Exercise 5: Mapping with R

Ben Davies

2022-05-11

Contents

Chapter 1

Making maps with R

With the help of many user generated packages, R can operate as a full-fledged geographic information system. One place where R truly shines over many desktop GIS platforms is the control a user has over the display of spatial information. While R's base graphics capabilities are useful for quickly visualizing data, packages like `ggplot2` and others offer cleaner default plotting and specialize in manipulating all aspects of a plot's appearance. Here, we will explore some of these capabilities for mapped data.

1.1 Adding packages

While `sf` and `terra` are useful for handling spatial data, they aren't built for visualization. To help us out, we're going to add some packages from the `ggplot2` family:

```
require("sf")
```

```
## Loading required package: sf
```

```
## Linking to GEOS 3.9.1, GDAL 3.3.2, PROJ 7.2.1; sf_use_s2() is TRUE
```

```
require("terra")
```

```
## Loading required package: terra
```

```
## terra 1.5.21
```

```
require("ggplot2") # main graphics package
```

```
## Loading required package: ggplot2
```

```
##
## Attaching package: 'ggplot2'
```

```
## The following object is masked from 'package:terra':
##
##     arrow
```

```
require("ggspatial") # map specific ggplot functionality
```

```
## Loading required package: ggspatial
```

Just a little background: the `ggplot2` package was developed in 2005 by Hadley Wickham as a software approach to visualization scheme called "The Grammar of Graphics" published by Leland Wilkinson. The major distinction between `base` graphics and `ggplot2` is the ability to control the elements of a plot as a series of layers that can be added, shuffled, or removed.

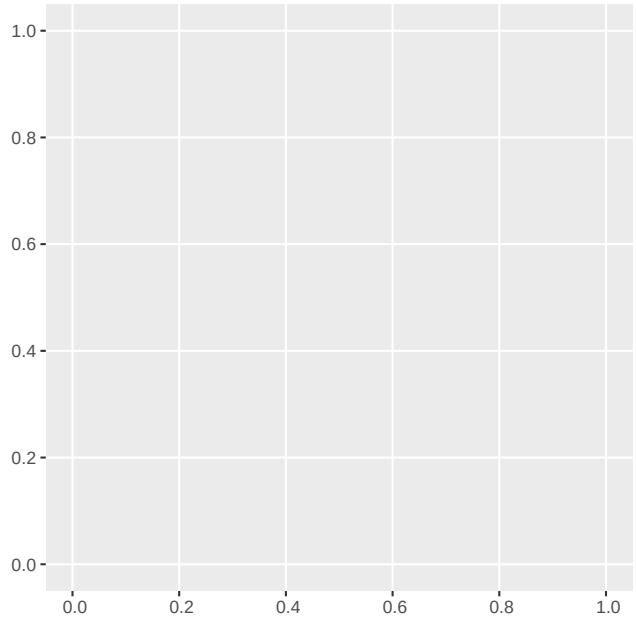
Chapter 2

Getting going with ggplot2 and geom_sf

Over the years, the developers of `ggplot2` have recognized widespread interest in using the package to make maps, and have updated the software to include handling and plotting of spatial data At the same time, efforts like the promotion of the Simple Features data structure are a recognition that data interoperability is a priority for many data users.

These two processes meet in the middle with the `geom_sf` object. This is a way of translating Simple Features data (like that from the `sf` package) into a format that can be manipulated in the `ggplot2` environment. The general scheme looks something like this:

```
#Plot a blank geom_sf object
ggplot() + geom_sf()
```



This begins with an empty `ggplot` object. To this we add (with the `+` operator) a `geom_sf` object. In order to plot data, though, this will need some data.

2.1 Add some data

The dataset we'll start with are white-bearded wildebeest tracks at three study areas in Kenya, downloaded from Movebank.

```
#Load data
wbTracks<-st_read("wildebeestTracks.shp")
```

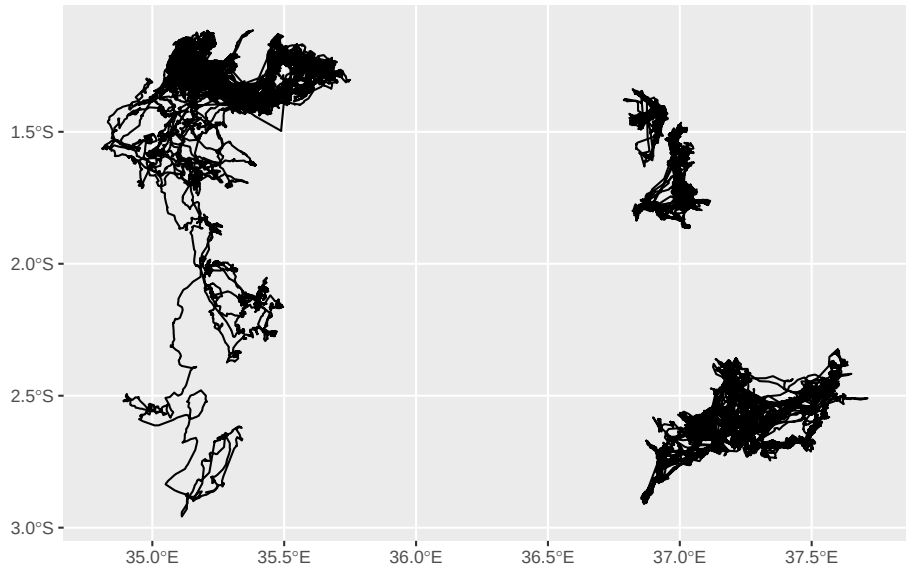
```
## Reading layer `wildebeestTracks' from data source
##   `C:\Users\bdav_\Dropbox\Teaching\Spatial R Short Course\Bookdown\Exercise5\Exercis
##   using driver `ESRI Shapefile'
## replacing null geometries with empty geometries
## Simple feature collection with 46 features and 1 field (with 10 geometries empty)
## Geometry type: LINESTRING
## Dimension:     XY
## Bounding box:  xmin: 34.80902 ymin: -2.957671 xmax: 37.71177 ymax: -1.114714
## Geodetic CRS:  WGS 84
```

This is linestring data, so we should be expecting lines in our plot.

2.2 Plotting a geom_sf object

We can add it to our blank object this way:

```
#Plot sf linestring data
ggplot() +
  geom_sf(data = wbTracks)
```



Looks like tracks to me! Note that we need to put `data=` before the dataset in our `geom_sf` call. This tells ggplot2 that what we're passing it is not meant to be something other than data (it's not so good at guessing).

