Maps - Natural Earth

Oscar A. Trevizo

Contents

Introduction

Examples using R Natural Earth package.

Example 1: Taiwan Real Estate Example

```
# https://archive.ics.uci.edu/ml/datasets/Real+estate+valuation+data+set
rawDat <- import("../data/RealEstate.xlsx")

# Explore
class(rawDat)
```

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

```
mode(rawDat)
```

```
## [1] "list"
```

```
str(rawDat)
```

```
## 'data.frame':    414 obs. of  8 variables:
##  $ No                                    : num  1 2 3 4 5 6 7 8 9 10 ...
##  $ X1 transaction date                   : num  2013 2013 2014 2014 2013 ...
##  $ X2 house age                          : num  32 19.5 13.3 13.3 5 7.1 34.5 20.3 31.7 17.9 ...
##  $ X3 distance to the nearest MRT station: num  84.9 306.6 562 562 390.6 ...
##  $ X4 number of convenience stores       : num  10 9 5 5 5 3 7 6 1 3 ...
##  $ X5 latitude                           : num  25 25 25 25 25 ...
##  $ X6 longitude                          : num  122 122 122 122 122 ...
##  $ Y house price of unit area            : num  37.9 42.2 47.3 54.8 43.1 32.1 40.3 46.7 18.8 22.1 ..
```

```
summary(rawDat)
```

```
##        No        X1 transaction date  X2 house age
##  Min.   :  1.0   Min.   :2013        Min.   : 0.000
##  1st Qu.:104.2   1st Qu.:2013        1st Qu.: 9.025
##  Median :207.5   Median :2013        Median :16.100
##  Mean   :207.5   Mean   :2013        Mean   :17.713
##  3rd Qu.:310.8   3rd Qu.:2013        3rd Qu.:28.150
##  Max.   :414.0   Max.   :2014        Max.   :43.800
##  X3 distance to the nearest MRT station X4 number of convenience stores
##  Min.   :  23.38                        Min.   : 0.000
##  1st Qu.: 289.32                        1st Qu.: 1.000
##  Median : 492.23                        Median : 4.000
##  Mean   :1083.89                        Mean   : 4.094
##  3rd Qu.:1454.28                        3rd Qu.: 6.000
##  Max.   :6488.02                        Max.   :10.000
##   X5 latitude     X6 longitude   Y house price of unit area
##  Min.   :24.93   Min.   :121.5   Min.   :  7.60
##  1st Qu.:24.96   1st Qu.:121.5   1st Qu.: 27.70
##  Median :24.97   Median :121.5   Median : 38.45
##  Mean   :24.97   Mean   :121.5   Mean   : 37.98
##  3rd Qu.:24.98   3rd Qu.:121.5   3rd Qu.: 46.60
##  Max.   :25.01   Max.   :121.6   Max.   :117.50
```

```
# First, remove the "No" (Number). The dataset has Row Names already
# "No" is not numeric. It is a key.
reDat <- rawDat[, !(names(rawDat) %in% c("No"))]
str(reDat)
```

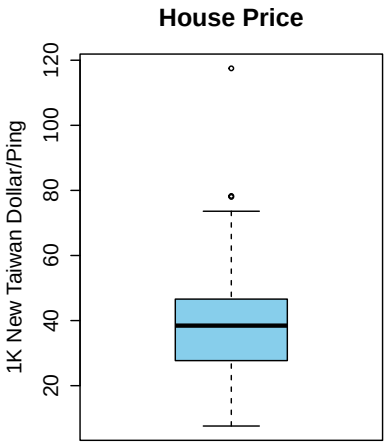
```
## 'data.frame':    414 obs. of  7 variables:
##  $ X1 transaction date                   : num  2013 2013 2014 2014 2013 ...
##  $ X2 house age                          : num  32 19.5 13.3 13.3 5 7.1 34.5 20.3 31.7 17.9 ...
##  $ X3 distance to the nearest MRT station: num  84.9 306.6 562 562 390.6 ...
##  $ X4 number of convenience stores       : num  10 9 5 5 5 3 7 6 1 3 ...
##  $ X5 latitude                           : num  25 25 25 25 25 ...
##  $ X6 longitude                          : num  122 122 122 122 122 ...
##  $ Y house price of unit area            : num  37.9 42.2 47.3 54.8 43.1 32.1 40.3 46.7 18.8 22.1 ..
```

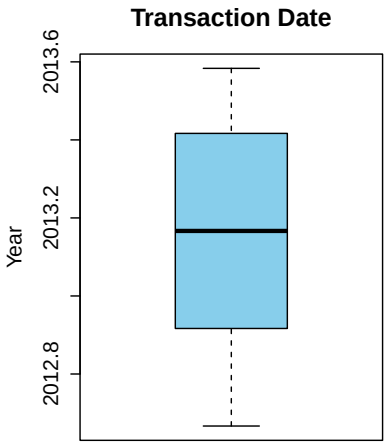
```
#
##
```

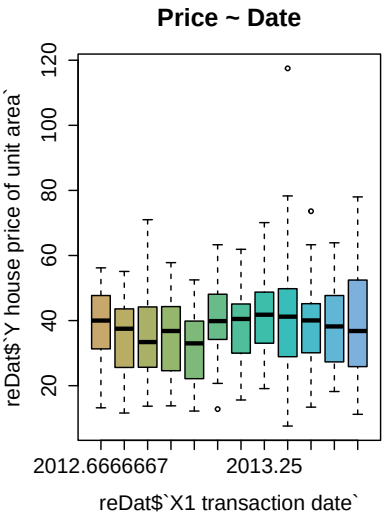
Exploratory Data Analysis: Boxplots

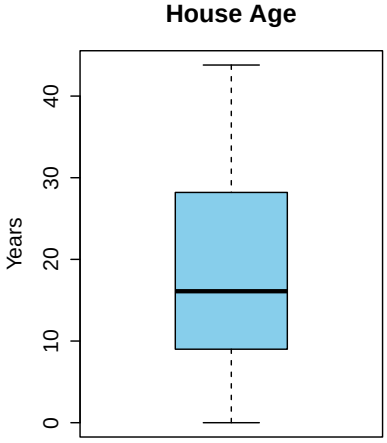
Now that I have `reDat`, I will explore with boxplots, pairs plots, correlations. I have `lat/log` variables that I will need to consider in a special way. There are several questions I will need to address before applying the `lat/log`

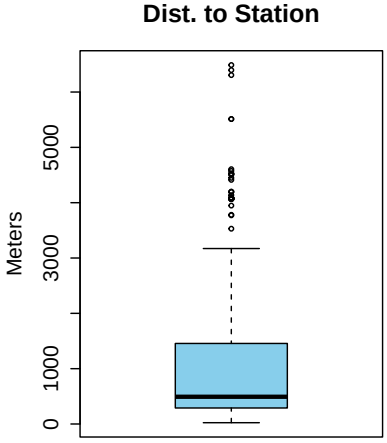
```
##

# Let boxplots expose some properties to aid the summary() tables
#

old.par <- par(mfrow=c(3,3),ps=16)

boxplot(reDat$`Y house price of unit area`,
         main = "House Price", ylab = "1K New Taiwan Dollar/Ping",
       col = "skyblue")
boxplot(reDat$`X1 transaction date`,
        main = "Transaction Date", ylab = "Year",
       col = "skyblue")
boxplot(reDat$`Y house price of unit area` ~ reDat$`X1 transaction date`,
        main="Price ~ Date",
       col=qualitative_hcl(12, palette = "Harmonic"))
boxplot(reDat$`X2 house age`,
        main = "House Age", ylab = "Years",
       col = "skyblue")
boxplot(reDat$`X3 distance to the nearest MRT station`,
        main = "Dist. to Station", ylab = "Meters",
       col = "skyblue")
boxplot(reDat$`X4 number of convenience stores`,
        main = "Stores", ylab = "Convenience stores",
       col = "skyblue")
boxplot(reDat$`X5 latitude`,
         main = "Latitude", ylab = "Degrees (Geo)",
       col = "skyblue")
boxplot(reDat$`X6 longitude`,
         main = "Longitude", ylab = "Degrees (Geo)",
       col = "skyblue")

par(old.par)
```

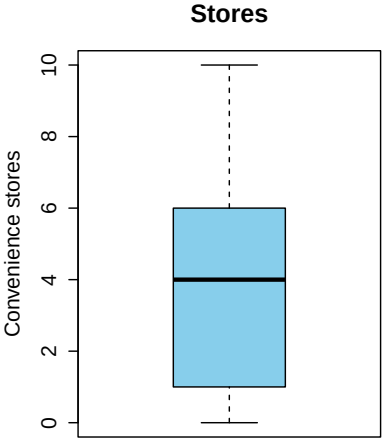

House Price
1K New Taiwan Dollar/Ping
20
40
60
80
100
120

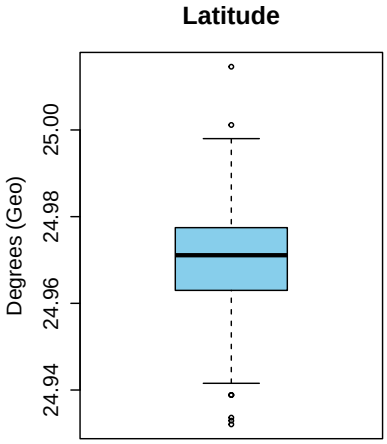

Transaction Date
Year
2012.8
2013.2
2013.6

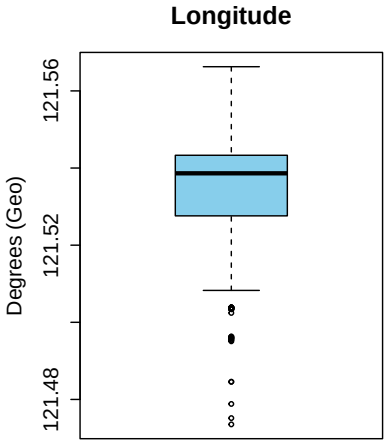

Price ~ Date
reDat$`Y house price of unit area`
20
40
60
80
100
120
2012.6666667
2013.25
reDat$`X1 transaction date`


House Age
Years
0
10
20
30
40


Dist. to Station
Meters
0
1000
3000
5000


Stores
Convenience stores
0
2
4
6
8
10


Latitude
Degrees (Geo)
24.94
24.96
24.98
25.00


Longitude
Degrees (Geo)
121.48
121.52
121.56

```
#
##
```

Takeaways from boxplots:

- Price: The price ranges widely from minimum 7.6 to maximum 117.5. It has 1st quartile at 27.7 and 3rd quartile at 46.6. It has significant outliers on both ends min and max. And the price is 5/3 times higher in the 3rd quartile than in the 1st quartile, not quite doubled, but almost there.
- Transaction date seems to be very uniform and does not appear to add any value to the data set other than confirming that the data was taken approximately within the same time period, relatively speaking. That is, data did not come from different decades that may have an impact on subsequent comparisons.
- House age may not be something that could have an impact on the price. Boxplot and summary exhibit a nicely distributed set of data.But there is not apparent pattern when plotting it against price.
- Distance to a station exhibits some particular patterns. The medium is moved towards the bottom of the data (in terms of meters to a station), and it exhibits some outliers for houses that are far from a station. This variable may require transformation.
- Number of stores seems to be very well distributed. It may not need to be transformed, but I reserve that decision to a subsequent step.
- Latitude and Longitude need to be analyzed as a pair. The boxplots do show that there are some outliers and that may enrich our analysis.
- Finally, our outcome variable for house price does exhibit some outliers that will need to be analyzed in the diagnostic plots.

Geolocation analysis

Now I will analyze the latitude and longitude data. Those are geographical coordinates in degrees. Latitude provide degrees running north to south (vertically), and longitude east to west (horizontally).

But the earth is not flat, it is rounded. The distance in meters between degrees depend on where we are, on the lat/long coordinates. It is longer to go around the world in the equator than it is say by the north pole (you can spin and go around the world if you are standing precisely on the north pole).

Where in the world does our data come from? And what does latitude and longitude mean in terms of distance (meters or Km)?

Our area of interest is New Taipei, a very large metropolis. And within New Taipei our area of interest is located at around 25 degree latitude (N/S) and 121.5 longitude (E/W). The central part of this polygon is heavily populated. Some parts of the polygon a rural. Google Earth provide a good introduction to this area: https://earth.google.com/web/@24.9774487,121.54564399,16.81296399a,21285.31191319d,35y,-0h,0t,0r".

In those coordinates:

1 longitude degree is around 100 Km moving east to west along the longitude, 1 latitude degree is around 111 Km moving north to south.

