Spatial Data Handling

Textbook: Chapter 3

https://ceiba.ntu.edu.tw/1072_Geog2017

授課教師：溫在弘
E-mail: wenthung@ntu.edu.tw

Contents

- Chapter 3: Spatial data handling
  - Introducing GISTools
  - Mapping spatial objects
  - Mapping spatial data attributes (thematic maps)

Learning Objectives

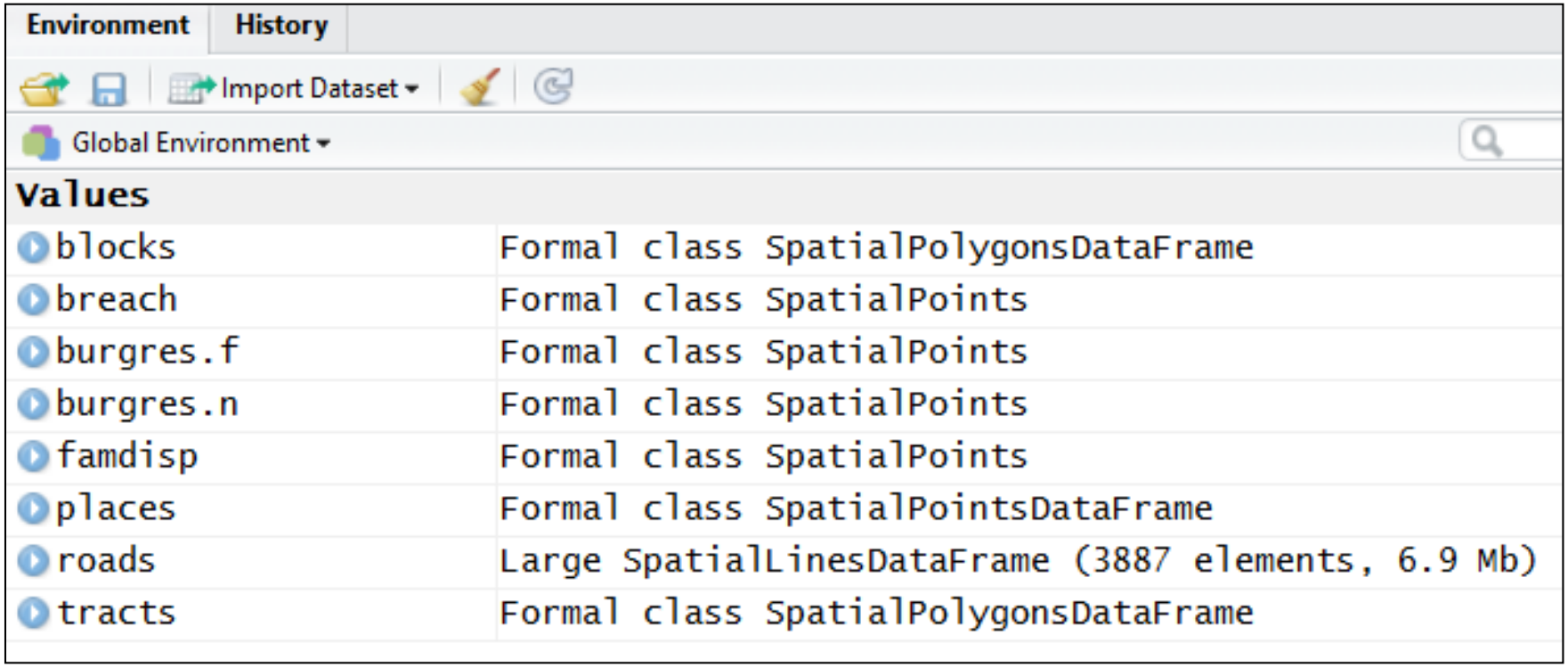
- R Package: GISTools
- Compile maps based on multiple layers
- Set different shading schemes
- Plot spatial data with different parameters

Data Classes for Spatial Data

Without Attributes	With attributes	ArcGIS Equivalent
SpatialPoints	SpatialPointsDataFrame	Point shapefiles
SpatialLines	SpatialLinesDataFrame	Line shapefiles
SpatialPoints	SpatialPolygonsDataFrame	Polygon shapefiles

Spatial Data in GISTools

```
> library(GISTools)
> data(newhaven)
> ls()
[1] "blocks"    "breach"    "burgres.f" "burgres.n" "famdisp"   "places"    "roads"     "tracts"
```



1. Mapping Spatial Objects: Exploring Spatial Data and Its Attributes

```
> plot(blocks)
```

```
block.attr<-data.frame(blocks)
```