2.3 Changing appearances

First, let's have a look at this data.

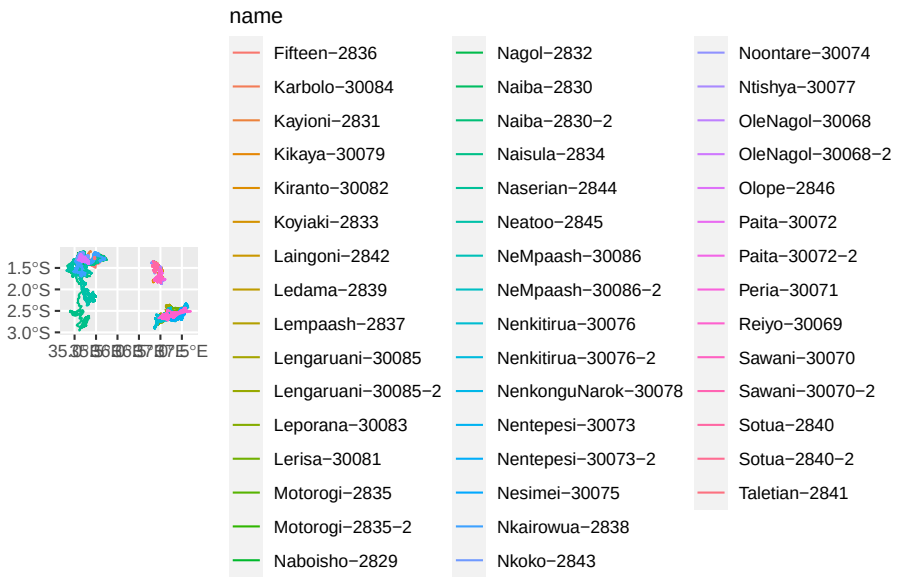
```
#Look at first six lines of data
head(wbTracks)
```

```
## Simple feature collection with 6 features and 1 field
## Geometry type: LINESTRING
## Dimension:     XY
```

```
## Bounding box:  xmin: 34.83135 ymin: -2.957671 xmax: 35.75016 ymax: -1.114714
## Geodetic CRS:  WGS 84
##            name                       geometry
## 1 Naboisho-2829 LINESTRING (35.30873 -1.343...
## 2    Naiba-2830 LINESTRING (35.37816 -1.381...
## 3  Kayioni-2831 LINESTRING (35.31887 -1.333...
## 4    Nagol-2832 LINESTRING (35.4046 -1.3339...
## 5  Koyiaki-2833 LINESTRING (35.32003 -1.390...
## 6  Naisula-2834 LINESTRING (35.67744 -1.299...
```

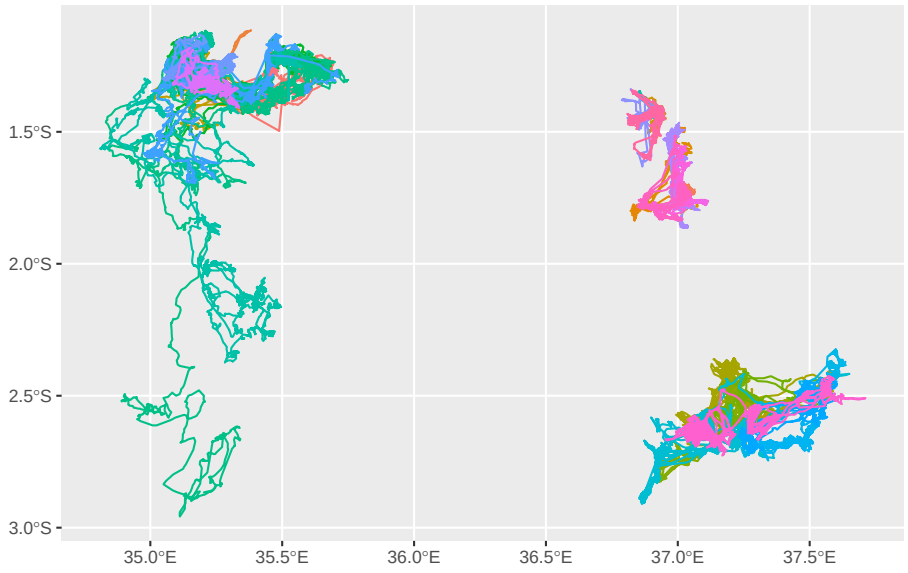
The name of the animal is the only piece of information here, so we'll use that to change the look of the plot.

```
#Plot with lines colored by animal
ggplot() +
  geom_sf(data = wbTracks, aes(color=name))
```



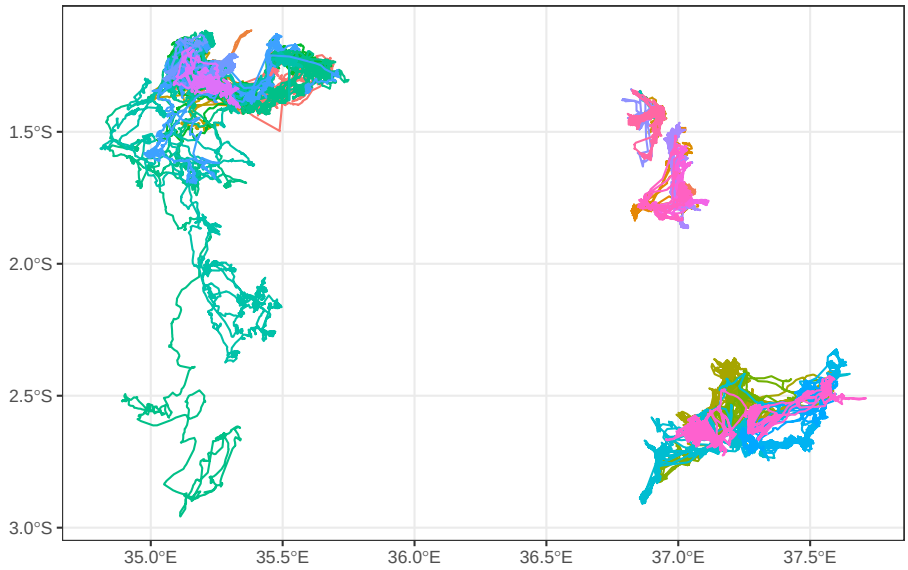
Eek! The legend takes up almost the whole image! Let's get rid of that for now. To do this, we add another element with `+`, and then use `theme()` to modify the legend settings:

```
#Plot without legend
ggplot() +
  geom_sf(data = wbTracks, aes(color=name)) +
  theme(legend.position=0)
```



