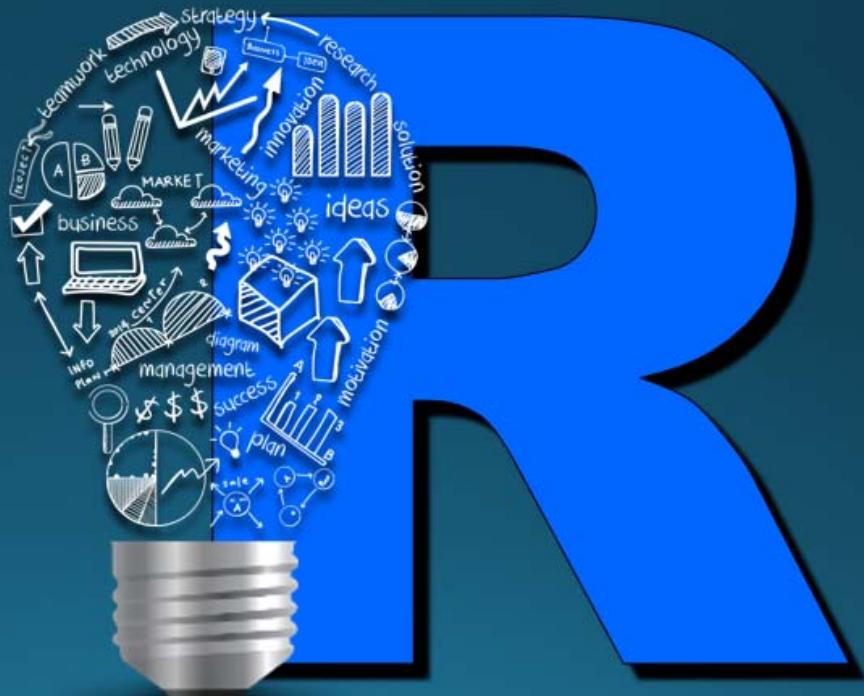


R語言畫地圖 (Maps)

吳漢銘
國立政治大學 統計學系



<https://hmwu.idv.tw>



- 使用Google Cloud Platform雲端服務
- **RgoogleMaps** : plotting on Google static maps in R
- **ggmap**: Spatial visualization with ggplot2
- 分級著色圖 /面量圖(Choropleth Maps)
- **choroletchr**: Simplify the creation of choropleth maps in R



使用Google Cloud Platform雲端服務

3/54

step1: Get API Key

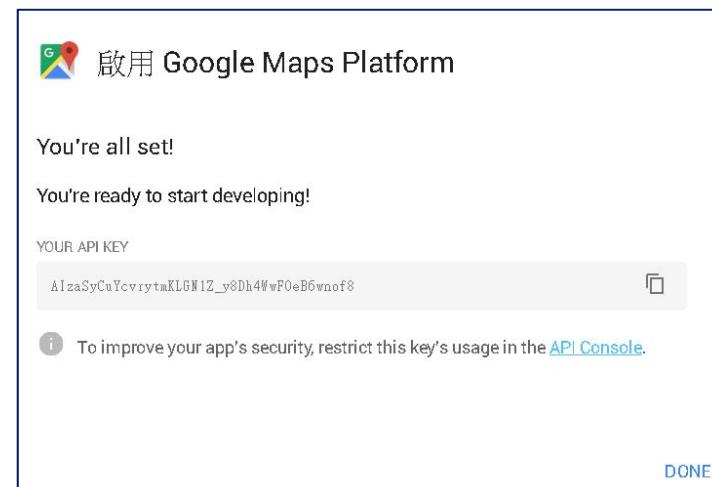
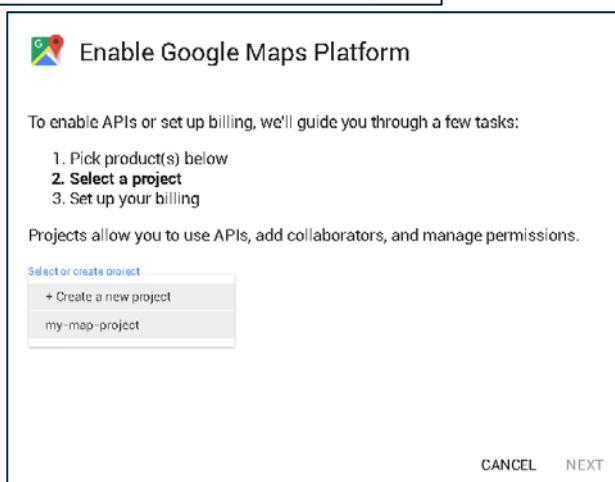
<https://cloud.google.com/maps-platform/>

<https://developers.google.com/maps/documentation/maps-static/get-api-key>

Step2: Enable Billing

https://console.cloud.google.com/project/_/billing/enable

As of mid-2018, the Google Maps Platform requires a registered API key.





使用Google Cloud Platform雲端服務

4/54

Google Cloud Platform

Han-Ming, 歡迎使用！

感謝您申請免費試用本服務 12 個月。

感謝您提出申請。免費試用方案提供 \$300 美元的抵免額，供您在未來 12 個月內使用。請放心，除非您明確同意，否則即使額度用盡，系統也不會自動向您收費。

我知道了

Google APIs my-map-project

← API 程式庫

Geocoding API

Google

Convert between addresses and geographic coordinates.

啟用

類型 API 和服務

上次更新時間 2019/9/13 上午7:39

類別 地圖

服務名稱 geocoding-backend.googleapis.com

總覽

Convert addresses into geographic coordinates (geocoding), which you can use to place markers or position the map. This API also allows you to convert geographic coordinates into an address (reverse geocoding).

Google 簡介

Google's mission is to organize the world's information and make it universally accessible and useful. Through products and platforms like Search, Maps, Gmail, Android, Google Play, Chrome and YouTube, Google plays a meaningful role in the daily lives of billions of people.

定價

彈性定價可根據您的需求進行調整。此外，每月可獲得 \$200.00 的免費用額 (用於地圖、路徑與地點)。

Geocoding	\$ 5.00 價格/1K requests 0 - 100K requests/個月	\$ 4.00 價格/1K requests 100K+ requests/個月
-----------	---	--

免費試用狀態：您的試用額度還剩 NT\$9,197.00，且免費試用期將在 366 天後結束。只要升級為完整帳戶，您就能使用所有 Google Cloud Platform 功能，不受任何限制。

關閉 啟用

Google Cloud Platform my-map-project

Google 地圖 API

請選取 API 來查看詳情。數值為最近 30 天的資料。

API ↑	要求	錯誤	平均延遲時間 (毫秒)
Geocoding API	4	3	83
Maps Embed API	0	0	-
Maps JavaScript API	0	0	-
Maps SDK for Android	0	0	-
Maps SDK for iOS	0	0	-
Maps Static API	2	1	210
Street View Static API	0	0	-

Google Cloud Platform my-map-project

← Google 地圖 Maps Static API 停用

指標 配額 憑證 網址簽署密鑰

如要查看所有憑證或建立新的憑證，請前往 API 和服務中的憑證頁面

建議您妥善保存新的 API 金鑰 確保憑證安全

API 金鑰

名稱	建立日期	限制 ↓	鍵	這項服務的用量 (天內) ?
API 金鑰 1	2019年10月 15日	無	AIzaSyCuYc...F0eB6wnof8	2



從RStudio測試是否已註冊GCP成功

5/54

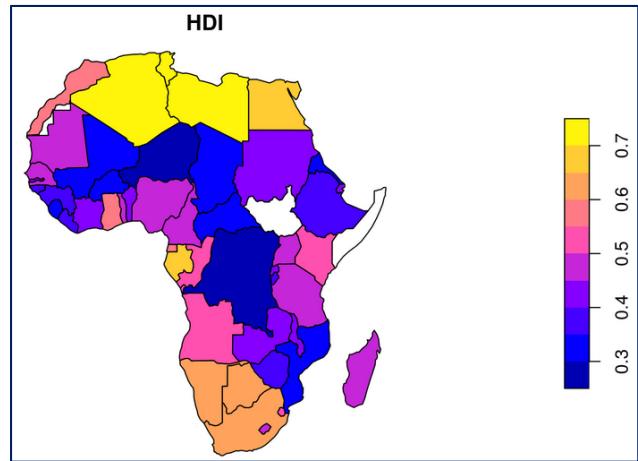
```
> library(ggmap)
> register_google(key = "AIzaSyCuYcvrytmKLGNxxxxxxxx", write = TRUE)
Replacing old key (AIzaSyCuYcvrytmKLGNxxxxxxxx) with new key in
C:/Users/userpc/Documents/.Renviron
> tw.xy <- geocode("Taiwan")
Source : https://maps.googleapis.com/maps/api/geocode/json?address=Taiwan&key=xxx
> tw.xy
# A tibble: 1 x 2
  lon    lat
  <dbl> <dbl>
1 121.   23.7
> tw.map <- get_map(location = 'Taiwan', zoom = 7, language = "zh-TW")
Source :
https://maps.googleapis.com/maps/api/staticmap?center=Taiwan&zoom=7&size=640x640&
scale=2&maptype=terrain&language=zh-TW&key=xxx
Source : https://maps.googleapis.com/maps/api/geocode/json?address=Taiwan&key=xxx
>

> has_google_key()
[1] TRUE
> google_key()
[1] "AIzaSyCuYcvrytmKLGNxxxxxxxx"
```

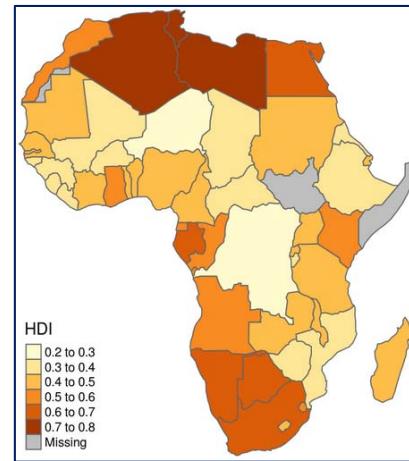
```
> tw.map <- get_map(location = 'Taiwan', zoom = 7, language = "zh-TW")
Error: Google now requires an API key.
See ?register_google for details.
```

