INFX 573: Problem Set 4 - Data Analysis

TAPASVI BANSAL

Due: Thursday, November 2, 2017

Collaborators:

Instructions:

1. Replace the "Insert Your Name Here" text in the `author:` field with your own full name. Any collaborators must be listed on the top of your assignment.
2. Be sure to include well-documented (e.g. commented) code chucks, figures and clearly written text chunk explanations as necessary. Any figures should be clearly labeled and appropriately referenced within the text.
3. Collaboration on problem sets is fun and useful! However, you must turn in your individual write-up in his or her own words and his or her own work. The names of your collaborators must be listed on each assignment. Do not copy-and-paste from other students' responses or code.
4. When you have completed the assignment and have **checked** that your code both runs in the Console and knits correctly, rename the R Markdown file to `YourLastName_YourFirstName_ps4.Rmd`, knit a PDF and submit both the markdown and the PDF file on Canvas.

The Task

The problem set is inspired by a real-world situation and is deliberately somewhat vague. Your task is to understand the data, convert it into a suitable format, and find the tools that produce the desired output. Note: You are asked to produce a map but you don't have to use dedicated mapping tools like *ggmap* and shapefiles, just ordinary plotting will do.

You are working at PredicitiveAnalytics LLC. One day your Most Important Customer comes to you and says:

I need a temperature and precipitation map of Europe for January and July. It must be based on the most recet NOAA long term means data from NOAA webpage, the v401 format. And I need it by Thursday, November 2nd, 5:30pm. I just need a color map, it does not have to be anything fancy with borders and cities and rivers on it. Just the temperature and rain, plotted in a way I can understand would do.

Download the data and produce such maps for temperature and precipitation (do not use the tools on the website). Make sure to explain and label your data sources and units of measurement. Try to tune the plot with suitable colors, scales, etc, to impress your Important Customer. Comment, or otherwise explain your code, and briefly discuss the results.

Suggestions:

- If you use *ggplot* for plotting, add coordinate transformation `+ coord_map()` (requires *mapproj* library). This ensures the map will be in a valid map projection. You may experiment with different projections.