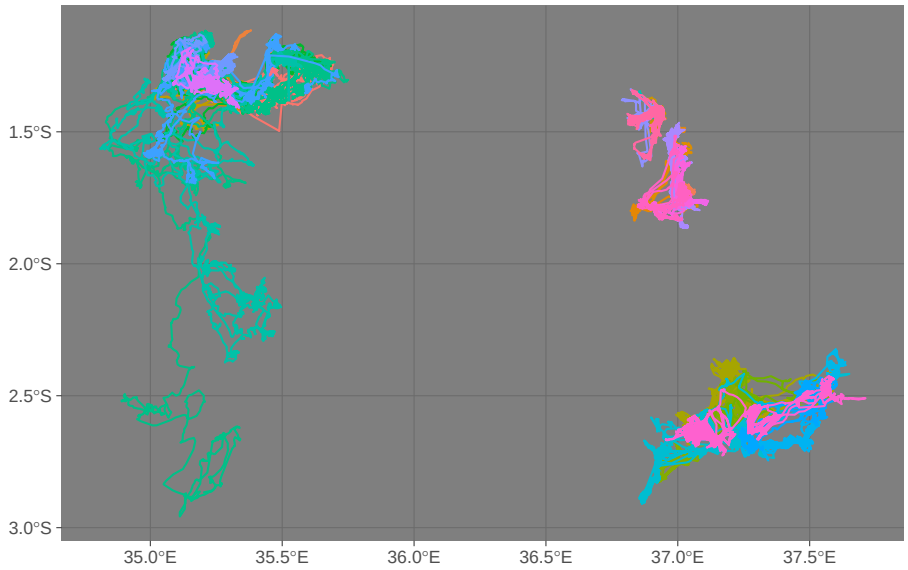
That's better. Each ggplot object has a *theme* that controls aspects of the plot that are unrelated to the data. What we did there was pass an argument about the position, which could be a set of coordinates The default is `theme_grey`, but there are a number of preset themes that we can use. For example…

```
#Plot with theme_bw() theme
ggplot() +
  geom_sf(data = wbTracks, aes(color=name)) +
  theme_bw() +
  theme(legend.position=0)
```



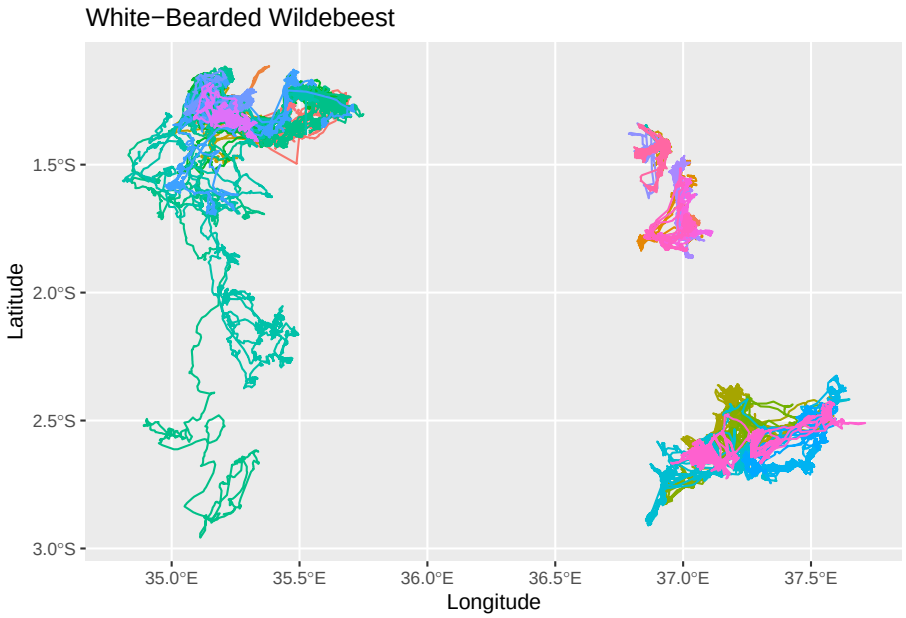
...or this...

```
#Plot with theme_dark() theme
ggplot() +
  geom_sf(data = wbTracks, aes(color=name)) +
  theme_dark() +
  theme(legend.position=0)
```



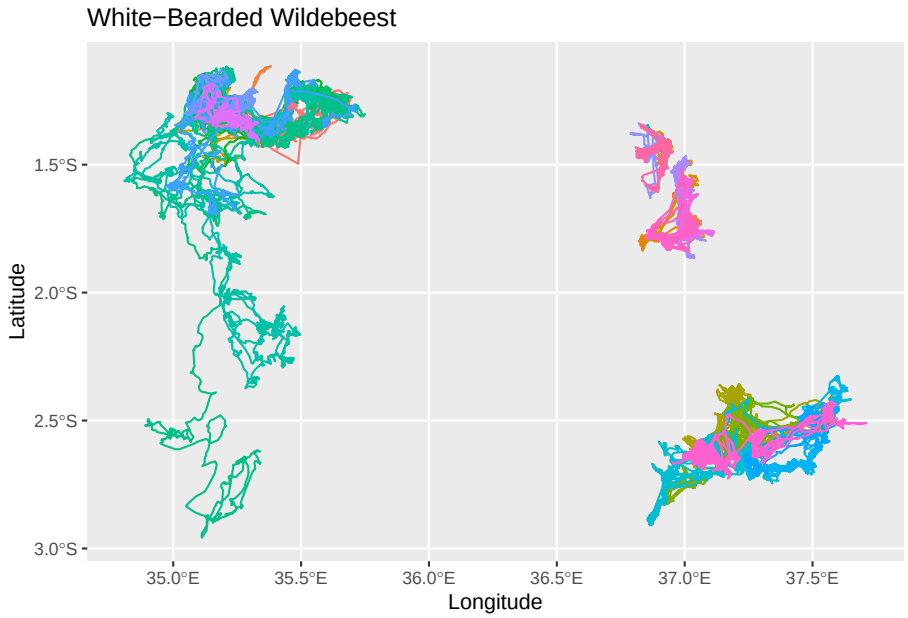
For now we'll stick with the grey theme. There are other elements we might add to this plot. For example, there are no axis labels or title here. There are specific calls we can make for these elements:

```
#Plot with title and axis labels
ggplot() +
  geom_sf(data = wbTracks, aes(color=name)) +
  theme_grey() +
  theme(legend.position=0) +
  xlab("Longitude") +
  ylab("Latitude") +
  ggtitle("White-Bearded Wildebeest")
```



Alternatively, these can be done with the `labs()` object:

```
#Plot with title and axis labels
ggplot() +
  geom_sf(data = wbTracks, aes(color=name)) +
  theme_grey() +
  theme(legend.position=0) +
  labs(x="Longitude",y="Latitude",title="White-Bearded Wildebeest")
```



This is just a brief introduction to plotting controls with `ggplot2`. You could take an entire course on data visualization with this package and still only scratch the surface. For more about this, the book ggplot2: Elegant Graphics for Data Analysis is free and offers a very thorough treatment.

