GIS in R: Tutorial 1

Marie Auger-Méthé

1 Good R practices

If you want to be able to reproduce your analyses, it is important to keep your track of your files and preferably do version control. Keeping track of your steps is both important to make your research reproducible and easily shareable, and just because it is nice not to have to start from scratch every time you want to do an analysis.

1.1 Keep all our files in one place with R Studio projects

This is particularly important for spatial analyses because spatial data files can be challenging to handle and keeping all files in the same folder facilitate the process. Keep originals somewhere else so only changes stuff.

1.2 Make reproducible examples with R scripts

Share complete analysis by sending an R script. Put comments!

For those that are taking the class for credit, I want you to send me by e-mail for each tutorial a `R` script that has all of the code from the tutorial, including the code found in the document.

1.3 Keep track of your changes with version control

2 `Spatial` class

Vector data (i.e., Points, Lines, and Polygons) are handle in `R` using the foundation class `Spatial` associated with the package `sp`. To get a feel for how `Spatial` objects differ from other types of `R` objects, we will import a .csv table with the locations of bioprobes (this is a simplified version of the data available on the Ocean Tracking Network (OTN) website).

```
# Read the .csv file with OTN Sable Island bioprobe data
bioPr <- read.csv("sableSealBioP.csv")
```

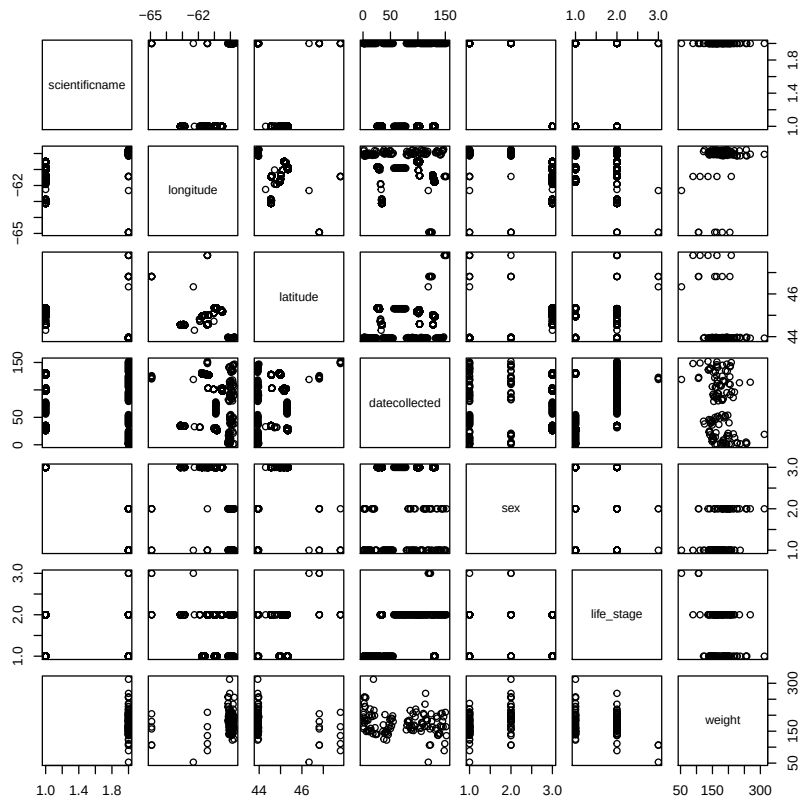
To get a sense for this object we will investigate the class, and some of the generic functions.

```
# What's the class?
class(bioPr)
```

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

# Can we summarise the information within it?
summary(bioPr)

##             scientificname   longitude         latitude
##  Gadus morhua      :609    Min.   :-64.93   Min.   :43.93
##  Halichoerus grypus:118    1st Qu.:-61.74   1st Qu.:44.56
##                            Median :-60.90   Median :45.01
##                            Mean   :-61.17   Mean   :44.89
##                            3rd Qu.:-60.51   3rd Qu.:45.18
##                            Max.   :-59.74   Max.   :47.80
##
##              datecollected sex         life_stage      weight
##  2011-06-09T00:00:00: 62   F: 81            :255   Min.   : 53.0
##  2012-11-19T00:00:00: 57   M: 37   ADULT    :469   1st Qu.:152.0
##  2012-11-20T00:00:00: 57   U:609   SUB-ADULT:  3   Median :169.0
##  2013-05-16T00:00:00: 49                           Mean   :174.7
##  2014-05-13T00:00:00: 45                           3rd Qu.:196.0
##  2013-11-16T00:00:00: 42                           Max.   :312.5
##  (Other)            :415                           NA's   :610
```

One of the generic function is the `plot` functions.

