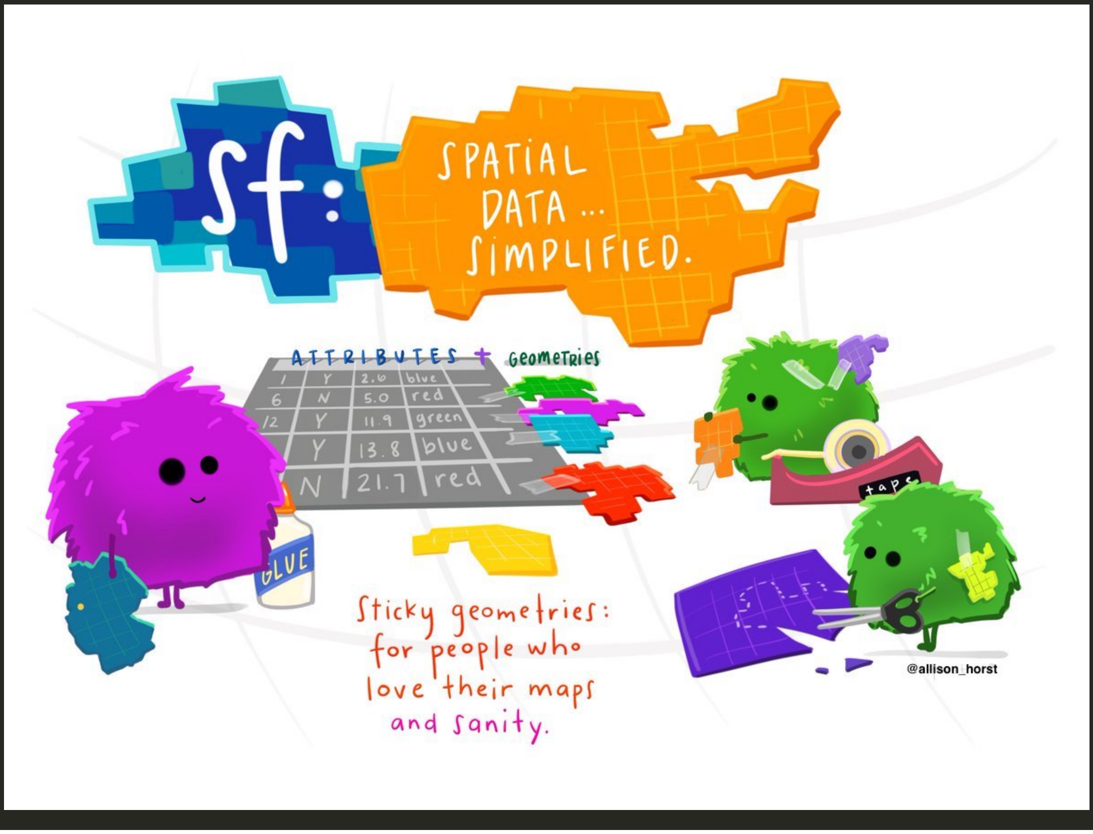
Geographic Information Systems and mapping in R

Introduction to the sf package, for DSCI351/353 class

Olivier Gimenez
2020-10-30

Simple Features for R: the sf package



Olivier Gimenez, and the description file of the sf package

Olivier Gimenez is a scientist

- at the French National Centre for Scientific Research (CNRS)

Support for simple features,

- a standardized way to encode spatial vector data.
- Binds to 'GDAL' for reading and writing data,
- to 'GEOS' for geometrical operations, and
- to 'PROJ' for projection conversions and datum transformations.
- Optionally uses the 's2' package for spherical geometry operations on geographic coordinates.

Author: Edzer Pebesma, Roger Bivand, Etienne Racine, Michael Sumner, Ian Cook, Tim Keitt, Robin Lovelace, Hadley Wickham, Jeroen Ooms, Kirill Müller, Thomas Lin Pedersen, Dan Baston

Introduction

What's so nice about `sf`?

- Easy to work with spatial data because the distinction between spatial data and other forms of data is minimized
- Spatial objects are stored as **dataframes**, with the feature geometries stored in list-columns
- This is similar to the way that **spatial databases** are structured
- All functions begin with `st_` for easy autofill with RStudio tab
- Functions are **pipe-friendly**
- `dplyr` and `tidyr` verbs have been defined for the `sf` objects
- `ggplot2` is able to plot `sf` objects directly

Load packages

```r
library(sf) # GIS package
```

```
## Linking to GEOS 3.8.0, GDAL 3.0.4, PROJ 6.3.1
```

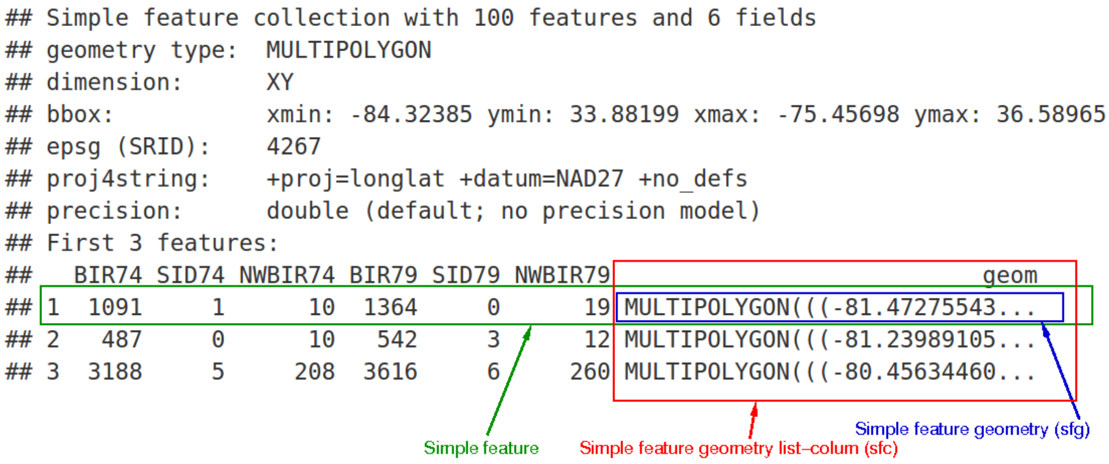
```r
library(tidyverse) # tidyverse packages, dplyr and ggplot2 among others
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --

## v ggplot2 3.3.2     v purrr   0.3.4
## v tibble  3.0.4     v dplyr   1.0.2
## v tidyr   1.1.2     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.0

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```r
theme_set(theme_minimal()) # set ggplot theme
```

Vector layers in `sf`

- The `sf` class is a hierarchical structure composed of 3 classes
  - In green, **sf** - Vector layer object, `data.frame` with ≥ 1 attribute columns and 1 geometry column
  - In red, **sfc** - Geometric part of vector layer - geometry column
  - In blue, **sfg** - Geometry of individual simple feature



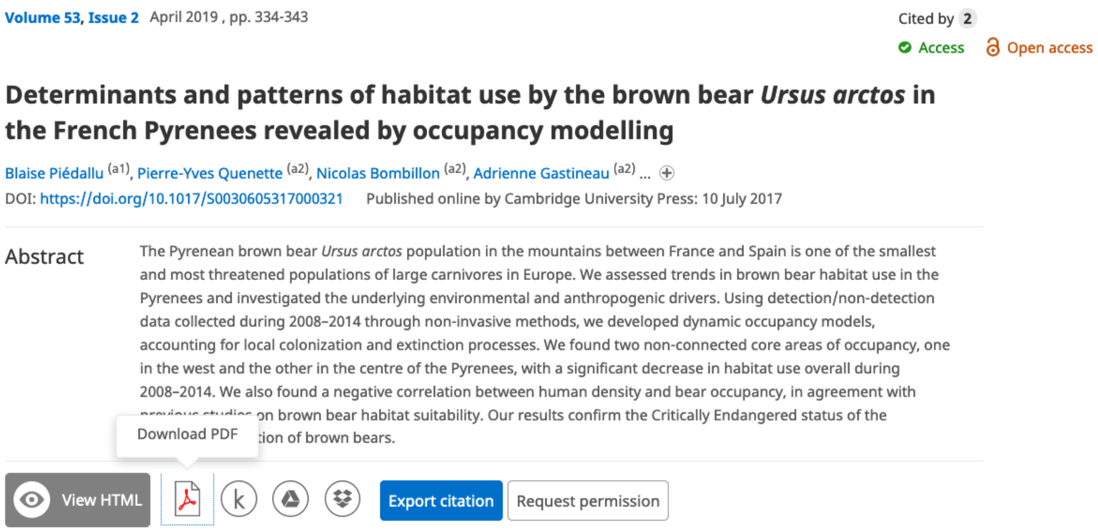
```
## Simple feature collection with 100 features and 6 fields
## geometry type:  MULTIPOLYGON
## dimension:      XY
## bbox:           xmin: -84.32385 ymin: 33.88199 xmax: -75.45698 ymax: 36.58965
## epsg (SRID):    4267
## proj4string:    +proj=longlat +datum=NAD27 +no_defs
## precision:      double (default; no precision model)
## First 3 features:
##   BIR74 SID74 NWBIR74 BIR79 SID79 NWBIR79                           geom
## 1  1091     1      10  1364     0      19 MULTIPOLYGON(((-81.47275543...
## 2   487     0      10   542     3      12 MULTIPOLYGON(((-81.23989105...
## 3  3188     5     208  3616     6     260 MULTIPOLYGON(((-80.45634460...
```

Simple feature
Simple feature geometry list-colum (sfc)
Simple feature geometry (sfg)

Simple feature geometry `sfg`

First steps

Case study



Volume 53, Issue 2 April 2019 , pp. 334-343

Cited by 2

Access Open access

Determinants and patterns of habitat use by the brown bear *Ursus arctos* in the French Pyrenees revealed by occupancy modelling

Blaise Piédallu (a1), Pierre-Yves Quenette (a2), Nicolas Bombillon (a2), Adrienne Gastineau (a2) ...