Making maps with R packages

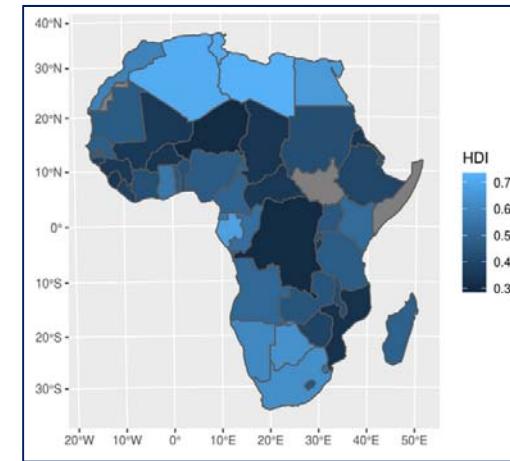
plot



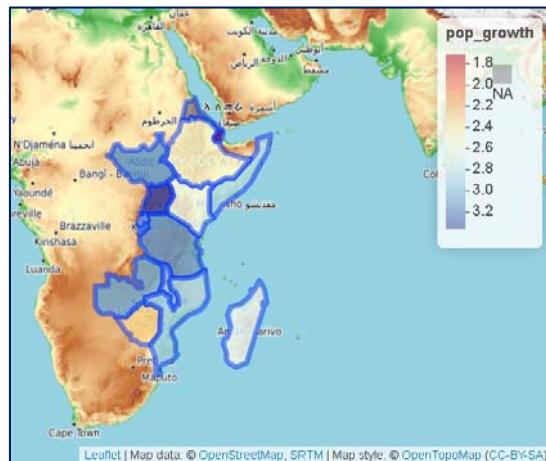
tmap



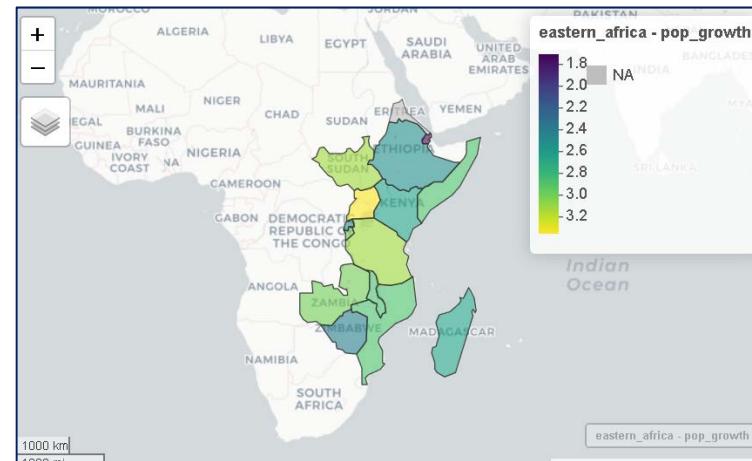
ggplot2



leaflet



mapview



Chapter 8: Making maps with R, Robin Lovelace, Jakub Nowosad, Jannes Muenchow, 2019-08-30

<https://geocompr.github.io/geocomprpkg/articles/solutions08.html>



Plotting on Google Static Maps in R: RgoogleMaps

```
library(RgoogleMaps)
TaiwanMap <- GetMap(center=c(lat = 23.58, lon =120.58), zoom =7, destfile = "Taiwan1.png")
TaiwanMap <- GetMap(center=c(lat = 23.58, lon =120.58), zoom = 10, destfile = "Taiwan2.png",
maptype = "terrain")
```



x = longitude 經度 (long)
y = latitude 緯度 (lat)



於地圖上標記

8/54

```
> my.lat <- c(25.175339, 25.082288, 25.042185, 25.046254)
> my.lon <- c(121.450003, 121.565481, 121.614548, 121.517532)
> bb = qbbox(my.lat, my.lon)
> print(bb)

$latR
[1] 25.04152 25.17600

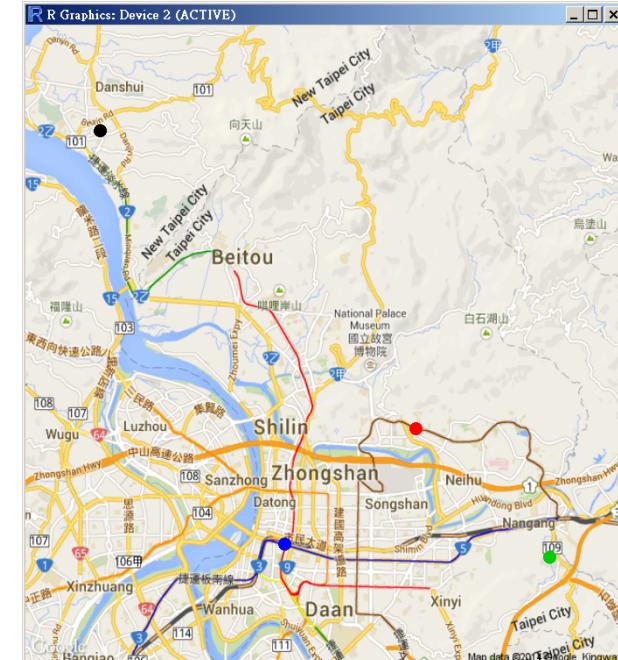
$lonR
[1] 121.4492 121.6154

> MyMap <- GetMap.bbox(bb$lonR, bb$latR, destfile = "my.png", maptype = "roadmap")
> My.markers <- cbind.data.frame(lat = my.lat, lon = my.lon)
> tmp <- PlotOnStaticMap(MyMap, lat = My.markers[, "lat"], lon = My.markers[, "lon"], destfile
= "my.png", cex=2.5, pch=20, col=1:4, add=F)
```

查詢經緯度

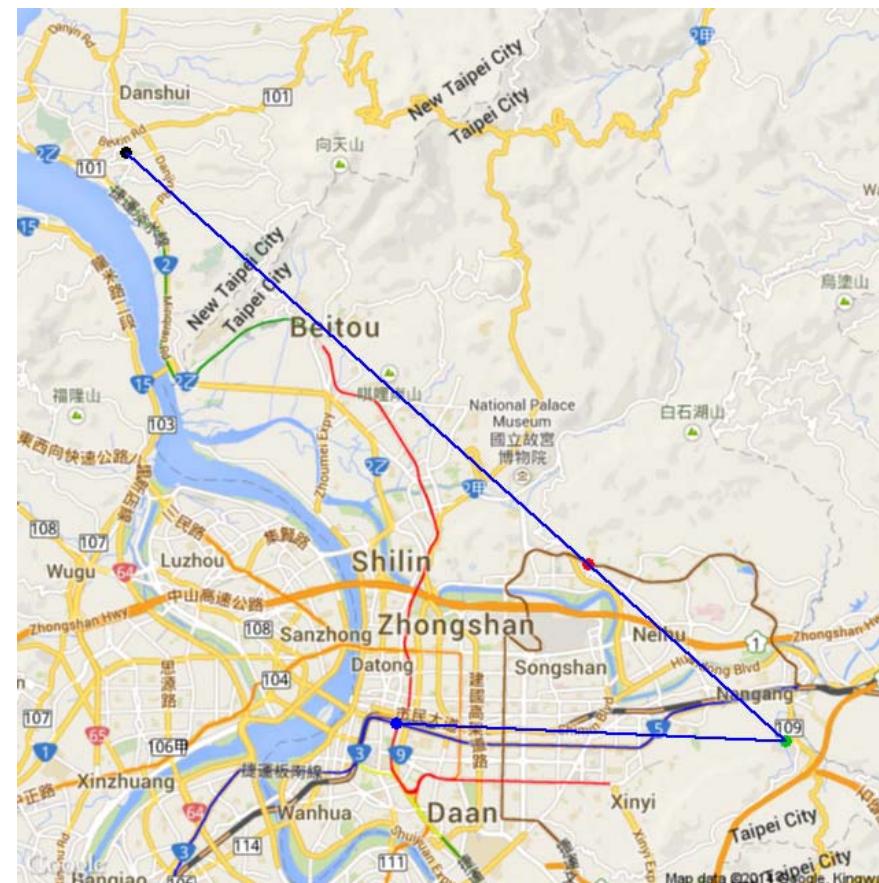
http://card.url.com.tw/realads/map_latlng.php

- 淡江大學 25.175339, 121.450003
- 台北市的地理中心位置: 內湖區環山路和內湖路一段
跟基湖路口: 25.082288, 121.565481
- 中研院 25.042185, 121.614548
- 捷運台北站: 25.046254, 121.517532



於地圖上標記

```
png("my2.png", 640, 640)
tmp <- PlotOnStaticMap(MyMap, lat = My.markers[, "lat"], lon = My.markers[, "lon"], cex=2.5,
pch=20, col=1:4, add=F)
tmp <- PlotOnStaticMap(MyMap, lat = My.markers[, "lat"], lon = My.markers[, "lon"], col="blue",
add=T, FUN = lines, lwd = 2)
dev.off()
```