Global Environment

Data

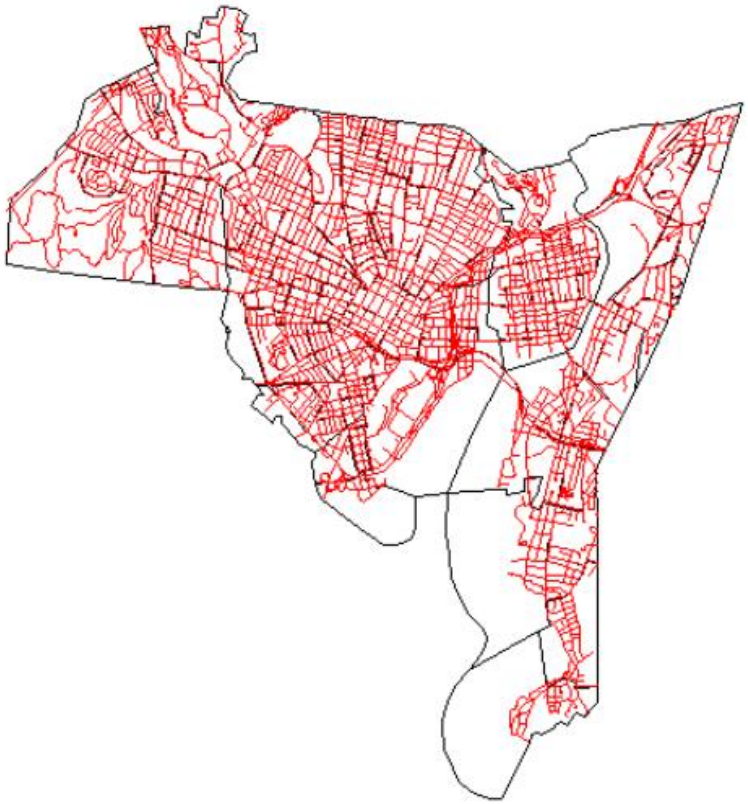
block.attr 129 obs. of 28 variables

Filter

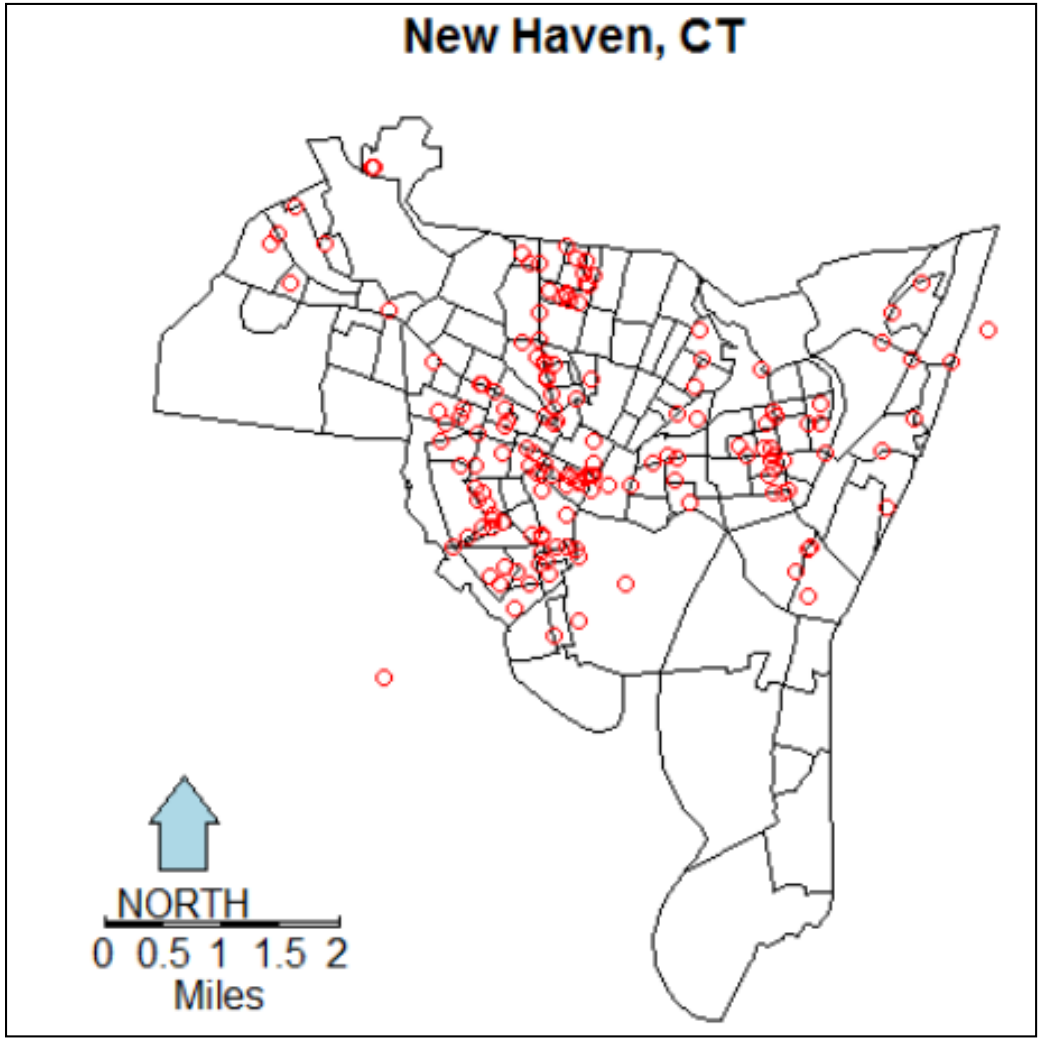
	NEWH075H_	NEWH075H_I	HSE_UNITS	OCCUPIED	VACANT	P_VACANT	P_OWNEROCC	P_RENTROCC
0	2	69	763	725	38	4.980341	0.393185	94.626474
1	3	72	510	480	30	5.882353	20.392157	73.725490
2	4	64	389	362	27	6.940874	57.840617	35.218509
3	5	68	429	397	32	7.459207	19.813520	72.727273
4	6	67	443	385	58	13.092551	80.361174	6.546275
5	7	133	588	548	40	6.802721	52.551020	40.646259
6	8	73	410	389	21	5.121951	57.804878	37.073171
7	9	134	615	562	53	8.617886	33.658537	57.723577
8	10	84	316	293	23	7.278481	49.367089	43.354430
9	11	80	365	337	28	7.671233	38.630137	53.698630
10	12	79	276	256	20	7.246377	41.666667	51.086957
11	13	136	393	377	16	4.071247	44.274809	51.653944
12	14	77	355	309	46	12.957746	32.394366	54.647887
13	15	97	595	534	61	10.252101	26.386555	63.361345

Plotting multiple layers

```
par(mar = c(0,0,0,0))
plot(blocks)
plot(roads, add=TRUE, col="red")
```