DOI: https://doi.org/10.1017/S0030605317000321 Published online by Cambridge University Press: 10 July 2017

Abstract

The Pyrenean brown bear *Ursus arctos* population in the mountains between France and Spain is one of the smallest and most threatened populations of large carnivores in Europe. We assessed trends in brown bear habitat use in the Pyrenees and investigated the underlying environmental and anthropogenic drivers. Using detection/non-detection data collected during 2008–2014 through non-invasive methods, we developed dynamic occupancy models, accounting for local colonization and extinction processes. We found two non-connected core areas of occupancy, one in the west and the other in the centre of the Pyrenees, with a significant decrease in habitat use overall during 2008–2014. We also found a negative correlation between human density and bear occupancy, in agreement with previous studies on brown bear habitat suitability. Our results confirm the Critically Endangered status of the tion of brown bears.

Read in spatial data

```
studysites_raw <- st_read("../shp/bearpyrenees.shp")
```

```
## Reading layer `bearpyrenees' from data source `D:\Git\shp\bearpyrenees.shp' using drive
## Simple feature collection with 138 features and 4 fields
## geometry type:  MULTIPOLYGON
## dimension:      XY
## bbox:           xmin: 315722.7 ymin: 1704775 xmax: 644368 ymax: 1800721
## projected CRS:  Lambert_Conformal_Conic
```

Examine structure

```
glimpse(studysites_raw)
```

```
## Rows: 138
## Columns: 5
## $ Numero     <dbl> 640101, 640102, 640203, 640204, 640202, 640201, 640301, ...
## $ pres_08_12 <fct> Absence, Absence, Occasionnelle, Occasionnelle, Absence,...
## $ Perimeter  <dbl> 53446.71, 60814.07, 38908.05, 32749.40, 36869.03, 37629....
## $ Iti2012    <int> NA, NA, 1, 1, NA, NA, NA, 1, 1, 1, 1, NA, 0, 1, 1, 1, 1,...
## $ geometry   <MULTIPOLYGON [m]> MULTIPOLYGON (((315785.1 17..., MULTIPOLYGO...
```

Examine structure

```
studysites_raw
```

```
## Simple feature collection with 138 features and 4 fields
## geometry type:  MULTIPOLYGON
## dimension:      XY
## bbox:           xmin: 315722.7 ymin: 1704775 xmax: 644368 ymax: 1800721
## projected CRS:  Lambert_Conformal_Conic
## First 10 features:
##    Numero    pres_08_12 Perimeter Iti2012                       geometry
## 1  640101       Absence  53446.71      NA MULTIPOLYGON (((315785.1 17...
## 2  640102       Absence  60814.07      NA MULTIPOLYGON (((326069.1 17...
## 3  640203 Occasionnelle  38908.05       1 MULTIPOLYGON (((328868.9 17...
## 4  640204 Occasionnelle  32749.40       1 MULTIPOLYGON (((338223.2 17...
## 5  640202       Absence  36869.03      NA MULTIPOLYGON (((348261.2 17...
## 6  640201       Absence  37629.04      NA MULTIPOLYGON (((347345.4 17...
## 7  640301       Absence  29586.76      NA MULTIPOLYGON (((350581.2 17...
## 8  640302       Absence  25304.44       1 MULTIPOLYGON (((353106.3 17...
## 9  640401 Occasionnelle  44181.28       1 MULTIPOLYGON (((353084 1783...
## 10 640402      Reguliere  43850.75       1 MULTIPOLYGON (((359081.4 17...
```

Select relevant columns

```
studysites <- studysites_raw %>%
  select('bearpresence' = pres_08_12,
         'idsite' = Numero)
studysites
```

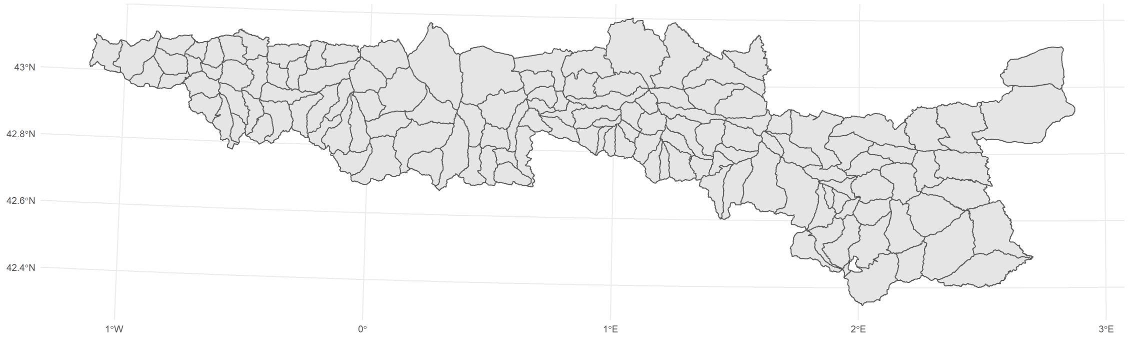
```
## Simple feature collection with 138 features and 2 fields
## geometry type:  MULTIPOLYGON
## dimension:      XY
## bbox:           xmin: 315722.7 ymin: 1704775 xmax: 644368 ymax: 1800721
## projected CRS:  Lambert_Conformal_Conic
## First 10 features:
##     bearpresence idsite                       geometry
## 1        Absence 640101 MULTIPOLYGON (((315785.1 17...
## 2        Absence 640102 MULTIPOLYGON (((326069.1 17...
## 3  Occasionnelle 640203 MULTIPOLYGON (((328868.9 17...
## 4  Occasionnelle 640204 MULTIPOLYGON (((338223.2 17...
## 5        Absence 640202 MULTIPOLYGON (((348261.2 17...
## 6        Absence 640201 MULTIPOLYGON (((347345.4 17...
## 7        Absence 640301 MULTIPOLYGON (((350581.2 17...
## 8        Absence 640302 MULTIPOLYGON (((353106.3 17...
## 9  Occasionnelle 640401 MULTIPOLYGON (((353084 1783...
## 10     Reguliere 640402 MULTIPOLYGON (((359081.4 17...
```

Our first map

```
studysites %>%
  ggplot() +
  geom_sf() +
  labs(title = 'Brown bear monitoring sites in the French Pyrenees mountains',
       subtitle = 'Source: French Game and Wildlife Agency')
```

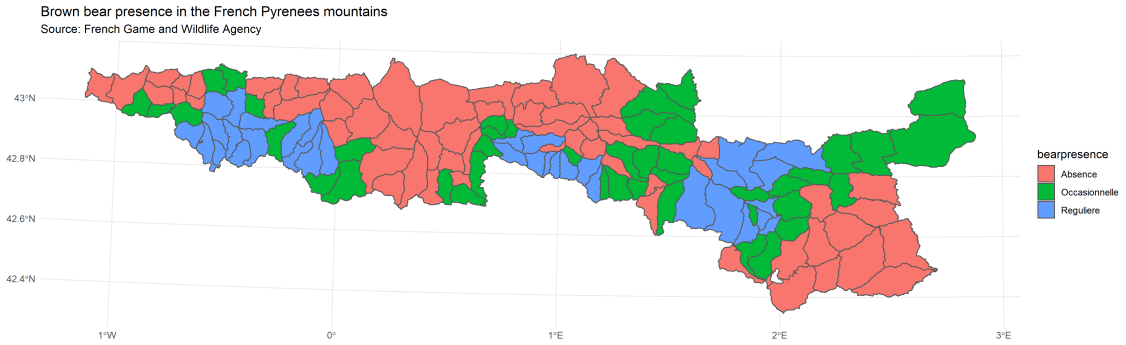
Brown bear monitoring sites in the French Pyrenees mountains

Source: French Game and Wildlife Agency



Where did the species occur?

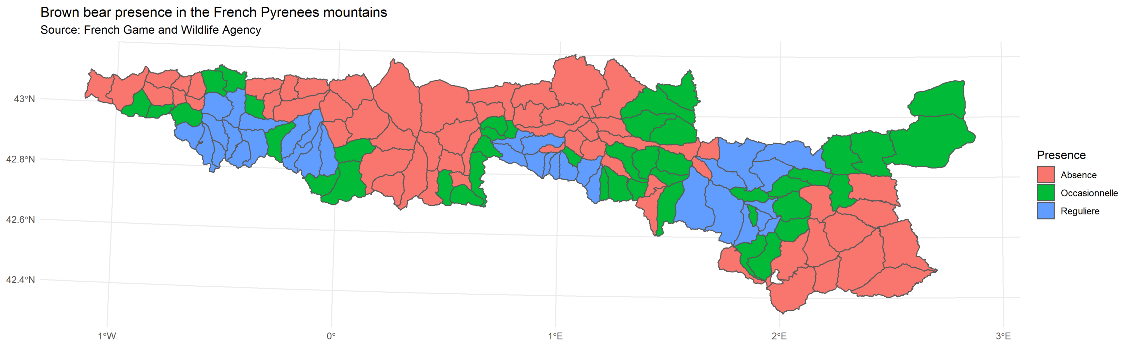
```r
studysites %>%
  ggplot() +
  geom_sf(aes(fill = bearpresence)) +
  labs(title = 'Brown bear presence in the French Pyrenees mountains',
       subtitle = 'Source: French Game and Wildlife Agency')
```


Brown bear presence in the French Pyrenees mountains

Source: French Game and Wildlife Agency

Where did the species occur?