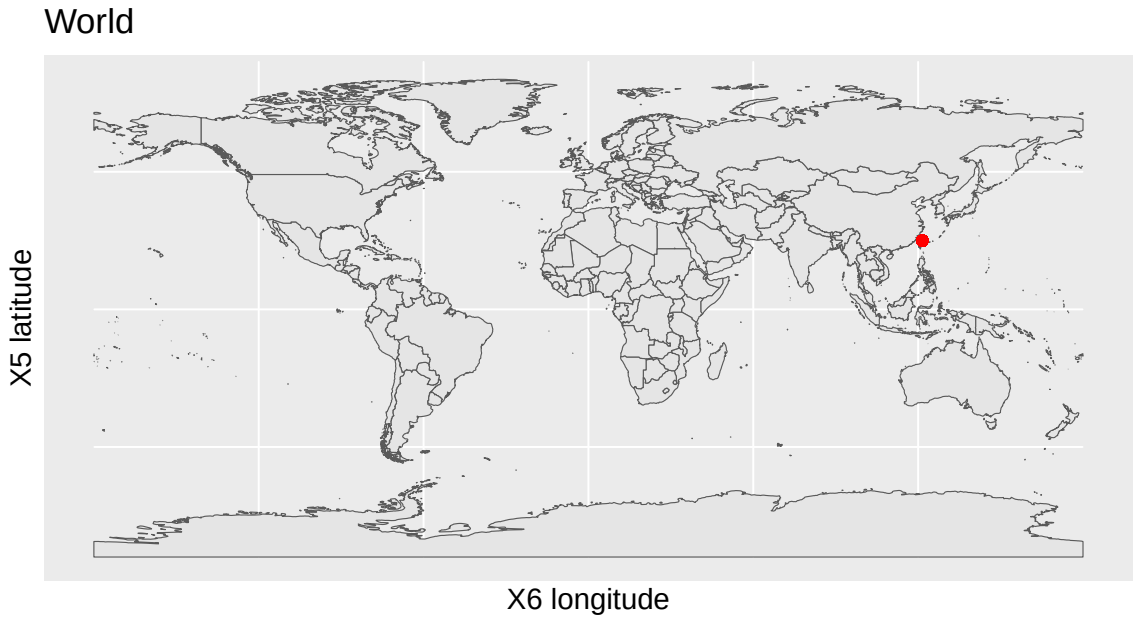
Source: https://www.nhc.noaa.gov/gccalc.shtml

Geo plots

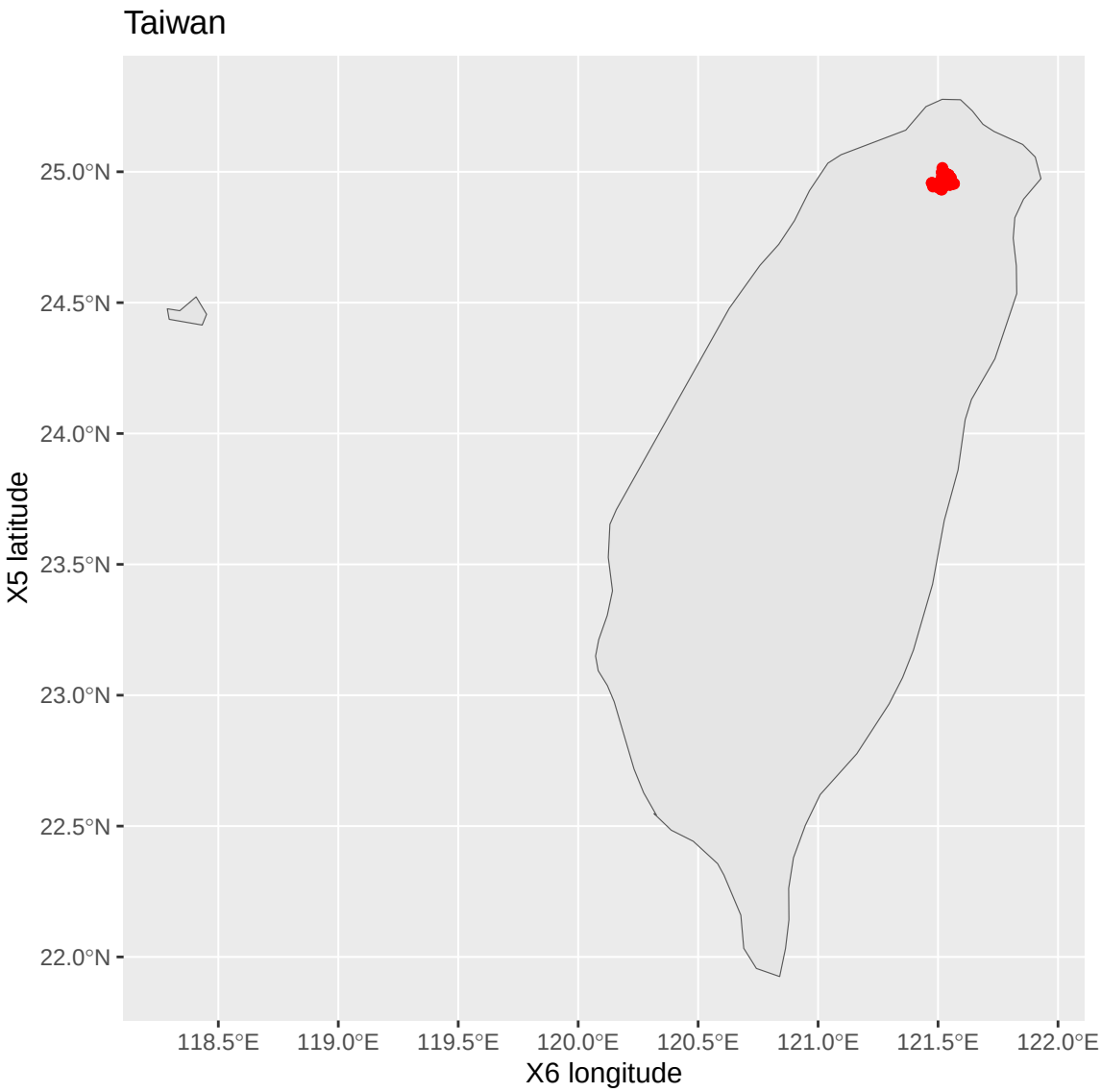
The following code provides plots related to latitude and longitude.

```
##
#
# Start with a map.
# Then plot data on a latitude / longitude chart
#
# For simplicity, make 1 degree to be equal to 100 Km in this exercise.
#
# Need to take a close look at the lat / long data.
# Determine how it could be used and in subsequent steps see if it should be used.
#
# The earth is not flat, it is rounded. Degree scale to meters depend on coordinates.
#
# Where in the world does our data come from,
# and what does latitude and longitude mean?
#
# Our area of interest is located at around 25 degree latitude (N/S) and 121.5 longitude (E/W).
# In those coordinates:
# 1 longitude degree is around 100 Km moving east to west along the longitude,
# 1 latitude degree is around 111 Km moving north to south.
#
# Source: https://www.nhc.noaa.gov/gccalc.shtml
#
#
#
# world <- map_data("world"). Reference: https://slcladal.github.io/maps.html
# Using rnaturalearth and rnaturaldata to use ggplot
world <- ne_countries(scale = "medium", returnclass = "sf")
taiwan <- ne_countries(scale = "medium", returnclass = "sf", country="Taiwan")

plotMap <- vector("list", length=2)

old.par <- par(mfrow=c(1,2),ps=16)

plotMap[[1]] <- ggplot(world) +
  geom_sf() +
  geom_point(data = reDat,
             aes( x = `X6 longitude`,
                  y = `X5 latitude`), col = "red") +
  labs(title = "World")

plotMap[[2]] <- ggplot(taiwan) +
  geom_sf() +
  geom_point(data = reDat,
             aes( x = `X6 longitude`,
                  y = `X5 latitude`), col = "red") +
  labs(title = "Taiwan")

plotMap
```

```
## [[1]]
```