Map Layout: scale-bar and north arrow



Map Layout: R codes

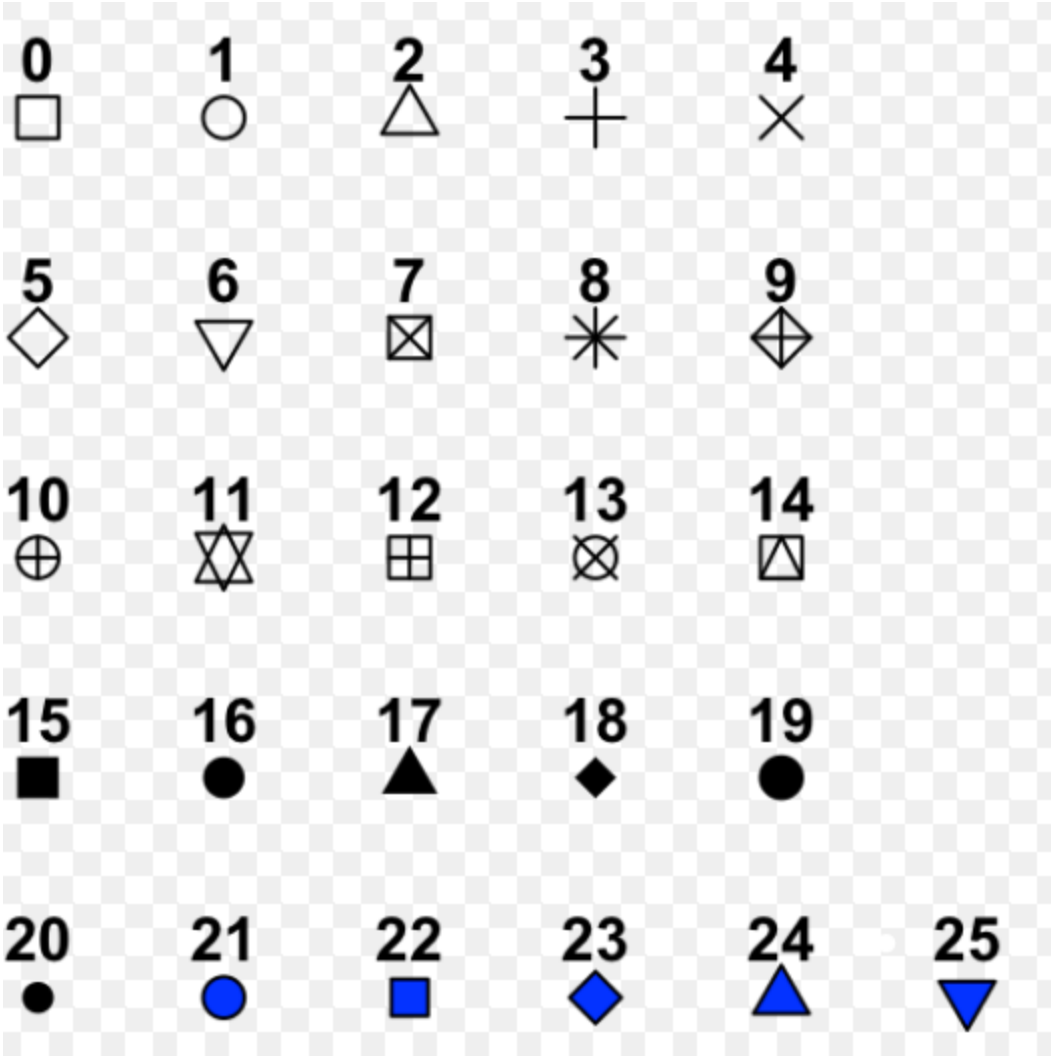
```
data(newhaven)
# plot spatial data
par(mar = c(0,0,2,0))
plot(blocks)
plot(breach,add=TRUE,col= 'red', pch = 1)
# embellish the map
map.scale(534750,152000,miles2ft(2), "Miles",4,0.5)
north.arrow(533043.9,154617.4,miles2ft(0.2),col= 'lightblue')
title('New Haven, CT')
```

setting margins c(bottom, left, top, right)

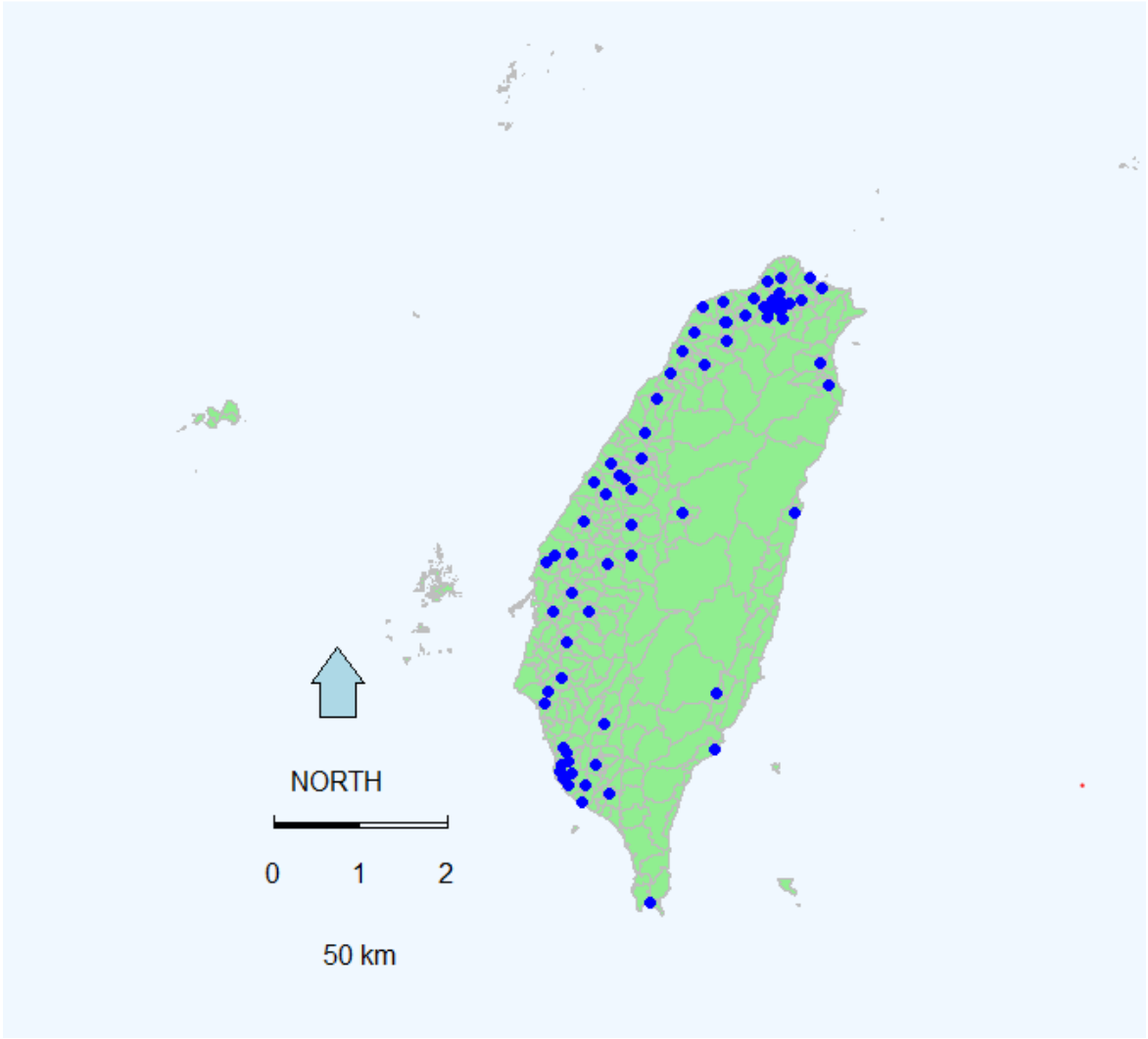
`mar`

A numerical vector of the form `c(bottom, left, top, right)` which gives the number of lines of margin to be specified on the four sides of the plot. The default is `c(5, 4, 4, 2) + 0.1`.