Solution:

```
# install.packages("ncdf4")
# install.packages("chron")
```

```r
# install.packages('mapproj')
library(tidyverse)
```

```
## Loading tidyverse: ggplot2
## Loading tidyverse: tibble
## Loading tidyverse: tidyr
## Loading tidyverse: readr
## Loading tidyverse: purrr
## Loading tidyverse: dplyr
```

```
## Warning: package 'tidyr' was built under R version 3.4.2
```

```
## Warning: package 'dplyr' was built under R version 3.4.2
```

```
## Conflicts with tidy packages ----------------------------------------------
```

```
## filter(): dplyr, stats
## lag():    dplyr, stats
```

```r
library(ncdf4)
library(chron)
```

```
## Warning: package 'chron' was built under R version 3.4.2
```

```r
library(lattice)
library(RColorBrewer)
library(mapproj)
```

```
## Loading required package: maps
```

```
##
## Attaching package: 'maps'
```

```
## The following object is masked from 'package:purrr':
##
##     map
```

```r
library(scales)
```

```
##
## Attaching package: 'scales'
```

```
## The following object is masked from 'package:purrr':
##
##     discard
```

```
## The following object is masked from 'package:readr':
##
##     col_factor
```

```r
#getwd()


noaa1 <- paste("air.mon.ltm.v401",".nc",sep = "")
noaaT <- nc_open(noaa1)
noaa2 <- paste("precip.mon.ltm.v401",".nc",sep = "")
noaaP <- nc_open(noaa2)
a <- attributes(noaaT)$names
b <- attributes(noaaP)$names
attributes(noaaT$var)$names
```

```
## [1] "climatology_bounds" "air"                "valid_yr_count"
```

```r
setdiff(a,b)

## character(0)

lon <- noaaT$dim$lon$vals
lon <- lon -180

long <- ncvar_get(noaaT,"lon", verbose = F) - 180

latt <- noaaT$dim$lat$vals
latt <- rev(latt)
lat <- ncvar_get(noaaT, "lat")
lat <- rev(lat)

t <- noaaT$dim$time$vals
time <- ncvar_get(noaaT,"time")
timeunits <- ncatt_get(noaaT,"time","units")
timed <- time/24
time_d <- as.Date(timed, format="%j", origin=as.Date("1900-01-01"))
tconverttime <- as.POSIXct(time*3600,"", tryFormats = c("%Y-%m-%d %H:%M:%OS","%Y/%m/%d %H:%M:%OS"),orig

temp <- ncvar_get(noaaT,"air")
ltemp <- ncatt_get(noaaT, "air","long_name")
utemp <- ncatt_get(noaaT, "air","units")
tmissingvalue <- ncatt_get(noaaT, "air","missing_value")
temp[temp==tmissingvalue$value] <- NA

m1 <- 1
temp_1 <- temp[,,m1]
dim(temp_1)

## [1] 720 360

m7 <- 7
temp_7 <- temp[,,m7]
#dim(temp_7)

longlat <-as.matrix(expand.grid(long,lat))
#dim(longlat)
v1temp <- as.vector(temp_1)
v2temp <- as.vector(temp_7)
temp_Jan <- data.frame(cbind(longlat,v1temp))
names(temp_Jan) <- c("lon","lat","temp")

temp_July <- data.frame(cbind(longlat,v2temp))
names(temp_July) <- c("lon","lat","temp")

cols <- brewer.pal(n = 5, name = "RdBu")

########## For Precipitation #############


plong <- ncvar_get(noaaP,"lon", verbose = F) - 180
plat <- ncvar_get(noaaP, "lat")
plat <- rev(lat)
```

```r
ptime <- ncvar_get(noaaP,"time")
ptimeunits <- ncatt_get(noaaP,"time","units")
ptimed <- time/24
ptime_d <- as.Date(ptimed, format="%j", origin=as.Date("1900-01-01"))
pconverttime <- as.POSIXct(ptime*3600,"", tryFormats = c("%Y-%m-%d %H:%M:%OS","%Y/%m/%d %H:%M:%OS"),orig

prep <- ncvar_get(noaaP,"precip")
lprep <- ncatt_get(noaaP, "precip","long_name")
uprep <- ncatt_get(noaaP, "precip","units")
pmissingvalue <- ncatt_get(noaaP, "precip","missing_value")
prep[prep==pmissingvalue$value] <- NA

pm1 <- 1
prep_1 <- prep[,,m1]
dim(prep_1)
```

```
## [1] 720 360
```

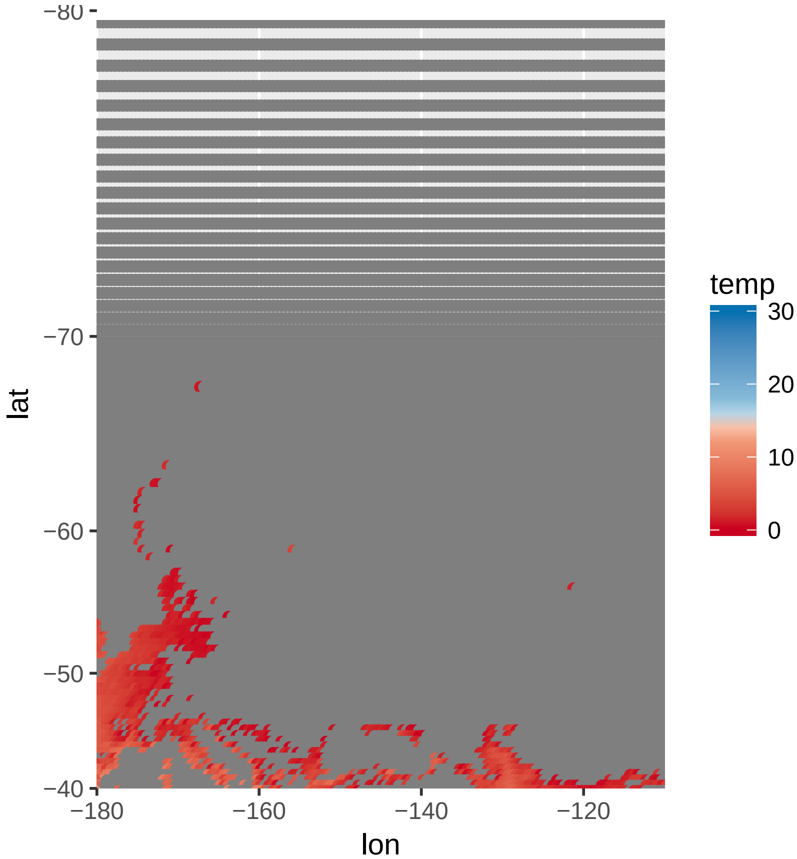
```r
pm7 <- 7
prep_7 <- prep[,,m7]
dim(prep_7)
```

```
## [1] 720 360
```