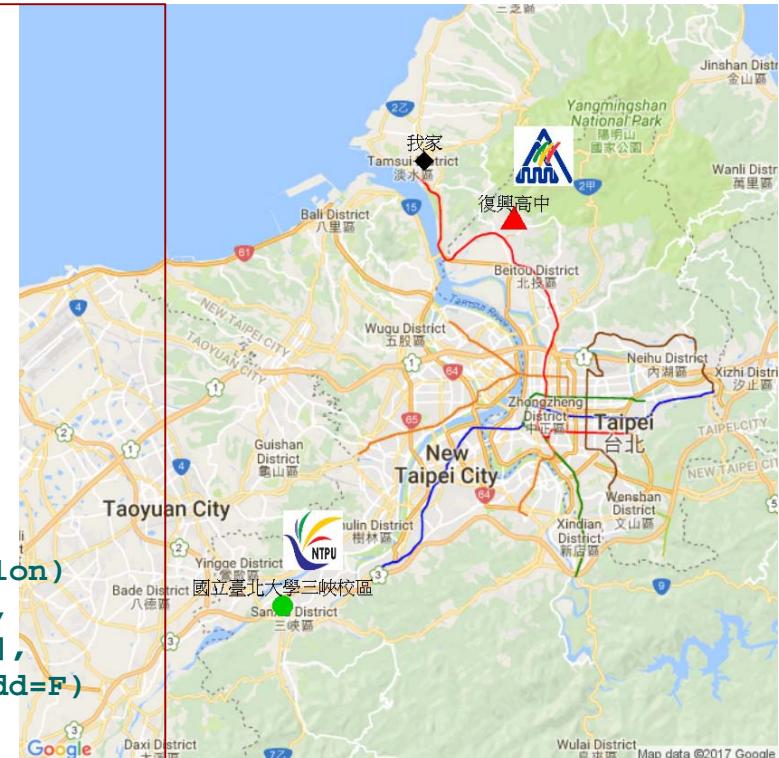


於地圖上加文字，加圖片

10/54

```
> library(RgoogleMaps)
> my.lat <- c(25.175339, 25.14362, 24.942605)
> my.lon <- c(121.450003, 121.501768, 121.368381)
>
> bb = qbbox(my.lat, my.lon)
> print(bb)
$latR
[1] 24.94144 25.17650

$lonR
[1] 121.3677 121.5024
> MyMap <- GetMap.bbox(bb$lonR, bb$latR,
+                         destfile = "my.png", maptype = "roadmap")
>
> My.markers <- cbind.data.frame(lat = my.lat, lon = my.lon)
> tmp <- PlotOnStaticMap(MyMap, lat = My.markers[, "lat"],
+                         lon = My.markers[, "lon"],
+                         destfile = "my.png", cex=2.5, pch=18:10, col=1:3, add=F)
> TextOnStaticMap(MyMap, lat = My.markers[, "lat"]+0.01,
+                   lon = My.markers[, "lon"],
+                   labels=c("我家", "復興高中", "國立臺北大學三峽校區"), add=T)
```



```
> library(EBImage)
> ntpu <- readImage("NTPUcolorlogo.jpg")
> loc <- LatLon2XY.centered(MyMap, lat=My.markers[3, 1], lon=My.markers[3, 2])
> rasterImage(ntpu, loc[[1]], loc[[2]]+30, loc[[1]]+50, loc[[2]]+80)
> Fuxing <- readImage("Fuxinglogo.jpg")
> loc <- LatLon2XY.centered(MyMap, lat=My.markers[2, 1], lon=My.markers[2, 2])
> rasterImage(Fuxing, loc[[1]], loc[[2]]+30, loc[[1]]+50, loc[[2]]+80)
```

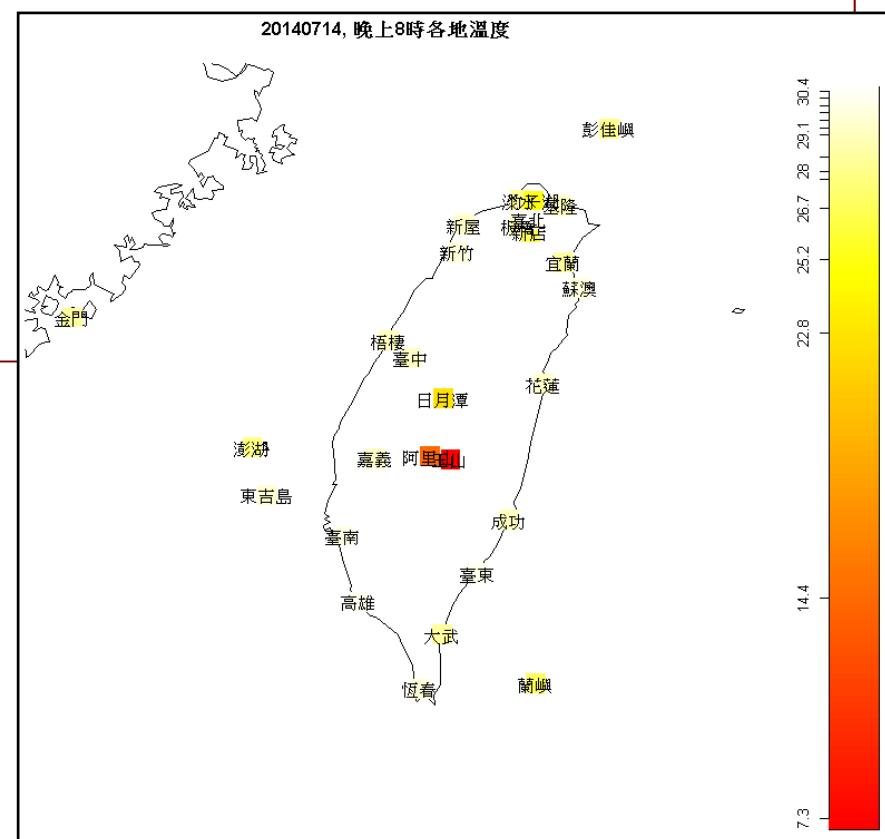
```
if (!requireNamespace("BiocManager", quietly = TRUE))
  install.packages("BiocManager")
BiocManager::install("EBImage")
```



Package: maps , mapdata

氣象局開放資料平台 <http://opendata.cwb.gov.tw/>

library(maps); library(maptools); library(mapdata); library(mapproj)
layout(matrix(c(1,1,1,0,2,0), ncol=2), widths=c(10, 1), heights=c(1, 10, 1))
map("world2Hires", xlim=c(118, 123), ylim=c(21, 26))
data <- read.table("20140714-weather.txt", sep="\t", header=TRUE, row.names=1)
x <- data\$TEMP
tm <- floor((100-1)/(max(x)-min(x))*(x-min(x)) + 1)
used.col <- heat.colors(100)[tm]
points(data\$lon, data\$lat, pch=15, col=used.col)
text(data\$lon, data\$lat, labels=row.names(data))
title("20140714, 晚上8時各地溫度")
par(mar=c(1,1,1,1))
image(t(matrix(c(1:100), ncol=1)),
 col=heat.colors(100), xaxt="n", yaxt="n")
axis(LEFT <- 2, at=tm/100,
 labels=as.character(x), cex.axis=1)



See also:

Spatial data in R: Using R as a GIS

<http://pakillo.github.io/R-GIS-tutorial/>



ggmap: Spatial Visualization with ggplot2

ggmap quickstart

For more functionality, see ggmap documentation and
<https://dl.dropboxusercontent.com/u/24648660/ggmap%20useR%202012.pdf>

There are 2 basic steps to making a map using ggmap:

Part 1: Download map raster → Part 2: Plot raster and overlay data

Part 1: Downloading the map raster

Start by loading the package: `library(ggmap)`

1. Define location: 3 ways

- location/address
`myLocation <- "University of Washington"`
- lat/long
`myLocation <- c(lon = -95.3632715, lat = 29.7632836)`
- bounding box lowerleftlon, lowerleftlat, upperrightlon, upperrightlat
(a little glitchy for google maps)
`myLocation <- c(-130, 30, -105, 50)`

Convert location/address its lat/long coordinates:
`geocode("University of Washington")`

2. Define map source, type, and color

The `get_map` function provides a general approach for quickly obtaining maps from multiple sources. I like this option for exploring different styles of maps.

There are 4 map "sources" to obtain a map raster, and each of these sources has multiple "map types" (displayed on right).

- `stamen`: `maptyle = c("terrain", "toner", "watercolor")`
- `google`: `maptyle = c("roadmap", "terrain", "satellite", "hybrid")`
- `osm`: open street map
- `cloudmade`: 1000s of maps, but an api key must be obtained from <http://cloudmade.com>

```
myMap <- get_map(location=myLocation,  
source="stamen", maptype="watercolor", crop=FALSE)  
ggmap(myMap)
```

This will produce a map that looks something like this

NOTE: `crop = FALSE` because otherwise, with stamen plots, the map is slightly shifted when I overlay data.

3. Fine tune the scale of the map using zoom

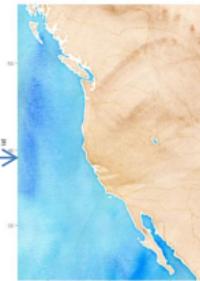
The `get_map` function takes a guess at the zoom level, but you can alter it:

`zoom = integer from 3-21`
3 = continent, 10=city, 21=building
(openstreetmap limit of 18)

The following maps show different map source/type options (except cloudmade)

The appearance of these maps may be very different depending on zoom/scale

stamen: watercolor



stamen: toner



stamen: terrain



If you can't get the map you want by adjusting the location/zoom variables, the functions designed for the different map sources provide more options:
`get_googlemap`,
`get_openstreetmap`,
`get_stamenmap`,
`get_cldmadeimap`

google: terrain



google: satellite



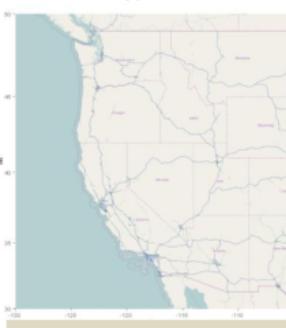
google: roadmap



google: hybrid



osm*



All maps can be displayed in black and white
`color = "bw"`



*Open street maps may return a '503 Service Unavailable' error. This means their server is unavailable, and you must wait for it to become available again.

`myMap <-
get_map(location=myLocation,
source="osm", color="bw")`

- If you use Rstudio:
Sometimes a plot will not display. Increasing the size of the plot window may help. `dev.off()` prior to plotting may also help.
- The `urlonly = TRUE` will return a url that will display your map in a web browser. Which is pretty cool and may be handy!
- `legend="topleft"` will inset the legend on the top left of the map if data is overlaid (page 2).

ggmap: Spatial Visualization with ggplot2



ggmap quickstart

Part 2: Plotting the maps and data

1 . Plot the raster:

```
ggmap(myMap)
```

2 . Add points with latitude/longitude coordinates:

```
ggmap(myMap)+  
  geom_point(aes(x = Longitude, y = Latitude), data = data,  
             alpha = .5, color = "darkred", size = 3)  
  alpha = transparency  
  color = color  
  size = size of points
```

The `size`, `color`, `alpha`, and `shape` of the points can be scaled relative to another variable (in this case `estArea`) within the `aes` function:

```
ggmap(myMap)+  
  geom_point(aes(x = Longitude, y = Latitude, size=sqrt(estArea)),  
             data = data, alpha = .5, color="darkred")
```



Additional functions can be added to control scaling, e.g.:

```
ggmap(myMap)+  
  geom_point(aes(x = Longitude, y = Latitude, size=sqrt(estArea)),  
             data = data, alpha = .5, color="darkred") +  
  scale_size(range=c(3,20))
```

3 . Add polygons from shp file

The shp file is imported into R using the rgdal package, and must be transformed to geographic coordinates (latitude/longitude) on the [World Geodetic System](#) of 1984 (WGS84) datum using the rgdal package:

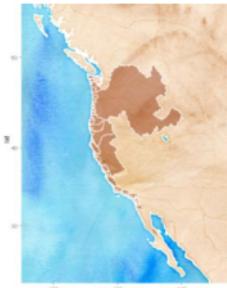
```
library(rgdal)  
shpData <- readOGR(dsn="C:\\Documents and Settings\\Watershed", layer="WS")  
proj4string(shpData) # describes data's current coordinate reference system  
# to change to correct projection:  
shpData <- spTransform(shpData,  
CRS("+proj=longlat +datum=WGS84"))
```