R Plot function: pch symbol



課堂練習

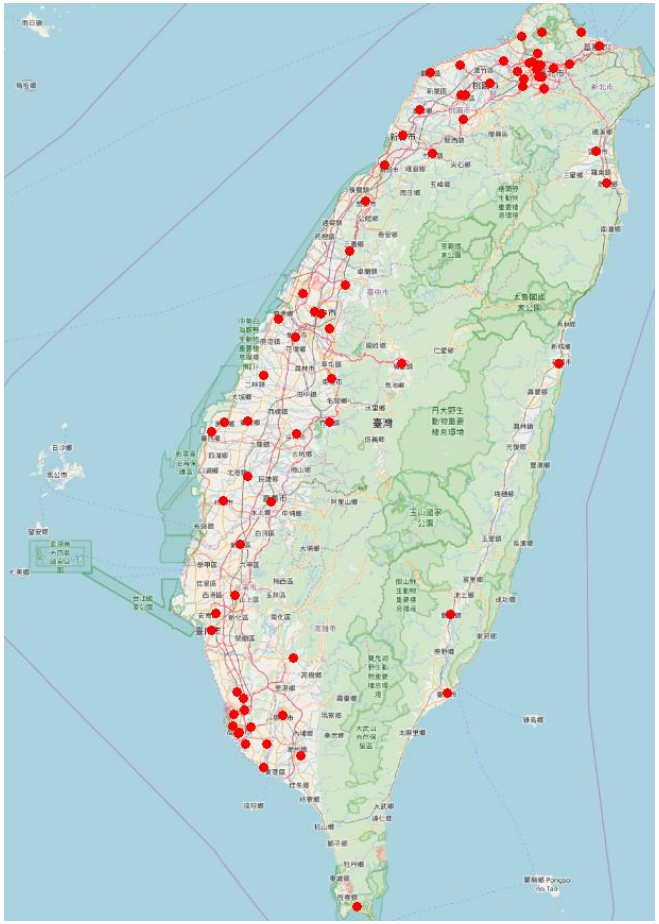


課堂練習: R codes

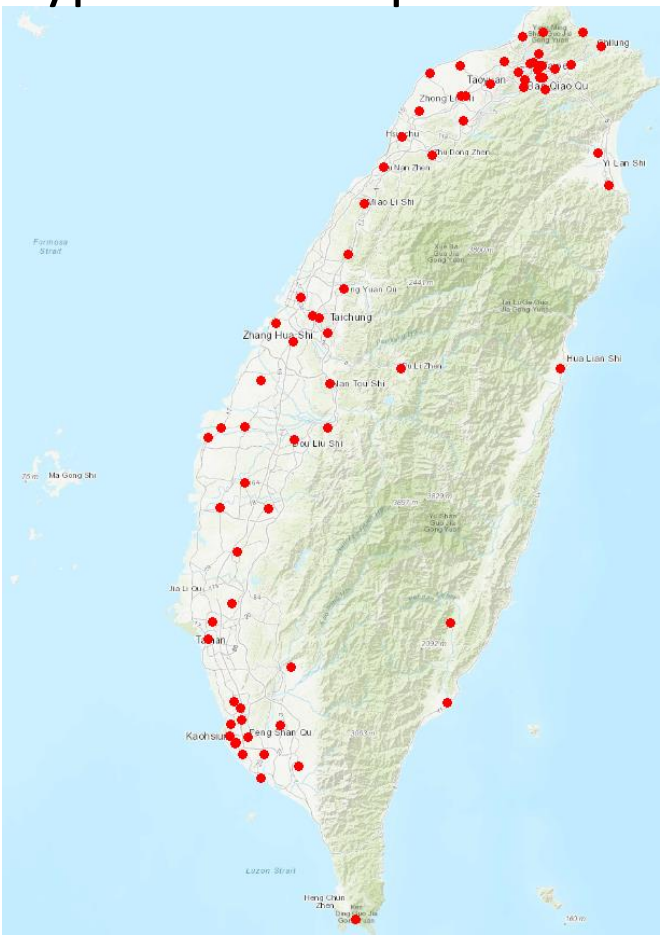
```
library(rgdal)
EPA.STN <- readOGR(dsn = "./data/SHP", layer = "EPA_STN1", encoding="unicode")
Popn.TWN <- readOGR(dsn = "./data/SHP", layer = "Popn_TWN2", encoding="unicode")
par(mar = c(0,2,2,2))
plot(Popn.TWN, col='lightgreen', border="grey", bg='aliceblue')
Popn.TWN.attr<-data.frame(Popn.TWN)
plot(EPA.STN,add=TRUE,col= 'blue', pch = 16)
map.scale(63030.22,2472112,100000, "50 km",2,1)
north.arrow(49646.41,2534913,10000,col= 'lightblue')
title('Taiwan EPA Stations')
```

Adding contexts (OpenStreetMap as Background)

Type = “osm”



Type = “esri-topo”



Adding contexts: R codes

openmap {OpenStreetMap} R Documentation

Get a map based on lat long coordinates

Description

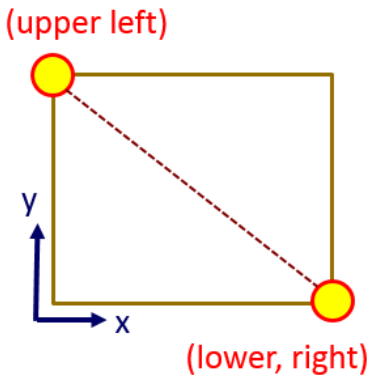
Get a map based on lat long coordinates

Usage

```
openmap(upperLeft, lowerRight, zoom = NULL, type = c("osm", "osm-bw",
  "maptoolkit-topo", "waze", "bing", "stamen-toner", "stamen-terrain",
  "stamen-watercolor", "osm-german", "osm-wanderreitkarte", "mapbox", "esri",
  "esri-topo", "nps", "apple-iphoto", "skobbler", "hillshade", "opencyclemap",
  "osm-transport", "osm-public-transport", "osm-bbike", "osm-bbike-german"),
  minNumTiles = 9L, mergeTiles = TRUE)
```

```
#####################
library(rgdal);library(sp)
proj4string(Popn.TWN)
TWN.LongLat <- spTransform(Popn.TWN, CRS("+proj=longlat +datum=WGS84 +no_defs +ellps=WGS84 +towgs84=0,0,0"))

##############
library(OpenStreetMap)
# define upper left, lower right corners
ul <- as.vector(cbind(bbox(TWN.LongLat)[2,2], bbox(TWN.LongLat)[1,1]))
lr <- as.vector(cbind(bbox(TWN.LongLat)[2,1], bbox(TWN.LongLat)[1,2]))
# download the map tile
MyMap <- openmap(ul,lr,9, "osm")
# now plot the layer and the backdrop
par(mar = c(0,0,0,0))
plot(MyMap, removeMargin=FALSE)
plot(spTransform(EPA.STN, osm()), pch = 16, add = TRUE, col="red", cex=1.2)
```



投影座標轉換 spTransform():
map projection and datum transformation

```
proj4string(Popn.TWN)
TWN.LongLat <- spTransform(Popn.TWN, CRS("+proj=longlat +datum=WGS84 +no_defs +ellps=WGS84 +towgs84=0,0,0"))
```