2.4 Try it yourself!

In this section we plotted the tracks using the default color scheme, but there's a lot of different options for this. * Try adding another element to this plot using the `scale_fill_brewer()` function. This can take a `palette=` argument, which you can plug in the name of a ColorBrewer palette. Try a few! The names can be found here.

Chapter 3

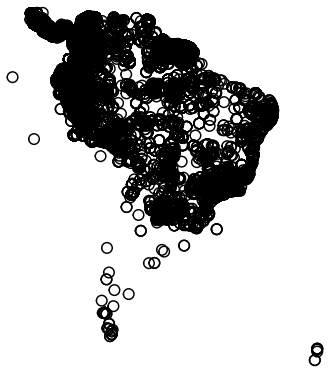
Mapping multiple datasets

OK, now we have seen how to make maps with a single dataset. There's some things to keep in mind when working with multiple datasets. This section will look at this process.

3.1 Add data

This first dataset is primate species observation data from the Global Biodiversity Information Facility (GBIF), covering all of South America. It is in CSV format, so we need to use `st_as_sf` to convert it to an `sf` object:

```
#Load data
primateObs<-read.csv("SAPrimateObservations.csv")
primates<-st_as_sf(primateObs,coords=c("decimalLongitude","decimalLatitude"))
st_crs(primates)<-"EPSG:4326"
plot(st_geometry(primates))
```



This is a lot of data, so we'll just cut it down to observations in Brazil.

```
brazilPrimates<-subset(primates,countryCode=="BR")
```

The second set of data we're going to use is a shapefile of Brazil's states.

```
states<-st_read("brazilStates.shp")
```

```
## Reading layer `brazilStates' from data source
##   `C:\Users\bdav_\Dropbox\Teaching\Spatial R Short Course\Bookdown\Exercise5\Exercis
##   using driver `ESRI Shapefile'
## Simple feature collection with 27 features and 5 fields
## Geometry type: MULTIPOLYGON
## Dimension:     XY
## Bounding box:  xmin: -73.99047 ymin: -33.75077 xmax: -28.84917 ymax: 5.271131
## Geodetic CRS:  WGS 84
```

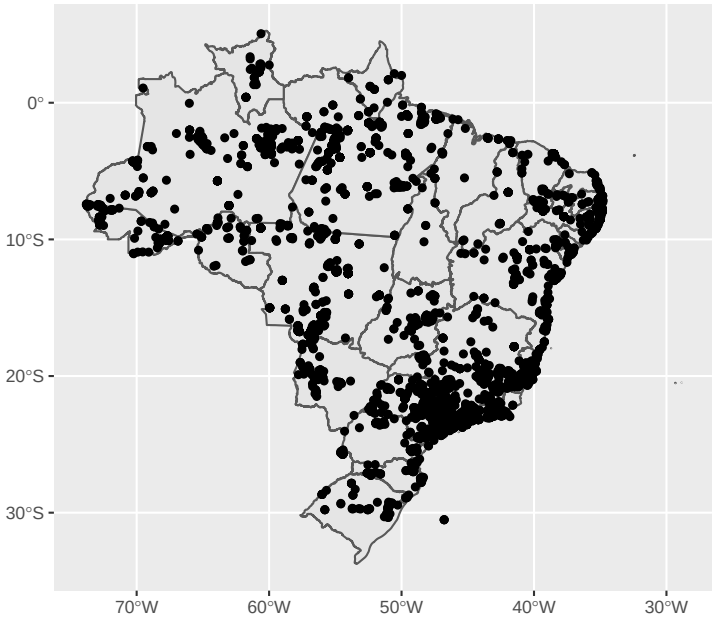
3.2 Combining datasets in a plot

OK, let's get plotting. It works just like our last plot, but we add two `geom_sf` objects rather than one.

```
#Plot two sf objects
primatePlot<-ggplot() +
  geom_sf(data = brazilPrimates) +
  geom_sf(data= states)
print(primatePlot)
```

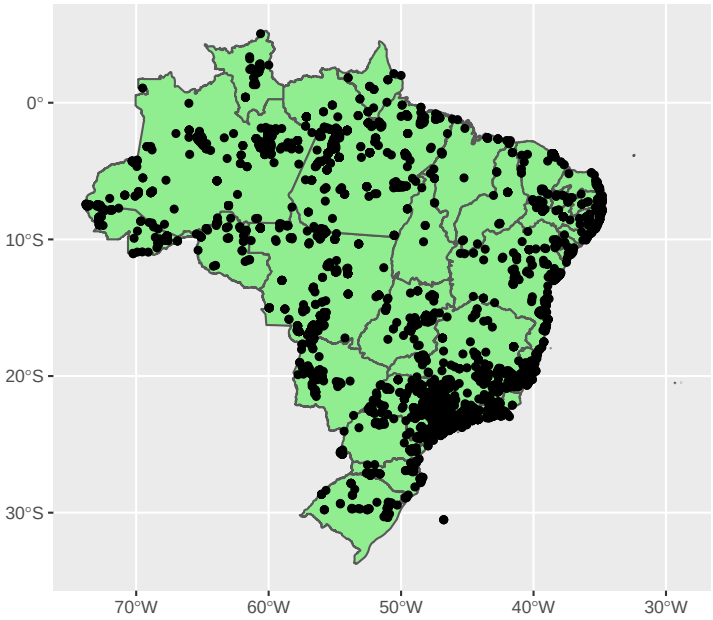
Oops! Our points are all sitting under the map of Brazil! This is the first thing we want to keep in mind: the ggplot object plots layers in order. If we want the points on top, they get added last.

```
#Plot in correct order
primatePlot<-ggplot() +
  geom_sf(data= states) +
  geom_sf(data = brazilPrimates)
print(primatePlot)
```