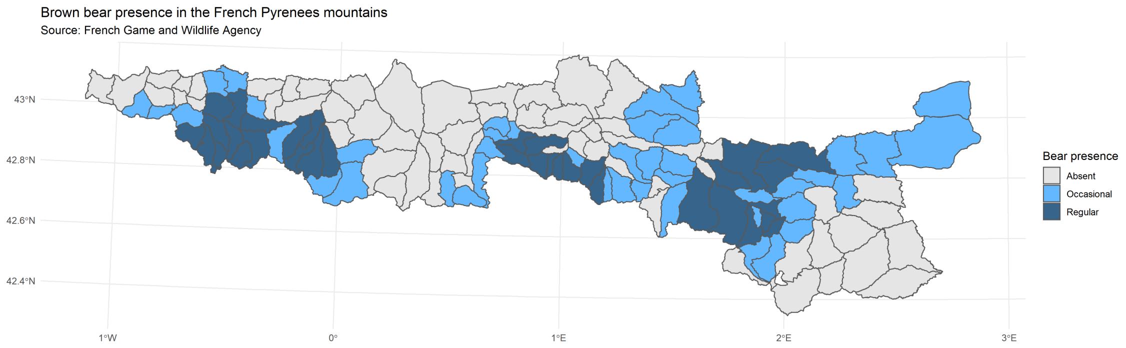
```r
studysites %>%
  ggplot() +
  geom_sf(aes(fill = bearpresence)) +
  labs(title = 'Brown bear presence in the French Pyrenees mountains',
       subtitle = 'Source: French Game and Wildlife Agency',
       fill = "Presence")
```


Brown bear presence in the French Pyrenees mountains

Source: French Game and Wildlife Agency

Take control of your legends

```
studysites %>%
  ggplot() +
  geom_sf(aes(fill = bearpresence)) +
  labs(title = 'Brown bear presence in the French Pyrenees mountains',
       subtitle = 'Source: French Game and Wildlife Agency') +
  scale_fill_manual(values = c('gray90','steelblue1','steelblue4'),
                    name = "Bear presence",
                    labels = c("Absent", "Occasional", "Regular"))
```


Brown bear presence in the French Pyrenees mountains
Source: French Game and Wildlife Agency
43°N
42.8°N
42.6°N
42.4°N
1°W
0°
1°E
2°E
3°E
Bear presence
Absent
Occasional
Regular

Spatial operations: transform, crop, intersect, join

Forest cover

- Forest cover might be a driver of brown bear distribution
- We use corine land cover (CLC) data (2012 version) to get forest cover
- Data can be downloaded from the official website
- An explanation of what's in the data is available here.

Read in data

```
clc2012 <- st_read("../shp/CLC12_FR_RGF.shp")
```

```
## Reading layer `CLC12_FR_RGF' from data source `D:\Git\shp\CLC12_FR_RGF.shp' using drive
## Simple feature collection with 274573 features and 3 fields
## geometry type:  POLYGON
## dimension:      XY
## bbox:           xmin: 73767.79 ymin: 6021170 xmax: 1267595 ymax: 7133490
## projected CRS:  RGF93_Lambert_93
```

Read in data

```
clc2012
```

```
## Simple feature collection with 274573 features and 3 fields
## geometry type:  POLYGON
## dimension:      XY
## bbox:           xmin: 73767.79 ymin: 6021170 xmax: 1267595 ymax: 7133490
## projected CRS:  RGF93_Lambert_93
## First 10 features:
##           ID CODE_12      AREA_HA                       geometry
## 1  FR-274573     523 4.879872e+06 POLYGON ((657640.6 7133490,...
## 2  FR-271961     423 8.555978e+02 POLYGON ((669050 7111012, 6...
## 3  FR-265902     331 3.092622e+02 POLYGON ((669132.3 7110834,...
## 4  FR-250624     322 7.593459e+01 POLYGON ((669181.5 7110656,...
## 5  FR-250623     322 1.826213e+02 POLYGON ((666715.4 7109580,...
## 6   FR-25381     112 3.282088e+02 POLYGON ((666715.4 7109580,...
## 7  FR-265901     331 2.503051e+01 POLYGON ((667326.7 7108770,...
## 8  FR-147959     242 6.157960e+01 POLYGON ((669097.7 7109171,...
## 9  FR-270463     333 2.949202e+01 POLYGON ((666773.8 7109345,...
## 10  FR-62513     211 9.808718e+05 POLYGON ((669737.1 7108874,...
```

Extract forest codes

```
forest <- clc2012 %>%
  filter(CODE_12 == 311 | CODE_12 == 312 | CODE_12 == 313)
forest
```

```
## Simple feature collection with 69111 features and 3 fields
## geometry type:  POLYGON
## dimension:      XY
## bbox:           xmin: 125951.9 ymin: 6021917 xmax: 1243919 ymax: 7100648
## projected CRS:  RGF93_Lambert_93
## First 10 features:
##           ID CODE_12    AREA_HA                       geometry
## 1  FR-211653     311  114.81939 POLYGON ((657867.2 7100629,...
## 2  FR-227070     312   29.01440 POLYGON ((632270.7 7100616,...
## 3  FR-211652     311   36.78086 POLYGON ((630186.3 7100313,...
## 4  FR-211651     311   90.71384 POLYGON ((625545 7098175, 6...
## 5  FR-211650     311   32.23925 POLYGON ((660260.8 7097323,...
## 6  FR-211649     311   43.00924 POLYGON ((607150.2 7090018,...
## 7  FR-211648     311   25.39949 POLYGON ((626532.2 7086540,...
## 8  FR-211647     311   45.10054 POLYGON ((669017.5 7085727,...
## 9  FR-211646     311 1005.69349 POLYGON ((618120.9 7082302,...
## 10 FR-211645     311   62.38459 POLYGON ((603423.4 7085365,...
```

Use same coordinates system for map of the Pyrénées and forest layer

```
studysites <- studysites %>%
  st_transform(crs = st_crs(forest))
studysites
```

```
## Simple feature collection with 138 features and 2 fields
## geometry type:  MULTIPOLYGON
## dimension:      XY
## bbox:           xmin: 362039 ymin: 6139034 xmax: 690160.2 ymax: 6235505
## projected CRS:  RGF93_Lambert_93
## First 10 features:
##     bearpresence idsite                       geometry
## 1        Absence 640101 MULTIPOLYGON (((362110.5 62...
## 2        Absence 640102 MULTIPOLYGON (((372352.1 62...
## 3  Occasionnelle 640203 MULTIPOLYGON (((375125.4 62...
## 4  Occasionnelle 640204 MULTIPOLYGON (((384447.4 62...
## 5        Absence 640202 MULTIPOLYGON (((394521 6219...
## 6        Absence 640201 MULTIPOLYGON (((393646.7 62...
## 7        Absence 640301 MULTIPOLYGON (((396923.9 62...
## 8        Absence 640302 MULTIPOLYGON (((399383.2 62...
## 9  Occasionnelle 640401 MULTIPOLYGON (((399338.3 62...
## 10     Reguliere 640402 MULTIPOLYGON (((405268.8 62...
```

Calculate area of each site

```
studysites %>%
  mutate(area = st_area(.),
         .before = 1)
```

```
## Simple feature collection with 138 features and 3 fields
## geometry type:  MULTIPOLYGON
## dimension:      XY
## bbox:           xmin: 362039 ymin: 6139034 xmax: 690160.2 ymax: 6235505
## projected CRS:  RGF93_Lambert_93
## First 10 features:
##               area  bearpresence idsite                       geometry
## 1    79618924 [m^2]       Absence 640101 MULTIPOLYGON (((362110.5 62...
## 2   122155210 [m^2]       Absence 640102 MULTIPOLYGON (((372352.1 62...
## 3    64012619 [m^2] Occasionnelle 640203 MULTIPOLYGON (((375125.4 62...
## 4    36208552 [m^2] Occasionnelle 640204 MULTIPOLYGON (((384447.4 62...
## 5    67904470 [m^2]       Absence 640202 MULTIPOLYGON (((394521 6219...
## 6    58455872 [m^2]       Absence 640201 MULTIPOLYGON (((393646.7 62...
## 7    30310636 [m^2]       Absence 640301 MULTIPOLYGON (((396923.9 62...
## 8    25866722 [m^2]       Absence 640302 MULTIPOLYGON (((399383.2 62...
## 9    72321198 [m^2] Occasionnelle 640401 MULTIPOLYGON (((399338.3 62...
## 10   67797321 [m^2]     Reguliere 640402 MULTIPOLYGON (((405268.8 62...
```