To plot the data:

```
geom_polygon(aes(x = long, y = lat, group=id),  
            data = shpData, color = "white", fill = "orangered4",  
            alpha = .4, size = .2)
```

`color`=outline color
`fill`=polygon color

`alpha`=transparency
`size`=outline size



4 . Annotate figure

```
baylor <- get_map("baylor university", zoom = 15, maptype = 'satellite')  
ggmap(baylor) +  
  annotate('rect', xmin=-97.11, ymin=31.54, xmax=-97.12, ymax=31.55, col="red", fill="white") +  
  annotate('text', x=-97.12, y=31.54, label = 'Statistical Sciences', colour = l('red'), size = 8) +  
  annotate('segment', x=-97.12, xend=-97.12, y=31.55, yend=31.55,  
          colour=l('red'), arrow = arrow(length=unit(0.3, "cm")), size = 1.5) +  
  labs(x = 'Longitude', y = 'Latitude') + ggtitle('Baylor University')
```

Controlling size and color

`size` scale_size(range = c(3, 20))
But, the following is better for display because it is based on area (rather than radius)
scale_area(range=c(3,20))

`color` Continuous variables: color gradient between n colors, e.g.:

scale_colour_gradientn(colours = rainbow_hcl(7))

Discrete variables, e.g.:

scale_colour_manual(values=rainbow_hcl(7))
scale_colour_manual(values=c("8" = "red", "4" = "blue", "6" = "green"))

*Use colorspace and RColorBrewer to choose color combinations



Page 2 ggmap,
Melanie Frazier

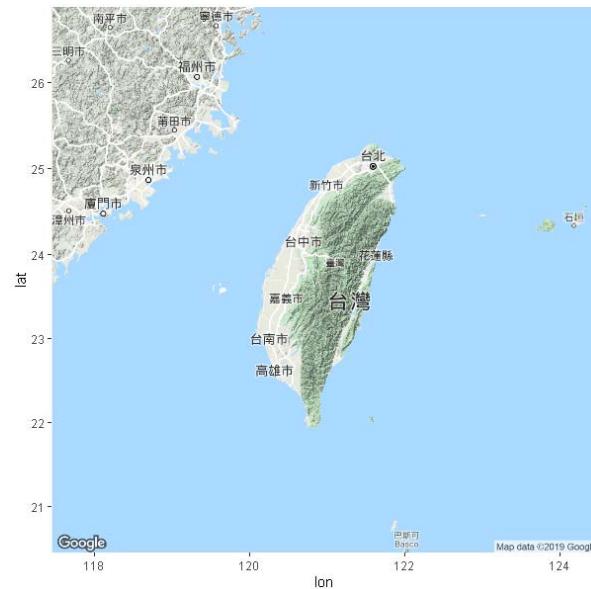
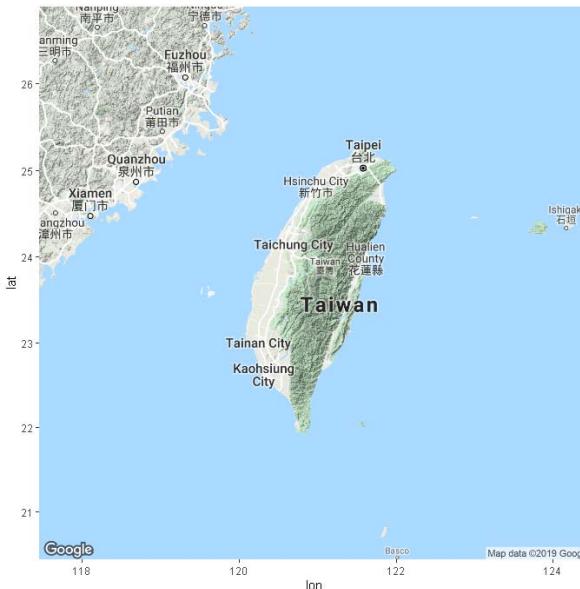


以ggmap獲取google提供的台灣地圖

14/54

ggmap: A collection of functions to visualize spatial data and models on top of static maps from various online sources (e.g Google Maps and Stamen Maps). It includes tools common to those tasks, including functions for geolocation and routing.

```
> library(ggmap)
> library(mapproj)
> tw.map <- get_map(location = 'Taiwan', zoom = 7)
Source :
https://maps.googleapis.com/maps/api/staticmap?center=Taiwan&zoom=7&size=640x640&scale=2&map
type=terrain&language=en-EN&key=AIzaSyCuYcvrytmKLGN
Source :
https://maps.googleapis.com/maps/api/geocode/json?address=Taiwan&key=AIzaSyCuYcvrytmKLGN
> ggmap(tw.map)
> tw.map.zh <- get_map(location = 'Taiwan', zoom = 7, language = "zh-TW")
> ggmap(tw.map.zh)
```

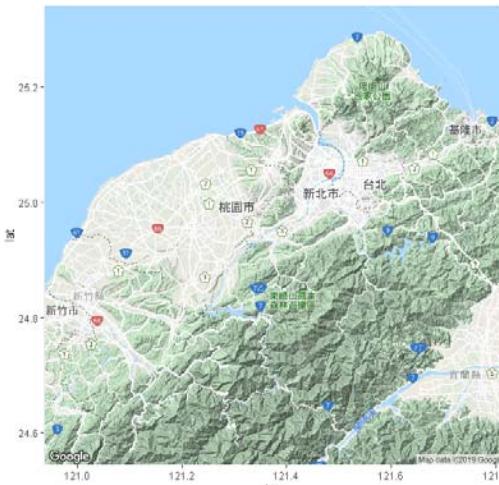




不同的地圖呈現方式

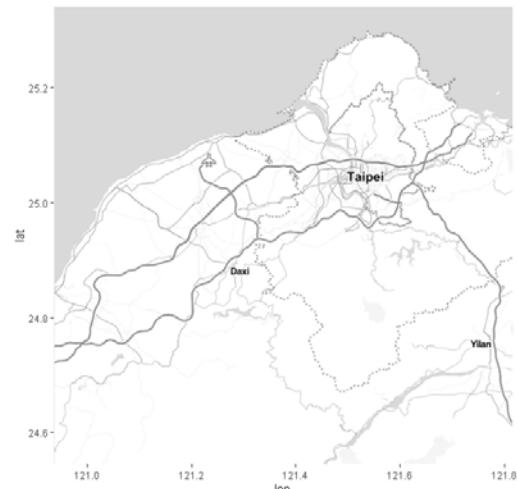
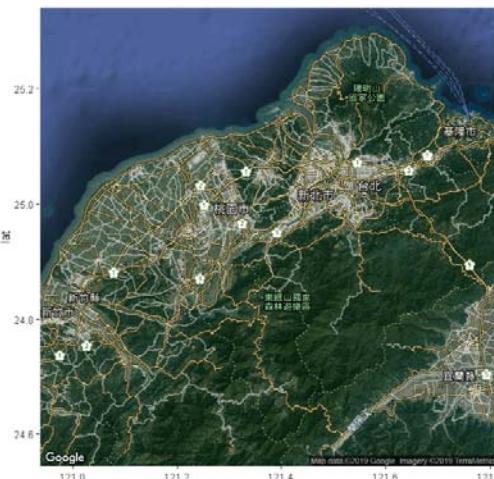
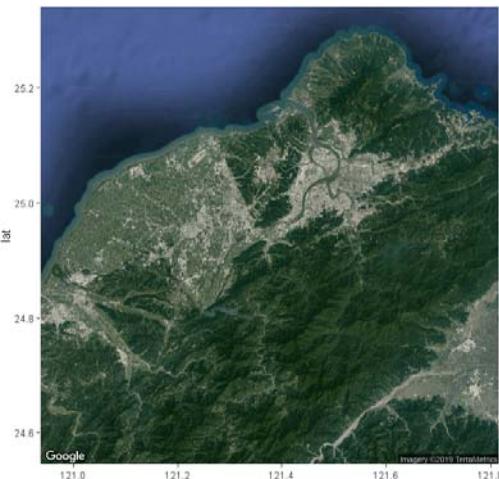
15/54

```
> tw.map.ntpu <- get_map(location = c(lon = 121.374925, lat = 24.943403),  
+ zoom = 10, language = "zh-TW" , maptype = "terrain")  
> ggmap(tw.map.ntpu)
```



roadmap
satellite
hybrid
toner-lite

國立臺北大學三峽校區。
新北市三峽區大學路151號。
經緯度 : (121.374925,24.943403)





下載台灣各地紫外線即時監測資料

16/54

```

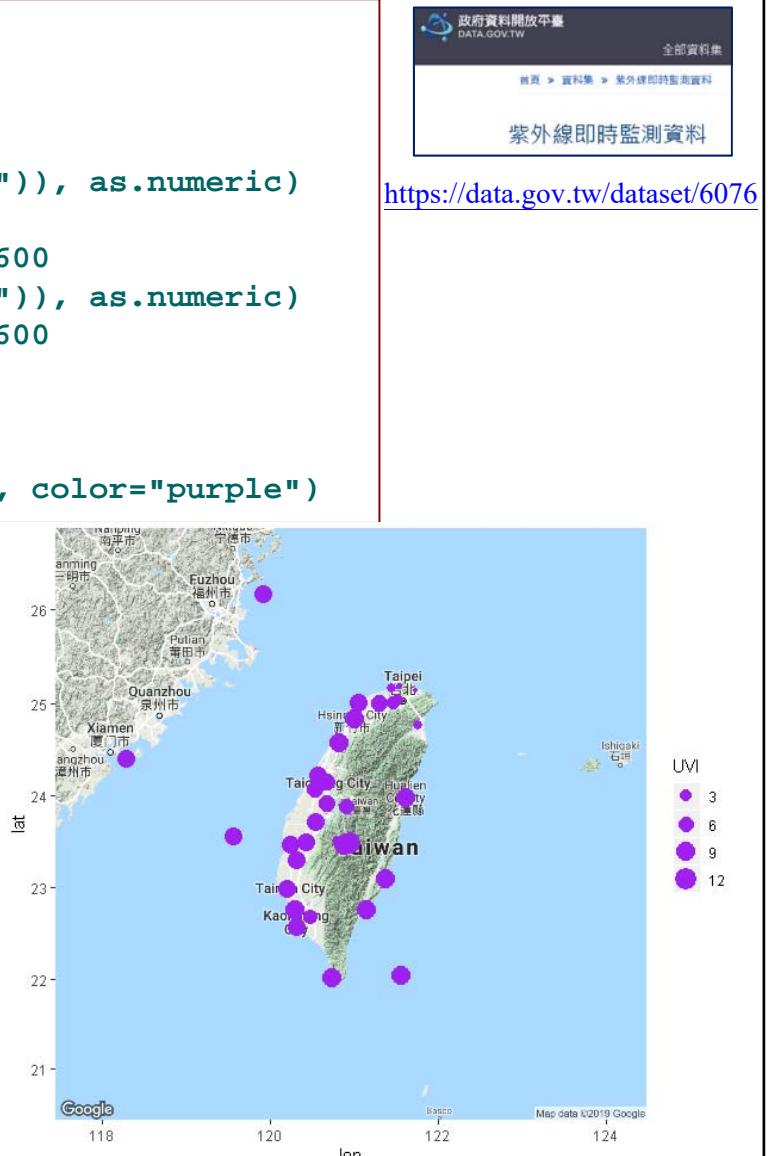
uv <- read.csv("data/UV_20191016130309.csv")
head(uv)

# 經緯度(度分秒) => 度數
lon.deg <- sapply(strsplit(as.character(uv$WGS84Lon), ",")), as.numeric)
lon.deg
uv$lon <- lon.deg[1, ] + lon.deg[2, ]/60 + lon.deg[3, ]/3600
lat.deg <- sapply(strsplit(as.character(uv$WGS84Lat), ",")), as.numeric)
uv$lat <- lat.deg[1, ] + lat.deg[2, ]/60 + lat.deg[3, ]/3600

tw.map <- get_map(location = 'Taiwan', zoom = 7)
ggmap(tw.map) +
  geom_point(data = uv, aes(x = lon, y = lat, size = UVI), color="purple")