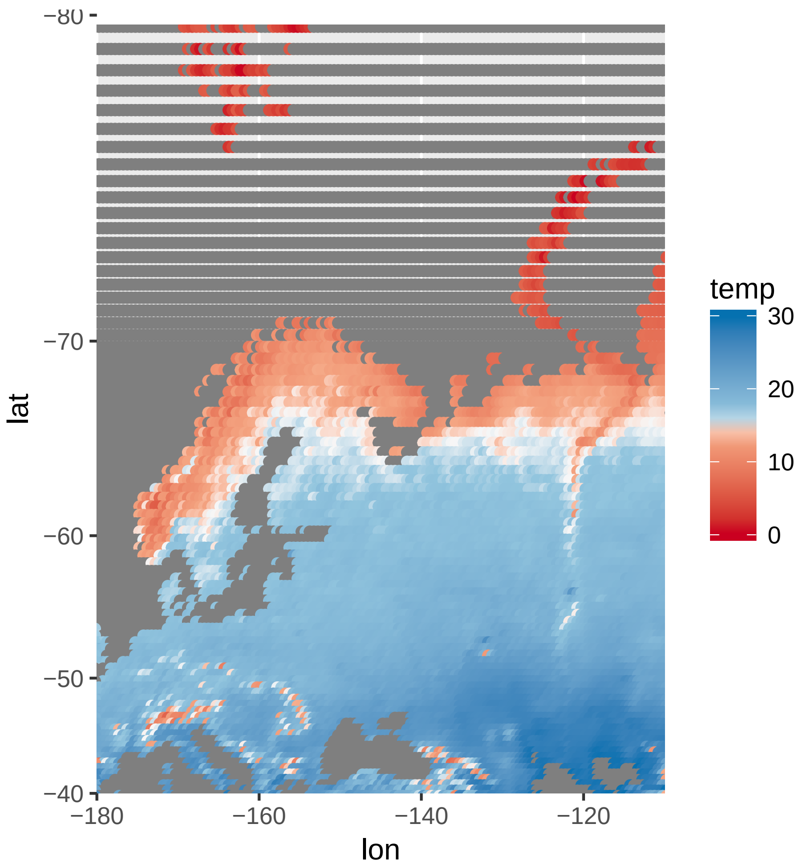
```r
longlat <-as.matrix(expand.grid(long,lat))
v1prep <- as.vector(prep_1)
v2prep <- as.vector(prep_7)

prep_Jan <- data.frame(cbind(longlat,v1prep))
names(prep_Jan) <- c("lon","lat","prep")

prep_July <- data.frame(cbind(longlat,v2prep))
names(prep_July) <- c("lon","lat","prep")

cols <- brewer.pal(n = 5, name = "RdBu")
```

```
## Plotting for Temperature
```