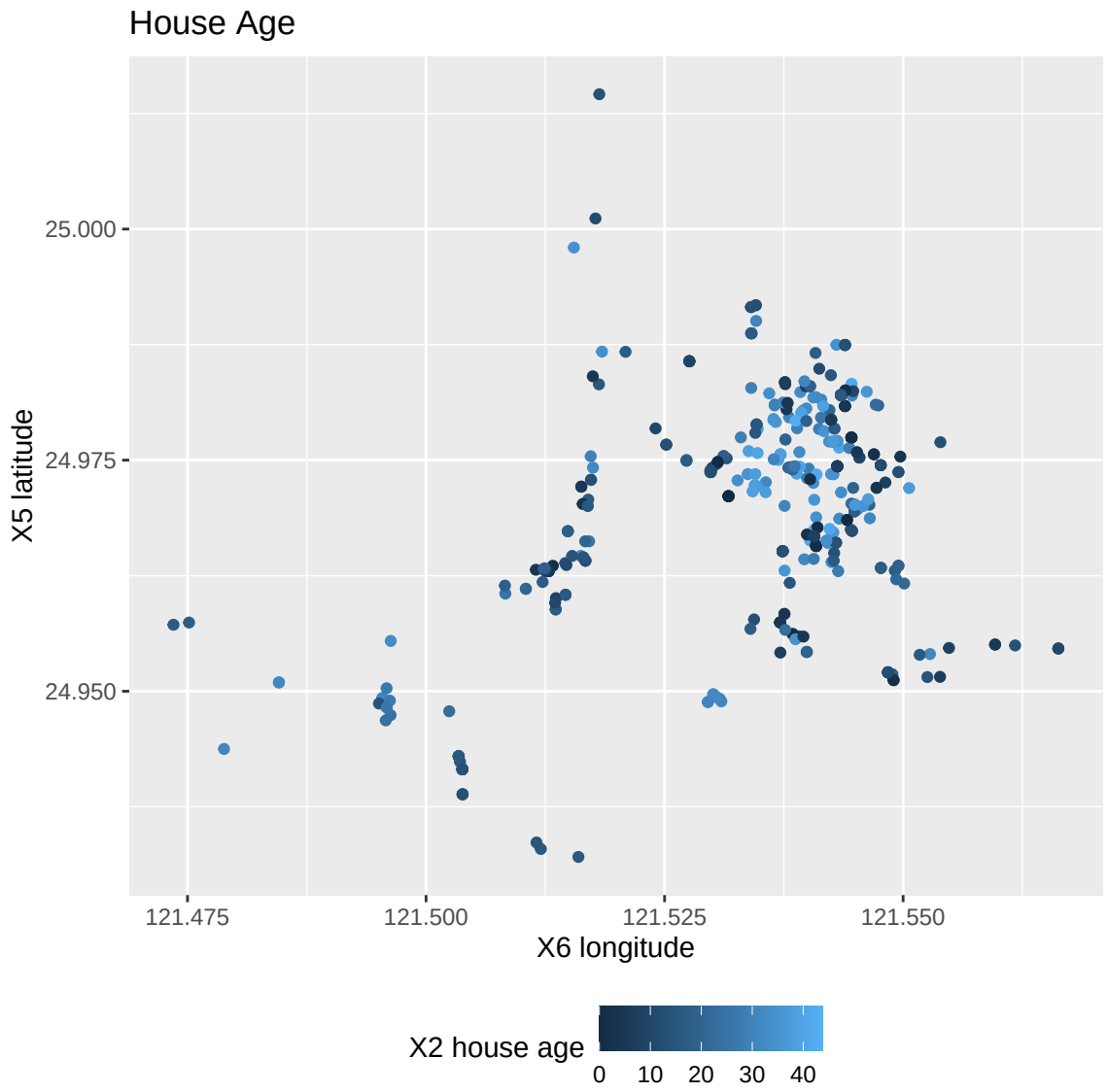

```
##
## [[2]]
```



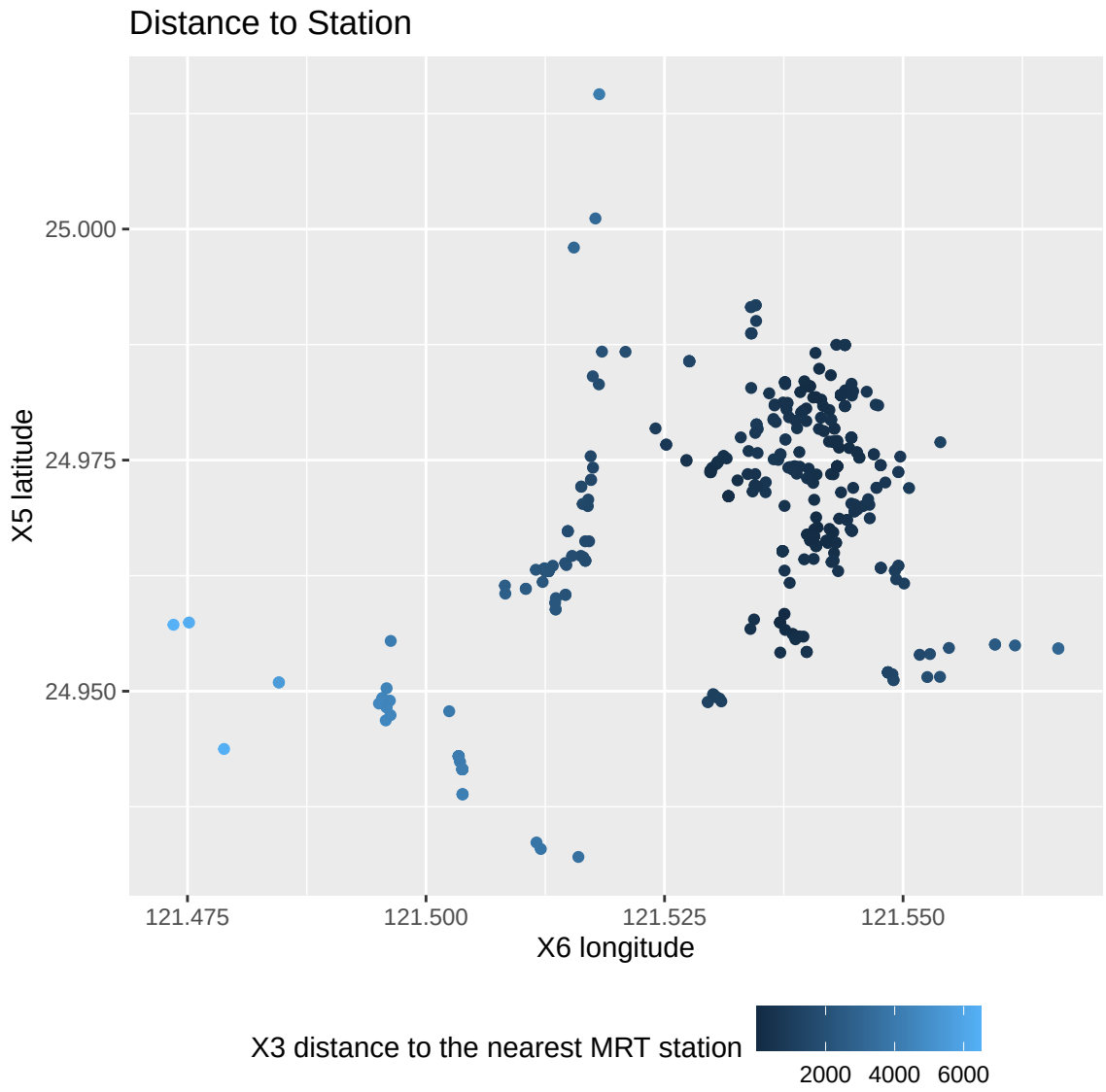

```
# Now plot on on longitude / latitude coordinates
#
# Latitude runs north to south (y axis). Longitude runs east to west (x axis).
# Simple scatterplot with horizontal axis as longitude, vertical as latitude.
# Four plots to visualize the spacial effect on each other variable.
#

plotSpatial <- vector("list", length=5)

plotMe <- ggplot(reDat,
       aes( x = `X6 longitude`,
            y = `X5 latitude`,
            colour = `X2 house age`)) +
  geom_point() +
  theme(legend.position = "bottom") +
  labs(title = "House Age")
```

```
plotSpatial[[1]] <- plotMe

plotMe <- ggplot(reDat,
       aes( x = `X6 longitude`,
            y = `X5 latitude`,
            colour = `X3 distance to the nearest MRT station`)) +
  geom_point() +
  theme(legend.position = "bottom") +
  labs(title = "Distance to Station")

plotSpatial[[2]] <- plotMe

plotMe <- ggplot(reDat,
       aes( x = `X6 longitude`,
            y = `X5 latitude`,
            colour = `X4 number of convenience stores`)) +
  geom_point() +
  theme(legend.position = "bottom") +
  labs(title = "Stores")

plotSpatial[[3]] <- plotMe

plotMe <- ggplot(reDat,
       aes( x = `X6 longitude`,
            y = `X5 latitude`,
            colour = `Y house price of unit area`)) +
  geom_point() +
  theme(legend.position = "bottom") +
  labs(title = "Price/Area")

plotSpatial[[4]] <- plotMe

plotBins <- ggplot(reDat,
       aes( x = `X6 longitude`,
            y = `X5 latitude`)) +
  stat_bin2d() +
  theme(legend.position = "bottom") +
  labs(title = "Bins with Counts of Houses")

plotSpatial
```

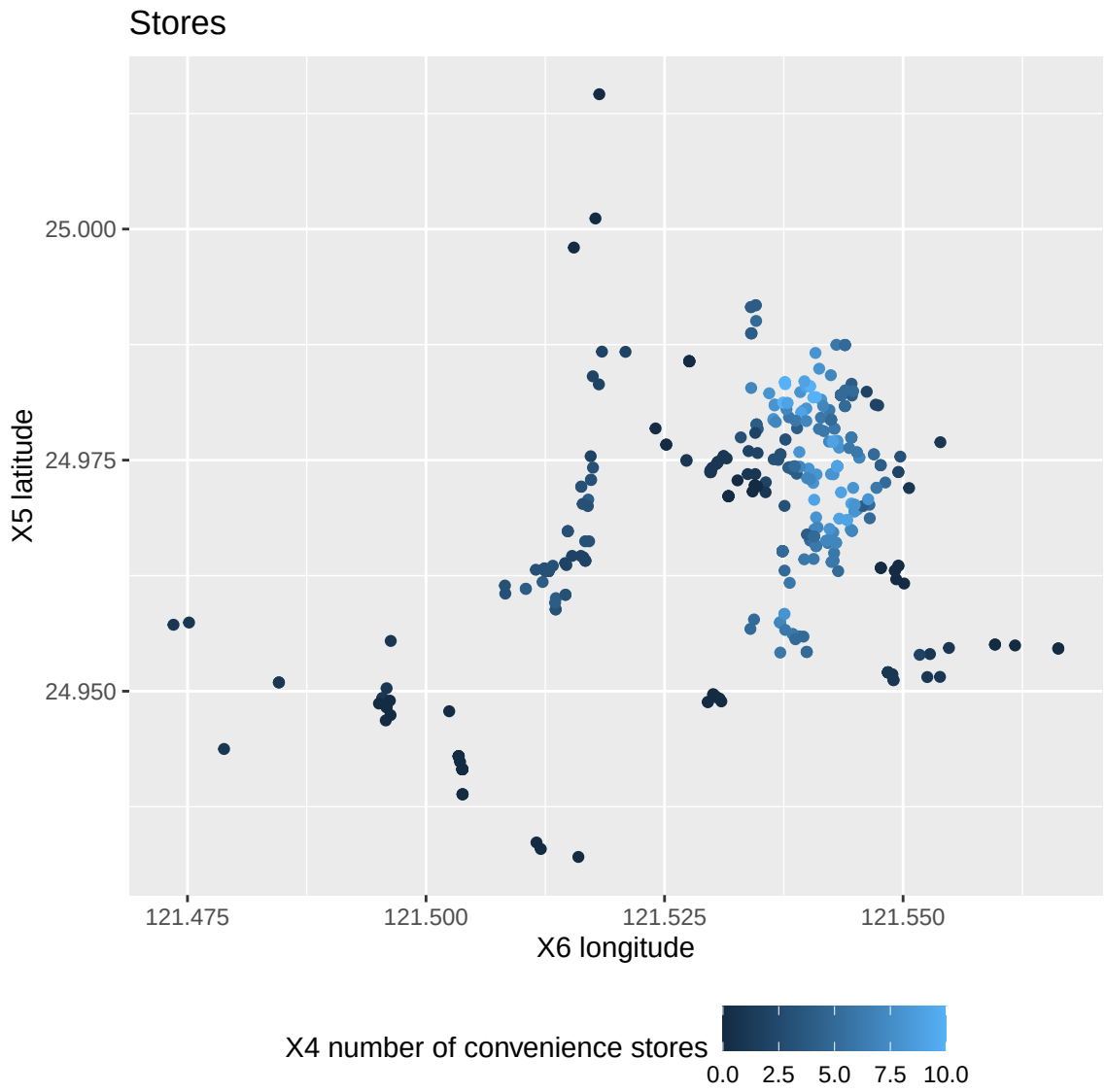
```
## [[1]]
```


House Age
25.000
24.975
24.950
X5 latitude
121.475
121.500
121.525
121.550
X6 longitude
X2 house age
0
10
20
30
40

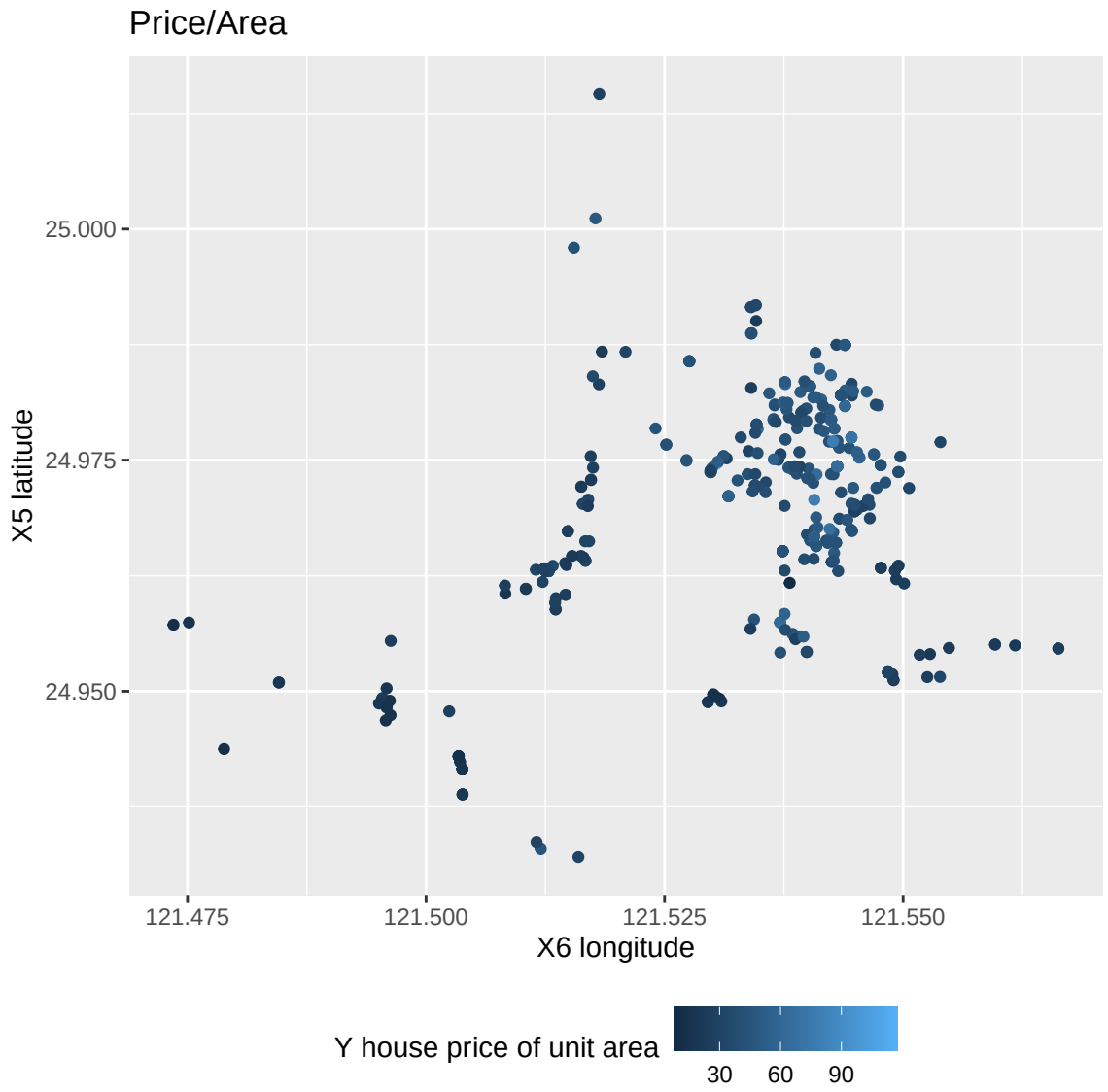
```
##
## [[2]]
```


Distance to Station
25.000
24.975
24.950
X5 latitude
121.475
121.500
121.525
121.550
X6 longitude
X3 distance to the nearest MRT station
2000
4000
6000

```
##
## [[3]]
```