```

```
> Ton.deg
 [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12]
[1,] 121 119 120 120 120 121 121 120 121 121 121 121
[2,] 36 55 18 57 12 0 31 44 30 22 44 2
[3,] 48 24 57 34 17 51 47 47 53 24 26 51
 [,21] [,22] [,23] [,24] [,25] [,26] [,27] [,28] [,29]
[1,] 120.00 120.00 120.0 120.00 120.00 120.00 120.00 120.0 120.00 12
[2,] 29.00 18.00 19.0 14.00 52.00 48.00 32.00 41.0 32.00 3
[3,] 16.92 20.48 2.1 52.12 50.06 5.02 41.98 7.1 29.47
```



參考: <https://blog.gtwang.org/r/r-ggmap-package-spatial-data-visualization/>

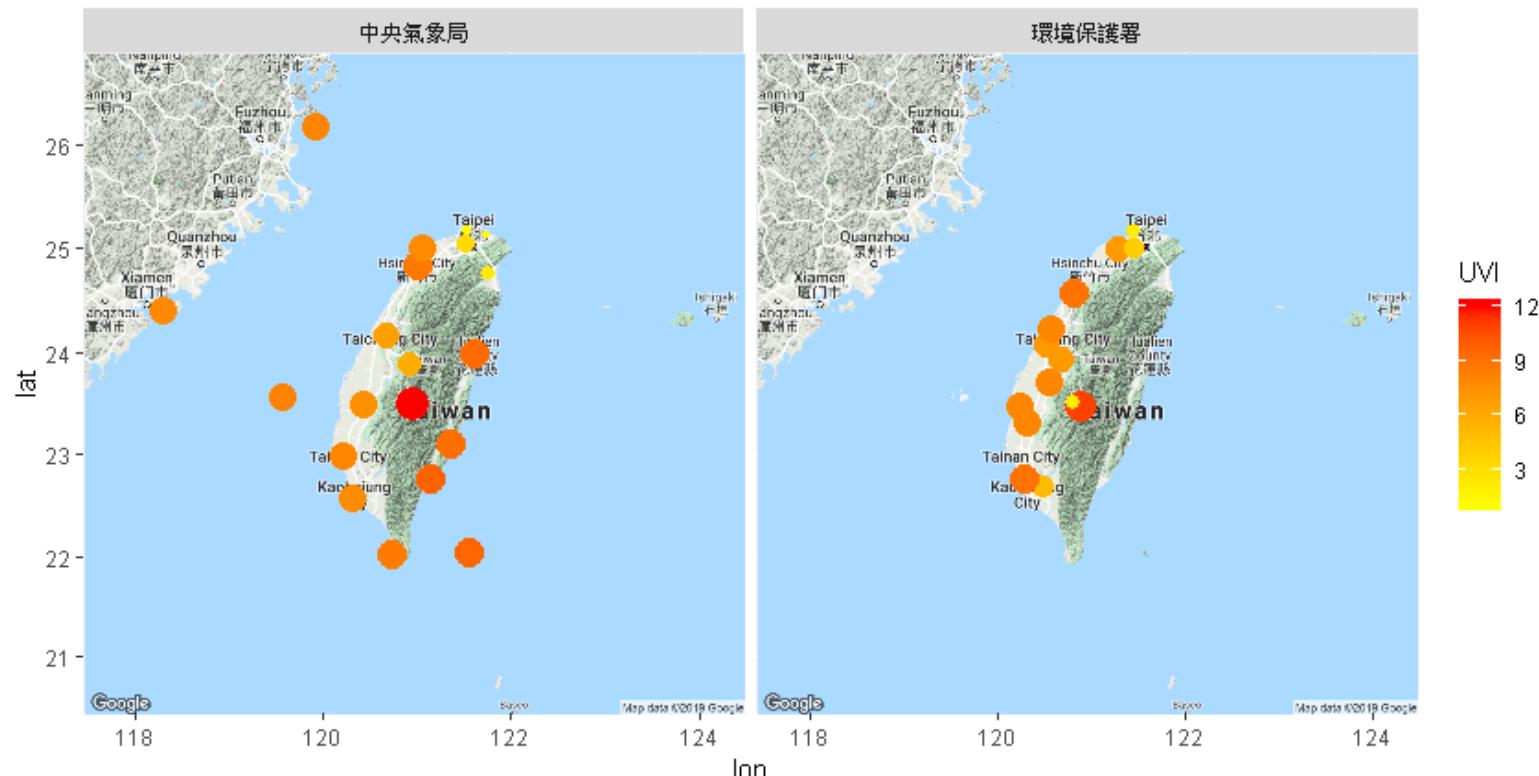
<https://hmwu.idv.tw>



繪製台灣各地紫外線指數地圖

17/54

```
ggmap(tw.map) +  
  geom_point(data = uv, aes(x = lon, y = lat, size = UVI, color = UVI)) +  
  scale_color_continuous(low = "yellow", high = "red") +  
  facet_grid(~PublishAgency) +  
  guides(size = FALSE)
```



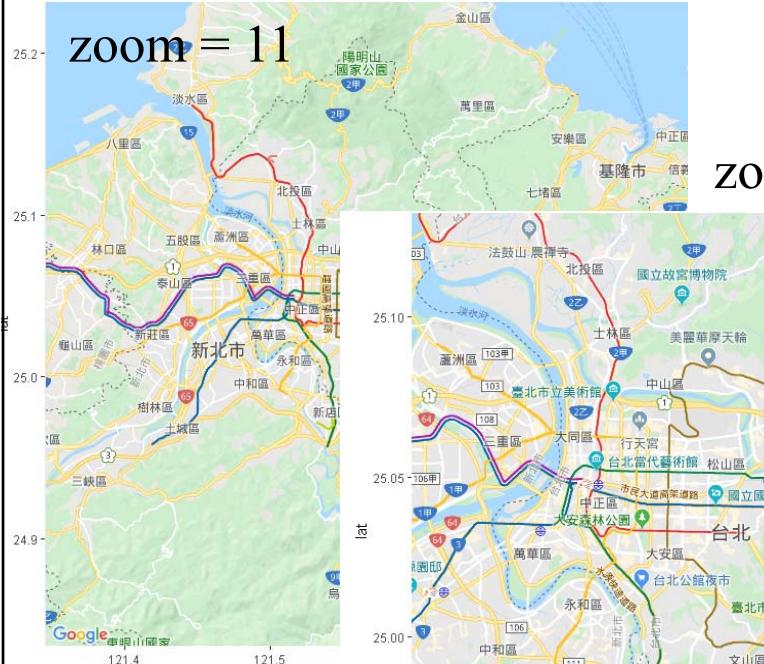
讀取中文檔問題:

```
Sys.setlocale(category = "LC_ALL", locale = "zh_TW.big5")
```

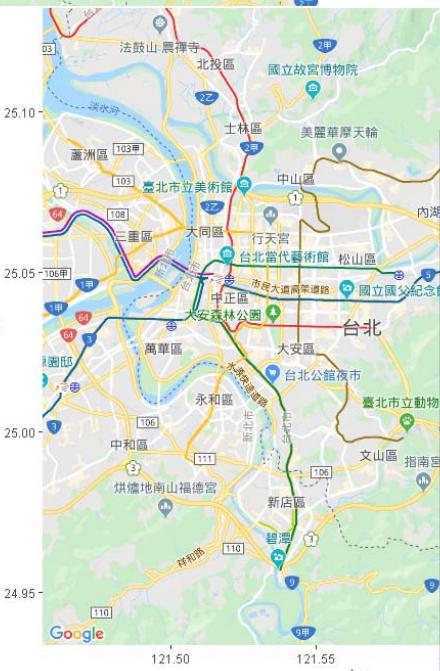


不同放大比例(zoom)的台北市地圖

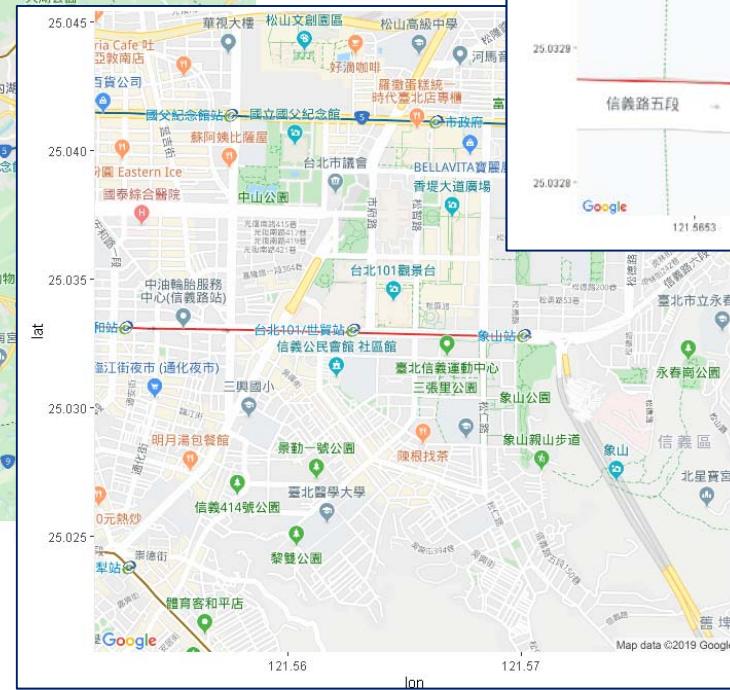
```
tpe.map.zh11 <- get_map(location = 'Taipei', zoom = 11, maptype = "roadmap",
language = "zh-TW")
ggmap(tpe.map.zh11)
```



zoom = 12



zoom = 15



zoom = 21

map zoom: an integer from 3 (continent) to 21 (building), default value 10 (city).