That's better. Now, let's give the states a different color.

```
#Plot with title and axis labels
primatePlot<-ggplot() +
  geom_sf(data= states, fill="light green") +
  scale_color_brewer(palette="Paired") +
  geom_sf(data = brazilPrimates)
print(primatePlot)
```



OK, now let's change the look of the primate data. Maybe we want to show the diversity of species in the dataset. Let's look at the data and find a good attribute for that:

```
head(brazilPrimates)
```

```
## Simple feature collection with 6 features and 48 fields
## Geometry type: POINT
## Dimension:     XY
## Bounding box:  xmin: -48.10679 ymin: -22.98356 xmax: -40.06505 ymax: -15.56004
## Geodetic CRS:  WGS 84
##        gbifID                              datasetKey
## 22 3760406607 50c9509d-22c7-4a22-a47d-8c48425ef4a7
## 24 3760402428 50c9509d-22c7-4a22-a47d-8c48425ef4a7
## 30 3760386387 50c9509d-22c7-4a22-a47d-8c48425ef4a7
## 38 3760360787 50c9509d-22c7-4a22-a47d-8c48425ef4a7
## 40 3760350394 50c9509d-22c7-4a22-a47d-8c48425ef4a7
## 43 3760344838 50c9509d-22c7-4a22-a47d-8c48425ef4a7
##                                          occurrenceID  kingdom   phylum
## 22 https://www.inaturalist.org/observations/112156242 Animalia Chordata
## 24 https://www.inaturalist.org/observations/110912409 Animalia Chordata
## 30 https://www.inaturalist.org/observations/111860218 Animalia Chordata
## 38 https://www.inaturalist.org/observations/111202030 Animalia Chordata
## 40 https://www.inaturalist.org/observations/111853748 Animalia Chordata
## 43 https://www.inaturalist.org/observations/112298162 Animalia Chordata
```

```
##       class    order         family      genus                species
## 22 Mammalia Primates Callitrichidae Callithrix   Callithrix geoffroyi
## 24 Mammalia Primates Callitrichidae Callithrix Callithrix penicillata
## 30 Mammalia Primates    Pitheciidae Callicebus  Callicebus personatus
## 38 Mammalia Primates        Cebidae    Sapajus       Sapajus nigritus
## 40 Mammalia Primates Callitrichidae Callithrix   Callithrix geoffroyi
## 43 Mammalia Primates Callitrichidae Callithrix Callithrix penicillata
##    infraspecificEpithet taxonRank
## 22                      SPECIES
## 24                      SPECIES
## 30                      SPECIES
## 38                      SPECIES
## 40                      SPECIES
## 43                      SPECIES
##                                          scientificName
## 22                     Callithrix geoffroyi (Humboldt, 1812)
## 24 Callithrix penicillata (\xc9.Geoffroy Saint-Hilaire, 1812)
## 30  Callicebus personatus (\xc9.Geoffroy Saint-Hilaire, 1812)
## 38                       Sapajus nigritus (Goldfuss, 1809)
## 40                     Callithrix geoffroyi (Humboldt, 1812)
## 43 Callithrix penicillata (\xc9.Geoffroy Saint-Hilaire, 1812)
##    verbatimScientificName verbatimScientificNameAuthorship countryCode locality
## 22   Callithrix geoffroyi                                            BR
## 24 Callithrix penicillata                                            BR
## 30  Callicebus personatus                                            BR
## 38       Sapajus nigritus                                            BR
## 40   Callithrix geoffroyi                                            BR
## 43 Callithrix penicillata                                            BR
##        stateProvince occurrenceStatus individualCount
## 22 Esp\xedrito Santo          PRESENT              NA
## 24  Distrito Federal          PRESENT              NA
## 30 Esp\xedrito Santo          PRESENT              NA
## 38    Rio de Janeiro          PRESENT              NA
## 40 Esp\xedrito Santo          PRESENT              NA
## 43  Distrito Federal          PRESENT              NA
##                        publishingOrgKey coordinateUncertaintyInMeters
## 22 28eb1a3f-1c15-4a95-931a-4af90ecb574d                           970
## 24 28eb1a3f-1c15-4a95-931a-4af90ecb574d                           967
## 30 28eb1a3f-1c15-4a95-931a-4af90ecb574d                         30562
## 38 28eb1a3f-1c15-4a95-931a-4af90ecb574d                         30237
## 40 28eb1a3f-1c15-4a95-931a-4af90ecb574d                           464
## 43 28eb1a3f-1c15-4a95-931a-4af90ecb574d                             4
##    coordinatePrecision elevation elevationAccuracy depth depthAccuracy
## 22                  NA        NA                NA    NA            NA
## 24                  NA        NA                NA    NA            NA
## 30                  NA        NA                NA    NA            NA
```