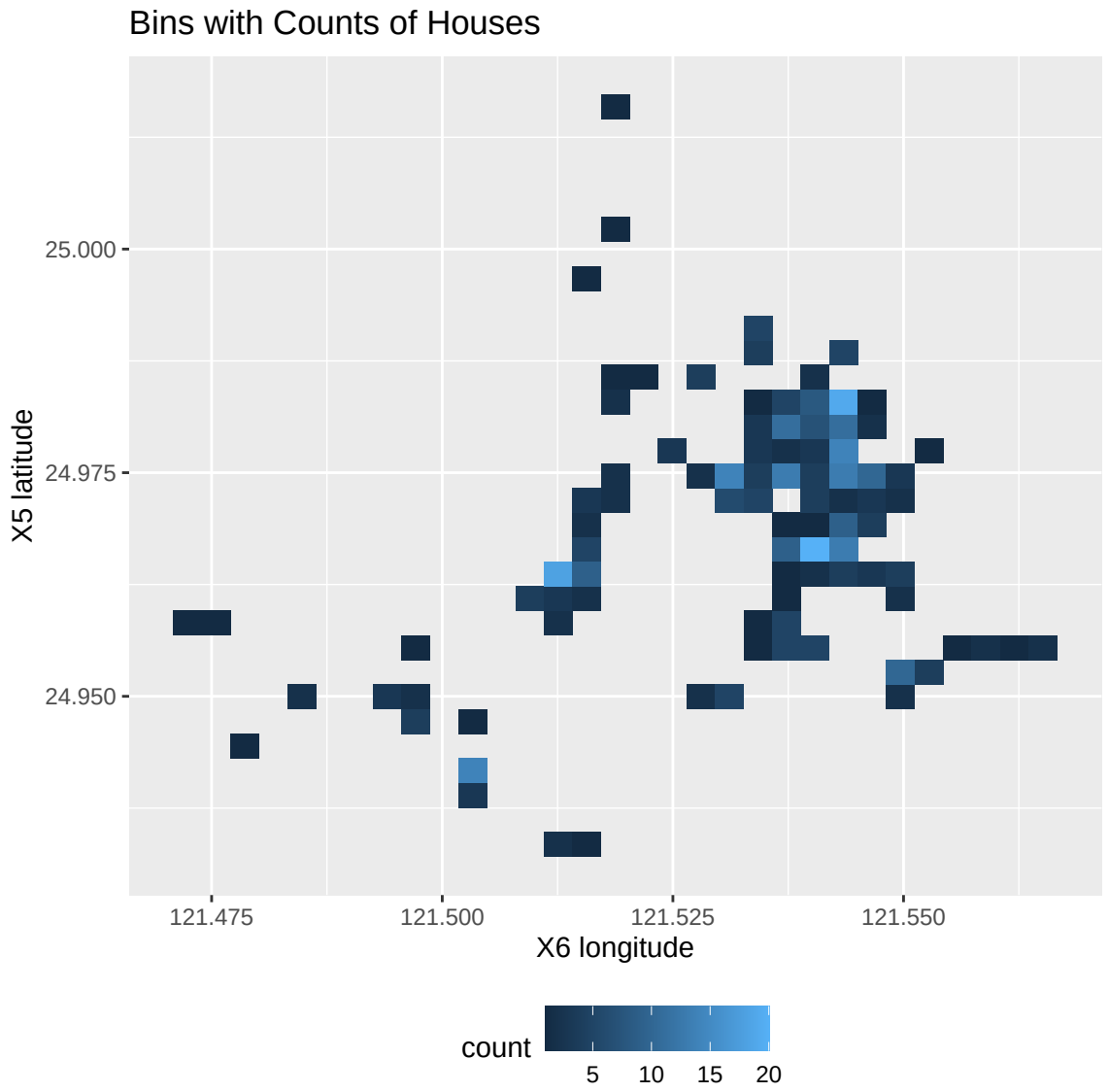

Stores
25.000
24.975
24.950
X5 latitude
121.475
121.500
121.525
121.550
X6 longitude
X4 number of convenience stores
0.0
2.5
5.0
7.5
10.0

```
##
## [[4]]
```




```
##
## [[5]]
## NULL
```

```
plotBins
```




```
par(old.par)

#
##
```

Initial takeaways from lat / long plot: The `lat` and `long` parameter cannot be used alone or by themselves. They need to be used in combination, as pairs. But they are one random variable, together. The challenge is how to define such one single random varable. One possible way to combine them is by creating some type of bins and then a dummy variable associated with each bin. An extended analysis may require access to additional databases and see street level differentiation between the houses. RIght now, I am interested on where each house is located. A geolocation within a vicinity that identifies the house and differentiates it. It would be nice to have a type of address, some level of a zip code even.

Here is my approach to handle this geolocation random variable. I create geographical bins of various sizes. Think of it a city blocks or a cluster of city block, or even a squared cluster in an area.

THe values of `lat/long` are strange. A fraction of a degree moves a long distance. I will need to transform those values without losing their information. I will start by creating two new columns: `newLat` and `newLong`.

The idea is to move latitude and longitude to a Cartesian chart that starts with (0, 0) or (`newLong` = 0, `newLat`=0). Then I will do a simply scale to fit my grid in a 10 by 10 chart. Keep it simple.

By doing so, the range say from south to north in the current latitude of 24.93 degrees to 25.01, spanning 0.08 degrees will roughly become 10 new points. Similarly, on the longitude or horizontal direction we are moving from 121.47 to 121.57, spanning 0.1 degrees.

Let's recall that for the latitude in New Taipei, 1 degree is 111 Km, and for the longitude 1 degree is around 100 Km (102 according to noaa.gov).

Therefore, in our new dimensions, 10 points of either direction newLat or newLong will be around 10 Km to think of these dimensions in human terms.

And that makes sense. If we measure the distance form the point farthest to the south (24.93207) to the point farthest to the north (25.01459), using the calculator from nooa.gov, we get 9 Km.

And we now measure the distance from the point farthest to the west (121.47353) to the point farthest to the east (121.56627) we also get 9 Km.

Therefore, I will make a new square 10 Km x 10 Km. And 1 point will be equal to 1 Km for simplicity in this analysis.

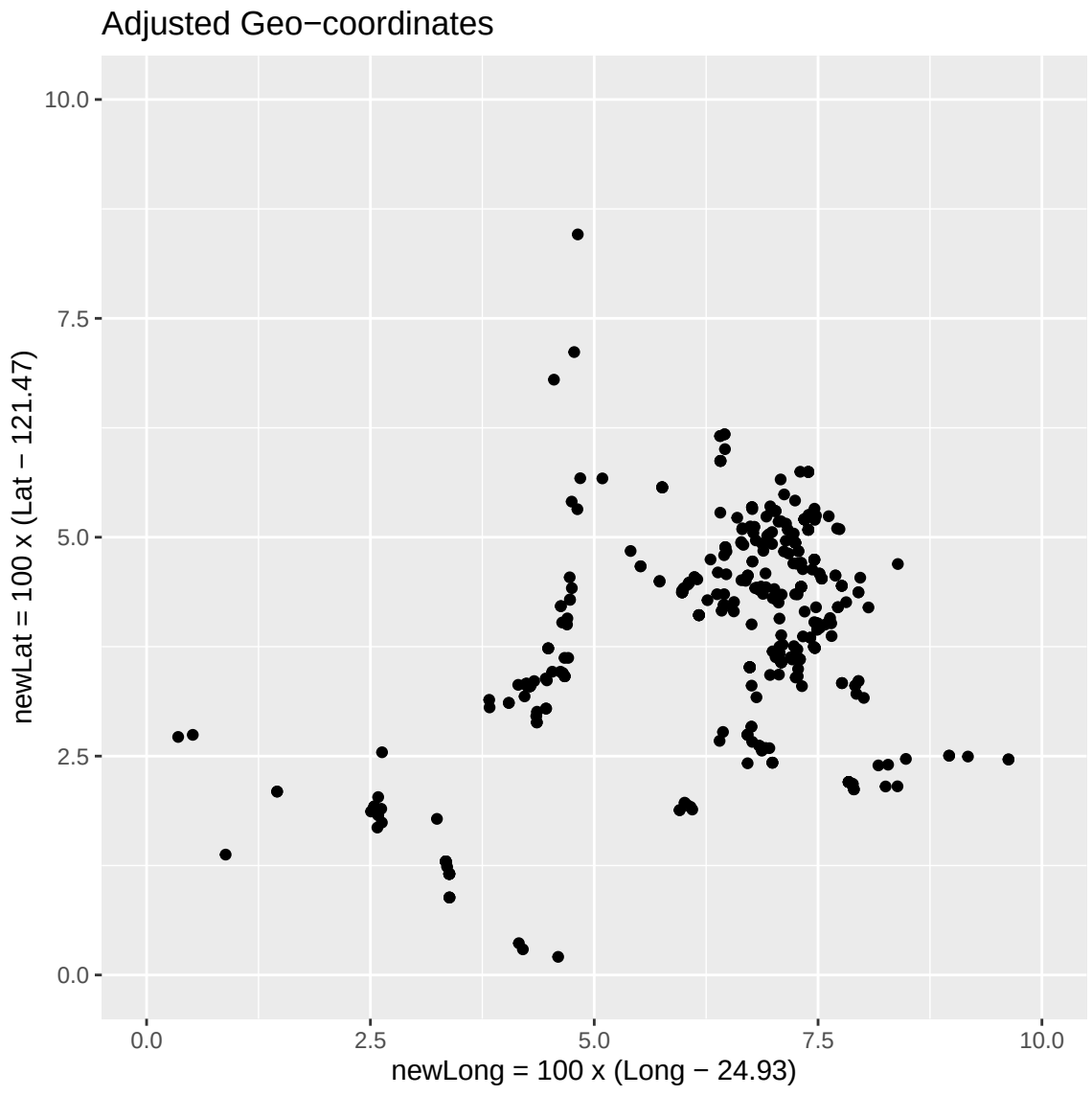
Geo bin changes

```
##
#
#

# Make two newcolumns, scaled after the lattitude and longitude.
# But make the data fall under a 100 x 100 grid

# Subtract the number just below the smallest latitude and multiply by 100
reDat$newLat <- 100*(reDat$`X5 latitude` - 24.93)

# Subtract the number just below the smallest longitude and multiply by 100
reDat$newLong <- 100*(reDat$`X6 longitude` - 121.470)

# Create new plots using adjusted geo-coordinates
#
old.par <- par(mfrow=c(1,2),ps=16)

plotNewSpatial <- vector("list", length=4)

#
# Plot and notice the date falls within a 10 x 10 grid
plotNewSpatial [[1]] <- ggplot(reDat,
       aes( x = newLong,
            y = newLat)) +
  xlim(0, 10) + ylim(0,10) +
  geom_point() +
  labs(title = "Adjusted Geo-coordinates",
       x = "newLong = 100 x (Long - 24.93)",
       y = "newLat = 100 x (Lat - 121.47)")

# Now lets create a new column called geobin10x10 for 10 by 10.
```

```
# This formula will create 100 bins from 0 to 99. I prefer to start with 0.
# The way I will do this is by going vertically, from left bottom (0, 0) to top (0, 9)
# Always start from the bottom, and then go left to right.
#
# Each of the following numbers is a geobin.
# I am in essence creating numeric dummy variables for each of my geobins
#
# 9 19 29 .. 99
# . .. .. .. ..
# . .. .. .. ..
# 2 12 22 .. 92
# 1 11 21 .. 91
# 0 10 20 .. 90
#
# That pattern follows this formula: geobin# = newLat + 10(newLong)
#
# Note the 10 is really square root of the number of geobins.
#
# So a general formula is:
#
# geobin# = new Lat + sqrt(number of geobins) x (newLong)
#
# Also, make them integers and roundup. Mathematically better.
#

# 100 bins 10 x 10: Bins are 1 Km x 1 Km
reDat$geobin10x10 <- as.integer(round(reDat$newLat + (10 * reDat$newLong), digits = 0))

plotNewSpatial [[2]] <- ggplot(reDat,
       aes( x = newLong,
            y = newLat,
            colour = geobin10x10)) +
  stat_bin2d(bins=10) +
  theme(legend.position = "bottom") +
  labs(title = "Bins 10 x 10 (~1 Km x 1 Km)",
       x = "newLong = 100 x (Long - 24.93)",
       y = "newLat = 100 x (Lat - 121.47)")

# 400 bins 20 x 20: Bins are 0.5 Km x 0.5 Km
reDat$geobin20x20 <- as.integer(round(reDat$newLat + (20 * reDat$newLong), digits = 0))

plotNewSpatial [[3]] <- ggplot(reDat,
       aes( x = newLong,
            y = newLat,
            colour = geobin20x20)) +
  stat_bin2d(bins=20) +
  theme(legend.position = "bottom") +
  labs(title = "Bins 20 x 20 (~0.5 Km x 0.5 Km)",
       x = "newLong = 100 x (Long - 24.93)",
       y = "newLat = 100 x (Lat - 121.47)")

# 2500 bins 50 x 50: Bins are 0.2 Km x 0.2 Km or 200 meter by 200 meters.
reDat$geobin50x50 <- as.integer(round(reDat$newLat + (50 * reDat$newLong), digits = 0))
```