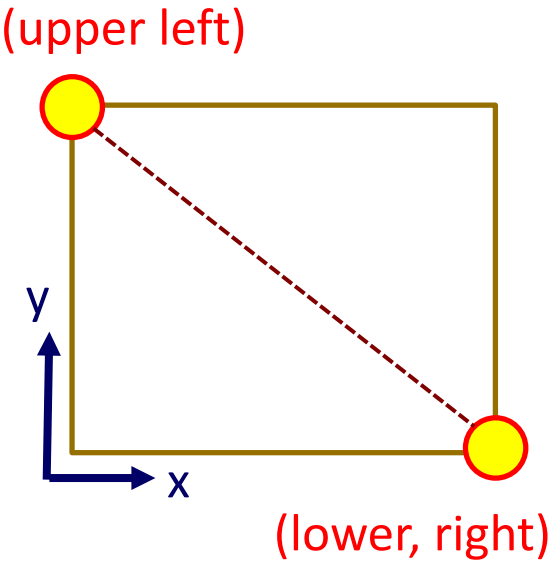
https://epsg.io/4326 EPSG:4326
WGS84 - World Geodetic System 1984

https://epsg.io/3826 EPSG:3826
TWD97 / TM2 zone 121

PROJ.4:
+proj=tmerc +lat_0=0 +lon_0=121 +k=0.9999 +x_0=250000 +y_0=0
+ellps=GRS80 +towgs84=0,0,0,0,0,0,0 +units=m +no_defs

bbox(): retrieves spatial bounding box from spatial data

```
> bbox(TWN.LongLat)
        min       max
x 119.30663 121.99868
y  21.89851  25.30293
```




```
> bbox(TWN.LongLat)[1,1]
[1] 119.3066
> bbox(TWN.LongLat)[1,2]
[1] 121.9987
```

2. Mapping Spatial Data Attributes

```
block.attr<-data.frame(blocks)
```

Global Environment

Data

block.attr 129 obs. of 28 variables

Filter

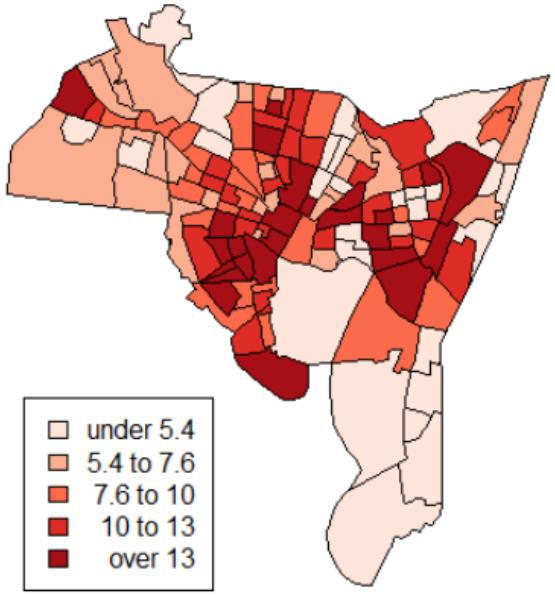
	NEWH075H_	NEWH075H_I	HSE_UNITS	OCCUPIED	VACANT	P_VACANT	P_OWNEROCC	P_RENTROCC
0	2	69	763	725	38	4.980341	0.393185	94.626474
1	3	72	510	480	30	5.882353	20.392157	73.725490
2	4	64	389	362	27	6.940874	57.840617	35.218509
3	5	68	429	397	32	7.459207	19.813520	72.727273
4	6	67	443	385	58	13.092551	80.361174	6.546275
5	7	133	588	548	40	6.802721	52.551020	40.646259
6	8	73	410	389	21	5.121951	57.804878	37.073171
7	9	134	615	562	53	8.617886	33.658537	57.723577
8	10	84	316	293	23	7.278481	49.367089	43.354430
9	11	80	365	337	28	7.671233	38.630137	53.698630
10	12	79	276	256	20	7.246377	41.666667	51.086957
11	13	136	393	377	16	4.071247	44.274809	51.653944
12	14	77	355	309	46	12.957746	32.394366	54.647887
13	15	97	595	534	61	10.252101	26.386555	63.361345

Spatial Data Attributes: R codes

```
library(GISTools)
data(newhaven)
data.frame(blocks)
head(data.frame(blocks))
colnames(data.frame(blocks))
data.frame(blocks)$P_VACANT
blocks$P_VACANT
attach(data.frame(blocks))
par(mar = c(3,5,3,3))
hist(P_VACANT)
```

Mapping Spatial Data Attributes

```
par(mar = c(1,5,3,3))
choropleth(blocks, blocks$P_VACANT)
vacant.shades = auto.shading(blocks$P_VACANT)
vacant.shades
choro.legend(533000,161000,vacant.shades)
```



Mapping Spatial Data Attributes: Data Query

The code used above includes logical operators and illustrates how they can be used to select elements that satisfy some condition. These can be used singularly or in combination to select in the following way:

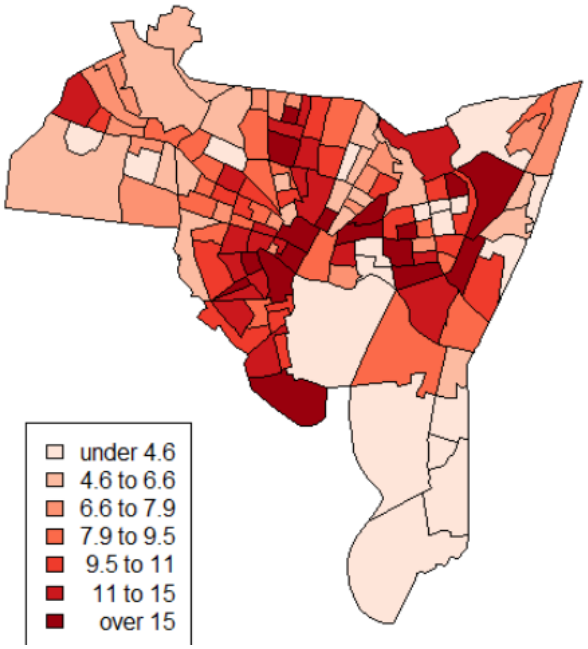
```
data <- c(3, 6, 9, 99, 54, 32, -102)
index <- (data == 32 | data <= 6)
data[index]
```

```
## [1]    3  6  32  -102
```

These are described in greater detail in Chapter 4.