Dot Density Maps

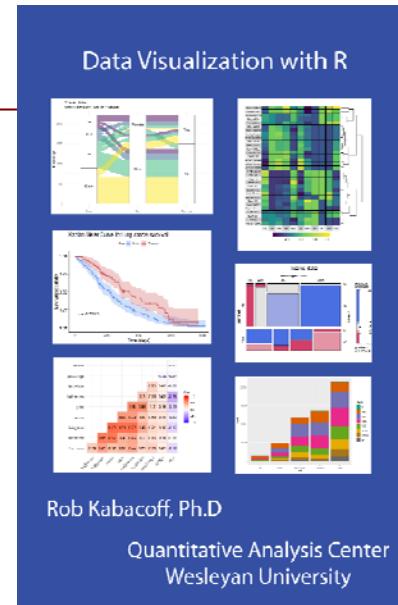
```
> head(crime)
```

	time	date	hour	premise	offense	beat	block	street	type	suffix	number	month
					murder	15E30	9600-9699	marlive	ln	-	1 january	
82729	2010-01-01	14:00:00	1/1/2010	0	18A							
82730	2010-01-01	14:00:00	1/1/2010	0	13R	robbery	13D10	4700-4799	telephone	rd	-	1 january
82731	2010-01-01	14:00:00	1/1/2010	0	20R	aggravated assault	16E20	5000-5099	wickview	ln	-	1 january
82732	2010-01-01	14:00:00	1/1/2010	0	20R	aggravated assault	2A30	1000-1099	ashland	st	-	1 january
82733	2010-01-01	14:00:00	1/1/2010	0	20A	aggravated assault	14D20	8300-8399	canyon	-	-	1 january
82734	2010-01-01	14:00:00	1/1/2010	0	20R	burglary	18F60	9300-9399	rowan	ln	-	1 january
	day		location		address	lon	lat					
82729	friday	apartment	parking lot	9650	marlive ln	-95.43739	29.67790					
82730	friday	road / street	/ sidewalk	4750	telephone rd	-95.29888	29.69171					
82731	friday	residence / house		5050	wickview ln	-95.45586	29.59922					
82732	friday	residence / house		1050	ashland st	-95.40334	29.79024					
82733	friday	apartment		8350	canyon	-95.37791	29.67063					
82734	friday	residence / house		9350	rowan ln	-95.54830	29.70223					

The crime dataset from the **ggmap** package, contains the time, date, and location of six types of crimes in Houston, Texas between January 2010 and August 2010.

```
> summary(crime$offense)
aggravated assault auto theft burglary murder rape robbery theft
    7177    7946      17802     157    378    6298    46556
>
> library(dplyr)
> rapes <- filter(crime, offense == "rape") %>%
+   select(date, offense, address, lon, lat)
> head(rapes)
  date offense           address      lon      lat
1 1/1/2010    rape 5950 glenmont dr -95.48498 29.72007
2 1/1/2010    rape 2350 sperber ln -95.34817 29.75505
...
6 1/2/2010    rape 1150 fidelity st -95.25535 29.74147
```

```
> # install.packages("ggmap")
> library(ggmap)
> head(crime)
> dim(crime)
[1] 86314    17
```



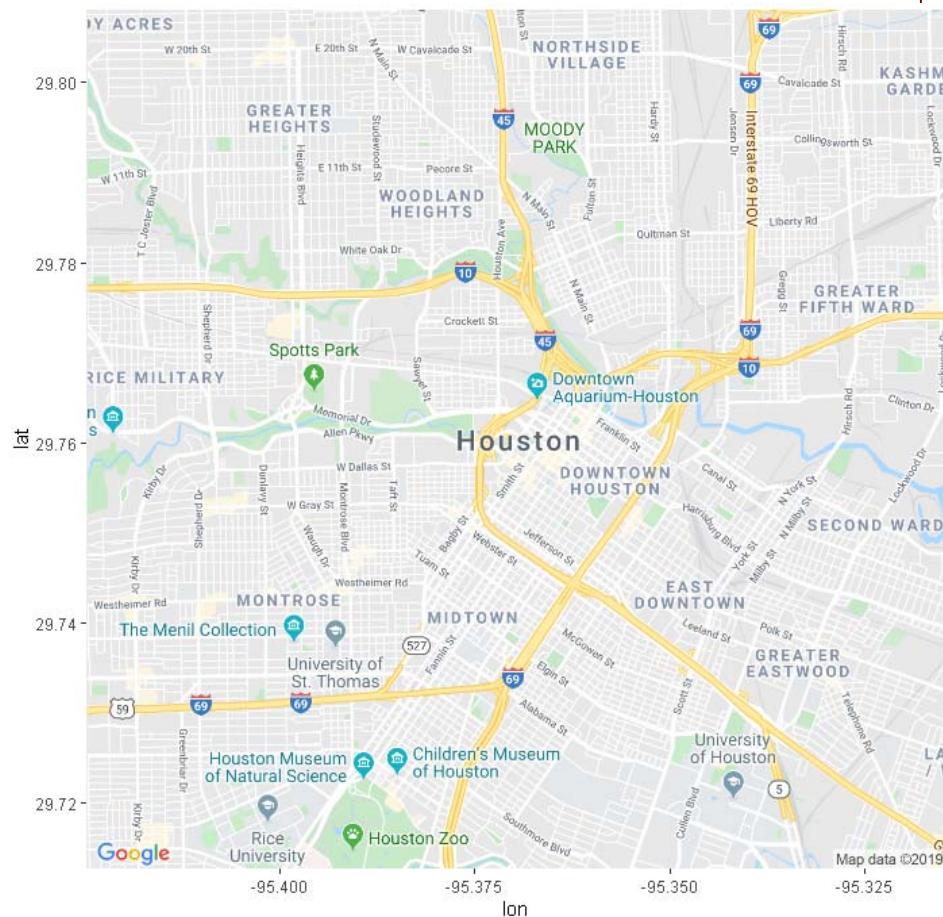


Get the background map image

```
> # (1) Find the center coordinates for Houston, TX  
> houston_center <- geocode("Houston, TX")  
Source :  
https://maps.googleapis.com/maps/api/geocode/json?address=Houston,+TX&key=AIzaSyCuYcvrytmKLGN  
> register_google(key = "AIzaSyCuYcvrytmKLGN1Z_y8Dh4WwFOeB6wnof8", write = TRUE)  
Replacing old key (AIzaSyCuYcvrytmKLGN) with new key in C:/Users/userpc/Documents/.Renviron  
> houston_center  
# A tibble: 1 x 2  
  lon     lat  
  <dbl> <dbl>  
1 -95.4   29.8
```

```
> has_google_key()  
[1] TRUE  
> google_key()  
[1] "AIzaSyCuYcvrytmKLGN"
```

```
houston_map <- get_map(houston_center,  
                        zoom = 13,  
                        maptype = "roadmap")  
ggmap(houston_map)
```

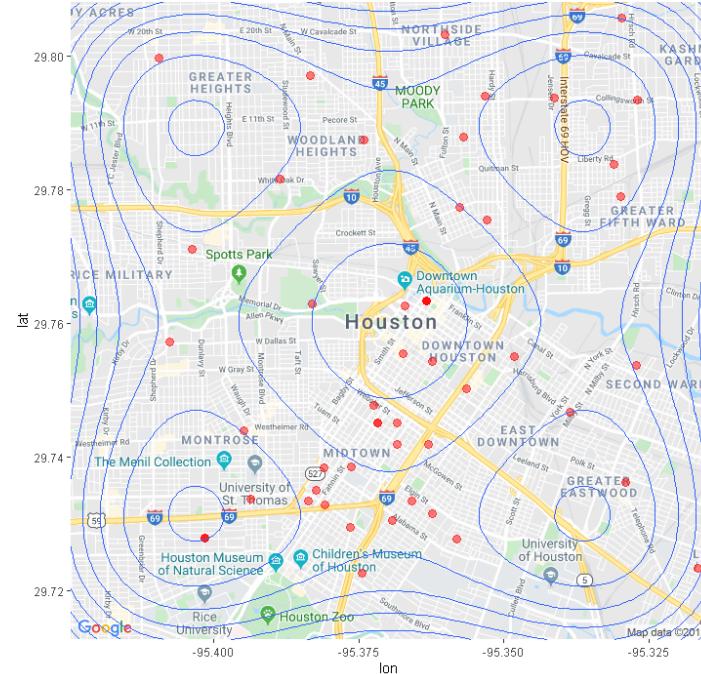
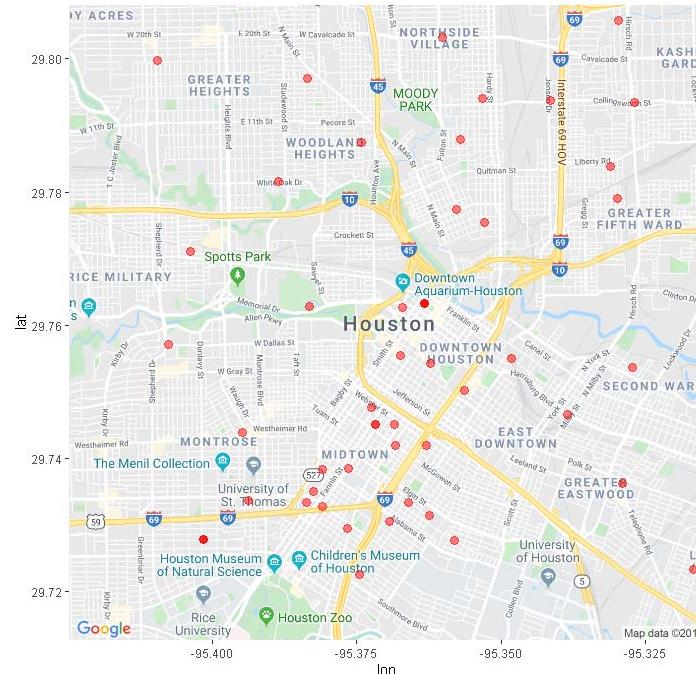