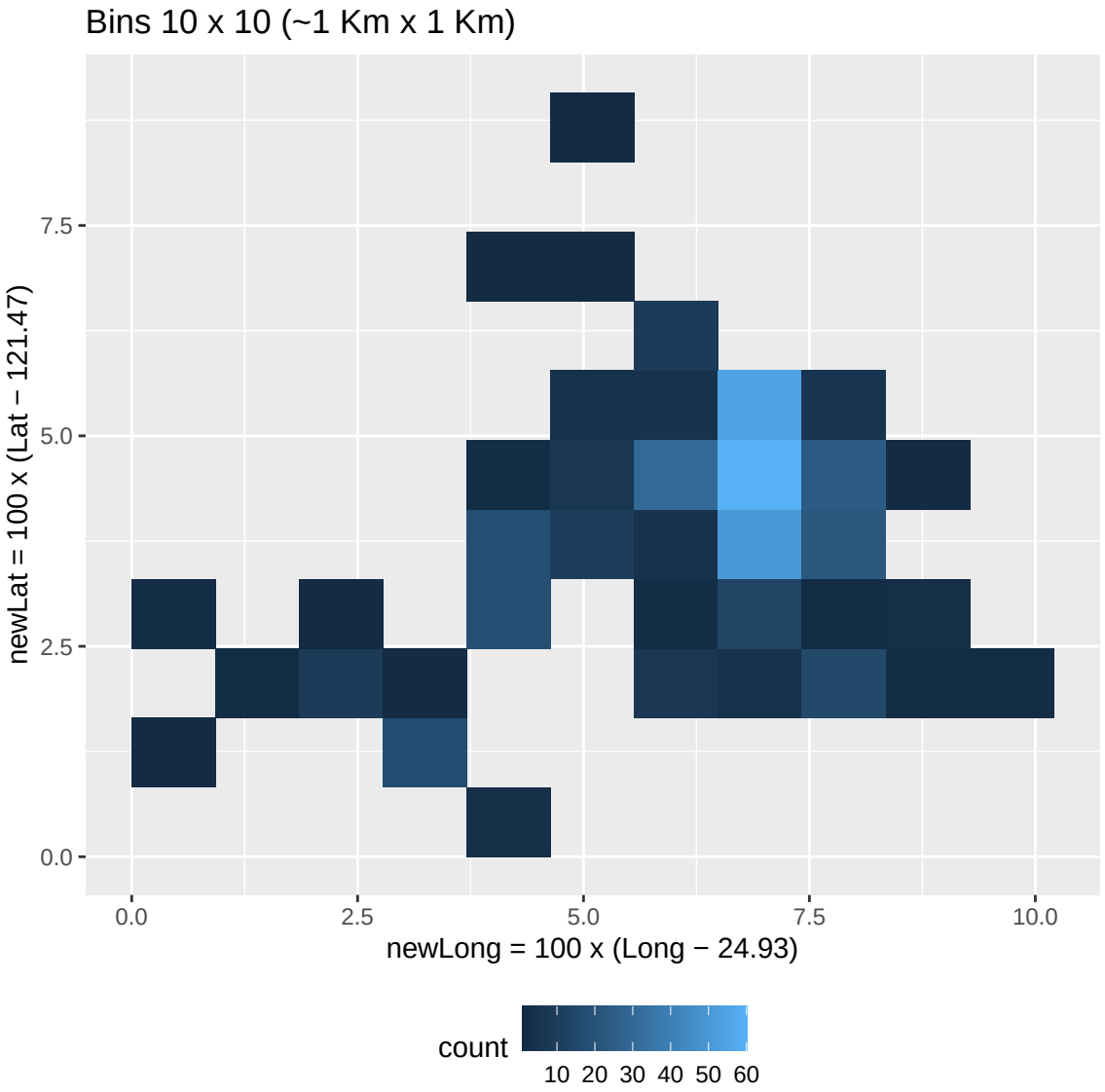
```
plotNewSpatial [[4]] <- ggplot(reDat,
       aes( x = newLong,
            y = newLat,
            colour = geobin50x50)) +
  stat_bin2d(bins=50) +
  theme(legend.position = "bottom") +
  labs(title = "Bins 20 x 20 (~0.2 Km x 0.2 Km)",
       x = "newLong = 100 x (Long - 24.93)",
       y = "newLat = 100 x (Lat - 121.47)")

plotNewSpatial
```

```
## [[1]]
```



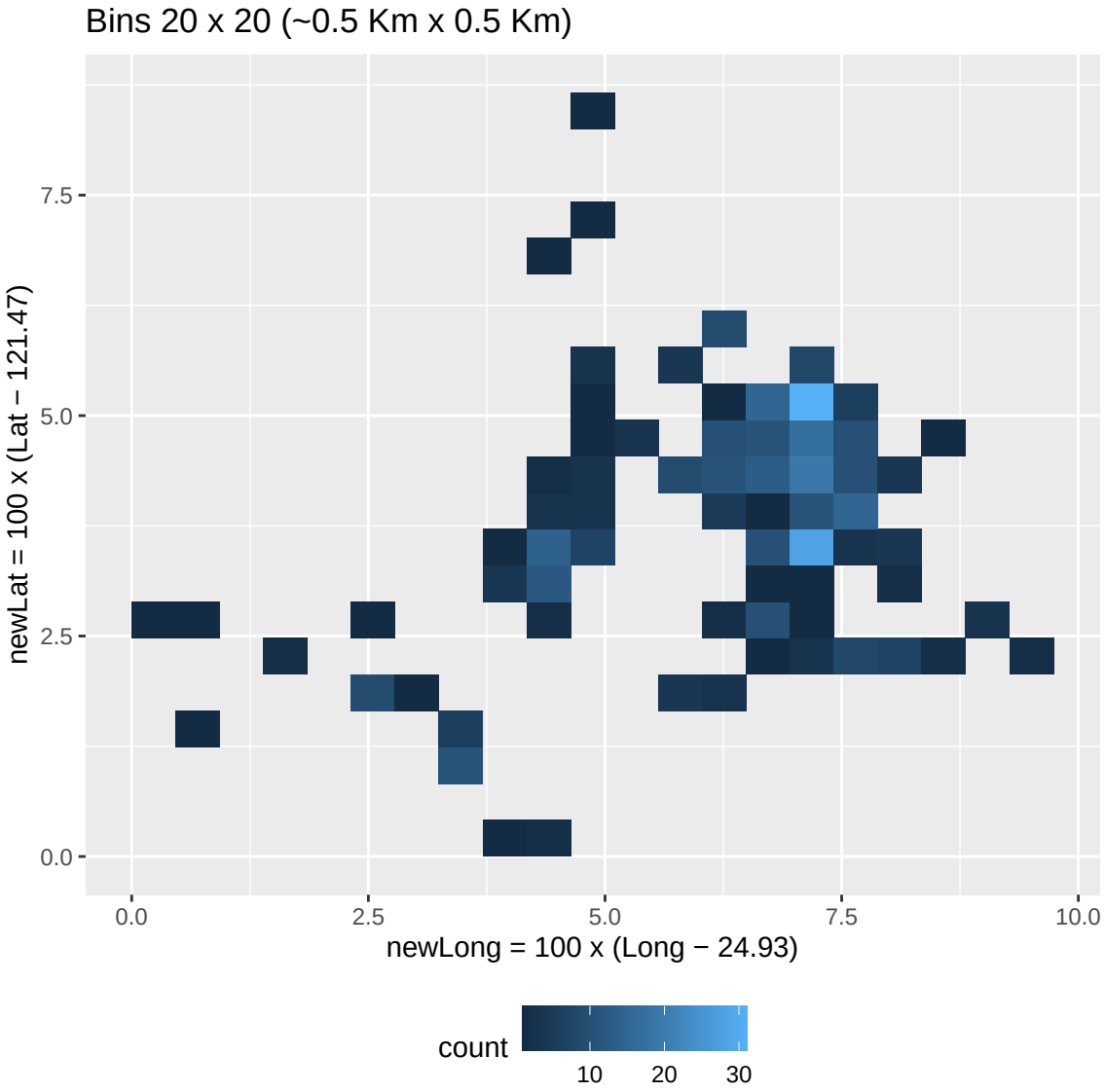

```
##
## [[2]]
```

```
## Warning: The following aesthetics were dropped during statistical transformation: colour
## i This can happen when ggplot fails to infer the correct grouping structure in
##   the data.
## i Did you forget to specify a 'group' aesthetic or to convert a numerical
##   variable into a factor?
```

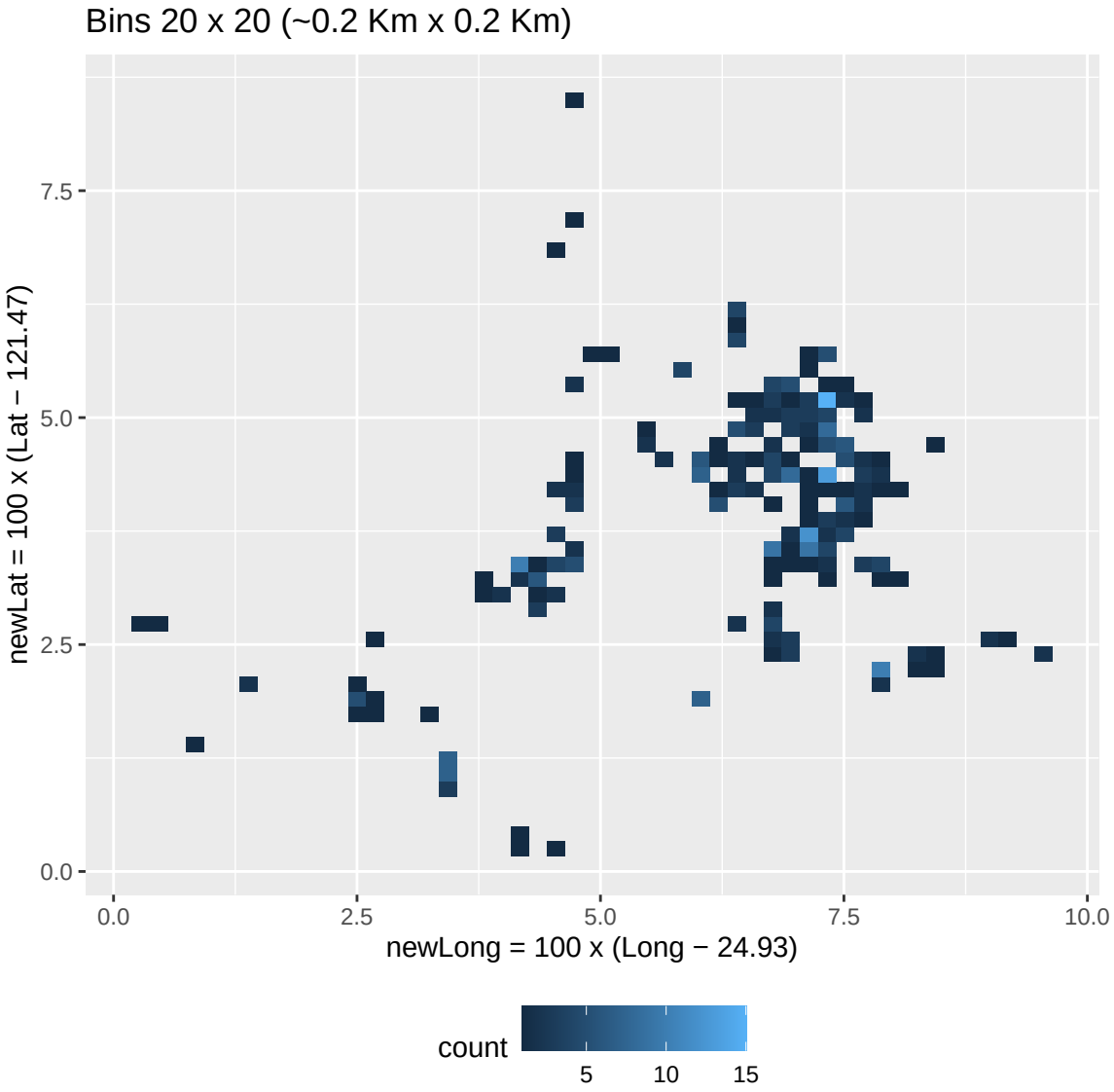



```
##
## [[3]]
```

```
## Warning: The following aesthetics were dropped during statistical transformation: colour
## i This can happen when ggplot fails to infer the correct grouping structure in
##   the data.
## i Did you forget to specify a 'group' aesthetic or to convert a numerical
##   variable into a factor?
```




```
##
## [[4]]

## Warning: The following aesthetics were dropped during statistical transformation: colour
## i This can happen when ggplot fails to infer the correct grouping structure in
##   the data.
## i Did you forget to specify a ‘group‘ aesthetic or to convert a numerical
##   variable into a factor?
```




```
par(old.par)

# Inspect our the updated dataset
str(reDat)
```

```
## 'data.frame':    414 obs. of  12 variables:
##  $ X1 transaction date                   : num  2013 2013 2014 2014 2013 ...
##  $ X2 house age                          : num  32 19.5 13.3 13.3 5 7.1 34.5 20.3 31.7 17.9 ...
##  $ X3 distance to the nearest MRT station: num  84.9 306.6 562 562 390.6 ...
##  $ X4 number of convenience stores       : num  10 9 5 5 5 3 7 6 1 3 ...
##  $ X5 latitude                           : num  25 25 25 25 25 ...
##  $ X6 longitude                          : num  122 122 122 122 122 ...
##  $ Y house price of unit area            : num  37.9 42.2 47.3 54.8 43.1 32.1 40.3 46.7 18.8 22.1 ...
##  $ newLat                                : num  5.3 5.03 5.75 5.75 4.94 ...
##  $ newLong                               : num  7.02 6.95 7.39 7.39 7.25 ...
##  $ geobin10x10                           : int  76 75 80 80 77 46 71 77 17 49 ...
##  $ geobin20x20                           : int  146 144 154 154 150 88 138 150 31 93 ...
```

```
##  $ geobin50x50                          : int  356 353 375 375 367 216 337 366 75 228 ...
```

```
summary(reDat)
```

```
##  X1 transaction date  X2 house age    X3 distance to the nearest MRT station
##  Min.   :2013        Min.   : 0.000   Min.   :  23.38
##  1st Qu.:2013        1st Qu.: 9.025   1st Qu.: 289.32
##  Median :2013        Median :16.100   Median : 492.23
##  Mean   :2013        Mean   :17.713   Mean   :1083.89
##  3rd Qu.:2013        3rd Qu.:28.150   3rd Qu.:1454.28
##  Max.   :2014        Max.   :43.800   Max.   :6488.02
##  X4 number of convenience stores  X5 latitude     X6 longitude
##  Min.   : 0.000                  Min.   :24.93   Min.   :121.5
##  1st Qu.: 1.000                  1st Qu.:24.96   1st Qu.:121.5
##  Median : 4.000                  Median :24.97   Median :121.5
##  Mean   : 4.094                  Mean   :24.97   Mean   :121.5
##  3rd Qu.: 6.000                  3rd Qu.:24.98   3rd Qu.:121.5
##  Max.   :10.000                  Max.   :25.01   Max.   :121.6
##  Y house price of unit area     newLat         newLong       geobin10x10
##  Min.   :  7.60             Min.   :0.207   Min.   :0.353   Min.   : 6.00
##  1st Qu.: 27.70             1st Qu.:3.300   1st Qu.:5.809   1st Qu.:62.00
##  Median : 38.45             Median :4.110   Median :6.863   Median :73.00
##  Mean   : 37.98             Mean   :3.903   Mean   :6.336   Mean   :67.29
##  3rd Qu.: 46.60             3rd Qu.:4.745   3rd Qu.:7.330   3rd Qu.:78.00
##  Max.   :117.50             Max.   :8.459   Max.   :9.627   Max.   :99.00
##   geobin20x20     geobin50x50
##  Min.   : 10.0   Min.   : 20.0
##  1st Qu.:121.0   1st Qu.:295.5
##  Median :141.5   Median :348.0
##  Mean   :130.6   Mean   :320.7
##  3rd Qu.:151.0   3rd Qu.:371.0
##  Max.   :195.0   Max.   :484.0
```

```
#
##
```