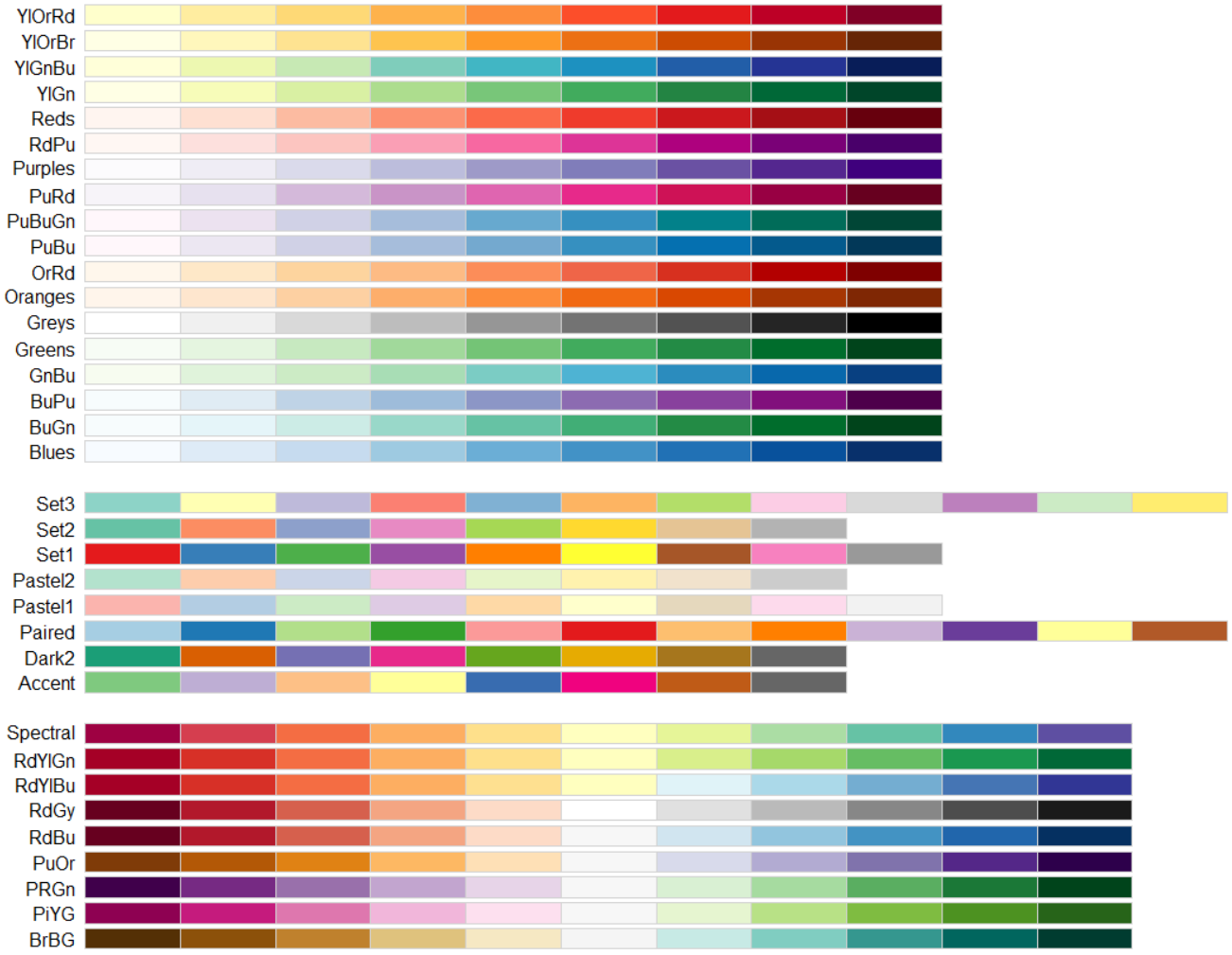
Set different shading schemes: No. of intervals

```
# set the shading
par(mar = c(0,5,0,3))
vacant.shades = auto.shading(blocks$P_VACANT,n=7)
# plot the map
choropleth(blocks,blocks$P_VACANT,shading=vacant.shades)
choro.legend(533000,161000,vacant.shades)
```



Set different shading schemes: colors

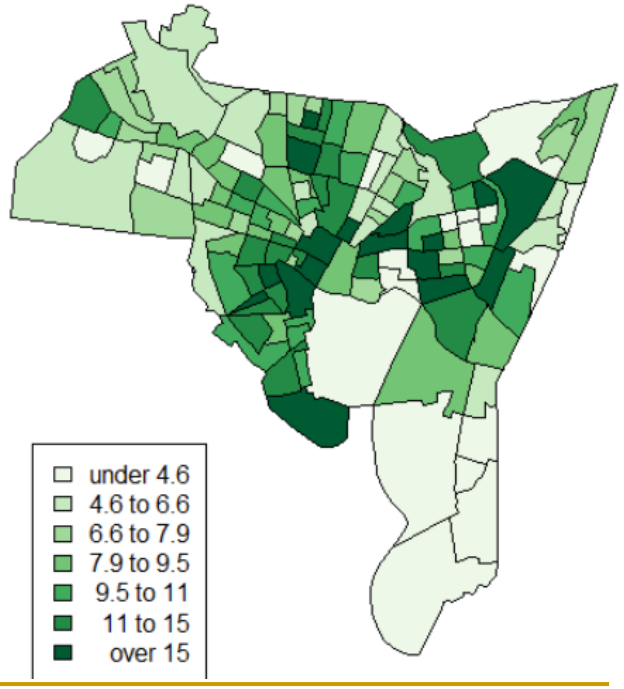
```
display.brewer.all()
```



Set different shading schemes: colors

```
brewer.pal(5,'Blues')

vacant.shades = auto.shading(blocks$P_VACANT, cols=brewer.pal(7,"Greens"), n=7)
choropleth(blocks, blocks$P_VACANT,shading=vacant.shades)
choro.legend(533000,161000,vacant.shades)
```



Set different shading schemes: Class break methods

```
vacant.shades = auto.shading(blocks$P_VACANT, n=5, cols=brewer.pal(5,"Blues"), cutter=rangeCuts)
choropleth(blocks,blocks$P_VACANT,shading=vacant.shades)
choro.legend(533000,161000,vacant.shades)
```

```
quantileCuts(x, n = 5, params = NA)
sdCuts(x, n = 5, params = NA)
rangeCuts(x, n = 5, params = NA)
```