Convert area in km^2

```
studysites %>%
  mutate(.before = 1,
         area = st_area(.),
         areakm2 = units::set_units(area, km^2))
```

```
## Simple feature collection with 138 features and 4 fields
## geometry type:  MULTIPOLYGON
## dimension:      XY
## bbox:           xmin: 362039 ymin: 6139034 xmax: 690160.2 ymax: 6235505
## projected CRS:  RGF93_Lambert_93
## First 10 features:
##               area          areakm2  bearpresence idsite
## 1    79618924 [m^2]  79.61892 [km^2]       Absence 640101
## 2   122155210 [m^2] 122.15521 [km^2]       Absence 640102
## 3    64012619 [m^2]  64.01262 [km^2] Occasionnelle 640203
## 4    36208552 [m^2]  36.20855 [km^2] Occasionnelle 640204
## 5    67904470 [m^2]  67.90447 [km^2]       Absence 640202
## 6    58455872 [m^2]  58.45587 [km^2]       Absence 640201
## 7    30310636 [m^2]  30.31064 [km^2]       Absence 640301
## 8    25866722 [m^2]  25.86672 [km^2]       Absence 640302
## 9    72321198 [m^2]  72.32120 [km^2] Occasionnelle 640401
## 10   67797321 [m^2]  67.79732 [km^2]     Reguliere 640402
##                          geometry
## 1  MULTIPOLYGON (((362110.5 62...
## 2  MULTIPOLYGON (((372352.1 62...
## 3  MULTIPOLYGON (((375125.4 62...
## 4  MULTIPOLYGON (((384447.4 62...
```

Define big sites (area > 300 km^2)

```
studysites <- studysites %>%
  mutate(.before = 1,
         area = st_area(.),
         areakm2 = units::set_units(area, km^2),
         bigsites = ifelse(as.numeric(areakm2) > 300, areakm2, NA))
studysites
```

```
## Simple feature collection with 138 features and 5 fields
## geometry type:  MULTIPOLYGON
## dimension:      XY
## bbox:           xmin: 362039 ymin: 6139034 xmax: 690160.2 ymax: 6235505
## projected CRS:  RGF93_Lambert_93
## First 10 features:
##               area          areakm2 bigsites  bearpresence idsite
## 1    79618924 [m^2]  79.61892 [km^2]       NA       Absence 640101
## 2   122155210 [m^2] 122.15521 [km^2]       NA       Absence 640102
## 3    64012619 [m^2]  64.01262 [km^2]       NA Occasionnelle 640203
## 4    36208552 [m^2]  36.20855 [km^2]       NA Occasionnelle 640204
## 5    67904470 [m^2]  67.90447 [km^2]       NA       Absence 640202
## 6    58455872 [m^2]  58.45587 [km^2]       NA       Absence 640201
## 7    30310636 [m^2]  30.31064 [km^2]       NA       Absence 640301
## 8    25866722 [m^2]  25.86672 [km^2]       NA       Absence 640302
## 9    72321198 [m^2]  72.32120 [km^2]       NA Occasionnelle 640401
## 10   67797321 [m^2]  67.79732 [km^2]       NA     Reguliere 640402
##                          geometry
## 1  MULTIPOLYGON (((362110.5 62...
## 2  MULTIPOLYGON (((372352.1 62...
```

Map again, with area on top of big sites

```
studysites %>%
  ggplot() +
  geom_sf() +
  geom_sf_label(aes(label = round(bigsites))) +
  labs(title = 'Brown bear big monitoring sites in the French Pyrenees mountains
       subtitle = 'Big sites have area > 300km2',
       caption = 'Data from: French Game and Wildlife Agency',
       x = "", y = "")
```

Crop forest to match study area boundaries

```
forest %>%
  st_crop(st_bbox(studysites)) %>%
  as_tibble()
```

```
## # A tibble: 2,504 x 4
##    ID       CODE_12 AREA_HA                                                geometry
##    <fct>    <fct>     <dbl>                                          <GEOMETRY [m]>
##  1 FR-1744~ 311       2807. MULTIPOLYGON (((479636.9 6235505, 479719.6 6235430,~
##  2 FR-1744~ 311       2409. POLYGON ((442252.5 6235505, 442429.7 6235438, 44254~
##  3 FR-1743~ 311       2843. MULTIPOLYGON (((396335.1 6235505, 396346.2 6235435,~
##  4 FR-1742~ 311        476. POLYGON ((471577.9 6235505, 471467.3 6235289, 47137~
##  5 FR-1742~ 311        588. POLYGON ((490987.1 6235505, 491006.8 6235337, 49098~
##  6 FR-1742~ 311        354. MULTIPOLYGON (((484715.2 6235505, 484694.9 6235499,~
##  7 FR-1742~ 311        579. MULTIPOLYGON (((500091.4 6235505, 500058.3 6235481,~
##  8 FR-1741~ 311        357. POLYGON ((509149.7 6235505, 509148.6 6235486, 50933~
##  9 FR-1741~ 311       6202. MULTIPOLYGON (((379133.9 6235505, 379132.4 6235483,~
## 10 FR-1741~ 311        292. POLYGON ((451256.3 6235505, 451260.6 6235447, 45126~
## # ... with 2,494 more rows
```

Then intersect the two layers

```
forest %>%
  st_crop(st_bbox(studysites)) %>%
  st_intersection(studysites) %>%
  as_tibble()
```

```
## # A tibble: 2,177 x 9
##    ID    CODE_12 AREA_HA     area  areakm2 bigsites bearpresence idsite
##    <fct> <fct>     <dbl>    [m^2]   [km^2]    <dbl> <fct>         <dbl>
##  1 FR-1~ 311       243.  79618924 79.61892       NA Absence      640101
##  2 FR-1~ 311     18041.  79618924 79.61892       NA Absence      640101
##  3 FR-1~ 311       289.  79618924 79.61892       NA Absence      640101
##  4 FR-1~ 311       235.  79618924 79.61892       NA Absence      640101
##  5 FR-1~ 311       153.  79618924 79.61892       NA Absence      640101
##  6 FR-1~ 311        83.7 79618924 79.61892       NA Absence      640101
##  7 FR-1~ 311        26.9 79618924 79.61892       NA Absence      640101
##  8 FR-1~ 311      2697.  79618924 79.61892       NA Absence      640101
##  9 FR-1~ 311        49.2 79618924 79.61892       NA Absence      640101
## 10 FR-1~ 311        52.6 79618924 79.61892       NA Absence      640101
## # ... with 2,167 more rows, and 1 more variable: geometry <GEOMETRY [m]>
```

Get forest area for each intersected `sfg`

```
forest %>%
  st_crop(st_bbox(studysites)) %>%
  st_intersection(studysites) %>%
  mutate(area = st_area(.)) %>%
  as_tibble()
```

```
## # A tibble: 2,177 x 9
##    ID    CODE_12 AREA_HA     area  areakm2 bigsites bearpresence idsite
##    <fct> <fct>     <dbl>    [m^2]   [km^2]    <dbl> <fct>         <dbl>
##  1 FR-1~ 311       243.    64094~ 79.61892       NA Absence      640101
##  2 FR-1~ 311     18041.   535496~ 79.61892       NA Absence      640101
##  3 FR-1~ 311       289.   258490~ 79.61892       NA Absence      640101
##  4 FR-1~ 311       235.    56759~ 79.61892       NA Absence      640101
##  5 FR-1~ 311       153.   153053~ 79.61892       NA Absence      640101
##  6 FR-1~ 311        83.7   83729~ 79.61892       NA Absence      640101
##  7 FR-1~ 311        26.9   26851~ 79.61892       NA Absence      640101
##  8 FR-1~ 311      2697.  2193312~ 79.61892       NA Absence      640101
##  9 FR-1~ 311        49.2   49232~ 79.61892       NA Absence      640101
## 10 FR-1~ 311        52.6   52602~ 79.61892       NA Absence      640101
## # ... with 2,167 more rows, and 1 more variable: geometry <GEOMETRY [m]>
```

Sum forest over all study sites

```
forestpyrenees <- forest %>%
  st_crop(st_bbox(studysites)) %>%
  st_intersection(studysites) %>%
  mutate(area = st_area(.)) %>%
  group_by(idsite) %>% # groups a data frame by variables
  summarise(areaforest = sum(area)) %>% # perform group-wise summaries
  as_tibble() %>%
  select(-geometry)
forestpyrenees
```

```
## # A tibble: 138 x 2
##    idsite areaforest
##     <dbl>      [m^2]
##  1  11403  157762282
##  2  11404   32526319
##  3  90101  137218858
##  4  90102   86944708
##  5  90103   82872880
##  6  90104   42241267
##  7  90201   19224150
##  8  90202   26309123
##  9  90203   31962484
## 10  90301    8958589
## # ... with 128 more rows
```

Join `sf` and `tibble` objects

More info here

```
studysites %>%
  inner_join(forestpyrenees, by = 'idsite')
```

```
## Simple feature collection with 138 features and 6 fields
## geometry type:  MULTIPOLYGON
## dimension:      XY
## bbox:           xmin: 362039 ymin: 6139034 xmax: 690160.2 ymax: 6235505
## projected CRS:  RGF93_Lambert_93
## First 10 features:
##               area          areakm2 bigsites  bearpresence idsite
## 1   79618924 [m^2]  79.61892 [km^2]       NA       Absence 640101
## 2  122155210 [m^2] 122.15521 [km^2]       NA       Absence 640102
## 3   64012619 [m^2]  64.01262 [km^2]       NA Occasionnelle 640203
## 4   36208552 [m^2]  36.20855 [km^2]       NA Occasionnelle 640204
## 5   67904470 [m^2]  67.90447 [km^2]       NA       Absence 640202
## 6   58455872 [m^2]  58.45587 [km^2]       NA       Absence 640201
## 7   30310636 [m^2]  30.31064 [km^2]       NA       Absence 640301
## 8   25866722 [m^2]  25.86672 [km^2]       NA       Absence 640302
## 9   72321198 [m^2]  72.32120 [km^2]       NA Occasionnelle 640401
## 10  67797321 [m^2]  67.79732 [km^2]       NA     Reguliere 640402
##         areaforest                       geometry
## 1  42105194 [m^2] MULTIPOLYGON (((362110.5 62...
## 2  60136750 [m^2] MULTIPOLYGON (((372352.1 62...
```