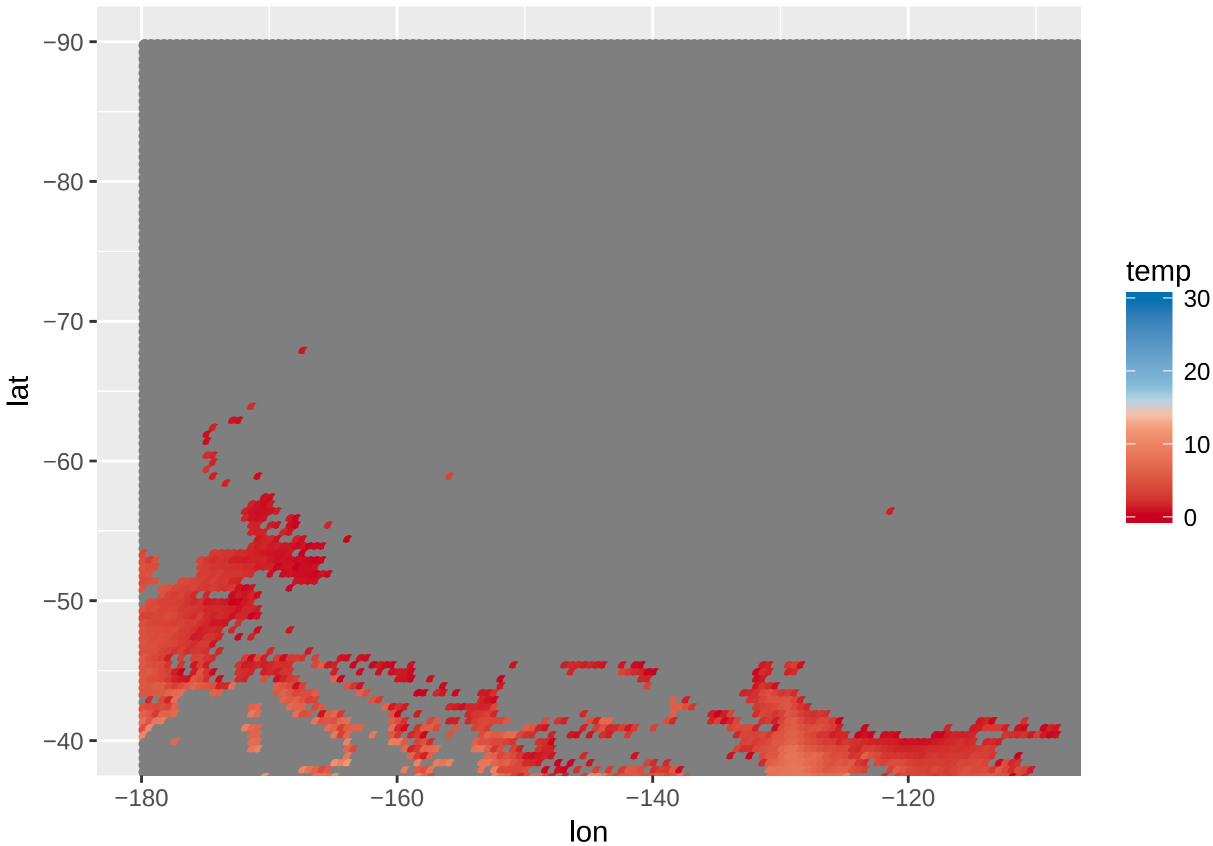
```r
ggplot() +
  geom_point(data = temp_Jan, aes(x=lon, y = lat, color=temp)) +
  coord_map(xlim =c(-180,-110),ylim=c(-90,-40)) +
  scale_colour_gradientn(colours = cols,
                         values = rescale(c(-10, -1, 0, 1, 10)),
                         guide = "colorbar", limits=c(0, 30)) +
  scale_y_reverse()
```