Binning makes a lot of sense. Note the `head(reDat)` command displays additional columns for the new binning variables for 'geobinNxN" (NxN is the number of geobins). Simple inspection indicates that binning is correctly done. Comparing the bin numbers for between geobin10x10 and geobin20x20, the bin number for the 20x20 case is roughly two times the bin number of the 10x10 case. And so is the comparison between 10x10 and 50x50, with the bin number of the latter being roughly 5 time that one of the 10x10 case.

The `summary()` also demonstrates that binning is done correctly. Let us take a close look at their min / max figures. The 10x10 case goes from 6 to 99, while 20x20 case goes from 10 to 195, and the 50x50 case goes from 20 to 484.

The 50x50 case is based on bins that roughly measure 200 meters by 200 meters. That is equivalent, in many cities, to around 2 block by 2 blocks, that amounts to a square containing 4 blocks.

The 20x20 case is based on bins that roughly measure 500 meters by 500 meters. That is equivalent, in many cities, to around 5 block by 5 blocks, that amounts to a square containing 25 blocks. Additionally, for as large city like New Taipei, the dimension of 500m x 500m is 0.25 of a Km^2.

To think about demographics, the 20x20 case forms bins that are not too far off from a dimension of a zip code in a large city like New York City. New Taipei covers a very large area and

many zip codes. (reference for area https://www.britannica.com/place/Tai-pei and for zip codes https://www.post.gov.tw/post/internet/U_english/index.jsp?page_type=2&search_city=%E6%96%B0%E5%8C%97%E5%B8%82&search_text=New%20Taipei%20City&topage=1&ID=24020901).

Similarly, the 10 x 10 case will contain squares with around 100 blocks. That square size may be a bit too big for the data analysis, but I will leave it in for now.

In the following step, I drop two of the geobin columns to avoid collinearity and redundancy. I am keeping the 20x20 bins. That analysis is coming up.

References

- Dataset: https://archive.ics.uci.edu/ml/datasets/Real+estate+valuation+data+set
- CSCI E-63C slides and Homework from Professors Andrey Sivachenko, PhD and Victor A. Farutin, PhD
- CSCI E-63C "Week-6-TF-Section: Beyond_linearity" by Jose M Pena
- CSCI E-63C "Week-5-TF-Section: Subset + Ridge + Lasso" by Alos Diallo
- CSCI E-63C "Week-4-TF-Section: Notes taken during recording by Sihong Ma
- Book "An Introduction to Statistical Learning with Applications in R" (ISLR) by Gareth James et al
- Book "R for Data Science" by Hadley Wickham and Garrett Grolemund
- Book "R Graphics Cookbook" by Winston Chang
- LinkedIn Learning "Wrangling and Visualizing Data" by Barton Poulson
- Stackoverflow reliable online coding tips: https://stackoverflow.com/
- National Oceanic and Atmospheric Administration: https://www.nhc.noaa.gov/gccalc.shtml
- Maps: https://slcladal.github.io/maps.html
- Britannica: https://www.britannica.com/place/Taipei
- Taiwan Postal Service https://www.post.gov.tw/post/internet/U_english/index.jsp?ID=24020501
- R help() function: Use this one a lot!!

Pledge of integrity: The code presented in this midterm is based on material from CSCI E-63C plus additional sources listed in the References. My code here is intended for my own own use and reference. I did not discuss it or shared it with any student in class or anyone outside of my class. I am not posting or sharing any of the CSCI solutions or excerpts with anyone.

Example 2: WPP

Load the data

Source United Nations WPP

```
# https://population.un.org/wpp/Download/Files/1_Indicators%20(Standard)/\
# EXCEL_FILES/1_General/WPP2022_GEN_F01_DEMOGRAPHIC_INDICATORS_REV1.xlsx
# wpp_raw <- read_excel('data/WPP2022_GEN_F01_DEMOGRAPHIC_INDICATORS_COMPACT_REV1.xlsx',
#                   sheet = 'Estimates',
#                   skip = 16,
#                   col_types = 'text')
# wpp_raw <- read.csv('../data/wpp.csv', check.names=FALSE)
```

Remove rows -> build new dataframe

```
# wpp <- wpp_raw %>% filter((Type == 'Country/Area') & Year > 1999) %>%
#   dplyr::select(-c('Variant',
#              'Notes',
#              'Location code',
#              'SDMX code**',
#              'Parent code',
#              'Total Population, as of 1 January (thousands)',
#              'Female Population, as of 1 July (thousands)'))
# dim(wpp)
# head(wpp)
```

Data types

Now we need to make numeric columns be type numeric.

```
# # Remove white spaces from numeric columns
# # https://www.geeksforgeeks.org/remove-all-whitespace-in-each-dataframe-column-in-r/
# wpp[, 7:58] <- as.data.frame(apply(wpp[, 7:58], 2, function(x) gsub("\\s+", "", x)))
#
# # Columns 7 to 58 should be numeric
# # Tips from:
# # https://stackoverflow.com/questions/2288485/how-to-convert-a-data-frame-column-to-numeric-type
# wpp[, 7:58] <- sapply(wpp[, 7:58], as.numeric)
#
# # Year should actually be Date as this:
# # The challenge with a Date is it will try to pin down the exact month and day too.
# # Even if the format is Year only, underneath the data will have exact days
# # wpp$Year <- as.Date(wpp$Year, '%Y')
#
# # But for now, leave it as numeric for simplicity.
# wpp$Year <- as.numeric(wpp$Year)
```

Build a clean WPP dataframe

First capture existing column names into a vector Then create a new vector with new names.

Save both so we can always map to the original names, just in case we need them

```
# wpp_columns_orig <- colnames(wpp)
# wpp_columns_new <- c('index', 'country', 'ISO3', 'ISO2', 'type', 'year',
#                      'pop', 'pop_m', 'pop_density', 'pop_sex_ratio', 'med_age',
#                      'natural_change', 'natural_change_rate', 'pop_change',
#                      'pop_growth_rate', 'pop_doubling', 'births', 'births_w15to19',
#                      'birth_rate', 'fertility_rate', 'net_reproduction_rate',
#                      'mean_age_childbearing', 'sex_ratio_birth', 'tot_deaths',
#                      'male_deaths', 'female_deaths', 'death_rate',
#                      'life_exp', 'life_exp_m', 'life_exp_f',
#                      'life_exp_15', 'life_exp_15_m', 'life_exp_15_f',
#                      'life_exp_65', 'life_exp_65_m', 'life_exp_65_f',
```

```
#                       'life_exp_80', 'life_exp_80_m', 'life_exp_80_f',
#                       'infant_deaths', 'under_five_mortality',
#                       'live_births', 'deaths_under_5', 'mortality_rate_under_5',
#                       'mortality_40', 'mortality_40_m', 'mortality_40_f',
#                       'mortality_60', 'mortality_60_m', 'mortality_60_f',
#                       'mortality_15_50', 'mortality_15_50_m', 'mortality_15_50_f',
#                       'mortality_15_60', 'mortality_15_60_m', 'mortality_15_60_f',
#                        'net_migrants', 'net_mig_rate'
#                       )
#
# wpp_colomns_table <- data.frame(wpp_columns_orig, wpp_columns_new)
# wpp_colomns_table
```

Rename wpp columns

```
# colnames(wpp) <- wpp_columns_new
# head(wpp)
```

Load the data

Source United Nations WPP

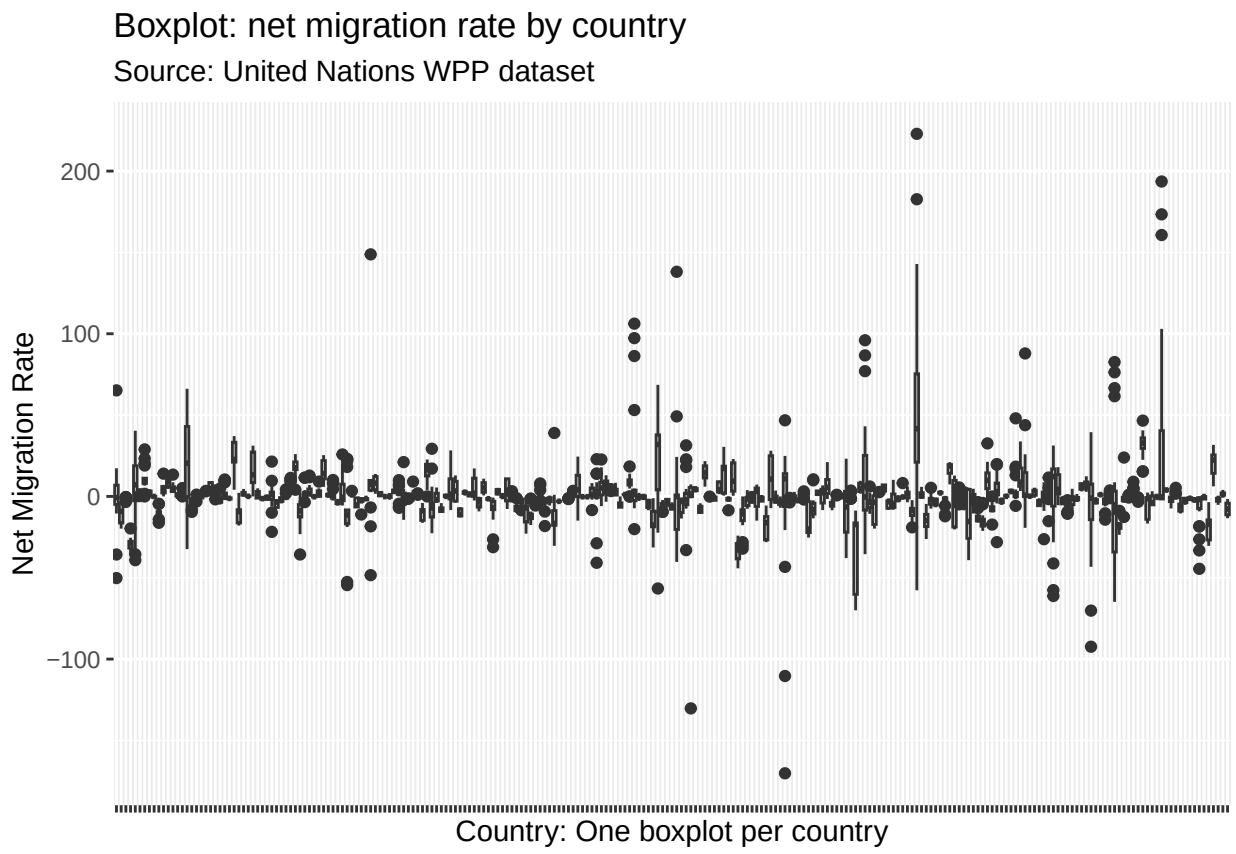
```
# From UN World Bank migration case study (March 2023)

wpp_wb <- read.csv('../data/wpp_wb.csv', check.names=FALSE)
```