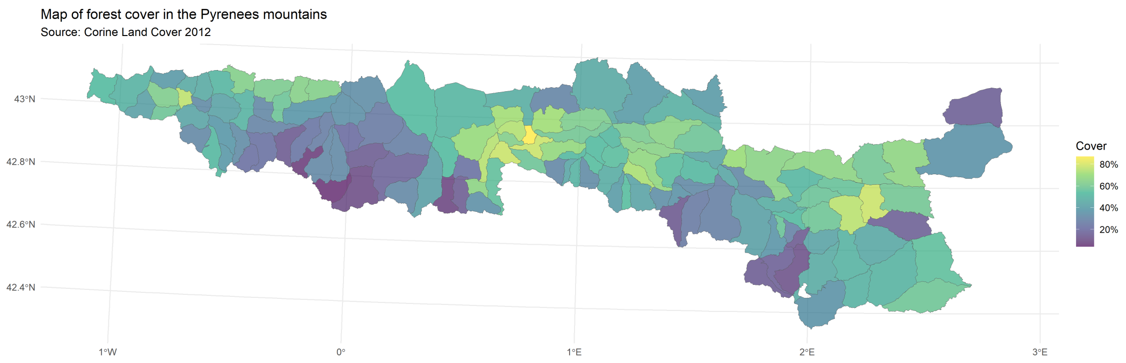
Calculate forest cover

```
covariates <- studysites %>%
  inner_join(forestpyrenees, by = 'idsite') %>%
  mutate(.before = 1,
         forestcover = areaforest / area)
```

Map forest cover

```
covariates %>%
  ggplot() +
  aes(fill = as.numeric(forestcover)) +
  geom_sf(lwd = 0.1) +
  scale_fill_viridis_c(
    labels = scales::percent_format(), #<< format percentage
    name = 'Cover',
    alpha = 0.7) +  #<< control transparency
  labs(title = 'Map of forest cover in the Pyrenees mountains',
       subtitle = 'Source: Corine Land Cover 2012')
```

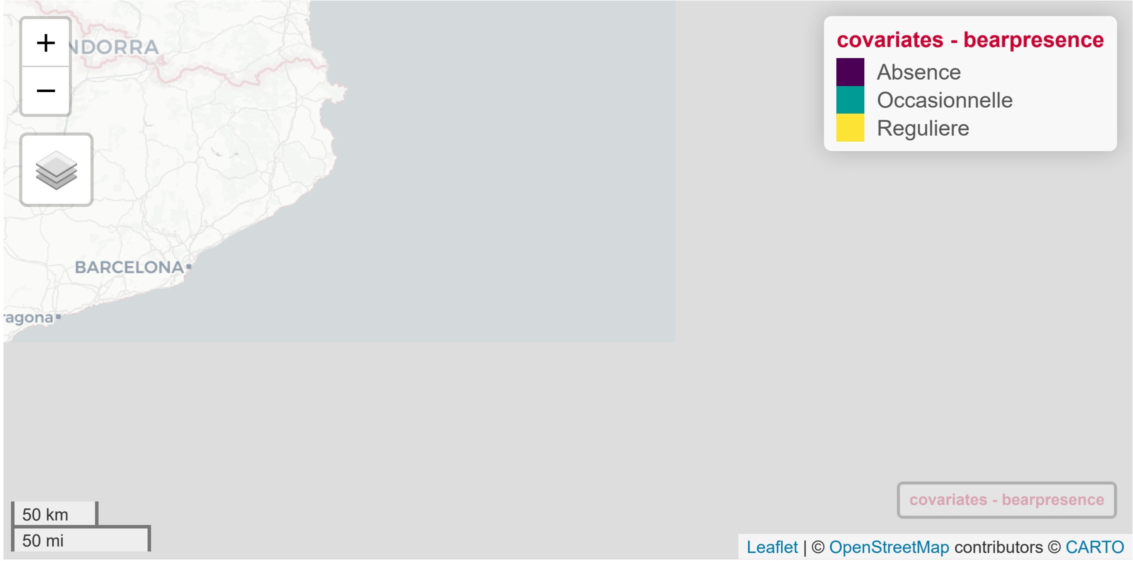
Map forest cover


Map of forest cover in the Pyrenees mountains
Source: Corine Land Cover 2012

Interactive map with `mapview`

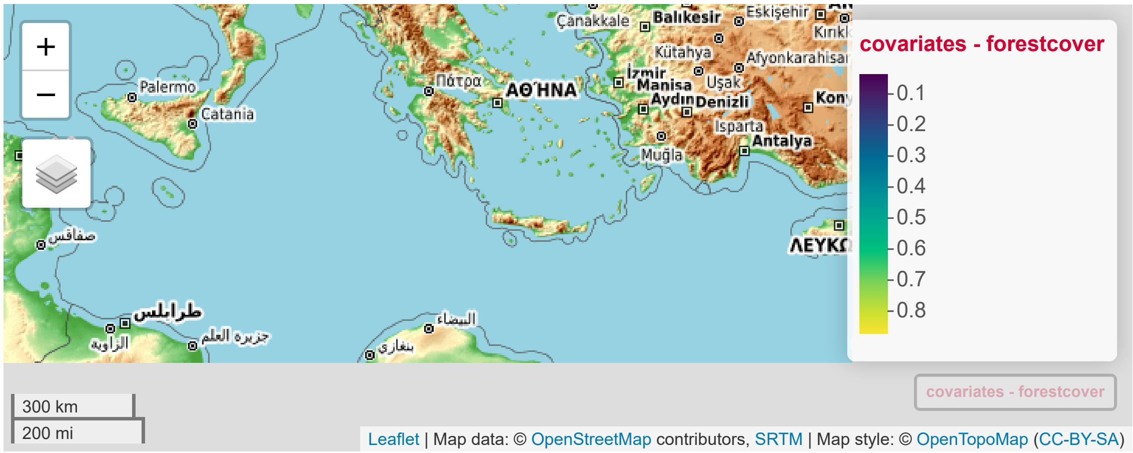
More info here

```
library(mapview)
mapview(covariates, zcol = "bearpresence")
```


+
−
NDORRA
BARCELONA
agona
covariates - bearpresence
Absence
Occasionnelle
Reguliere
covariates - bearpresence
50 km
50 mi
Leaflet | © OpenStreetMap contributors © CARTO

Interactive map with `mapview`

```
covariates <- covariates %>% mutate(forestcover = as.numeric(forestcover))
mapview(covariates, zcol = "forestcover", map.types = "OpenTopoMap")
```



Leaflet | Map data: © OpenStreetMap contributors, SRTM | Map style: © OpenTopoMap (CC-BY-SA)

```
# map.types = "CartoDB.Positron", "CartoDB.DarkMatter", "OpenStreetMap",
# "Esri.WorldImagery"
```

Human density

- Human density might be a driver of brown bear distribution
- We use data on population size of cities in France
- A more detailed analysis of human density in France is available from @SharpSightLabs here

Human density

```
# url.france_pop <- url("https://vrzkj25a871bpq7t1ugcgmn9-wpengine.netdna-ssl.co
# load(url.france_pop)
load("../shp/france_population_data_2016.RData")
glimpse(df.france)
```

```
## Rows: 35,798
## Columns: 18
## $ ID_GEOFLA  <fct> COMMUNE0000000000000001, COMMUNE0000000000000002, COMM...
## $ CODE_COM   <fct> 216, 033, 009, 225, 890, 018, 113, 319, 097, 070, 046, 5...
## $ INSEE_COM  <fct> 32216, 47033, 32009, 38225, 62890, 08018, 32113, 10319, ...
## $ NOM_COM    <fct> LOURTIES-MONBRUN, BOUDY-DE-BEAUREGARD, ARMOUS-ET-CAU, AU...
## $ STATUT     <fct> Commune simple, Commune simple, Commune simple, Commune ...
## $ X_CHF_LIEU <int> 500820, 516424, 472979, 898640, 640049, 824246, 461332, ...
## $ Y_CHF_LIEU <int> 6264958, 6384852, 6278963, 6450689, 7028672, 6908952, 63...
## $ X_CENTROID <int> 500515, 515575, 473004, 898625, 640115, 824391, 460721, ...
## $ Y_CENTROID <int> 6265413, 6385938, 6278937, 6451597, 7029900, 6908954, 63...
## $ Z_MOYEN    <int> 252, 112, 221, 1234, 79, 125, 134, 167, 752, 438, 1276, ...
## $ SUPERFICIE <dbl> 966, 1019, 932, 3371, 1023, 438, 919, 1904, 2217, 2667, ...
## $ POPULATION <dbl> 139, 414, 95, 2973, 178, 80, 97, 362, 296, 901, 10, 166,...
## $ CODE_ARR   <fct> 3, 3, 3, 1, 4, 4, 2, 3, 2, 2, 2, 3, 2, 1, 1, 3, 2, 1, 3,...
## $ CODE_DEPT  <fct> 32, 47, 32, 38, 62, 08, 32, 10, 06, 42, 31, 71, 53, 16, ...
## $ NOM_DEPT   <fct> GERS, LOT-ET-GARONNE, GERS, ISERE, PAS-DE-CALAIS, ARDENN...
## $ CODE_REG   <fct> 76, 75, 76, 84, 32, 44, 76, 44, 93, 84, 76, 27, 52, 75, ...
## $ NOM_REG    <fct> LANGUEDOC-ROUSSILLON-MIDI-PYRENEES, AQUITAINE-LIMOUSIN-P...
## $ geometry   <MULTIPOLYGON [m]> MULTIPOLYGON (((499484.6 62..., MULTIPOLYGO...
```