Add crime locations to the map

```
ggmap(houston_map,  
       base_layer = ggplot(data = rapes, aes(x=lon, y = lat))) +  
       geom_point(color = "red", size = 3, alpha = 0.5)
```

```
ggmap(houston_map) +  
  geom_point(data = rapes, aes(x=lon, y = lat),  
             color = "red", size = 3, alpha = 0.5) +  
  geom_density2d(size = 0.3)
```





Remove long and lat numbers and add titles

```
ggmap(houston_map,
  base_layer = ggplot(data = rapes, aes(x=lon, y = lat))) +
  geom_point(color = "red", size = 3, alpha = 0.5) +
  theme_void() +
  labs(title = "Location of reported rapes",
       subtitle = "Houston Jan - Aug 2010",
       caption = "source: - Roads: I-45, I-69, I-10, FM 527, FM 59, FM 1960, and various local streets like Allen Parkway, Main Street, and Congress Avenue.


Map data ©2019



source: http://www.houstontx.gov/police/cs/


```

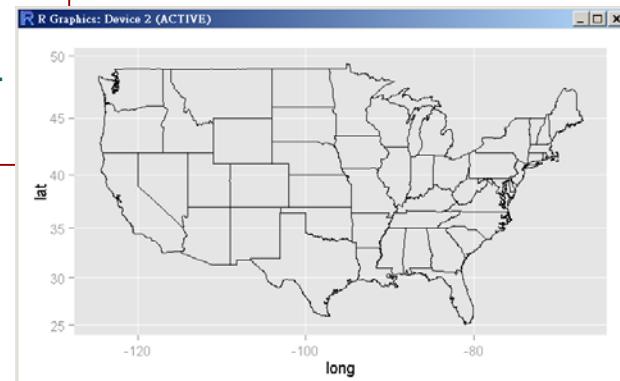
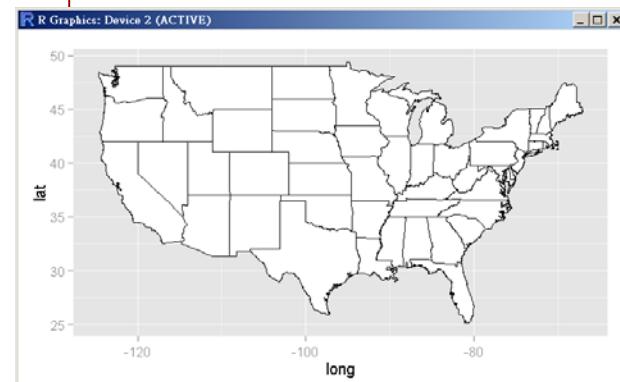


分級著色圖 /面量圖(Choropleth Maps)

Creating a Map

23/54

```
> library(ggplot2)
> library(maps)
> library(mapproj)
> states.map <- map_data("state")
> head(states.map, 3)
  long      lat group order  region subregion
1 -87.46201 30.38968     1    1 alabama      <NA>
2 -87.48493 30.37249     1    2 alabama      <NA>
3 -87.52503 30.37249     1    3 alabama      <NA>
> tail(states.map, 3)
  long      lat group order  region subregion
15597 -107.9223 41.01805   63 15597 wyoming      <NA>
15598 -109.0568 40.98940   63 15598 wyoming      <NA>
15599 -109.0511 40.99513   63 15599 wyoming      <NA>
> ggplot(states.map, aes(x=long, y=lat, group=group)) +
  geom_polygon(fill="white", colour="black")
> ggplot(states.map, aes(x=long, y=lat, group=group)) +
  geom_path() + coord_map("mercator")
```



mercator: equally spaced straight meridians, conformal, straight compass courses

Source: 13.17. Creating a Map, R Graphics Cookbook 2nd

Creating a Map



```

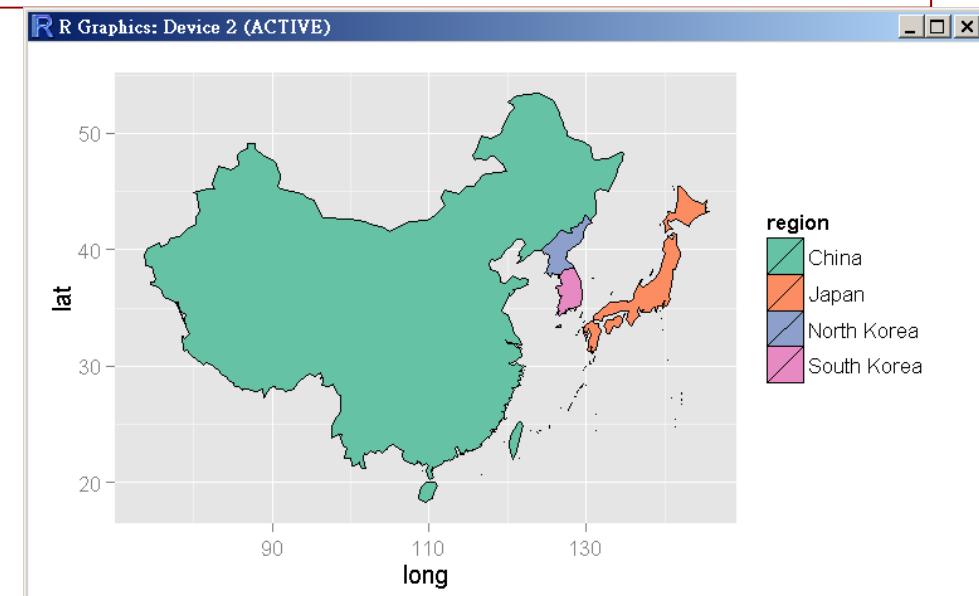
> # map_data: county, france, italy, nz, state, usa, world, world2.
> # region names
> world.map <- map_data("world")
> sort(unique(world.map$region))
[1] "Afghanistan"           "Albania"
[3] "Algeria"                "American Samoa"
[5] "Andaman Islands"        "Andorra"
...
> east.asia <- map_data("world", region=c("Japan", "China", "North Korea",
"South Korea"))
> ggplot(east.asia, aes(x=long, y=lat, group=group, fill=region)) +
  geom_polygon(colour="black") +
  scale_fill_brewer(palette="Set2")

```

NOTE:

See the **mapdata** package for more map data sets. It includes maps of China and Japan, as well as a high-resolution world map, **worldHires**.

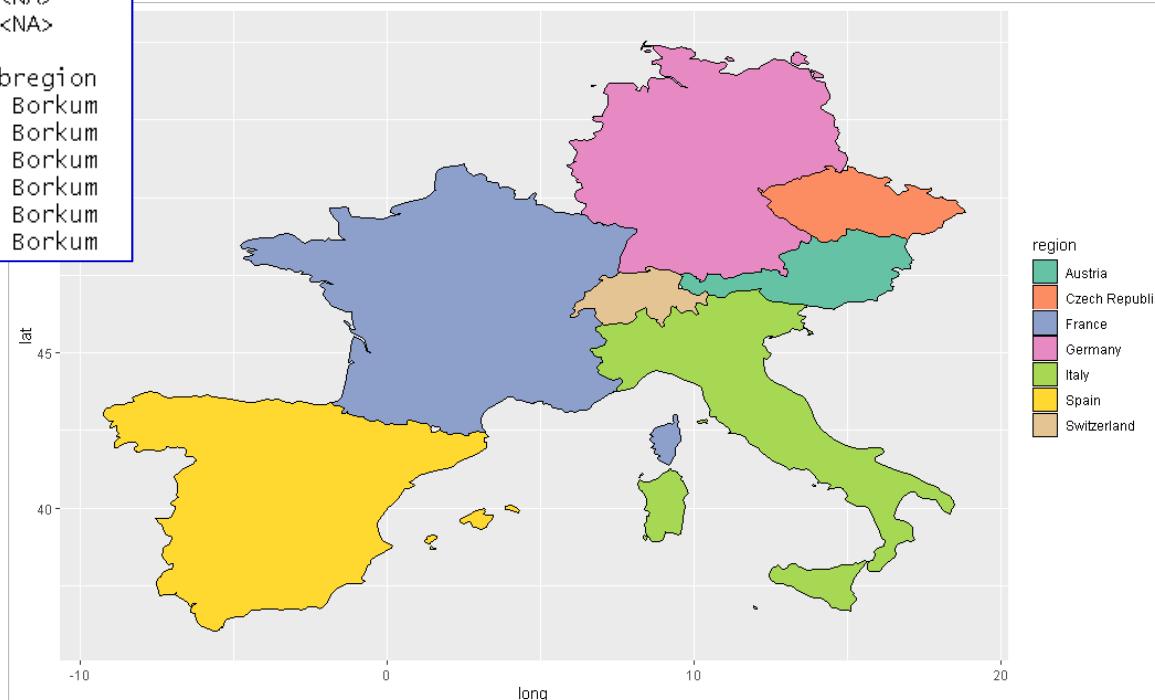
See Also: **mapproject**, **map**



分級著色圖 / 面量圖(Choropleth Maps)

```
> country <- c("France", "Austria", "Italy", "Switzerland", "Germany", "Spain", "Czech Republic")
> mymapdata <- map_data("world", region = country)
> ggplot(mymapdata, aes(x = long, y = lat, group = group, fill = region)) +
  geom_polygon(colour = "black") +
  scale_fill_brewer(palette = "Set2")
```

```
> head(mymapdata)
#> #>   long     lat group order region subregion
#> #> 1 16.95312 48.59883    1     1 Austria      <NA>
#> #> 2 16.94883 48.58858    1     2 Austria      <NA>
#> #> 3 16.94336 48.55093    1     3 Austria      <NA>
#> #> 4 16.90449 48.50352    1     4 Austria      <NA>
#> #> 5 16.86270 48.44141    1     5 Austria      <NA>
#> #> 6 16.86543 48.38691    1     6 Austria      <NA>
> tail(mymapdata)
#> #>   long     lat group order region subregion
#> #> 2941 6.734766 53.58252    26  2941 Germany    Borkum
#> #> 2942 6.642090 53.57920    26  2942 Germany    Borkum
#> #> 2943 6.668555 53.60566    26  2943 Germany    Borkum
#> #> 2944 6.754590 53.62549    26  2944 Germany    Borkum
#> #> 2945 6.800879 53.62549    26  2945 Germany    Borkum
#> #> 2946 6.734766 53.58252    26  2946 Germany    Borkum
```





Violent Crime Rates by US State (USArrests)