Exploratory Data Analysis

Boxplot

```
wpp_wb %>% ggplot(aes(x = country, y = net_mig_rate)) +
  geom_boxplot() +
  ggtitle('Boxplot: net migration rate by country', 'Source: United Nations WPP dataset') +
  xlab('Country: One boxplot per country') +
  ylab('Net Migration Rate') +
  theme(axis.text.x = element_blank())
```

```
## Warning: Removed 22 rows containing non-finite values (‘stat_boxplot()‘).
```



Get world countried from GitHub

https://gist.github.com/cpl/3dc2d19137588d9ae202d67233715478

```
# Reference https://developers.google.com/public-data/docs/canonical/countries_csv
# Reference (has dups) https://gist.github.com/tadast/8827699
# Reference (w/o dups) https://gist.github.com/cpl/3dc2d19137588d9ae202d67233715478

w <- read.csv('../data/countries_codes_and_coordinates.csv')
w <- w %>% rename(country = Country,
                          ISO2 = Alpha.2.code,
                          ISO3 = Alpha.3.code,
                          group = Numeric.code,
                          lat = Latitude..average.,
                          long = Longitude..average.)

w <- w %>% select(ISO3, group, lat, long)

# Strip leading spaces
# https://www.geeksforgeeks.org/remove-all-whitespace-in-each-dataframe-column-in-r/
w <- as.data.frame(apply(w, 2, function(x) gsub("\\s+", "", x)))

w[, 3:4] <- sapply(w[, 3:4], as.numeric)

head(w)
```

```
##   ISO3 group      lat   long
## 1  AFG     4  33.0000   65.0
## 2  ALB     8  41.0000   20.0
## 3  DZA    12  28.0000    3.0
## 4  ASM    16 -14.3333 -170.0
## 5  AND    20  42.5000    1.6
## 6  AGO    24 -12.5000   18.5
```

Merg geo long lat from rnaturalearth to wpp

```
# wpp_geo <- wpp %>% left_join(w)
# wpp_geo <- wpp_wb %>% dplyr::filter(year == 2015) %>%
#   dplyr::select(ISO3, net_migrants, net_mig_rate, net_mig_rate_mean, emigrates, migration_swings) %>%
#   left_join(w)
wpp_geo <- wpp_wb %>% group_by(ISO3, country) %>%
  summarize(net_mig_mu = mean(net_migrants),
            net_mig_rt_mu = mean(net_mig_rate, na.rm = TRUE),
            log_abs_net_mig_rt = log(abs(net_mig_rt_mu))) %>%
  left_join(w, by = join_by(ISO3))
```

```
## `summarise()` has grouped output by 'ISO3'. You can override using the
## `.groups` argument.
```

Geo plots

First a world map with dots on countries from WPP

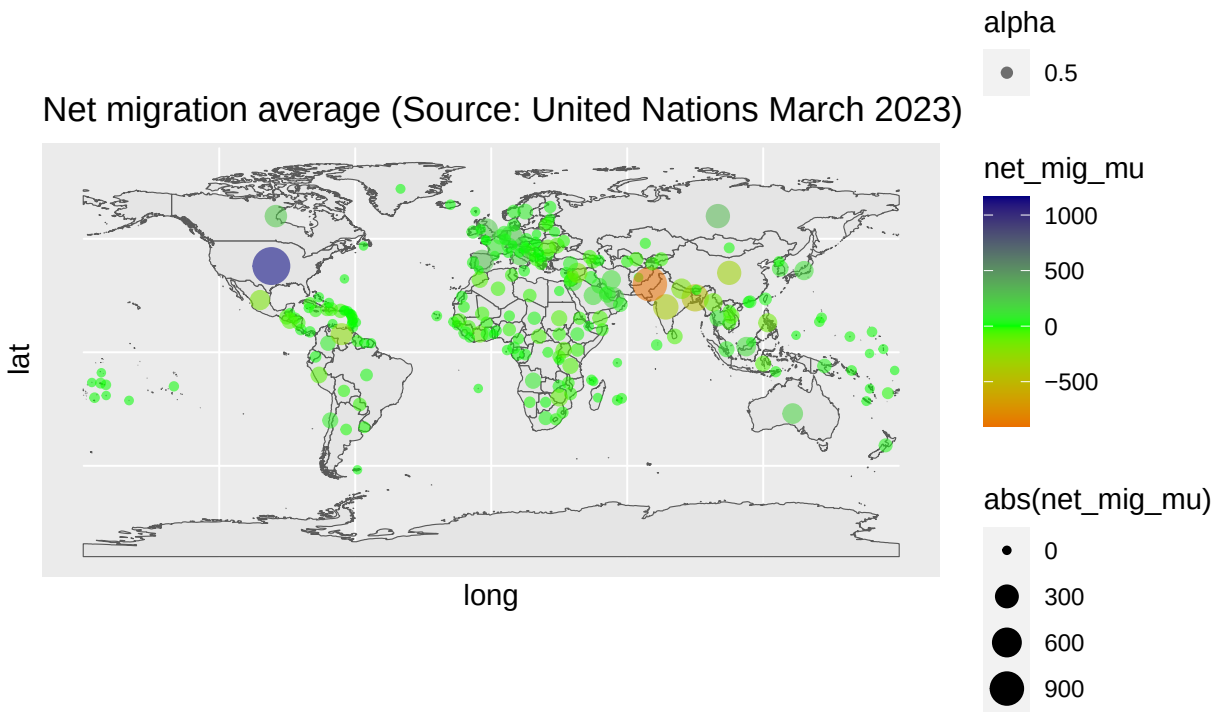
```
# world <- ne_countries(scale = "medium", returnclass = "sf")
#
# plotMapWPP <- ggplot(world) +
#   geom_sf() +
#   geom_point(data = wpp_geo,
#              aes( x = `long`,
#                   y = `lat`,
#                   size = abs(net_mig_mu),
#                   alpha = 0.5), col = "red") +
#   labs(title = "Net migration average (Source: United Nations March 2023)")
#
# plotMapWPP
```

Migration rate map: Net

```
plotMapWPP <- ggplot(world) +
  geom_sf() +
  geom_point(data = wpp_geo,
             aes( x = `long`,
                  y = `lat`,
```

```
                  size = abs(net_mig_mu),
                  alpha = 0.5,
                  color = net_mig_mu)) +
#  scale_colour_gradient2() +
  scale_colour_gradient2(low = "red", mid='green', high = "navyblue", na.value = NA) +
  labs(title = "Net migration average (Source: United Nations March 2023)")

plotMapWPP
```

```
## Warning: Removed 7 rows containing missing values (`geom_point()`).
```

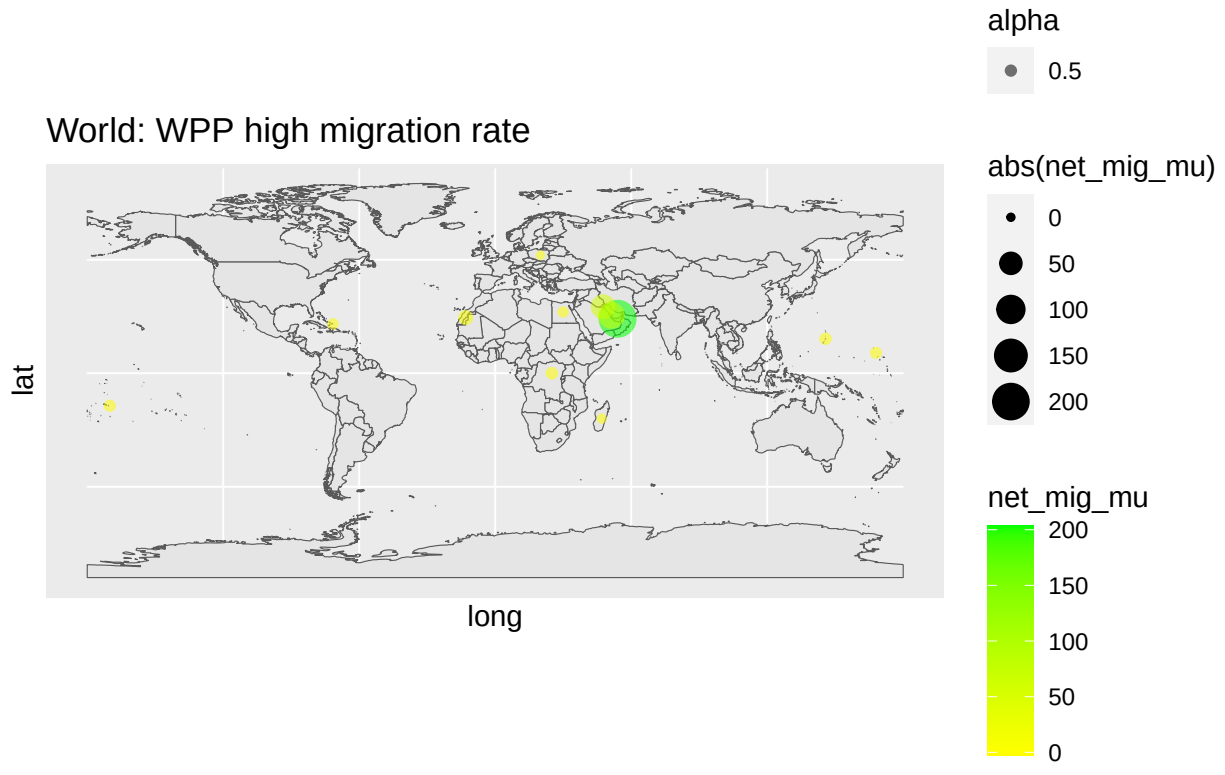


Very high migration rate, positive or negative

```
wpp_geo_hot <- wpp_geo %>% filter(abs(log_abs_net_mig_rt) > 3)

plotMapWPP <- ggplot(world) +
  geom_sf() +
  geom_point(data = wpp_geo_hot,
             aes( x = `long`,
                  y = `lat`,
                  size = abs(net_mig_mu),
                  alpha = 0.5,
                  color = net_mig_mu)) +
```

```
#  scale_colour_gradient2() +
  scale_colour_gradient2(low = "red", mid='yellow', high = "green", na.value = NA) +
  labs(title = "World: WPP high migration rate")

plotMapWPP
```

```
## Warning: Removed 1 rows containing missing values (`geom_point()`).
```




```
wpp_geo_hot$country
```