Transform into lower case

```
colnames(df.france) <- colnames(df.france) %>%
  str_to_lower()
colnames(df.france)
```

```
##  [1] "id_geofla"  "code_com"   "insee_com"  "nom_com"    "statut"
##  [6] "x_chf_lieu" "y_chf_lieu" "x_centroid" "y_centroid" "z_moyen"
## [11] "superficie" "population" "code_arr"   "code_dept"  "nom_dept"
## [16] "code_reg"   "nom_reg"    "geometry"
```

Have a look to the distribution

```
df.france$population %>%
  summary()
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       0     197     442    1779    1100  458298
```

Calculate density

```
df.france <- df.france %>%
  mutate(density = population/superficie*100)
as_tibble(df.france)
```

```
## # A tibble: 35,798 x 19
##    id_geofla code_com insee_com nom_com statut x_chf_lieu y_chf_lieu x_centroid
##    <fct>     <fct>    <fct>     <fct>   <fct>       <int>      <int>      <int>
##  1 COMMUNE0~ 216      32216     LOURTI~ Commu~     500820    6264958     500515
##  2 COMMUNE0~ 033      47033     BOUDY-~ Commu~     516424    6384852     515575
##  3 COMMUNE0~ 009      32009     ARMOUS~ Commu~     472979    6278963     473004
##  4 COMMUNE0~ 225      38225     AUTRAN~ Commu~     898640    6450689     898625
##  5 COMMUNE0~ 890      62890     WILLEM~ Commu~     640049    7028672     640115
##  6 COMMUNE0~ 018      08018     ARDEUI~ Commu~     824246    6908952     824391
##  7 COMMUNE0~ 113      32113     CRAVEN~ Commu~     461332    6300782     460721
##  8 COMMUNE0~ 319      10319     RIGNY-~ Commu~     746925    6790005     747181
##  9 COMMUNE0~ 097      06097     PIERRE~ Commu~    1028827    6315717    1027327
## 10 COMMUNE0~ 070      42070     CORDEL~ Commu~     782215    6538794     782159
## # ... with 35,788 more rows, and 11 more variables: y_centroid <int>,
## #   z_moyen <int>, superficie <dbl>, population <dbl>, code_arr <fct>,
## #   code_dept <fct>, nom_dept <fct>, code_reg <fct>, nom_reg <fct>,
## #   geometry <MULTIPOLYGON [m]>, density <dbl>
```

Show density

```
df.france %>%
  pull(density) %>%
  head()
```

```
## [1] 14.38923 40.62807 10.19313 88.19341 17.39980 18.26484
```

Sum population size over sites

```
df.pyrenees <- df.france %>%
  st_transform(crs = st_crs(forest)) %>%
  st_crop(st_bbox(covariates)) %>%
  st_intersection(covariates) %>%
  group_by(idsite) %>%
  summarise(humpop = sum(population)) %>%
  as_tibble()
df.pyrenees
```

```
## # A tibble: 138 x 3
##    idsite humpop                                                geometry
##     <dbl>  <dbl>                                          <GEOMETRY [m]>
##  1  11403  10711 POLYGON ((675659.1 6192498, 675629.2 6192477, 675591.8 6192478~
##  2  11404   7576 POLYGON ((686245.1 6212172, 686121.8 6211996, 685755 6211899, ~
##  3  90101  33154 POLYGON ((549629.4 6212088, 549618 6212091, 549505.9 6212175, ~
##  4  90102  17391 POLYGON ((550438.4 6212074, 550408.3 6212065, 550303 6212029, ~
##  5  90103  19126 MULTIPOLYGON (((586211.4 6209840, 586169.9 6209762, 586160.9 6~
##  6  90104  28000 POLYGON ((586455.6 6214769, 586476.5 6214741, 586664.2 6214764~
##  7  90201   8203 POLYGON ((549096.2 6203616, 549051.8 6203591, 549032.7 6203528~
##  8  90202   2554 POLYGON ((545831.6 6198483, 545777.3 6198374, 545625.9 6198143~
##  9  90203   3505 MULTIPOLYGON (((545041.7 6197354, 545031.8 6197359, 545041.8 6~
## 10  90301    285 POLYGON ((533167.3 6195384, 533210.4 6195336, 533210.1 6195304~
## # ... with 128 more rows
```

Join, then calculate density

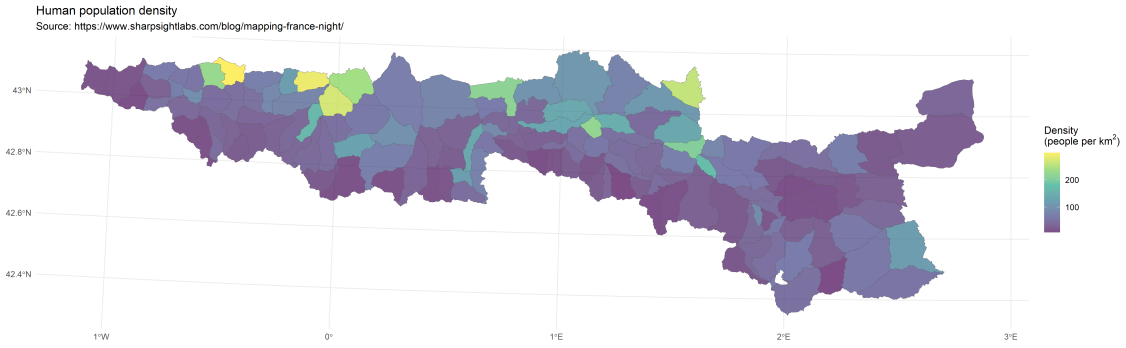
```
covariates <- covariates %>%
  inner_join(df.pyrenees, by = 'idsite') %>%
  mutate(.before = 1,
         humdens = humpop / (area/1000000))
as_tibble(covariates)
```

```
## # A tibble: 138 x 11
##      humdens forestcover     area  areakm2 bigsites bearpresence idsite
##      [1/m^2]       <dbl>    [m^2]   [km^2]    <dbl> <fct>         <dbl>
##  1 15.73746        0.529  796189~  79.618~       NA Absence      640101
##  2 16.38080        0.492 1221552~ 122.155~       NA Absence      640102
##  3 10.01365        0.416  640126~  64.012~       NA Occasionnel~ 640203
##  4 51.83858        0.580  362085~  36.208~       NA Occasionnel~ 640204
##  5 30.42510        0.623  679044~  67.904~       NA Absence      640202
##  6 68.13687        0.385  584558~  58.455~       NA Absence      640201
##  7 58.79124        0.460  303106~  30.310~       NA Absence      640301
##  8 70.63129        0.760  258667~  25.866~       NA Absence      640302
##  9 39.78087        0.433  723211~  72.321~       NA Occasionnel~ 640401
## 10 12.74387        0.312  677973~  67.797~       NA Reguliere    640402
## # ... with 128 more rows, and 4 more variables: areaforest [m^2], humpop <dbl>,
## #   geometry.x <MULTIPOLYGON [m]>, geometry.y <GEOMETRY [m]>
```

Map human population density

```
covariates %>%
  ggplot() +
  aes(fill = as.numeric(humdens)) +
  geom_sf(lwd = 0.1) +
  scale_fill_viridis_c(
    name = bquote('Density\n(people per km'^2*')'),
    alpha = 0.7) +
  labs(title = 'Human population density',
       subtitle = 'Source: https://www.sharpsightlabs.com/blog/mapping-france-ni
```

Map human population density



Human population density

Source: https://www.sharpsightlabs.com/blog/mapping-france-night/

Spatial operations: distance

Distance to highways

- Distance to highways might be a driver of brown bear distribution
- We use data from Route500 database

Read in data

```r
roads <- st_read("../shp/TRONCON_ROUTE.shp")
```

```
## Reading layer `TRONCON_ROUTE' from data source `D:\Git\shp\TRONCON_ROUTE.shp' using dri
## Simple feature collection with 339250 features and 12 fields
## geometry type:  LINESTRING
## dimension:      XY
## bbox:           xmin: 100076.5 ymin: 6021027 xmax: 1329738 ymax: 7120638
## projected CRS:  RGF93_Lambert_93
```

Read in data

```
as_tibble(roads)
```

```
## # A tibble: 339,250 x 13
##    ID_RTE500 VOCATION NB_CHAUSSE NB_VOIES ETAT  ACCES RES_VERT SENS  RES_EUROPE
##        <int> <fct>    <fct>      <fct>    <fct> <fct> <fct>    <fct> <fct>
##  1         1 "Liaiso~ "1 chauss~ "1 voie~ "Rev~ Libre N'appar~ Doub~ <NA>
##  2         2 "Liaiso~ "1 chauss~ "1 voie~ "Rev~ Libre N'appar~ Doub~ <NA>
##  3         3 "Liaiso~ "1 chauss~ "1 voie~ "Rev~ Libre N'appar~ Doub~ <NA>
##  4         4 "Liaiso~ "1 chauss~ "1 voie~ "Rev~ Libre N'appar~ Doub~ <NA>
##  5         5 "Liaiso~ "1 chauss~ "1 voie~ "Rev~ Libre N'appar~ Doub~ <NA>
##  6         6 "Liaiso~ "1 chauss~ "1 voie~ "Rev~ Libre N'appar~ Doub~ <NA>
##  7         7 "Liaiso~ "1 chauss~ "2 voie~ "Rev~ Libre N'appar~ Sens~ <NA>
##  8         8 "Liaiso~ "1 chauss~ "1 voie~ "Rev~ Libre N'appar~ Doub~ <NA>
##  9         9 "Liaiso~ "1 chauss~ "1 voie~ "Rev~ Libre N'appar~ Doub~ <NA>
## 10        10 "Liaiso~ "1 chauss~ "1 voie~ "Rev~ Libre N'appar~ Doub~ <NA>
## # ... with 339,240 more rows, and 4 more variables: NUM_ROUTE <fct>,
## #   CLASS_ADM <fct>, LONGUEUR <dbl>, geometry <LINESTRING [m]>
```