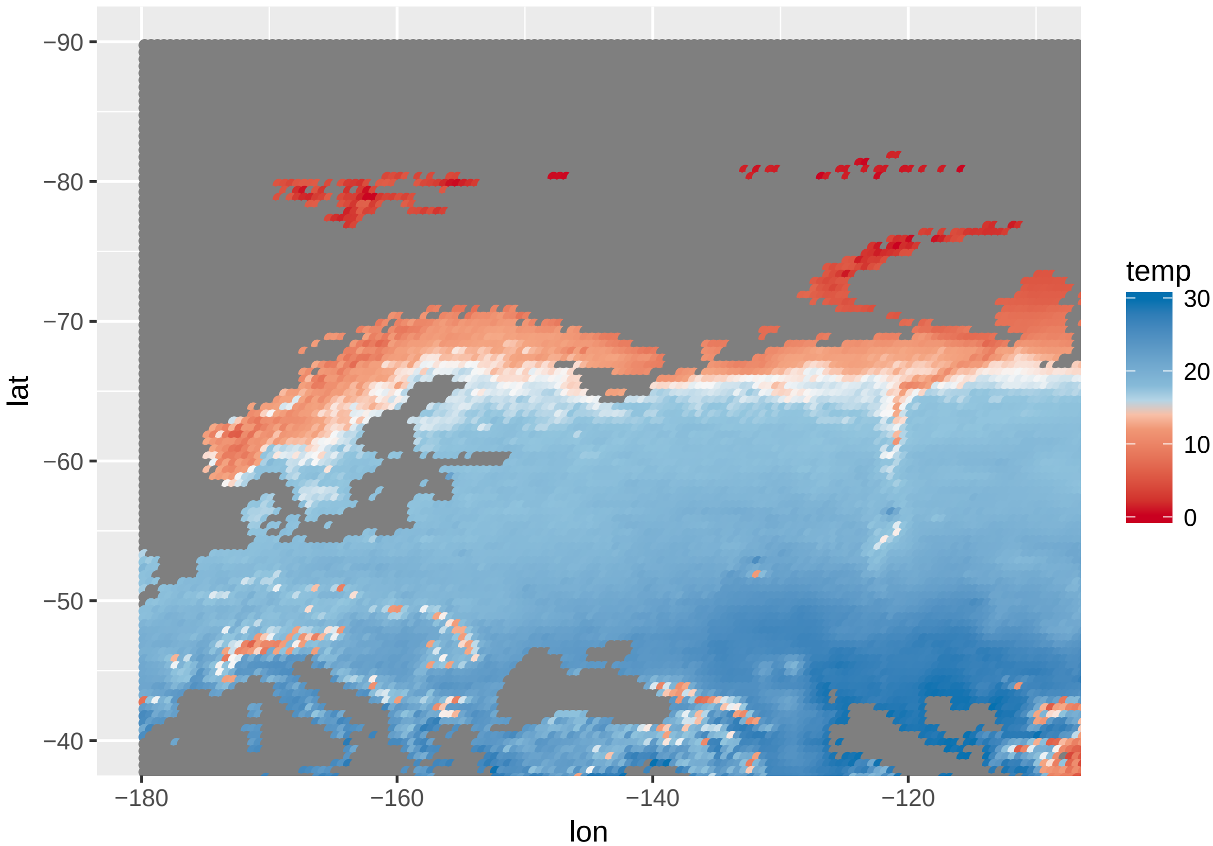

```r
ggplot() +
  geom_point(data = temp_July, aes(x=lon, y = lat, color=temp)) +
  coord_map(xlim =c(-180,-110),ylim=c(-90,-40)) +
  scale_colour_gradientn(colours = cols,
                         values = rescale(c(-10, -1, 0, 1, 10)),
                         guide = "colorbar", limits=c(0, 30)) +
  scale_y_reverse()
```



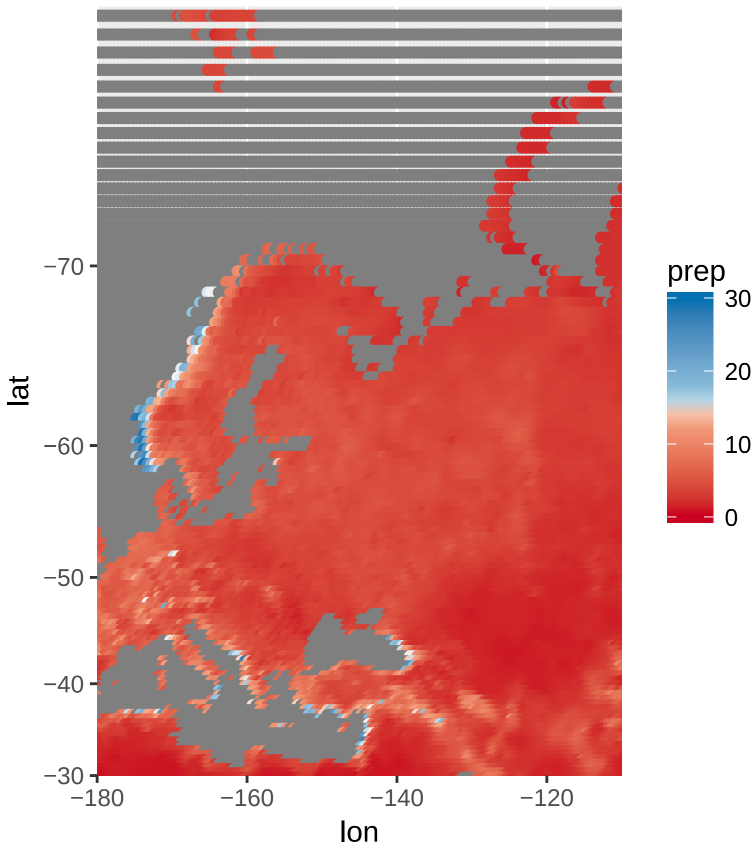

```
ggplot() +
  geom_point(data = temp_Jan, aes(x=lon, y = lat, color=temp)) + coord_cartesian(xlim =c(-180,-110),ylin
  scale_colour_gradientn(colours = cols,
                         values = rescale(c(-10, -1, 0, 1, 10)),
                         guide = "colorbar", limits=c(0, 30)) +
  scale_y_reverse()
```