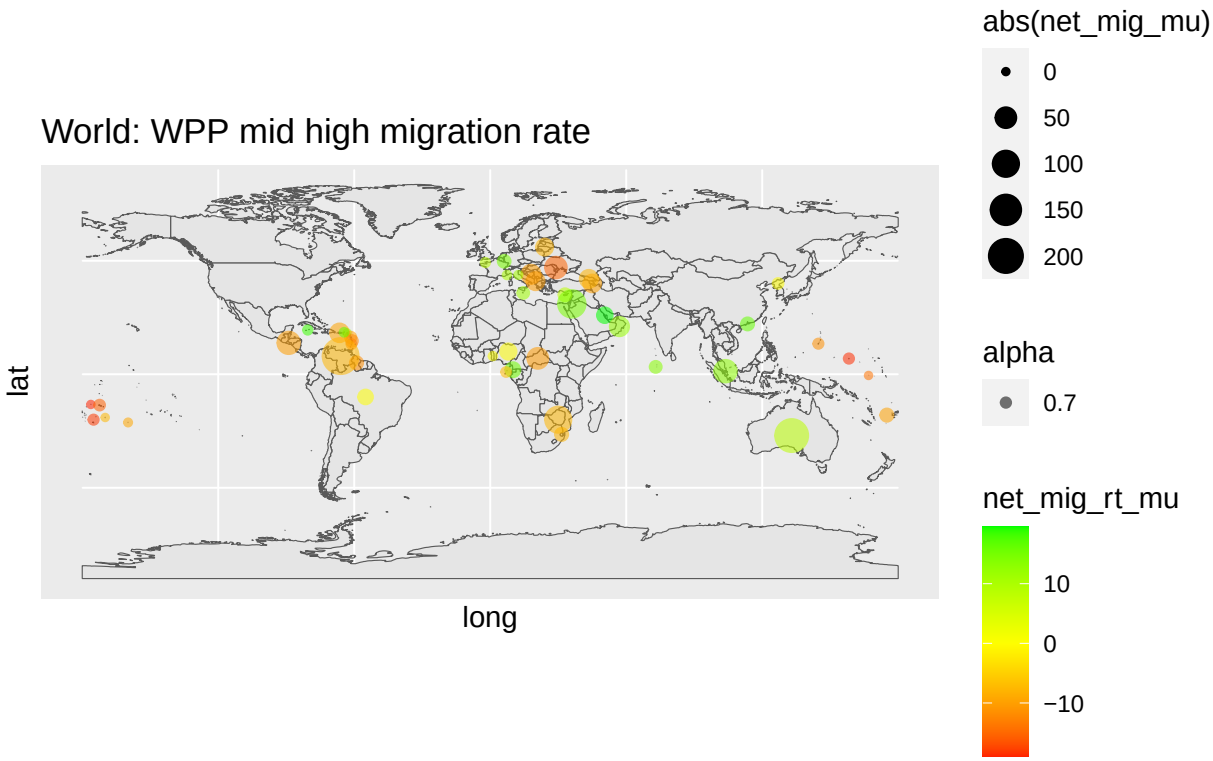
```
##  [1] "United Arab Emirates"             "American Samoa"
##  [3] "Bonaire, Sint Eustatius and Saba" "Democratic Republic of the Congo"
##  [5] "Egypt"                            "Western Sahara"
##  [7] "Kuwait"                           "Madagascar"
##  [9] "Marshall Islands"                 "Northern Mariana Islands"
## [11] "Poland"                           "Qatar"
## [13] "Turks and Caicos Islands"
```

Medium high migration rate, positive or negative

```
wpp_geo_midhigh <- wpp_geo %>% filter((abs(log_abs_net_mig_rt) < 3)
                                      & (abs(log_abs_net_mig_rt) > 2))
```

```
plotMapWPP <- ggplot(world) +
  geom_sf() +
  geom_point(data = wpp_geo_midhigh,
             aes( x = `long`,
                  y = `lat`,
                  size = abs(net_mig_mu),
                  alpha = 0.7,
                  color = net_mig_rt_mu)) +
#  scale_colour_gradient2() +
  scale_colour_gradient2(low = "red", mid='yellow', high = "green", na.value = NA) +
  labs(title = "World: WPP mid high migration rate")

plotMapWPP
```

```
## Warning: Removed 5 rows containing missing values (‘geom_point()‘).
```



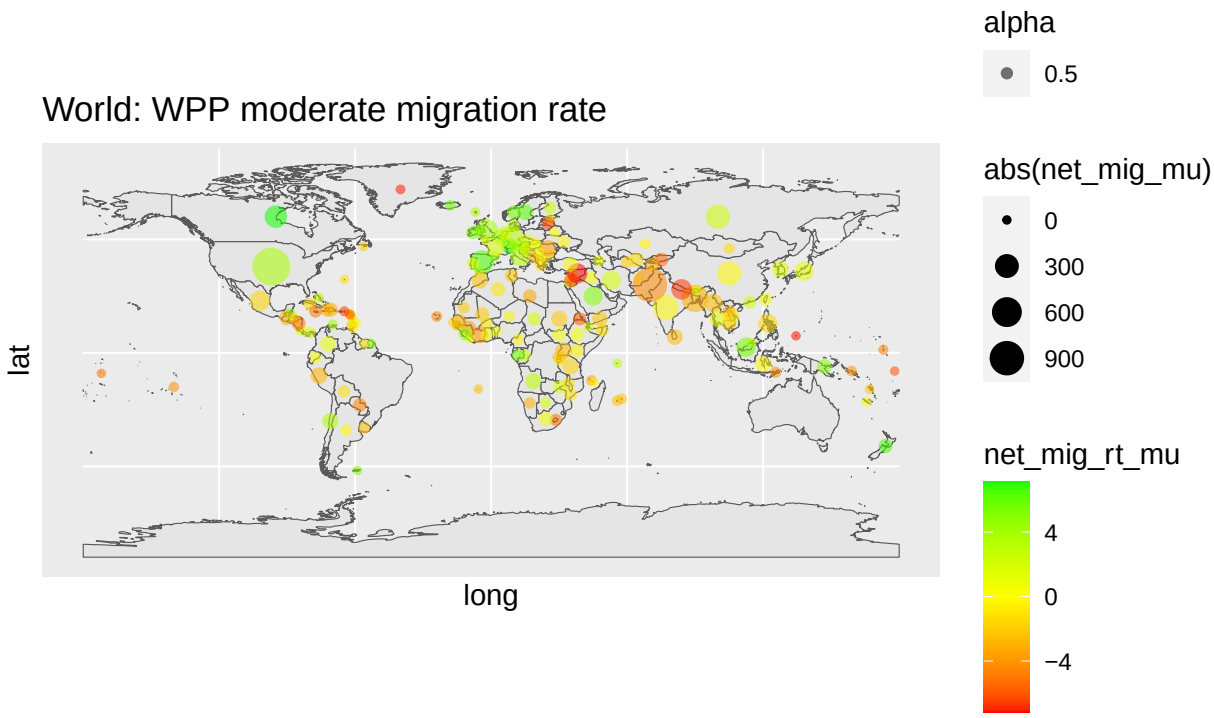

```
wpp_geo_midhigh$country
```

```
##  [1] "Anguilla"                  "Albania"
##  [3] "Armenia"                   "Australia"
##  [5] "Bahrain"                   "Bosnia and Herzegovina"
##  [7] "Saint Barthélemy"          "Brazil"
##  [9] "Central African Republic"  "Cook Islands"
```

```
## [11] "Curaçao"                              "Cayman Islands"
## [13] "Cyprus"                               "Fiji"
## [15] "Micronesia (Fed. States of)"          "Georgia"
## [17] "Guadeloupe"                           "Equatorial Guinea"
## [19] "Guam"                                 "Guyana"
## [21] "Jersey"                               "Jordan"
## [23] "Lithuania"                            "Luxembourg"
## [25] "China, Macao SAR"                     "Saint Martin (French part)"
## [27] "Monaco"                               "Republic of Moldova"
## [29] "Maldives"                             "Malta"
## [31] "Martinique"                           "Nigeria"
## [33] "Niue"                                 "Nauru"
## [35] "Oman"                                 "Puerto Rico"
## [37] "Dem. People's Republic of Korea"      "Singapore"
## [39] "El Salvador"                          "San Marino"
## [41] "Sao Tome and Principe"                "Eswatini"
## [43] "Sint Maarten (Dutch part)"            "Togo"
## [45] "Tonga"                                "Saint Vincent and the Grenadines"
## [47] "Venezuela (Bolivarian Republic of)" "British Virgin Islands"
## [49] "Wallis and Futuna Islands"            "Samoa"
## [51] "Kosovo (under UNSC res. 1244)"        "Zimbabwe"
```

Moderate migration rate, positive or negative

```
wpp_geo_mod <- wpp_geo %>% filter((abs(log_abs_net_mig_rt) < 2))

plotMapWPP <- ggplot(world) +
  geom_sf() +
  geom_point(data = wpp_geo_mod,
             aes( x = `long`,
                  y = `lat`,
                  size = abs(net_mig_mu),
                  alpha = 0.5,
                  color = net_mig_rt_mu)) +
#  scale_colour_gradient2() +
  scale_colour_gradient2(low = "red", mid='yellow', high = "green", na.value = NA) +
  labs(title = "World: WPP moderate migration rate")

plotMapWPP
```




```
wpp_geo_mod$country
```

```
##   [1] "Aruba"                                "Afghanistan"
##   [3] "Angola"                               "Andorra"
##   [5] "Argentina"                            "Antigua and Barbuda"
##   [7] "Austria"                              "Azerbaijan"
##   [9] "Burundi"                              "Belgium"
##  [11] "Benin"                                "Burkina Faso"
##  [13] "Bangladesh"                           "Bulgaria"
##  [15] "Bahamas"                              "Belarus"
##  [17] "Belize"                               "Bermuda"
##  [19] "Bolivia (Plurinational State of)"     "Barbados"
##  [21] "Brunei Darussalam"                    "Bhutan"
##  [23] "Botswana"                             "Canada"
##  [25] "Switzerland"                          "Chile"
##  [27] "China"                                "Côte d'Ivoire"
##  [29] "Cameroon"                             "Congo"
##  [31] "Colombia"                             "Comoros"
##  [33] "Cabo Verde"                           "Costa Rica"
##  [35] "Cuba"                                 "Czechia"
##  [37] "Germany"                              "Djibouti"
##  [39] "Dominica"                             "Denmark"
##  [41] "Dominican Republic"                   "Algeria"
##  [43] "Ecuador"                              "Eritrea"
##  [45] "Spain"                                "Estonia"
```

```
##  [47] "Ethiopia"                          "Finland"
##  [49] "Falkland Islands (Malvinas)"       "France"
##  [51] "Faroe Islands"                     "Gabon"
##  [53] "United Kingdom"                    "Guernsey"
##  [55] "Ghana"                             "Gibraltar"
##  [57] "Guinea"                            "Gambia"
##  [59] "Guinea-Bissau"                     "Greece"
##  [61] "Grenada"                           "Greenland"
##  [63] "Guatemala"                         "French Guiana"
##  [65] "China, Hong Kong SAR"              "Honduras"
##  [67] "Croatia"                           "Haiti"
##  [69] "Hungary"                           "Indonesia"
##  [71] "Isle of Man"                       "India"
##  [73] "Ireland"                           "Iran (Islamic Republic of)"
##  [75] "Iraq"                              "Iceland"
##  [77] "Israel"                            "Italy"
##  [79] "Jamaica"                           "Japan"
##  [81] "Kazakhstan"                        "Kenya"
##  [83] "Kyrgyzstan"                        "Cambodia"
##  [85] "Kiribati"                          "Saint Kitts and Nevis"
##  [87] "Republic of Korea"                 "Lao People's Democratic Republic"
##  [89] "Lebanon"                           "Liberia"
##  [91] "Libya"                             "Saint Lucia"
##  [93] "Liechtenstein"                     "Sri Lanka"
##  [95] "Lesotho"                           "Latvia"
##  [97] "Morocco"                           "Mexico"
##  [99] "North Macedonia"                   "Mali"
## [101] "Myanmar"                           "Montenegro"
## [103] "Mongolia"                          "Mozambique"
## [105] "Mauritania"                        "Montserrat"
## [107] "Mauritius"                         "Malawi"
## [109] "Malaysia"                          "Mayotte"
## [111] "Namibia"                           "New Caledonia"
## [113] "Niger"                             "Nicaragua"
## [115] "Netherlands"                       "Norway"
## [117] "Nepal"                             "New Zealand"
## [119] "Pakistan"                          "Panama"
## [121] "Peru"                              "Philippines"
## [123] "Palau"                             "Papua New Guinea"
## [125] "Portugal"                          "Paraguay"
## [127] "State of Palestine"                "French Polynesia"
## [129] "Réunion"                           "Romania"
## [131] "Russian Federation"                "Rwanda"
## [133] "Saudi Arabia"                      "Sudan"
## [135] "Senegal"                           "Saint Helena"
## [137] "Solomon Islands"                   "Sierra Leone"
## [139] "Somalia"                           "Saint Pierre and Miquelon"
## [141] "Serbia"                            "South Sudan"
## [143] "Suriname"                          "Slovakia"
## [145] "Slovenia"                          "Sweden"
## [147] "Seychelles"                        "Syrian Arab Republic"
## [149] "Chad"                              "Thailand"
## [151] "Tajikistan"                        "Tokelau"
## [153] "Turkmenistan"                      "Timor-Leste"
```

```
## [155] "Trinidad and Tobago"              "Tunisia"
## [157] "Türkiye"                          "Tuvalu"
## [159] "China, Taiwan Province of China"  "United Republic of Tanzania"
## [161] "Uganda"                           "Ukraine"
## [163] "Uruguay"                          "United States of America"
## [165] "Uzbekistan"                       "United States Virgin Islands"
## [167] "Viet Nam"                         "Vanuatu"
## [169] "Yemen"                            "South Africa"
## [171] "Zambia"
```