Focus on highways

```
highways <- roads %>%
  filter(CLASS_ADM == "Autoroute")
as_tibble(highways)
```

```
## # A tibble: 4,506 x 13
##    ID_RTE500 VOCATION NB_CHAUSSE NB_VOIES ETAT  ACCES RES_VERT SENS  RES_EUROPE
##        <int> <fct>    <fct>      <fct>    <fct> <fct> <fct>    <fct> <fct>
##  1        32 Type au~ "2 chauss~ "Sans o~ "Rev~ "A p~ Apparti~ Doub~ <NA>
##  2        33 Type au~ "1 chauss~ "2 voie~ "Rev~ "A p~ Apparti~ Doub~ <NA>
##  3      1041 Type au~ "1 chauss~ "1 voie~ "Rev~ "Lib~ Apparti~ Sens~ <NA>
##  4      2460 Type au~ "2 chauss~ "Sans o~ "Rev~ "Lib~ N'appar~ Doub~ E60
##  5      3074 Type au~ "1 chauss~ "1 voie~ "Rev~ "Lib~ N'appar~ Sens~ <NA>
##  6      3198 Type au~ "1 chauss~ "1 voie~ "Rev~ "A p~ Apparti~ Sens~ E712
##  7      3200 Type au~ "2 chauss~ "Sans o~ "Rev~ "Lib~ Apparti~ Doub~ <NA>
##  8      3201 Type au~ "2 chauss~ "Sans o~ "Rev~ "Lib~ Apparti~ Doub~ <NA>
##  9      3202 Type au~ "2 chauss~ "Sans o~ "Rev~ "Lib~ Apparti~ Doub~ <NA>
## 10      3363 Type au~ "2 chauss~ "Sans o~ "Rev~ "A p~ Apparti~ Doub~ E713
## # ... with 4,496 more rows, and 4 more variables: NUM_ROUTE <fct>,
## #   CLASS_ADM <fct>, LONGUEUR <dbl>, geometry <LINESTRING [m]>
```

Reproject and crop to match France extent

```
highways <- highways %>%
  st_transform(crs = st_crs(forest)) %>%
  st_crop(st_bbox(df.france))
as_tibble(highways)
```

```
## # A tibble: 4,494 x 13
##    ID_RTE500 VOCATION NB_CHAUSSE NB_VOIES ETAT  ACCES RES_VERT SENS  RES_EUROPE
##        <int> <fct>    <fct>      <fct>    <fct> <fct> <fct>    <fct> <fct>
##  1        32 Type au~ "2 chauss~ "Sans o~ "Rev~ "A p~ Apparti~ Doub~ <NA>
##  2        33 Type au~ "1 chauss~ "2 voie~ "Rev~ "A p~ Apparti~ Doub~ <NA>
##  3      1041 Type au~ "1 chauss~ "1 voie~ "Rev~ "Lib~ Apparti~ Sens~ <NA>
##  4      2460 Type au~ "2 chauss~ "Sans o~ "Rev~ "Lib~ N'appar~ Doub~ E60
##  5      3074 Type au~ "1 chauss~ "1 voie~ "Rev~ "Lib~ N'appar~ Sens~ <NA>
##  6      3198 Type au~ "1 chauss~ "1 voie~ "Rev~ "A p~ Apparti~ Sens~ E712
##  7      3200 Type au~ "2 chauss~ "Sans o~ "Rev~ "Lib~ Apparti~ Doub~ <NA>
##  8      3201 Type au~ "2 chauss~ "Sans o~ "Rev~ "Lib~ Apparti~ Doub~ <NA>
##  9      3202 Type au~ "2 chauss~ "Sans o~ "Rev~ "Lib~ Apparti~ Doub~ <NA>
## 10      3363 Type au~ "2 chauss~ "Sans o~ "Rev~ "A p~ Apparti~ Doub~ E713
## # ... with 4,484 more rows, and 4 more variables: NUM_ROUTE <fct>,
## #   CLASS_ADM <fct>, LONGUEUR <dbl>, geometry <LINESTRING [m]>
```

Get centroids of each monitoring sites

```
centroids <- covariates %>%
  st_centroid()
as_tibble(centroids)
```

```
## # A tibble: 138 x 11
##     humdens forestcover    area  areakm2 bigsites bearpresence idsite
##     [1/m^2]       <dbl>   [m^2]   [km^2]    <dbl> <fct>         <dbl>
##  1 15.73746       0.529  796189~  79.618~       NA Absence      640101
##  2 16.38080       0.492 1221552~ 122.155~       NA Absence      640102
##  3 10.01365       0.416  640126~  64.012~       NA Occasionnel~ 640203
##  4 51.83858       0.580  362085~  36.208~       NA Occasionnel~ 640204
##  5 30.42510       0.623  679044~  67.904~       NA Absence      640202
##  6 68.13687       0.385  584558~  58.455~       NA Absence      640201
##  7 58.79124       0.460  303106~  30.310~       NA Absence      640301
##  8 70.63129       0.760  258667~  25.866~       NA Absence      640302
##  9 39.78087       0.433  723211~  72.321~       NA Occasionnel~ 640401
## 10 12.74387       0.312  677973~  67.797~       NA Reguliere    640402
## # ... with 128 more rows, and 4 more variables: areaforest [m^2], humpop <dbl>,
## #   geometry.x <POINT [m]>, geometry.y <GEOMETRY [m]>
```

Then distance from centroids to highways

```
dtohighways <- highways %>%
  st_distance(centroids, by_element = F)
head(dtohighways)
```

```
## Units: [m]
## [1] 490490.3 515331.5 718550.3 898734.7 919059.7 679572.2
```

Convert distance to highways into numeric values and keep only minimal distance to highways

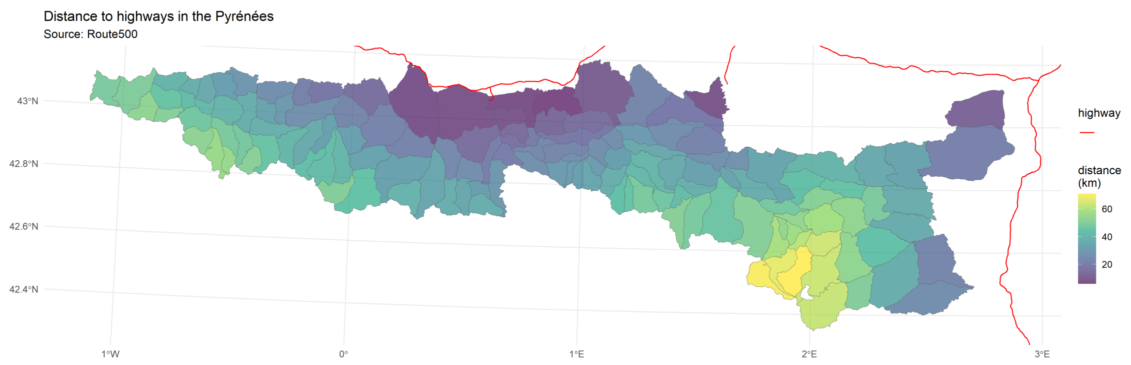
```
dtohighwaysnum <- matrix(as.numeric(dtohighways),
                         nrow = nrow(dtohighways),
                         ncol = ncol(dtohighways))
dtohighwaysnum <- apply(dtohighwaysnum,2,min)/1000
head(dtohighwaysnum)
```

```
## [1] 48.67387 49.63361 53.25704 50.65845 45.78141 39.89380
```

Map the distance to highways

```
covariates %>%
  ggplot() +
  geom_sf(lwd = 0.1, aes(fill = dtohighwaysnum)) +
  scale_fill_viridis_c(name = 'distance\n(km)',alpha = 0.7) +
  geom_sf(data = highways, aes(color = 'red'), show.legend = "line") +
  scale_color_manual(values = "red", labels = "", name = "highway") +
  coord_sf(xlim = st_bbox(covariates)[c(1,3)],
           ylim = st_bbox(covariates)[c(2,4)]) + # what if you turn this off?
  labs(title = 'Distance to highways in the Pyrénées',
       subtitle = 'Source: Route500')
```

Map the distance to highways



Wrap up

Geometric calculations

Geometric operations on vector layers can conceptually be divided into **three groups** according to their output:

- **Numeric** values: Functions that summarize geometrical properties of:
    - A **single layer** (e.g. area, length)
    - A **pair of layers** (e.g. distance)
- **Logical** values: Functions that evaluate whether a certain condition holds true, regarding:
    - A **single layer** (e.g. geometry is valid)
    - A **pair of layers** (e.g. feature A intersects feature B)
- **Spatial** layers: Functions that create a new layer based on:
    - A **single layer** (e.g. centroids)
    - A **pair of layers** (e.g. intersection area)