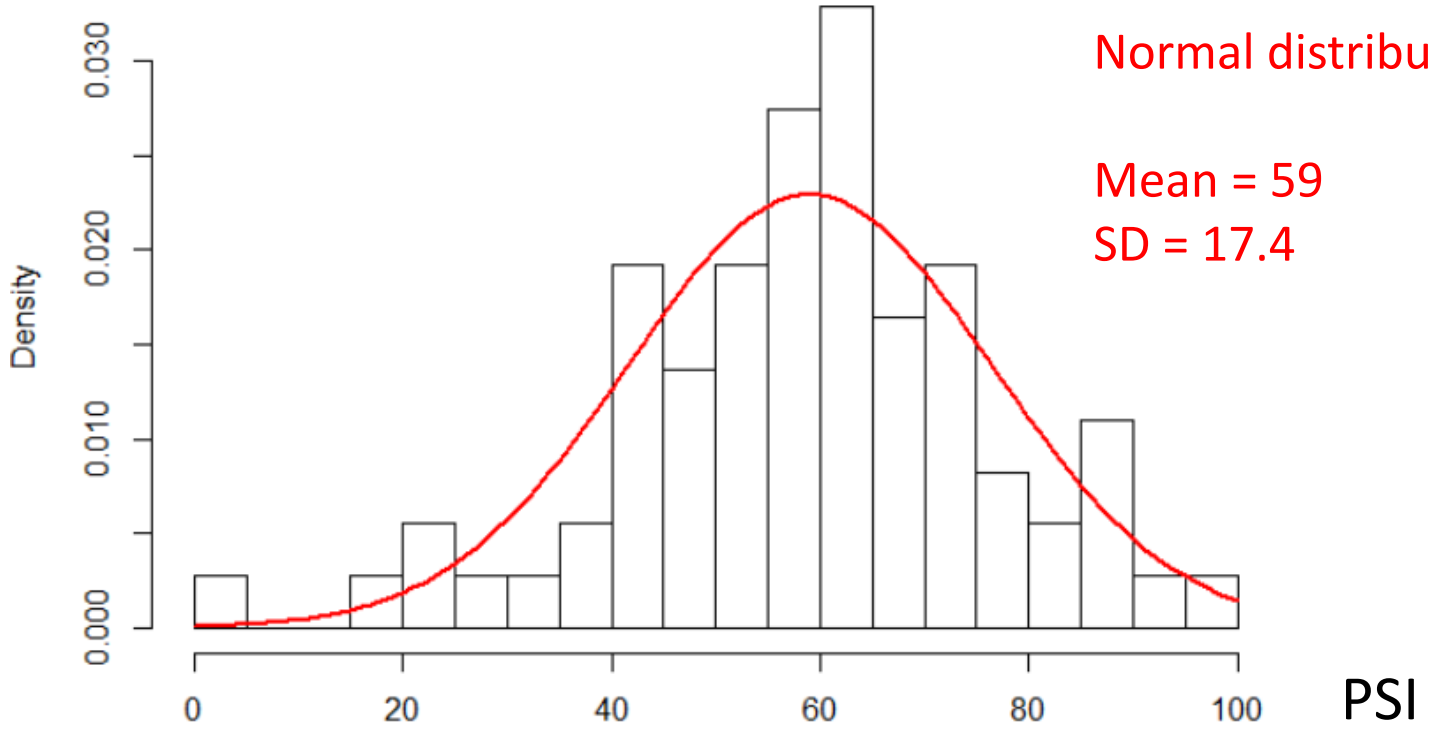
Review

- #1: Choropleth mapping
- #2: Choropleth mapping with different cutters
- #3: Selecting spatial data: Using logical functions
- #4: Transforming map projections: `spTransform()`
- #5: Retrieving spatial bounding: `bbox()`

實習:建立特定超越機率的空汙地圖

EPA_STN1.shp

PSI is a type of air quality index



Normal distribution:

Mean = 59

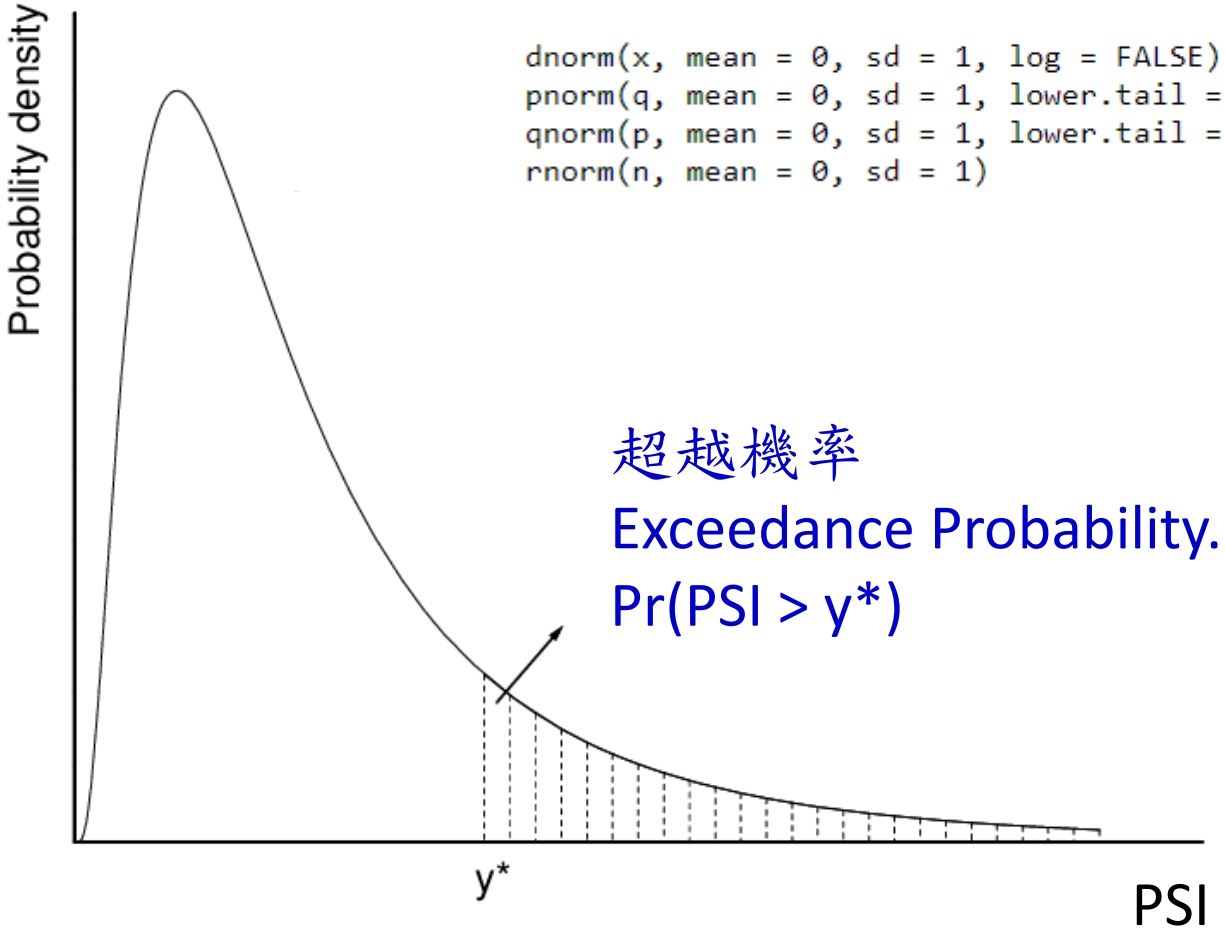
SD = 17.4

實習:超越機率的概念

PSI is a type of air quality index

複習**R**的機率函數使用

```
dnorm(x, mean = 0, sd = 1, log = FALSE)
pnorm(q, mean = 0, sd = 1, lower.tail = TRUE, log.p = FALSE)
qnorm(p, mean = 0, sd = 1, lower.tail = TRUE, log.p = FALSE)
rnorm(n, mean = 0, sd = 1)
```



超越機率

Exceedance Probability.

