO CLARO   SP       3
## 2      3543907   RIO CLARO   SP      35 nascente     3543907   RIO CLARO   SP       3
## 3      3543907   RIO CLARO   SP      35 nascente     3543907   RIO CLARO   SP       3
## 4      3543907   RIO CLARO   SP      35 nascente     3543907   RIO CLARO   SP       3
## 5      3543907   RIO CLARO   SP      35 nascente     3543907   RIO CLARO   SP       3
## 6      3543907   RIO CLARO   SP      35 nascente     3543907   RIO CLARO   SP       3
## 7      3543907   RIO CLARO   SP      35 nascente     3543907   RIO CLARO   SP       3
## 8      3543907   RIO CLARO   SP      35 nascente     3543907   RIO CLARO   SP       3
## 9      3543907   RIO CLARO   SP      35 nascente     3543907   RIO CLARO   SP       3
```

# 6.9 Operações espaciais

## 2. Join espacial

```
# 2. spatial join
rc_spr_use %>%
  sf::st_drop_geometry() %>%
  dplyr::count(CLASSE_USO)
```

```
##            CLASSE_USO    n
## 1               água    7
## 2   área antropizada 1027
## 3     área edificada   15
## 4 formação florestal  169
## 5       silvicultura    2
```

## 2. Join de atributos

```
# 2. attribute join
col ← tibble::tibble(CLASSE_USO = c("água", "área antropizada", "área edificada",
                     color = c("blue", "orange", "gray30", "forestgreen", "green"
col
```

```
## # A tibble: 5 x 2
##   CLASSE_USO         color
##   <chr>              <chr>
## 1 água               blue
## 2 área antropizada   orange
## 3 área edificada     gray30
## 4 formação florestal forestgreen
## 5 silvicultura       green
```

# 6.9 Operações espaciais

## 2. Join de atributos

```
# 2. attribute join
rc_spr_use ← dplyr::left_join(rc_spr_use, col, by = "CLASSE_USO")
rc_spr_use
```

```
## Simple feature collection with 1220 features and 12 fields
## geometry type:  POINT
## dimension:      XY
## bbox:           xmin: 217622.9 ymin: 7504132 xmax: 246367.4 ymax: 7537855
## projected CRS:  SIRGAS 2000 / UTM zone 23S
## First 10 features:
##    GEOCODIGO.x MUNICIPIO.x UF.x CD_UF.x    HIDRO GEOCODIGO.y MUNICIPIO.y UF.y CD_UF.
## 1      3543907   RIO CLARO   SP      35 nascente     3543907   RIO CLARO   SP       3
## 2      3543907   RIO CLARO   SP      35 nascente     3543907   RIO CLARO   SP       3
## 3      3543907   RIO CLARO   SP      35 nascente     3543907   RIO CLARO   SP       3
## 4      3543907   RIO CLARO   SP      35 nascente     3543907   RIO CLARO   SP       3
## 5      3543907   RIO CLARO   SP      35 nascente     3543907   RIO CLARO   SP       3
## 6      3543907   RIO CLARO   SP      35 nascente     3543907   RIO CLARO   SP       3
## 7      3543907   RIO CLARO   SP      35 nascente     3543907   RIO CLARO   SP       3
## 8      3543907   RIO CLARO   SP      35 nascente     3543907   RIO CLARO   SP       3
## 9      3543907   RIO CLARO   SP      35 nascente     3543907   RIO CLARO   SP       3
## 10     3543907   RIO CLARO   SP      35 nascente     3543907   RIO CLARO   SP       3
```

# 6.9 Operações espaciais

## 2. Join espacial

```
# plot
plot(rc_2019_sirgas2000_utm23s$geom, col = "gray", main = NA, axes = TRUE, graticu
plot(rc_spr_use[10], col = rc_spr_use$color, pch = 20, add = TRUE)
```

## 3. Pontos aleatórios

```
# 3. random points
rc_2019_sirgas2000_utm23s_rp ← sf::st_sample(rc_2019_sirgas2000_utm23s, 1e3)
rc_2019_sirgas2000_utm23s_rp
```

```
## Geometry set for 1000 features
## geometry type:  POINT
## dimension:      XY
## bbox:           xmin: 215513.9 ymin: 7503984 xmax: 246236.7 ymax: 7537780
## projected CRS:  SIRGAS 2000 / UTM zone 23S
## First 5 geometries:
```

# 6.9 Operações espaciais

## 3. Pontos aleatórios