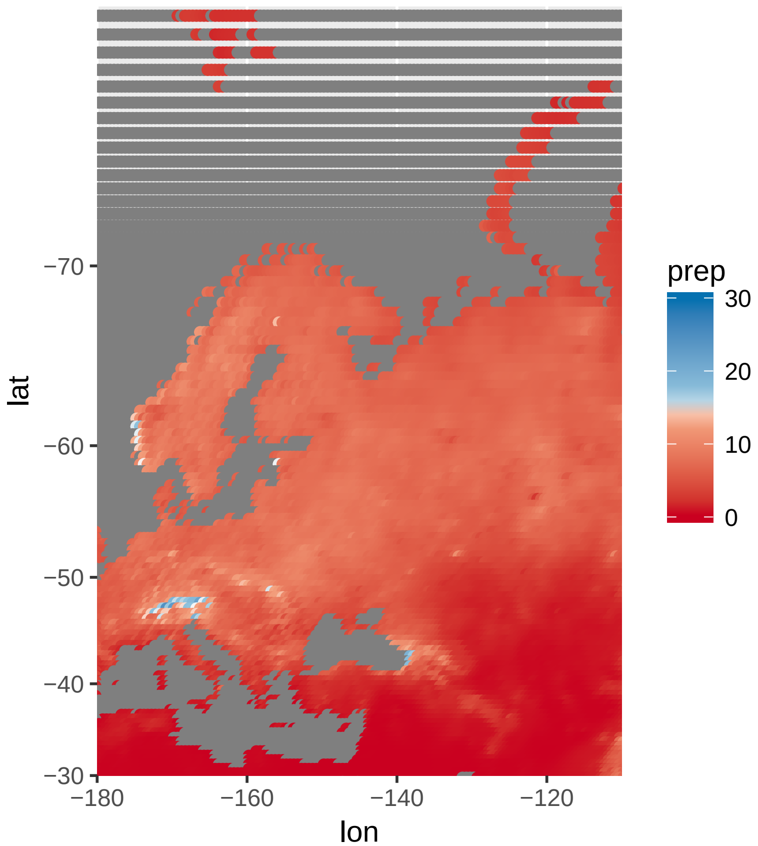

```
ggplot() +
  geom_point(data = temp_July, aes(x=lon, y = lat, color=temp)) + coord_cartesian(xlim =c(-180,-110),yl
  scale_colour_gradientn(colours = cols,
                         values = rescale(c(-10, -1, 0, 1, 10)),
                         guide = "colorbar", limits=c(0, 30)) +
  scale_y_reverse()
```