Pr(PSI > y*)

實習:建立特定超越機率的空汙地圖

- **利用 GISTools** 建立繪製地圖的函數

  Pollution_Map (agr1)；

  引數agr1 是可自行設定的超越機率 (e.g. 0.2)

  - (1) 該函數會回傳該超越機率所對應的PSI值。
  - (2) 以此數值為臨界值，繪製空氣污染地圖，

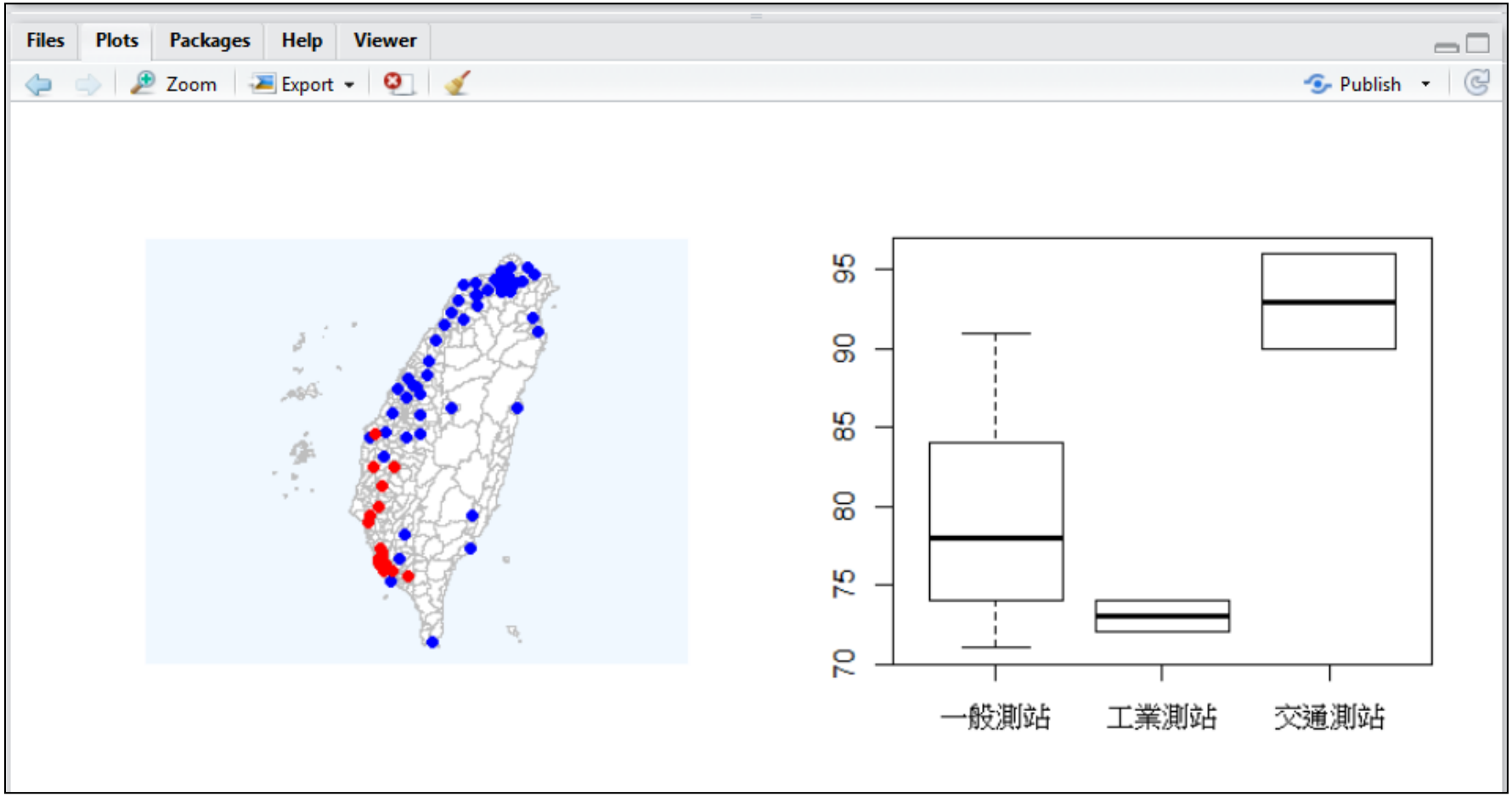
    超過該數值的測站，表示紅色，其餘為藍色。
  - (3) 以此數值為臨界值，針對超過該數值的測站，

    按照測站類別(SiteType)，依照「一般測站、工業測站、交通測站」這三類，以box plot呈現PSI分布。

實習的預期結果

* 執行Pollution_Map(0.3)與Pollution_Map(0.5)來檢核函數結果

```
> Pollution_Map(0.3)
[1] 68.12457
```



作業

- Q1：**利用 ggplot2** 完成實習的繪製地圖函數。

- Q2：繪製人口老化地圖與統計圖表。

  (不限使用R的繪圖套件)

  - 2-1: 台灣人口密度地圖
  - 2-2: 大台北人口老化地圖
  - 2-3: Boxplot: 比較老年人口分布

Q2：繪製人口老化地圖與統計圖表

Data: Popn_TWN2.shp

- 繪製**台灣**鄉鎮人口密度的面量圖 (Popn/Area)

  [按照Quantile 分成6級，含圖例、比例尺、圖名和指北針]

- 在大台北地區 (含台北、新北、基隆、桃園、宜蘭等)範圍內，以紅色標示老年人口比例 (Age_L65/Popn)在**top20%**的鄉鎮，繪製**大台北區**的人口老化地圖 。

  (設定60%紅色透明度，疊合OSM底圖)

- 繪製boxplot，比較台灣全島的高度密集 (鄉鎮人口密度 > 10,000/km2) vs. 低度密集 (鄉鎮人口密度 < 2,000/km2) 的老年人口比例的分布。