```
# plot
plot(rc_2019_sirgas2000_utm23s$geom, col = "gray", main = NA, axes = TRUE, graticu
plot(rc_2019_sirgas2000_utm23s_rp, pch = 20, cex = .5, col = "red", add = TRUE)
```

## 3. Pontos aleatórios

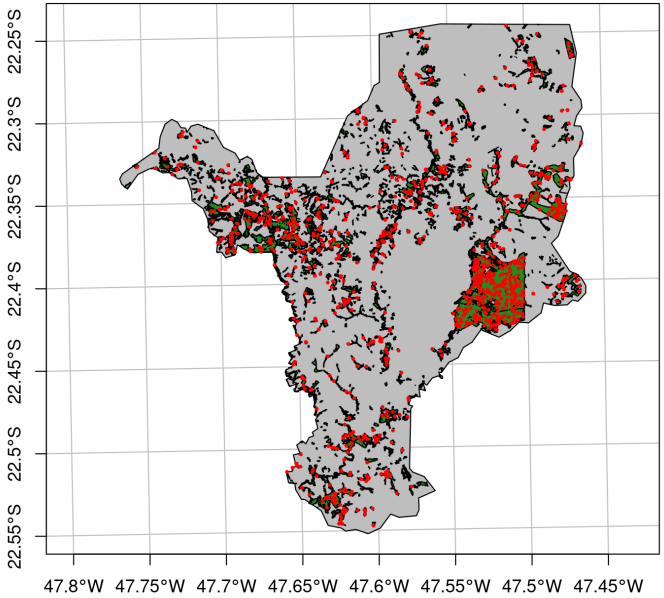
```
# 3. random points
rc_use_forest_rp ← sf::st_sample(rc_use_forest, 1e3)
rc_use_forest_rp
```

```
## Geometry set for 1000 features
## geometry type:  POINT
## dimension:      XY
## bbox:           xmin: 216320 ymin: 7504284 xmax: 246227.3 ymax: 7537870
## projected CRS:  SIRGAS 2000 / UTM zone 23S
## First 5 geometries:
```

# 6.9 Operações espaciais

## 3. Pontos aleatórios

```
# plot
plot(rc_2019_sirgas2000_utm23s$geom, col = "gray", main = NA, axes = TRUE, graticu
plot(rc_use_forest$geom, col = "forestgreen", add = TRUE)
plot(rc_use_forest_rp, pch = 20, cex = .5, col = "red", add = TRUE)
```



# 6.9 Operações espaciais

## 4. Quadrículas

```
# 4. grid
rc_2019_sirgas2000_utm23s_grid ← rc_2019_sirgas2000_utm23s %>%
  sf::st_make_grid(cellsize = 2000) %>%
  sf::st_as_sf()
rc_2019_sirgas2000_utm23s_grid
```

```
## Simple feature collection with 288 features and 0 fields
## geometry type:  POLYGON
## dimension:      XY
## bbox:           xmin: 215151.7 ymin: 7503729 xmax: 247151.7 ymax: 7539729
## projected CRS:  SIRGAS 2000 / UTM zone 23S
## First 10 features:
##                                x
## 1  POLYGON ((215151.7 7503729, ...
## 2  POLYGON ((217151.7 7503729, ...
## 3  POLYGON ((219151.7 7503729, ...
## 4  POLYGON ((221151.7 7503729, ...
## 5  POLYGON ((223151.7 7503729, ...
## 6  POLYGON ((225151.7 7503729, ...
## 7  POLYGON ((227151.7 7503729, ...
## 8  POLYGON ((229151.7 7503729, ...
```

# 6.9 Operações espaciais

## 4. Quadrículas

```
# plot
plot(rc_2019_sirgas2000_utm23s$geom, col = "gray", main = NA, axes = TRUE, graticu
plot(rc_2019_sirgas2000_utm23s_grid, col = adjustcolor("red", .1), add = TRUE)
```

# 6.9 Operações espaciais

## 4. Quadrículas

```
# 4. grid - subset
rc_2019_sirgas2000_utm23s_grid_in ← rc_2019_sirgas2000_utm23s_grid[rc_2019_sirgas
rc_2019_sirgas2000_utm23s_grid_in
```

```
## Simple feature collection with 167 features and 0 fields
## geometry type:  POLYGON
## dimension:      XY
## bbox:           xmin: 215151.7 ymin: 7503729 xmax: 247151.7 ymax: 7539729
## projected CRS:  SIRGAS 2000 / UTM zone 23S
## First 10 features:
##                                x
## 1  POLYGON ((227151.7 7503729, ...
## 2  POLYGON ((229151.7 7503729, ...
## 3  POLYGON ((231151.7 7503729, ...
## 4  POLYGON ((233151.7 7503729, ...
## 5  POLYGON ((235151.7 7503729, ...
## 6  POLYGON ((225151.7 7505729, ...
## 7  POLYGON ((227151.7 7505729, ...
## 8  POLYGON ((229151.7 7505729, ...
## 9  POLYGON ((231151.7 7505729, ...
## 10 POLYGON ((233151.7 7505729, ...
```

# 6.9 Operações espaciais

## 4. Quadrículas