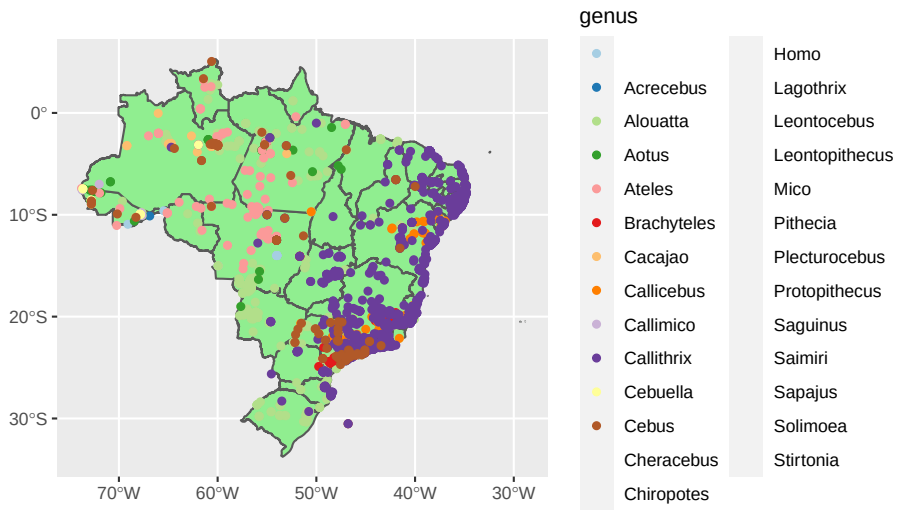
```
## 38                  NA        NA                NA    NA            NA
## 40                  NA        NA                NA    NA            NA
## 43                  NA        NA                NA    NA            NA
##          eventDate day month year taxonKey speciesKey     basisOfRecord
## 22  6/5/2016 13:47   5     6 2016  5219539    5219539 HUMAN_OBSERVATION
## 24  4/9/2022 11:19   9     4 2022  5219541    5219541 HUMAN_OBSERVATION
## 30 2/16/2015 13:49  16     2 2015  2436398    2436398 HUMAN_OBSERVATION
## 38   4/7/2019 0:00   7     4 2019  7519053    7519053 HUMAN_OBSERVATION
## 40 12/1/2013 12:40   1    12 2013  5219539    5219539 HUMAN_OBSERVATION
## 43  4/21/2022 9:33  21     4 2022  5219541    5219541 HUMAN_OBSERVATION
##    institutionCode collectionCode catalogNumber recordNumber       identifiedBy
## 22     iNaturalist   Observations     112156242           NA      Gabriel Bonfa
## 24     iNaturalist   Observations     110912409           NA Enrico A. R. Tosto
## 30     iNaturalist   Observations     111860218           NA      Gabriel Bonfa
## 38     iNaturalist   Observations     111202030           NA     Michal Sloviak
## 40     iNaturalist   Observations     111853748           NA      Gabriel Bonfa
## 43     iNaturalist   Observations     112298162           NA    Maria Clara Gil
##     dateIdentified      license      rightsHolder        recordedBy typeStatus
## 22 4/20/2022 15:08 CC_BY_NC_4_0     Gabriel Bonfa     Gabriel Bonfa
## 24  4/10/2022 1:04 CC_BY_NC_4_0  abelardomendesjr  abelardomendesjr
## 30  4/18/2022 0:02 CC_BY_NC_4_0     Gabriel Bonfa     Gabriel Bonfa
## 38 4/12/2022 15:13 CC_BY_NC_4_0 Wagner Fiorentino Wagner Fiorentino
## 40 4/17/2022 23:08 CC_BY_NC_4_0     Gabriel Bonfa     Gabriel Bonfa
## 43 4/21/2022 18:50 CC_BY_NC_4_0   Maria Clara Gil   Maria Clara Gil
##    establishmentMeans lastInterpreted  mediaType              issue
## 22                     5/2/2022 16:23 StillImage COORDINATE_ROUNDED
## 24                     5/2/2022 15:55 StillImage COORDINATE_ROUNDED
## 30                     5/2/2022 14:50 StillImage COORDINATE_ROUNDED
## 38                     5/2/2022 16:04 StillImage COORDINATE_ROUNDED
## 40                     5/2/2022 16:18 StillImage COORDINATE_ROUNDED
## 43                     5/2/2022 16:26 StillImage COORDINATE_ROUNDED
##                       geometry
## 22 POINT (-40.06505 -19.15138)
## 24 POINT (-47.86165 -15.73703)
## 30  POINT (-40.32546 -19.4921)
## 38 POINT (-43.20546 -22.98355)
## 40  POINT (-40.2999 -19.40386)
## 43 POINT (-48.10679 -15.56004)
```

OK, let's go with `genus`. To access this for plotting, we need to tell ggplot that this will be used as an aesthetic, so we call `aes`. Everything we ask for inside `aes` will be evaluated for each point, while everything outside that call gets applied the same way to all points.

```
#Plot with title and axis labels
primatePlot<-ggplot() +
  geom_sf(data= states, fill="light green") +
  scale_color_brewer(palette="Paired") +
  geom_sf(data = brazilPrimates,aes(color=genus))
print(primatePlot)
```

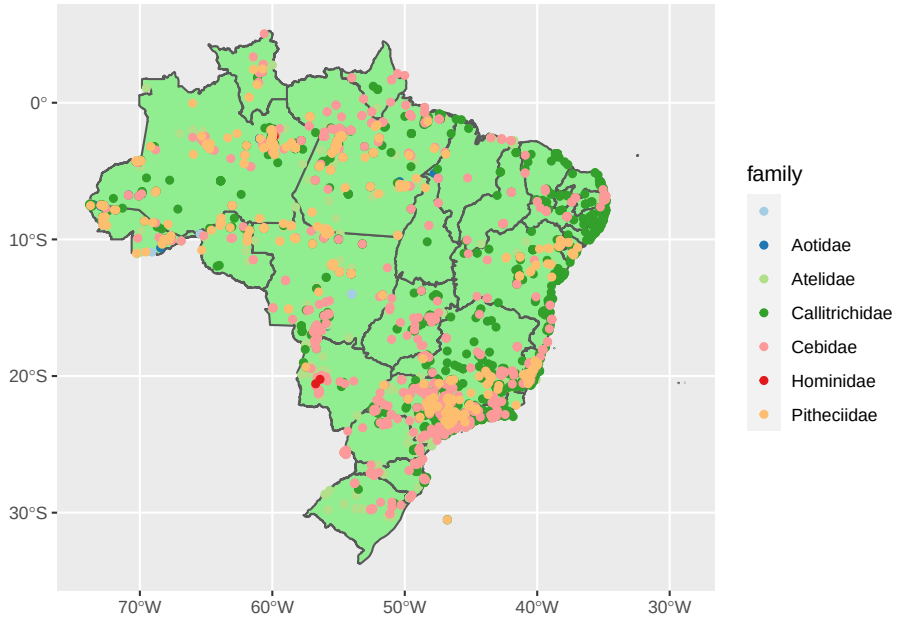
```
## Warning in RColorBrewer::brewer.pal(n, pal): n too large, allowed maximum for palett
## Returning the palette you asked for with that many colors
```

```
## Warning: Removed 2077 rows containing missing values (geom_sf).
```



Ack! The giant legend strikes again! There's a couple more problems here. First, there's a bunch of points that don't have a value for genus. And second we ran out of colors! We could find other ways to get more colors, but all of these problems could potentially be solved by plotting by `family` instead.

```
#Plot with title and axis labels
primatePlot<-ggplot() +
  geom_sf(data= states, fill="light green") +
  scale_color_brewer(palette="Paired") +
  geom_sf(data = brazilPrimates,aes(color=family))
print(primatePlot)
```