```
plot(bioPr)
```



Since we know that the data includes the locations of bioprobes, it would be much better to be able to plot it spatially. To do this we will trasform this `data.frame` into a `Spatial` object. In particular, since our data is point data, we will transform the data.frame into a `SpatialPointsDataFrame`. The simplest way to do so is by assigning values to `coords` slot, which is the component that contains the coordinate values.

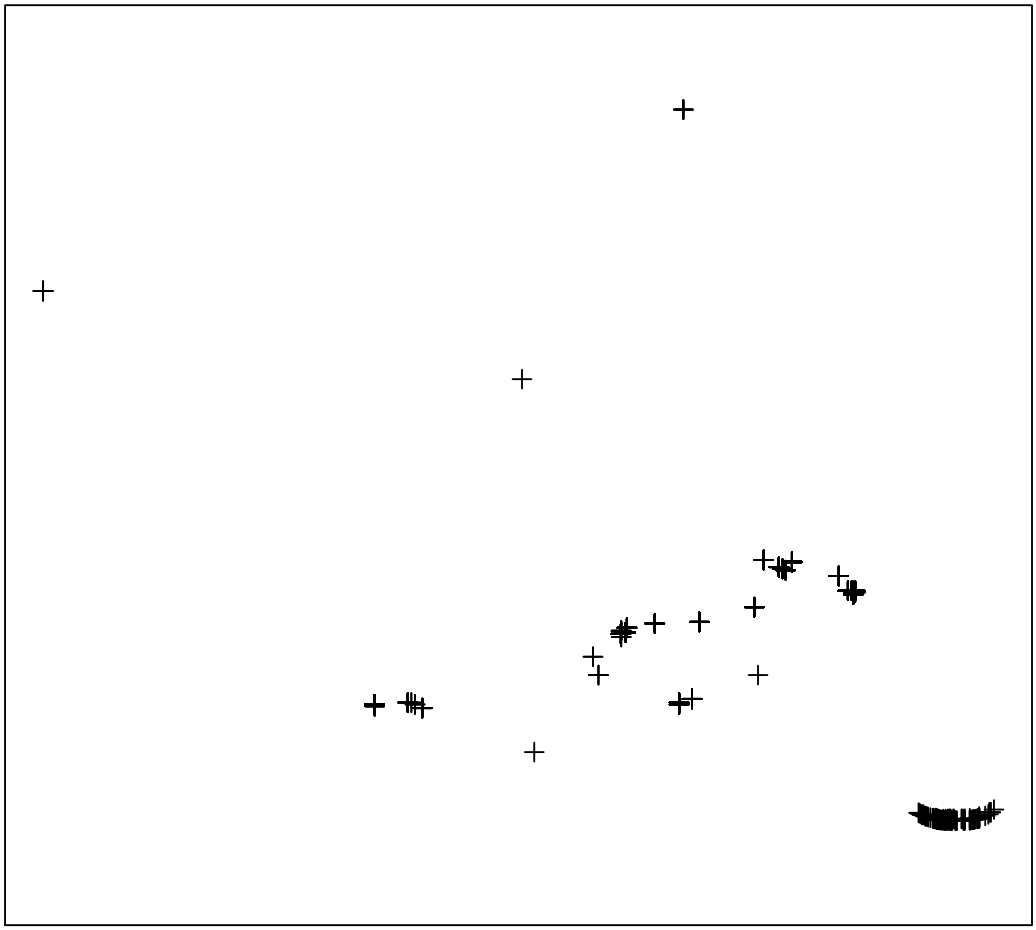
```
# Load package
library(sp)
# Create new data.frame with the exact same value as bioPr
bioPrSP <- bioPr
# Transform this new object into a SpatialPointsDataFrame using the columns that contain the latitude a
colnames(bioPrSP)

## [1] "scientificname" "longitude"      "latitude"       "datecollected"
## [5] "sex"            "life_stage"     "weight"

coordinates(bioPrSP) <- ~longitude+latitude
```

Now let's compare the results from the same generic functions.

```
class(bioPrSP)

## [1] "SpatialPointsDataFrame"
## attr(,"package")
## [1] "sp"

summary(bioPrSP)

## Object of class SpatialPointsDataFrame
## Coordinates:
##                 min       max
## longitude -64.92774 -59.74377
## latitude   43.92560  47.80300
## Is projected: NA
## proj4string : [NA]
## Number of points: 727
## Data attributes:
##             scientificname             datecollected sex
##  Gadus morhua      :609    2011-06-09T00:00:00: 62   F: 81
##  Halichoerus grypus:118    2012-11-19T00:00:00: 57   M: 37
##                            2012-11-20T00:00:00: 57   U:609
##                            2013-05-16T00:00:00: 49
##                            2014-05-13T00:00:00: 45
##                            2013-11-16T00:00:00: 42
##                            (Other)            :415
##      life_stage      weight
##           :255   Min.   : 53.0
##  ADULT    :469   1st Qu.:152.0
##  SUB-ADULT:  3   Median :169.0
##                  Mean   :174.7
##                  3rd Qu.:196.0
##                  Max.   :312.5
##                  NA's   :610
```

But more interestingly, let's look at the `plot` function.

```
plot(bioPrSP)
# With an added box around the figure
box()
```



Here instead of plotting the data values from all the columns the plot shows the location of the row in space.

Exercise 1

Create a spatial object with the data found in nbCod.csv, which is a slightly altered version from the

Animal file available on the Shippagan, NB: Code tagging project from OTN, see project website.