```
# plot
plot(rc_2019_sirgas2000_utm23s$geom, col = "gray", main = NA, axes = TRUE, graticu
plot(rc_2019_sirgas2000_utm23s_grid_in, col = adjustcolor("red", .1), add = TRUE)
```

# 6.9 Operações espaciais

## 4. Quadrículas

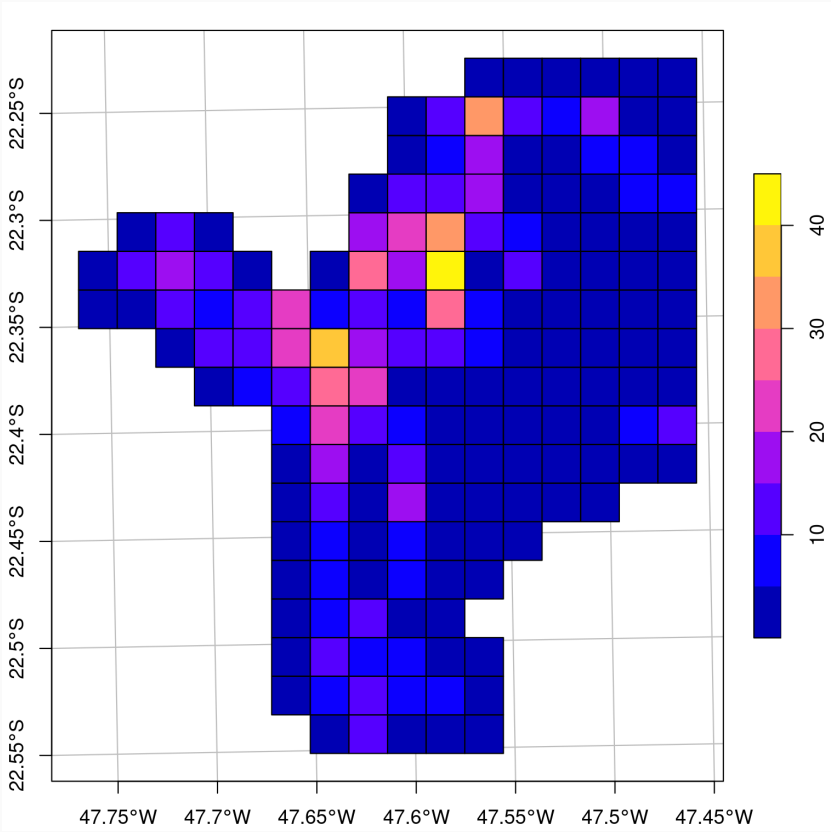
```
# 4. grid - count spring by cell
rc_2019_sirgas2000_utm23s_grid_in_spr_count ← rc_2019_sirgas2000_utm23s_grid_in %
  dplyr::mutate(n = sf::st_intersects(x = ., rc_spr) %>% lengths())
rc_2019_sirgas2000_utm23s_grid_in_spr_count
```

```
## Simple feature collection with 167 features and 1 field
## geometry type:  POLYGON
## dimension:      XY
## bbox:           xmin: 215151.7 ymin: 7503729 xmax: 247151.7 ymax: 7539729
## projected CRS:  SIRGAS 2000 / UTM zone 23S
## First 10 features:
##                                x  n
## 1  POLYGON ((227151.7 7503729, ...  1
## 2  POLYGON ((229151.7 7503729, ... 13
## 3  POLYGON ((231151.7 7503729, ...  5
## 4  POLYGON ((233151.7 7503729, ...  2
## 5  POLYGON ((235151.7 7503729, ...  0
## 6  POLYGON ((225151.7 7505729, ...  0
## 7  POLYGON ((227151.7 7505729, ...  6
## 8  POLYGON ((229151.7 7505729, ... 15
## 9  POLYGON ((231151.7 7505729, ...  9
```

# 6.9 Operações espaciais

## 4. Quadrículas

```
# plot
plot(rc_2019_sirgas2000_utm23s_grid_in_spr_count["n"], main = NA, axes = TRUE, gra
```



# 6.9 Operações espaciais

## 5. Hexágonos

```
# 5. hexagon
rc_2019_sirgas2000_utm23s_hex ← rc_2019_sirgas2000_utm23s %>%
  sf::st_make_grid(cellsize = 2000, square = FALSE) %>%
  sf::st_as_sf()
rc_2019_sirgas2000_utm23s_hex
```