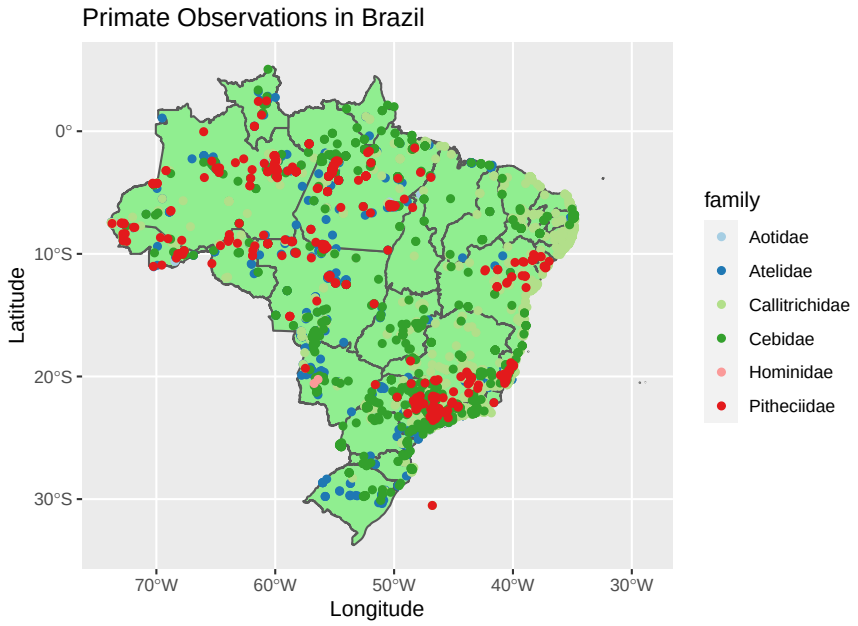
Much better. There's a couple of points here where family was recorded as " ", so those have no label. We can eliminate those quickly with `subset`:

```
brazilPrimates<-subset(brazilPrimates,family != "")
```

Also, someone apparently took the time to send in some observations of Hominidae! We'll leave that one in there. Ecologists are a fun bunch.

Let's wrap this section up by adding some axis labels and a title.

```
#Plot with title and axis labels
primatePlot<-ggplot() +
  geom_sf(data= states, fill="light green") +
  scale_color_brewer(palette="Paired") +
  geom_sf(data = brazilPrimates,aes(color=family)) +
  labs(x="Longitude",y="Latitude",title="Primate Observations in Brazil")
print(primatePlot)
```


Primate Observations in Brazil
0°
10°S
20°S
30°S
Latitude
70°W
60°W
50°W
40°W
30°W
Longitude
family
Aotidae
Atelidae
Callitrichidae
Cebidae
Hominidae
Pitheciidae

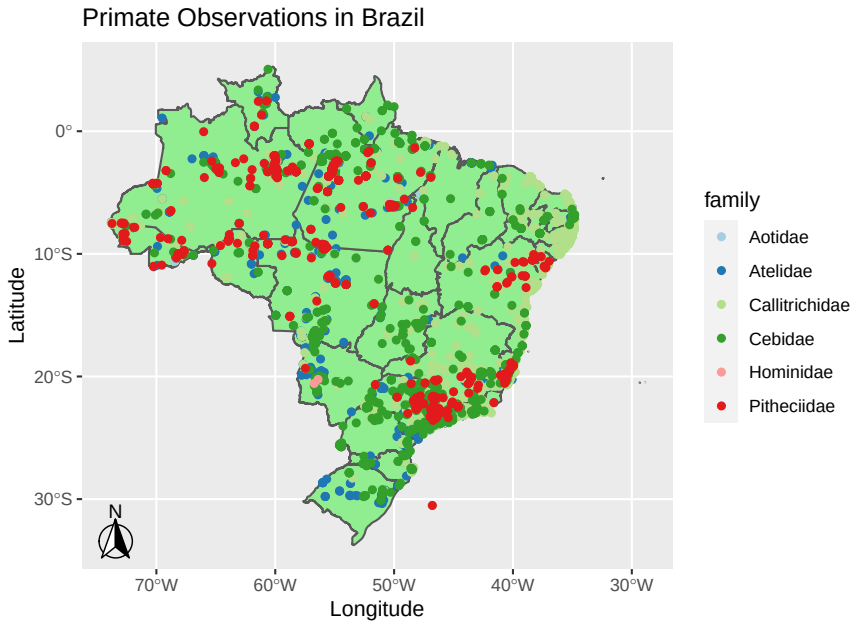
Chapter 4

Adding map elements

In addition to the plot itself, we want to make sure we're able to add other necessary plot elements. To get access to these, we'll use the `ggspatial` package.

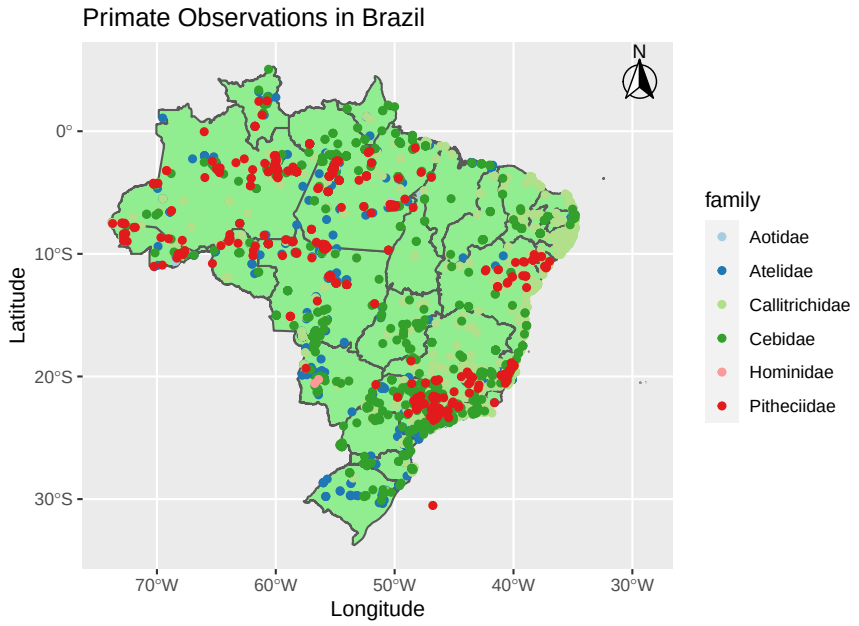
Let's start with a north arrow:

```
#Plot with title and axis labels
primatePlot<-ggplot() +
  geom_sf(data= states, fill="light green") +
  scale_color_brewer(palette="Paired") +
  geom_sf(data = brazilPrimates,aes(color=family)) +
  labs(x="Longitude",y="Latitude",title="Primate Observations in Brazil")  +
   annotation_north_arrow(location = "bl",  height = unit(1, "cm"), width = unit(1, "cm"),pad_x =


print(primatePlot)
```