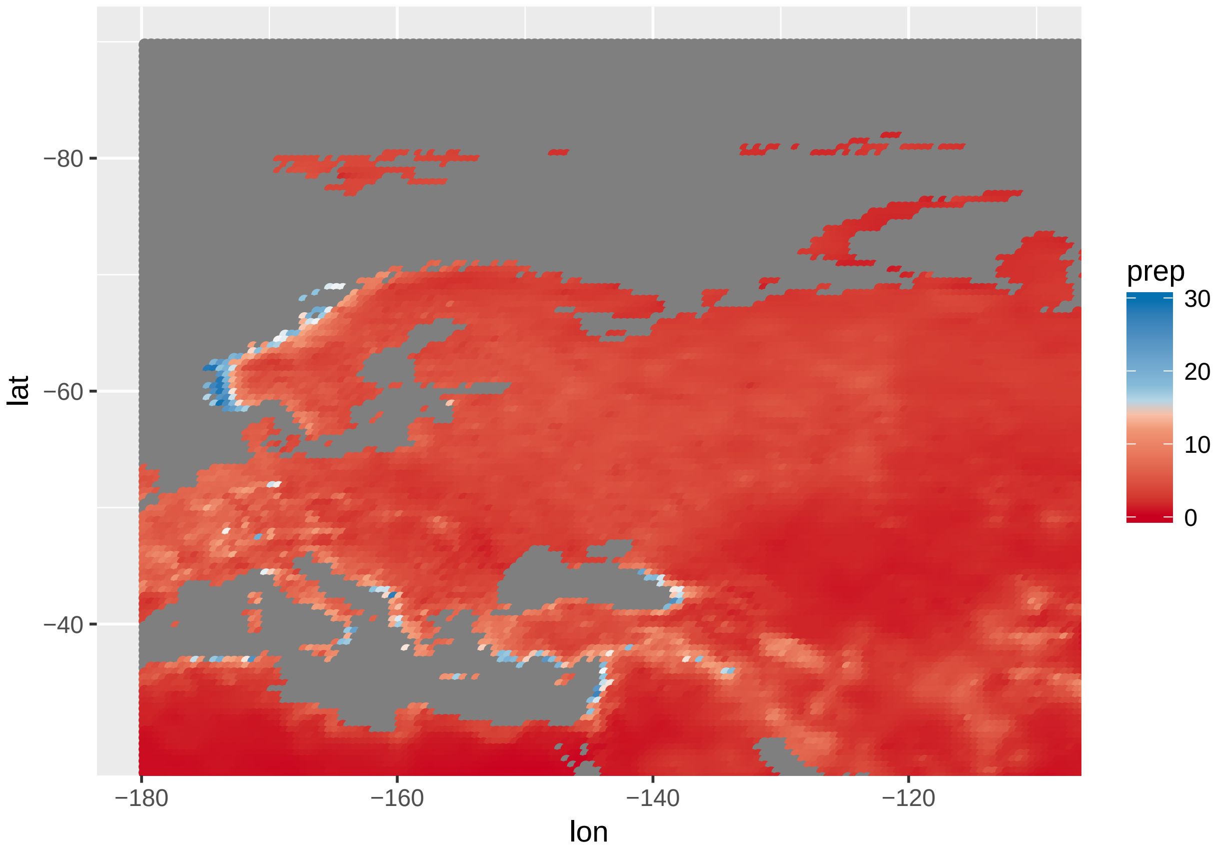

```
# Plotting for Precipitation

ggplot() +
  geom_point(data = prep_Jan, aes(x=lon, y = lat, color=prep)) +
  coord_map(xlim =c(-180,-110),ylim=c(-90,-30)) +
  scale_colour_gradientn(colours = cols,
                         values = rescale(c(-10, -1, 0, 1, 10)),
                         guide = "colorbar", limits=c(0, 30)) +
  scale_y_reverse()
```