```
## Simple feature collection with 187 features and 0 fields
## geometry type:  POLYGON
## dimension:      XY
## bbox:           xmin: 214151.7 ymin: 7502574 xmax: 248151.7 ymax: 7539525
## projected CRS:  SIRGAS 2000 / UTM zone 23S
## First 10 features:
##                                x
## 1  POLYGON ((215151.7 7526823, ...
## 2  POLYGON ((216151.7 7525091, ...
## 3  POLYGON ((216151.7 7528555, ...
## 4  POLYGON ((217151.7 7526823, ...
## 5  POLYGON ((217151.7 7530287, ...
## 6  POLYGON ((218151.7 7528555, ...
## 7  POLYGON ((219151.7 7526823, ...
## 8  POLYGON ((219151.7 7530287, ...
```

# 6.9 Operações espaciais

## 5. Hexágonos

```
# plot
plot(rc_2019_sirgas2000_utm23s$geom, col = "gray", main = NA, axes = TRUE, graticu
plot(rc_2019_sirgas2000_utm23s_hex, col = adjustcolor("blue", .1), add = TRUE)
```

# 6.9 Operações espaciais

## 5. Hexágonos

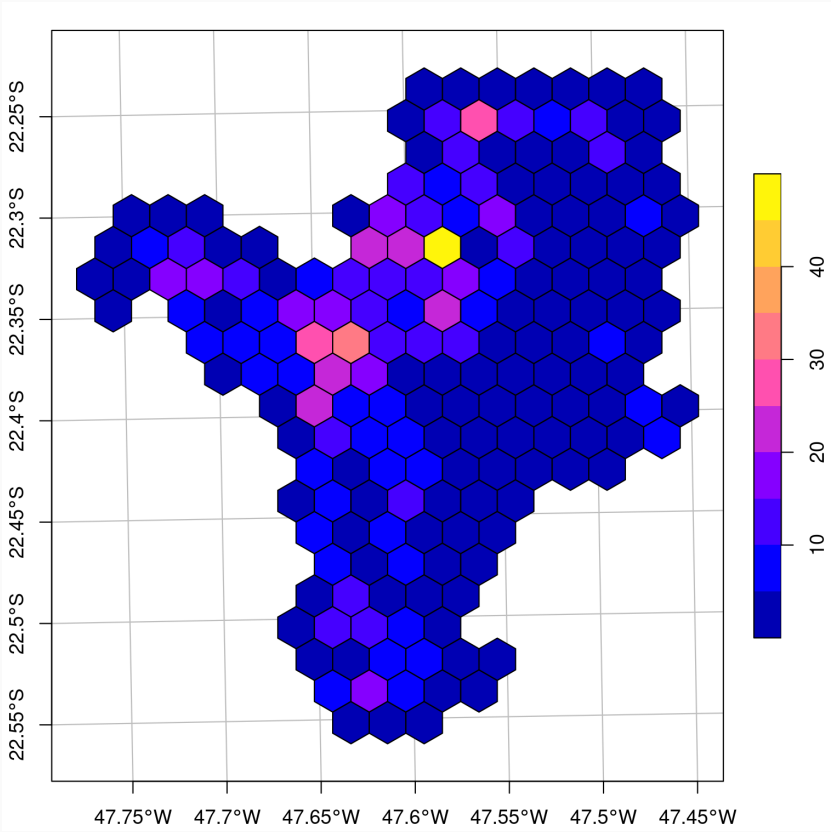
```
# 5. hexagon - count spring by hexagon
rc_2019_sirgas2000_utm23s_hex_spr_count ← rc_2019_sirgas2000_utm23s_hex %>%
  dplyr::mutate(n = sf::st_intersects(x = ., rc_spr) %>% lengths())
rc_2019_sirgas2000_utm23s_hex_spr_count
```

```
## Simple feature collection with 187 features and 1 field
## geometry type:  POLYGON
## dimension:      XY
## bbox:           xmin: 214151.7 ymin: 7502574 xmax: 248151.7 ymax: 7539525
## projected CRS:  SIRGAS 2000 / UTM zone 23S
## First 10 features:
##                                x  n
## 1  POLYGON ((215151.7 7526823, ...  0
## 2  POLYGON ((216151.7 7525091, ...  0
## 3  POLYGON ((216151.7 7528555, ...  0
## 4  POLYGON ((217151.7 7526823, ...  3
## 5  POLYGON ((217151.7 7530287, ...  0
## 6  POLYGON ((218151.7 7528555, ...  6
## 7  POLYGON ((219151.7 7526823, ... 17
## 8  POLYGON ((219151.7 7530287, ...  0
## 9  POLYGON ((220151.7 7525091, ...  6
```

# 6.9 Operações espaciais

## 5. Hexágonos