Numeric

- Several functions to calculate **numeric geometric properties** of vector layers:
  - `st_length`
  - `st_area`
  - `st_distance`
  - `st_bbox`
  - …

Logical

- Given two layers, `x` and `y`, the following **logical geometric functions** check whether each feature in `x` maintains the specified **relation** with each feature in `y`:
  - `st_intersects`
  - `st_disjoint`
  - `st_touches`
  - `st_crosses`
  - `st_within`
  - `st_contains`
  - `st_overlaps`
  - `st_covers`
  - `st_equals`
  - …

Spatial

- Common **geometry-generating** functions applicable to **individual** geometries:
  - `st_centroid`
  - `st_buffer`
  - `st_union`
  - `st_sample`
  - `st_convex_hull`
  - `st_voronoi`
  - …

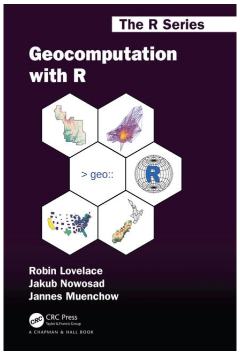
All `sf` methods

```
##  [1] $<-                   [                   [[<-
##  [4] aggregate             anti_join           arrange
##  [7] as.data.frame         cbind               coerce
## [10] dbDataType            dbWriteTable        distinct
## [13] dplyr_reconstruct     filter              full_join
## [16] gather                group_by            group_split
## [19] identify              initialize          inner_join
## [22] left_join             mapView             merge
## [25] mutate                nest                plot
## [28] print                 rbind               rename
## [31] right_join            sample_frac         sample_n
## [34] select                semi_join           separate
## [37] separate_rows         show                slice
## [40] slotsFromS3           spread              st_agr
## [43] st_agr<-              st_area             st_as_s2
## [46] st_as_sf              st_bbox             st_boundary
## [49] st_buffer             st_cast             st_centroid
## [52] st_collection_extract st_convex_hull      st_coordinates
## [55] st_crop               st_crs              st_crs<-
## [58] st_difference         st_filter           st_geometry
## [61] st_geometry<-         st_interpolate_aw   st_intersection
## [64] st_intersects         st_is               st_is_valid
## [67] st_join               st_line_merge       st_m_range
## [70] st_make_valid         st_nearest_points   st_node
## [73] st_normalize          st_point_on_surface st_polygonize
## [76] st_precision          st_reverse          st_sample
## [79] st_segmentize         st_set_precision    st_shift_longitude
## [82] st_simplify           st_snap             st_sym_difference
## [85] st_transform          st_triangulate      st_union
```

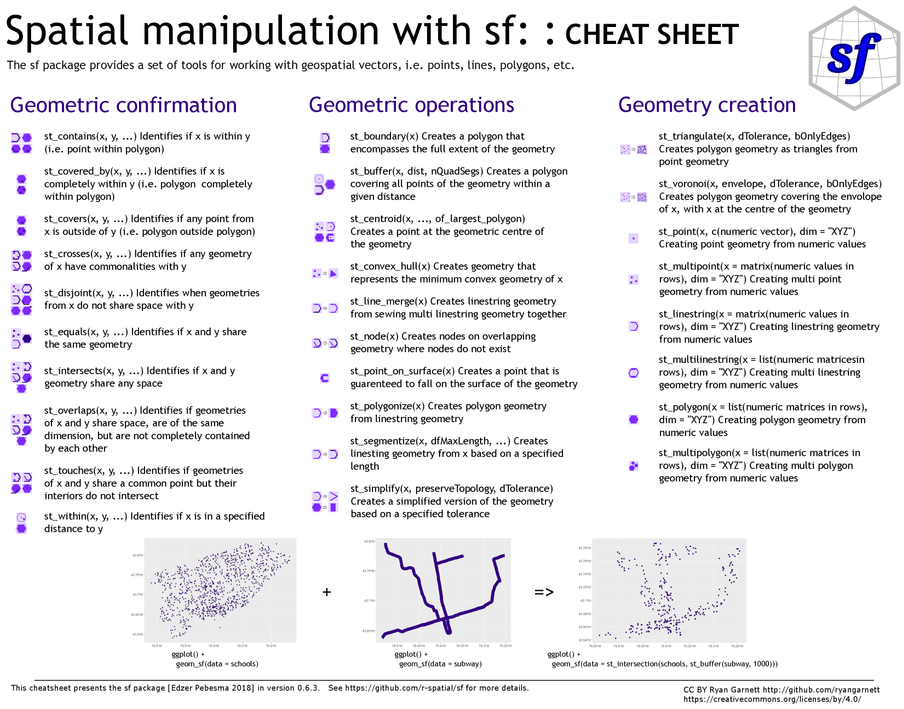
To go further

To dive even deeper into sf

- Detailed sf package vignettes
- Blog posts: here, here, here, here and there (in French)
- wiki page describing sp-sf migration
- Awesome online book Geocomputation with R by Lovelace, Nowosad and Muenchow



The RStudio Cheat Sheets



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 linesting 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 envolope 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

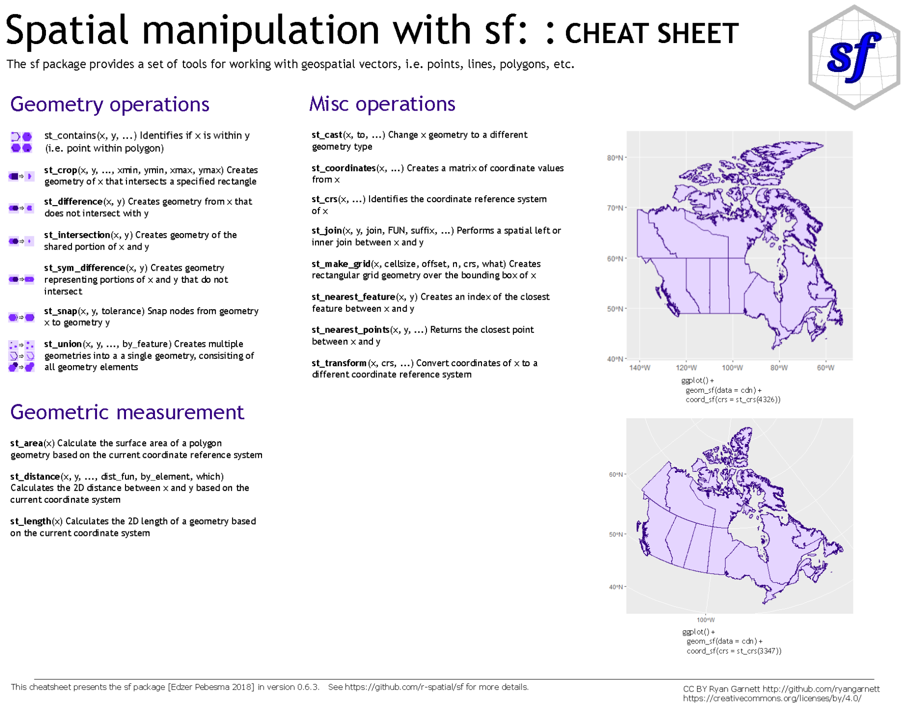
```
ggplot() +
  geom_sf(data = schools)
```

```
ggplot() +
  geom_sf(data = subway)
```

```
ggplot() +
  geom_sf(data = st_intersection(schools, st_buffer(subway, 1000)))
```

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

The RStudio Cheat Sheets



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.

Geometry operations

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

st_crop(x, y, ..., xmin, ymin, xmax, ymax) Creates geometry of x that intersects a specified rectangle

st_difference(x, y) Creates geometry from x that does not intersect with y

st_intersection(x, y) Creates geometry of the shared portion of x and y

st_sym_difference(x, y) Creates geometry representing portions of x and y that do not intersect

st_snap(x, y, tolerance) Snap nodes from geometry x to geometry y

st_union(x, y, ..., by_feature) Creates multiple geometries into a a single geometry, consisiting of all geometry elements

Geometric measurement

st_area(x) Calculate the surface area of a polygon geometry based on the current coordinate reference system

st_distance(x, y, ..., dist_fun, by_element, which) Calculates the 2D distance between x and y based on the current coordinate system

st_length(x) Calculates the 2D length of a geometry based on the current coordinate system

Misc operations

st_cast(x, to, ...) Change x geometry to a different geometry type

st_coordinates(x, ...) Creates a matrix of coordinate values from x

st_crs(x, ...) Identifies the coordinate reference system of x

st_join(x, y, join, FUN, suffix, ...) Performs a spatial left or inner join between x and y

st_make_grid(x, cellsize, offset, n, crs, what) Creates rectangular grid geometry over the bounding box of x

st_nearest_feature(x, y) Creates an index of the closest feature between x and y

st_nearest_points(x, y, ...) Returns the closest point between x and y

st_transform(x, crs, ...) Convert coordinates of x to a different coordinate reference system

```
ggplot() +
  geom_sf(data = cdn) +
  coord_sf(crs = st_crs(4326))
```

```
ggplot() +
  geom_sf(data = cdn) +
  coord_sf(crs = st_crs(3347))
```

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

Thanks!

I created these slides with xaringan and RMarkdown using the rutgers css that I slightly modified.

Credits: I used material from @StrimasMackey, @jafflerbach, @StatnMap, @SharpSightLabs and @edzerpebesma

| email: | olivier.gimenez@cefe.cnrs.fr | | Website: https://oliviergimenez.github.io/ | | Twitter: @oaggimenez | | Github @oliviergimenez