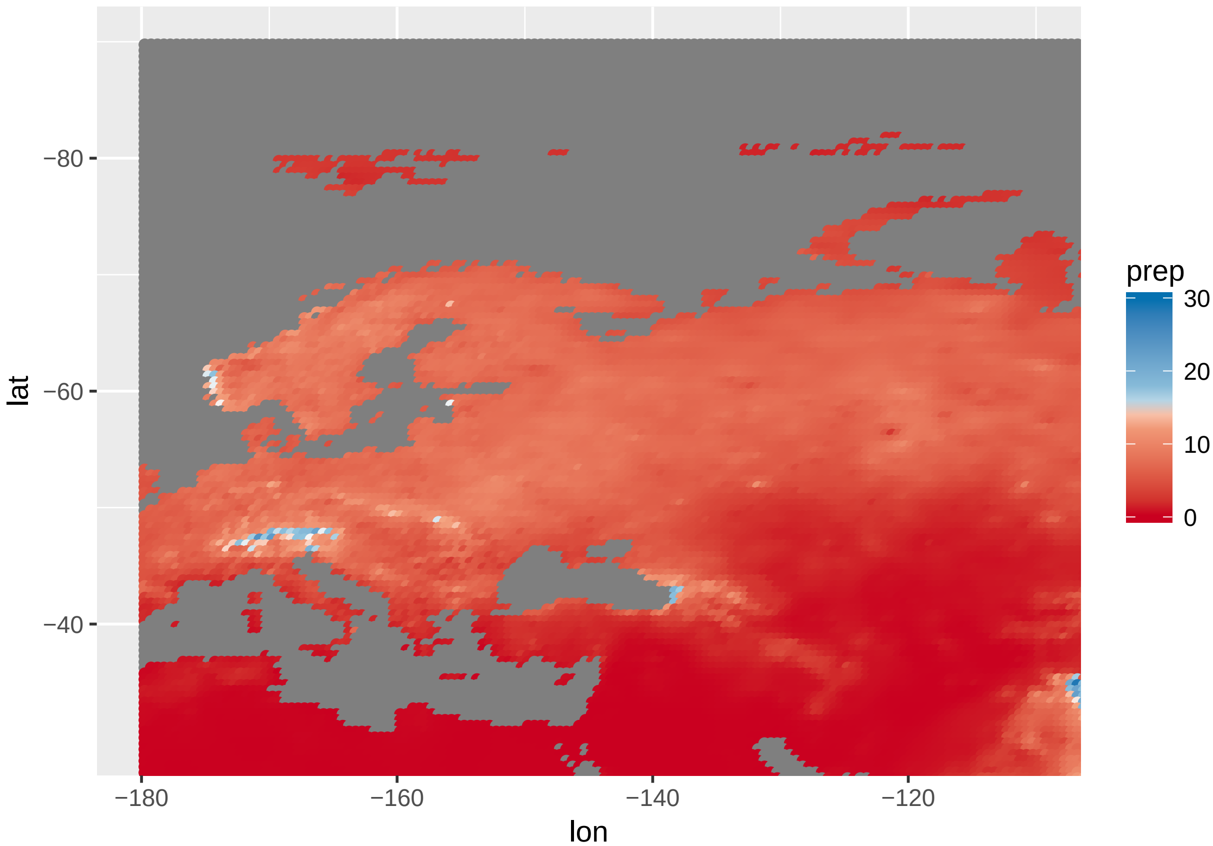

```
ggplot() +
  geom_point(data = prep_July, aes(x=lon, y = lat, color=prep)) +
  coord_map(xlim =c(-180,-110),ylim=c(-90,-30)) +
  scale_colour_gradientn(colours = cols,
                         values = rescale(c(-10, -1, 0, 1, 10)),
                         guide = "colorbar", limits=c(0, 30)) +
  scale_y_reverse()
```




```
ggplot() +
  geom_point(data = prep_Jan, aes(x=lon, y = lat, color=prep)) + coord_cartesian(xlim =c(-180,-110),ylim
  scale_colour_gradientn(colours = cols,
                         values = rescale(c(-10, -1, 0, 1, 10)),
                         guide = "colorbar", limits=c(0, 30)) +
  scale_y_reverse()
```




```
ggplot() +
  geom_point(data = prep_July, aes(x=lon, y = lat, color=prep)) + coord_cartesian(xlim =c(-180,-110),yli
  scale_colour_gradientn(colours = cols,
                         values = rescale(c(-10, -1, 0, 1, 10)),
                         guide = "colorbar", limits=c(0, 30)) +
  scale_y_reverse()
```



In Europe, the Mean temperatures are generally higher (lighter in color) in July compared to January, which makes sense given the summer timing of the Northern Hemisphere

The Mountain regions are relatively colder than rest of the places, which makes sense given temperature generally decreases with elevation

Precipitation varies more in comparision to temperature across Europe: in some places it rains more in January/the winter time (e.g. Italy, Greek coasts, western edge of Norway) and in other places it rains more in July/the summer time (e.g. spots of Russia, Romania/Ukraine, most of Scandinavia)