```
# plot
plot(rc_2019_sirgas2000_utm23s_hex_spr_count["n"], main = NA, axes = TRUE, graticu
```



# Operações geométricas

# 6.10 Operações geométricas

Modificação de objetos espaciais baseado em operações que mudam a **geometria do vetor**

Operações:

1. Simplificação
2. Centroides
3. Buffers
4. União (Dissolve)
5. Corte (Clip)

# 6.10 Operações geométricas

## 1. Simplificação

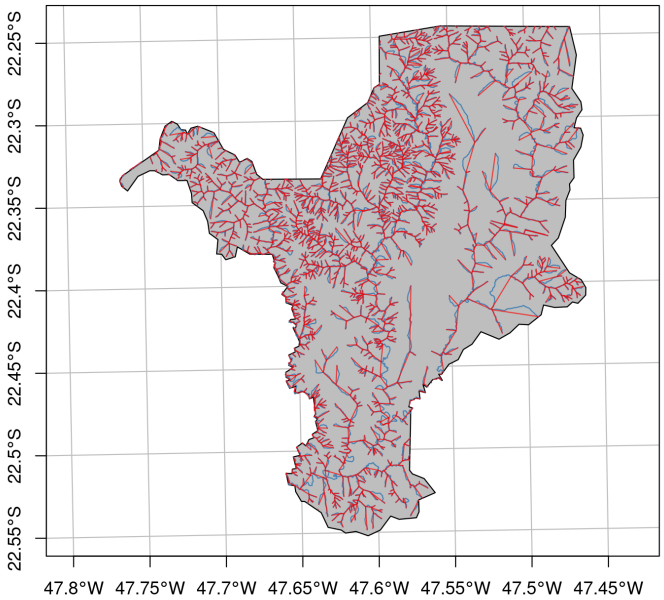
```
# 1. simplification
rc_riv_simp ← sf::st_simplify(rc_riv, dTolerance = 1e4)
rc_riv_simp
```

```
## Simple feature collection with 1 feature and 6 fields
## geometry type:  MULTILINESTRING
## dimension:      XY
## bbox:           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
## 1   3543907 RIO CLARO SP    35 curso d'água (0 - 10m) 1142.98 MULTILINESTRING ((2318
```

# 6.10 Operações geométricas

## 1. Simplificação

```
# plot
plot(rc_2019_sirgas2000_utm23s$geom, col = "gray", main = NA, axes = TRUE, graticu
plot(rc_riv$geometry, col = "steelblue", add = TRUE)
plot(rc_riv_simp$geometry, col = adjustcolor("red", .7), add = TRUE)
```



# 6.10 Operações geométricas

## 2. Centroides

```
# 2. centroids
rc_2019_cent ← sf::st_centroid(rc_2019_sirgas2000_utm23s)
rc_2019_cent
```

```
## Simple feature collection with 1 feature and 7 fields
## geometry type:  POINT
## dimension:      XY
## bbox:           xmin: 234336.7 ymin: 7523513 xmax: 234336.7 ymax: 7523513
## projected CRS:  SIRGAS 2000 / UTM zone 23S
##     code_muni name_muni code_state abbrev_state name_state code_region name_region
## 493   3543907 Rio Claro         35           SP  São Paulo           3     Sudeste P
```

# 6.10 Operações geométricas

## 2. Centroides

```
# plot
plot(rc_2019_sirgas2000_utm23s$geom, col = "gray", main = NA, axes = TRUE, graticu
plot(rc_2019_cent$geom, cex = 3, col = "red", pch = 20, add = TRUE)
```

# 6.10 Operações geométricas

## 2. Centroides

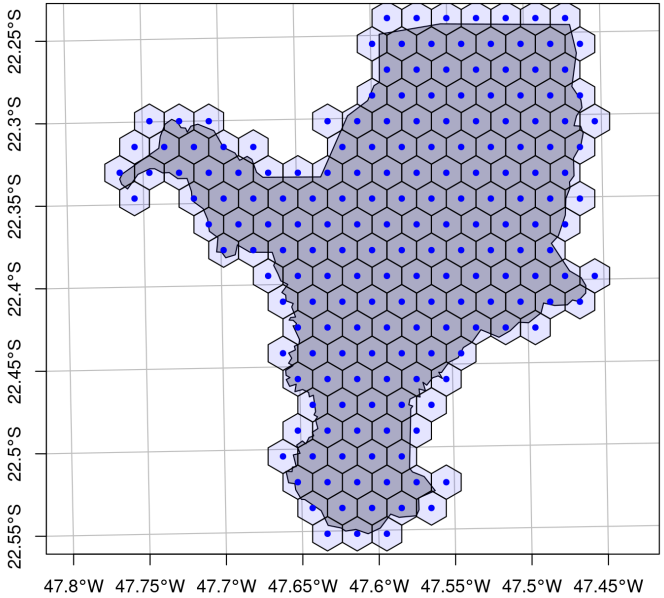
```
# 2. centroids
rc_2019_sirgas2000_utm23s_hex_cent ← sf::st_centroid(rc_2019_sirgas2000_utm23s_he
rc_2019_sirgas2000_utm23s_hex_cent
```