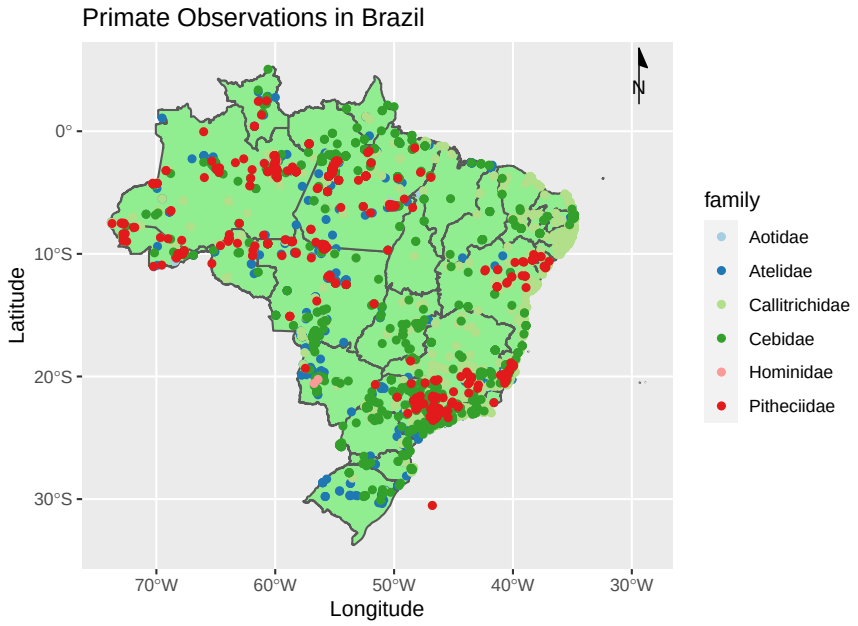
Looks pretty good. There are a number of parameters here that control how big it is, where it is on the page, and what kind of arrow it is. Let's move it from the bottom left (`'bl'`) to the top right (`'tr'`) with the `location` parameter:

```
#Plot with title and axis labels
primatePlot<-ggplot() +
  geom_sf(data= states, fill="light green") +
  scale_color_brewer(palette="Paired") +
  geom_sf(data = brazilPrimates,aes(color=family)) +
  labs(x="Longitude",y="Latitude",title="Primate Observations in Brazil")  +
   annotation_north_arrow(location = "tr",  height = unit(1, "cm"), width = unit(1, "cm


print(primatePlot)
```



We can probably do with a less fancy north arrow, frankly. You can find the different arrows (and the parameters that control them) by running `?north_arrow_orienteering` Lets go with `north_arrow_minimal` here.

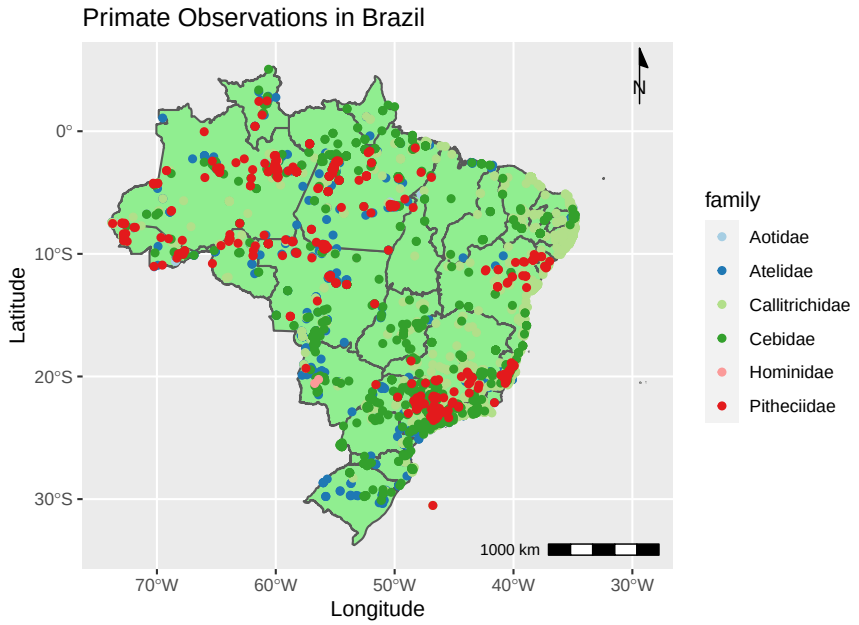
```
#Plot with title and axis labels
primatePlot<-ggplot() +
  geom_sf(data= states, fill="light green") +
  scale_color_brewer(palette="Paired") +
  geom_sf(data = brazilPrimates,aes(color=family)) +
  labs(x="Longitude",y="Latitude",title="Primate Observations in Brazil")  +
   annotation_north_arrow(location = "tr",  height = unit(1, "cm"), width = unit(1, "cm"),pad_x =


print(primatePlot)
```



Classy. Finally, our map needs a scale bar. We add that with `annotation_scale`:

```
#Plot with title and axis labels
primatePlot<-ggplot() +
  geom_sf(data= states, fill="light green") +
  scale_color_brewer(palette="Paired") +
  geom_sf(data = brazilPrimates,aes(color=family)) +
  labs(x="Longitude",y="Latitude",title="Primate Observations in Brazil")  +
    annotation_north_arrow(location = "tr",  height = unit(1, "cm"), width = unit(1, "
annotation_scale(location = "br", height = unit(0.2, "cm"))


print(primatePlot)
```

```
## Scale on map varies by more than 10%, scale bar may be inaccurate
```



Notice that error? That's because we're plotting this using a Geographic (elliptical) coordinate system, but it's being plotted on a 2D plane. That means that the distance relationships vary somewhat across the map. The `ggspatial` package does it's best job to find a reasonable middle point for the scale, but it lets us know that we need to be careful.

Chapter 5

Bringing it all together

Working with spatial data often involves finding a sequence of steps to create a desired endproduct. For this exercise, make a map of primate observations in Ecuador, with observations plotted by year. Use the data available in the working directory. Play around with a few different options for appearances, and don't forget to include a north arrow and scale bar!