```
## Simple feature collection with 187 features and 0 fields
## geometry type:  POINT
## dimension:      XY
## bbox:           xmin: 215151.7 ymin: 7503729 xmax: 247151.7 ymax: 7538370
## projected CRS:  SIRGAS 2000 / UTM zone 23S
## First 10 features:
##                            x
## 1  POINT (215151.7 7527978)
## 2  POINT (216151.7 7526245)
## 3  POINT (216151.7 7529710)
## 4  POINT (217151.7 7527978)
## 5  POINT (217151.7 7531442)
## 6  POINT (218151.7 7529710)
## 7  POINT (219151.7 7527978)
## 8  POINT (219151.7 7531442)
## 9  POINT (220151.7 7526245)
## 10 POINT (220151.7 7529710)
```

# 6.10 Operações geométricas

## 2. Centroides

```
# plot
plot(rc_2019_sirgas2000_utm23s$geom, col = "gray", main = NA, axes = TRUE, graticu
plot(rc_2019_sirgas2000_utm23s_hex, col = adjustcolor("blue", .1), add = TRUE)
plot(rc_2019_sirgas2000_utm23s_hex_cent, col = "blue", pch = 20, add = TRUE)
```



# 6.10 Operações geométricas

## 3. Buffer

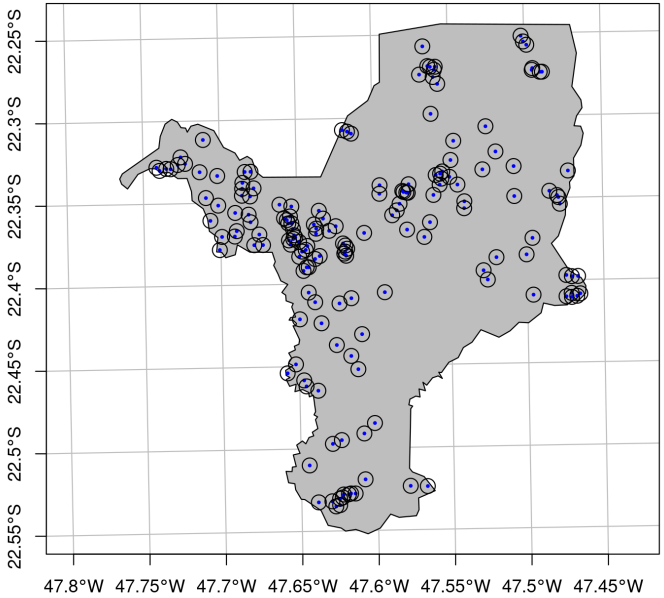
```
# 3. buffer
rc_spr_forest_buf ← rc_spr_forest %>%
  sf::st_buffer(dist = 500)
rc_spr_forest_buf
```

```
## Simple feature collection with 169 features and 5 fields
## geometry type:  POLYGON
## dimension:      XY
## bbox:           xmin: 217122.9 ymin: 7505068 xmax: 246681.4 ymax: 7537685
## projected CRS:  SIRGAS 2000 / UTM zone 23S
## First 10 features:
##    GEOCODIGO MUNICIPIO UF CD_UF    HIDRO                       geometry
## 1    3543907 RIO CLARO SP    35 nascente POLYGON ((218122.9 7528315, ...
## 2    3543907 RIO CLARO SP    35 nascente POLYGON ((218336.5 7528103, ...
## 4    3543907 RIO CLARO SP    35 nascente POLYGON ((218788.9 7528237, ...
## 9    3543907 RIO CLARO SP    35 nascente POLYGON ((219083.5 7528215, ...
## 15   3543907 RIO CLARO SP    35 nascente POLYGON ((219562.2 7528499, ...
## 19   3543907 RIO CLARO SP    35 nascente POLYGON ((219746.2 7529011, ...
## 23   3543907 RIO CLARO SP    35 nascente POLYGON ((220054.5 7528573, ...
## 46   3543907 RIO CLARO SP    35 nascente POLYGON ((221034.8 7528014, ...
## 50   3543907 RIO CLARO SP    35 nascente POLYGON ((221254.5 7530181, ...
```

# 6.10 Operações geométricas

## 3. Buffer

```
# plot
plot(rc_2019_sirgas2000_utm23s$geom, col = "gray", main = NA, axes = TRUE, graticu
plot(rc_spr_forest$geom, pch = 20, cex = .5, col = "blue", add = TRUE)
plot(rc_spr_forest_buf$geom, col = adjustcolor("white", 0), add = TRUE)
```



## 4. União (Dissolve)

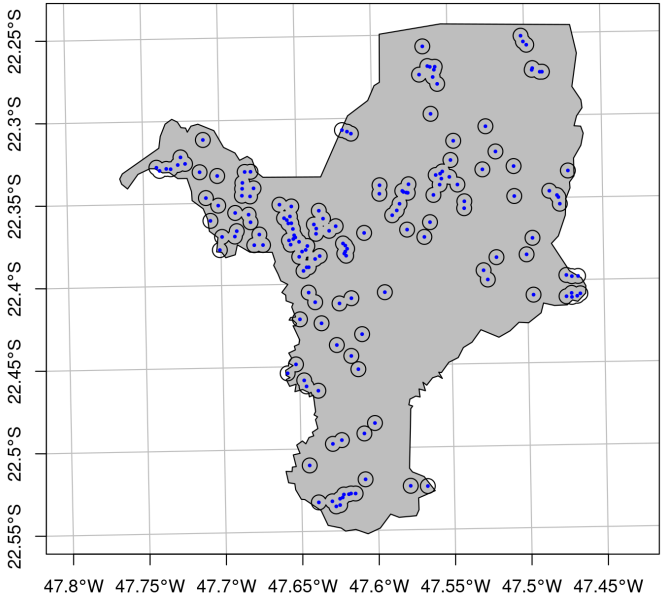
```
# 4. union
rc_spr_forest_buf_union ← sf::st_union(rc_spr_forest_buf)
rc_spr_forest_buf_union
```

```
## Geometry set for 1 feature
## geometry type:  MULTIPOLYGON
## dimension:      XY
## bbox:           xmin: 217122.9 ymin: 7505068 xmax: 246681.4 ymax: 7537685
## projected CRS:  SIRGAS 2000 / UTM zone 23S
```

# 6.10 Operações geométricas

## 4. União (Dissolve)

```
# plot
plot(rc_2019_sirgas2000_utm23s$geom, col = "gray", main = NA, axes = TRUE, graticu
plot(rc_spr_forest$geom, pch = 20, cex = .5, col = "blue", add = TRUE)
plot(rc_spr_forest_buf_union, col = adjustcolor("white", 0), add = TRUE)
```



## 5. Corte (Clip) - intersecção

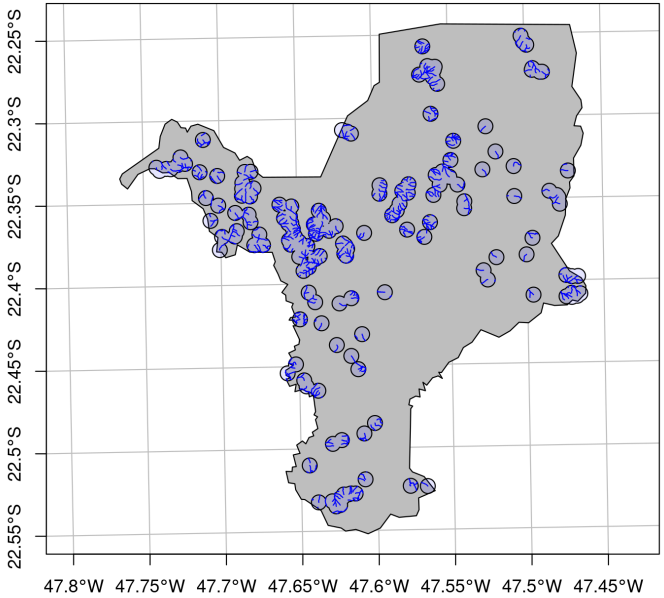
```
# 5. clipping - intersection
rc_riv_spr_forest_buf_int ← sf::st_intersection(x = rc_riv, y = rc_spr_forest_buf
rc_riv_spr_forest_buf_int
```

```
## Simple feature collection with 1 feature and 6 fields
## geometry type:  MULTILINESTRING
## dimension:      XY
## bbox:           xmin: 217364 ymin: 7505072 xmax: 246259.4 ymax: 7537676
## projected CRS:  SIRGAS 2000 / UTM zone 23S
##   GEOCODIGO MUNICIPIO UF CD_UF                   HIDRO COMP_KM
## 1   3543907 RIO CLARO SP    35 curso d'água (0 - 10m) 1142.98 MULTILINESTRING ((2173
```

# 6.10 Operações geométricas

## 5. Corte (Clip) - intersecção

```
# plot
plot(rc_2019_sirgas2000_utm23s$geom, col = "gray", main = NA, axes = TRUE, graticu
plot(rc_spr_forest_buf_union, col = adjustcolor("blue", .1), add = TRUE)
plot(rc_riv_spr_forest_buf_int$geometry, col = "blue", add = TRUE)
```



## 5. Corte (Clip) - diferença

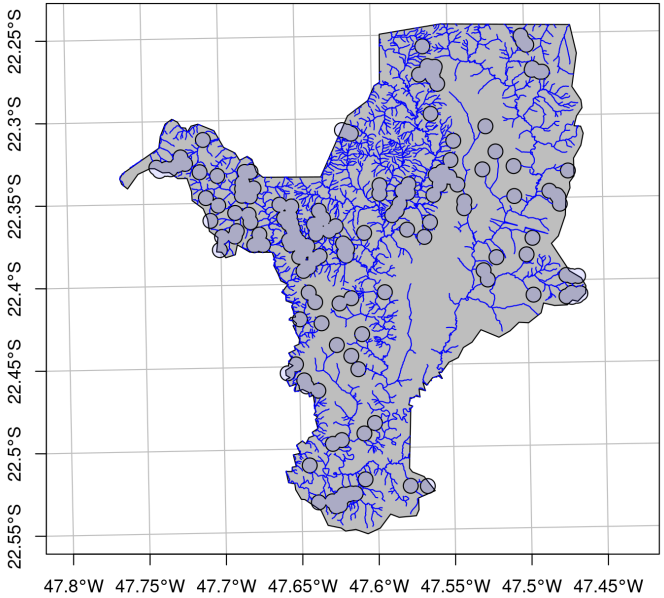
```
# 5. clipping - difference
rc_riv_spr_forest_buf_dif ← sf::st_difference(x = rc_riv, y = rc_spr_forest_buf_u
rc_riv_spr_forest_buf_dif
```

```
## Simple feature collection with 1 feature and 6 fields
## geometry type:  MULTILINESTRING
## dimension:      XY
## bbox:           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
## 1   3543907 RIO CLARO SP    35 curso d'água (0 - 10m) 1142.98 MULTILINESTRING ((2151
```

# 6.10 Operações geométricas

## 5. Corte (Clip)

```
# plot
plot(rc_2019_sirgas2000_utm23s$geom, col = "gray", main = NA, axes = TRUE, graticu
plot(rc_spr_forest_buf_union, col = adjustcolor("blue", .1), add = TRUE)
plot(rc_riv_spr_forest_buf_dif$geometry, col = "blue", add = TRUE)
```

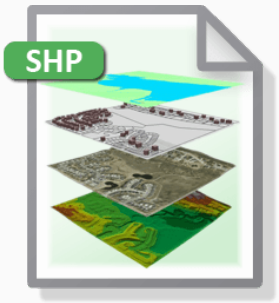


# Exportar dados

# 6.11 Exportar dados

## Exportar como shapefile

```
# export shapefile
sf::st_write(obj = rc_spr_forest_buf,
             dsn = here::here("03_dados", "vetor", "rc_spr_forest_buf.shp"),
             layer = "rc_spr_forest_buf.shp",
             driver = "ESRI Shapefile")
```




# 6.11 Exportar dados

## Exportar como geopackage

```
# export geopackage
sf::st_write(obj = rc_spr_forest_buf,
             dsn = here::here("03_dados", "vetor", "rc_spr_forest_buf.gpkg"),
             layer = "rc_spr_forest_buf")
```



[*] http://switchfromshapefile.org/

[*] https://jsta.rbind.io/blog/2016-07-14-geopackage-r/

# 6.11 Exportar dados

## Exportar como geopackage

```
# export geopackage
sf::st_write(obj = rc_spr_forest_buf_union,
             dsn = here::here("03_dados", "vetor", "rc_spr_forest_buf.gpkg"),
             layer = "rc_spr_forest_buf_union")
```



[*] http://switchfromshapefile.org/

[*] https://jsta.rbind.io/blog/2016-07-14-geopackage-r/

# Dúvidas?

# Maurício Vancine

Contatos:

mauricio.vancine@gmail.com

@mauriciovancine

mauriciovancine

mauriciovancine.github.io

Slides criados via pacote xaringan e tema Metropolis