Violent Crime Rates by US State (USArrests): the data set contains statistics, in arrests per 100,000 residents for assault, murder, and rape in each of the 50 US states in 1973. Also given is the percent of the population living in urban areas.

```
> head(USArrests, 3)
      Murder Assault UrbanPop Rape
Alabama     13.2     236      58 21.2
Alaska      10.0     263      48 44.5
Arizona      8.1     294      80 31.0

> crimes <- data.frame(state = tolower(rownames(USArrests)), USArrests)
> head(crimes, 3)
      state Murder Assault UrbanPop Rape
Alabama    alabama   13.2     236      58 21.2
Alaska      alaska    10.0     263      48 44.5
Arizona     arizona    8.1     294      80 31.0

> library(maps); library(ggmap)
> states.map <- map_data("state")
> head(states.map, 3)
      long     lat group order  region subregion
1 -87.46201 30.38968     1     1 alabama      <NA>
2 -87.48493 30.37249     1     2 alabama      <NA>
3 -87.52503 30.37249     1     3 alabama      <NA>

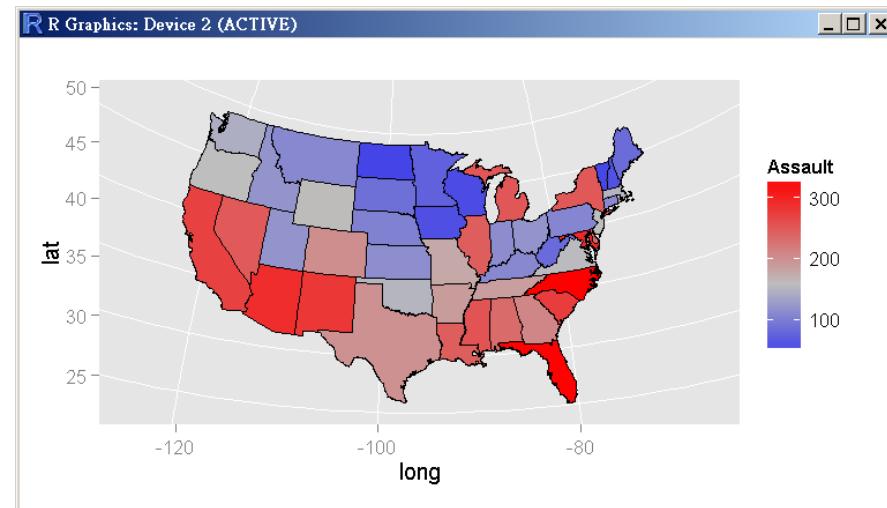
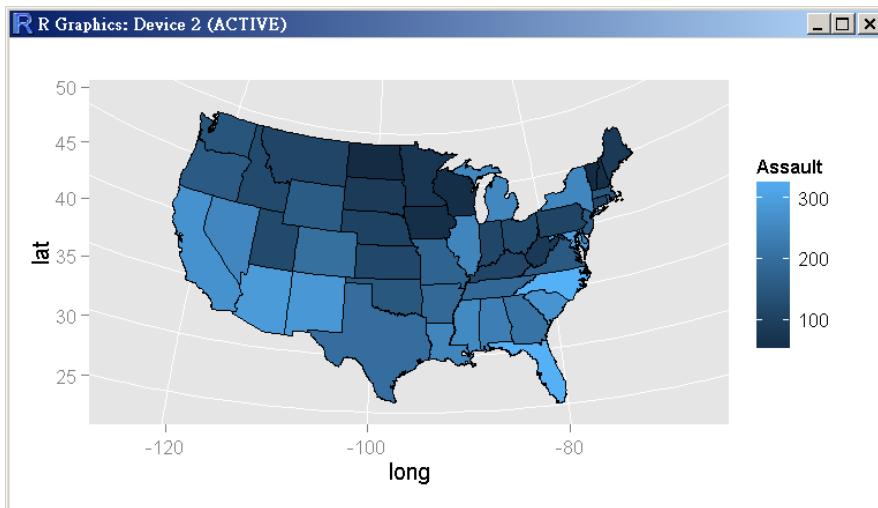
> crime.map <- merge(states.map, crimes, by.x="region", by.y="state")
> head(crime.map, 3)
      region     long     lat group order subregion Murder Assault UrbanPop Rape
1 alabama -87.46201 30.38968     1     1      <NA>   13.2     236      58 21.2
2 alabama -87.48493 30.37249     1     2      <NA>   13.2     236      58 21.2
3 alabama -87.95475 30.24644     1    13      <NA>   13.2     236      58 21.2
```

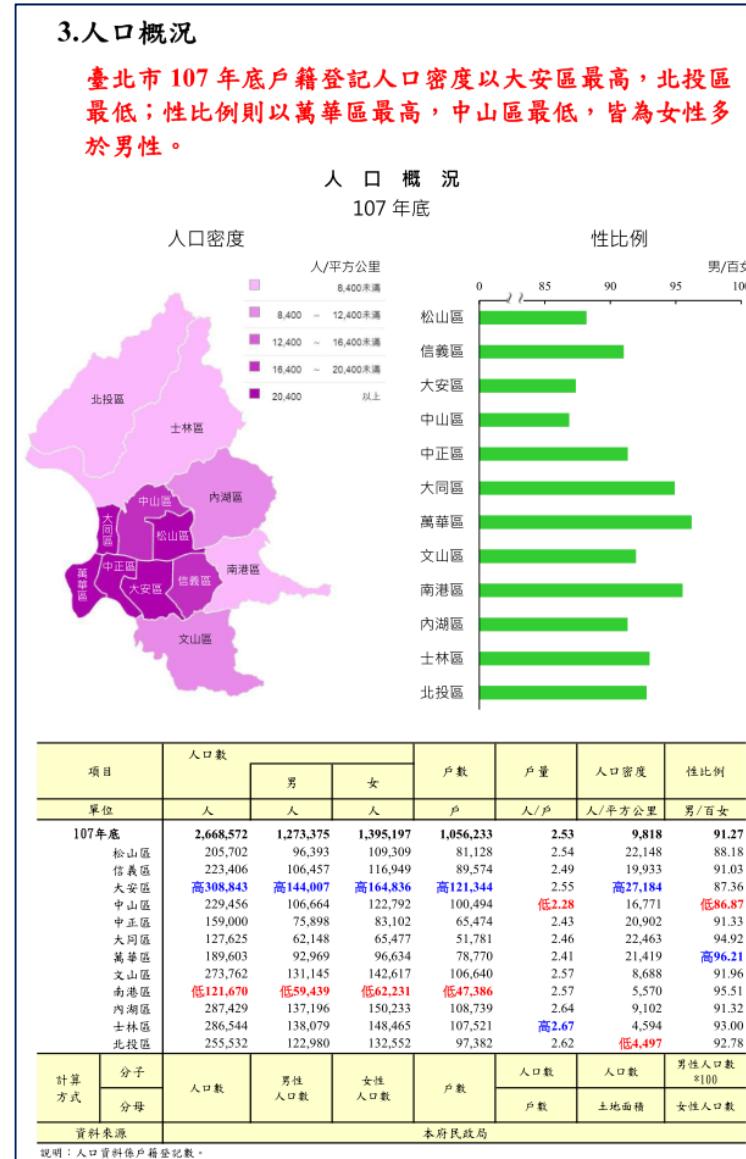
分級著色圖 /面量圖(Choropleth Maps)



```
> library(dplyr)
> crime.map <- arrange(crime.map, group, order)
> head(crime.map, 3)
  region      long     lat group order subregion Murder Assault UrbanPop Rape
1 alabama -87.46201 30.38968     1     1       <NA>   13.2    236      58 21.2
2 alabama -87.48493 30.37249     1     2       <NA>   13.2    236      58 21.2
3 alabama -87.52503 30.37249     1     3       <NA>   13.2    236      58 21.2
> ggplot(crime.map, aes(x=long, y=lat, group=group, fill=Assault)) +
  geom_polygon(colour="black") +
  coord_map("polyconic")
```

```
ggplot(crime.map, aes(x=long, y=lat, group=group, fill=Assault)) +
  geom_polygon(colour="black") +
  coord_map("polyconic") +
  scale_fill_gradient2(low="blue", mid="grey", high="red",
  midpoint=median(crimes$Assault))
```







Shapefile圖形格式

ESRI Shapefile (shp)，或簡稱shapefile，是美國環境系統研究所公司（ESRI）開發的空間資料開放格式 (<https://zh.wikipedia.org/wiki/Shapefile>)

GADM.org is a spatial database of the location of the world's administrative areas (or administrative boundaries) for use in GIS and similar software.

https://gadm.org/download_country_v3.html

Download GADM data (version 3.6)

Country: Taiwan

Geopackage
Shapefile
R(sp): level-0, level1, level2
R(sf): level-0, level1, level2
KMZ: level-0, level1, level2

three levels of SpatialPolygonDataFrame for Taiwan: gadm36_TWN_shp.zip:

- gadm36_TWN_2.cpg
- gadm36_TWN_2.dbf
- gadm36_TWN_2.prj
- gadm36_TWN_2.shp
- gadm36_TWN_2.shx

The coordinate reference system is [longitude/latitude](#) and the [WGS84](#) datum.
Description of [File Formats](#).

地圖Shape資料:

- shp: 用於儲存地圖元素的幾何資料。
- shx: 幾何資料索引，記錄每一shp檔案之中的位置。
- dbf: 以dBase IV的資料表格式儲存每個幾何形狀的屬性資料。
- prj (選項): shp檔中幾何資料使用的經緯度座標系統。

政府資料開放平台

鄉鎮市區界線(TWD97經緯度)

<https://data.gov.tw/dataset/7441>

直轄市、縣市界線(TWD97經緯度)

<https://data.gov.tw/dataset/7442>

mapdata201907310953.zip

Metadata.xml

TOWN_MOI_1080726.dbf

TOWN_MOI_1080726.prj

TOWN_MOI_1080726.shp

TOWN_MOI_1080726.shx

TW-07-301000100G-614001.xml

相關資料集

資料集評分: ★★★★☆ 平均3.8 (68 人次投票)

資料集描述: 我國各鄉(鎮、市、區)行政區域界線圖資

主要欄位說明: TOWNID · TOWNCODE · COUNTYNAME · TOWNNAME · TOWNENG · COUNTYID · COUNTYCODE

資料下載地址: [SHP](#) [檢視資料](#) 鄉(鎮、市、區)界線(TWD97經緯度).....

提供機關: 內政部國土測繪中心

零時政府: 台灣行政區域圖(Country, Town, Village)

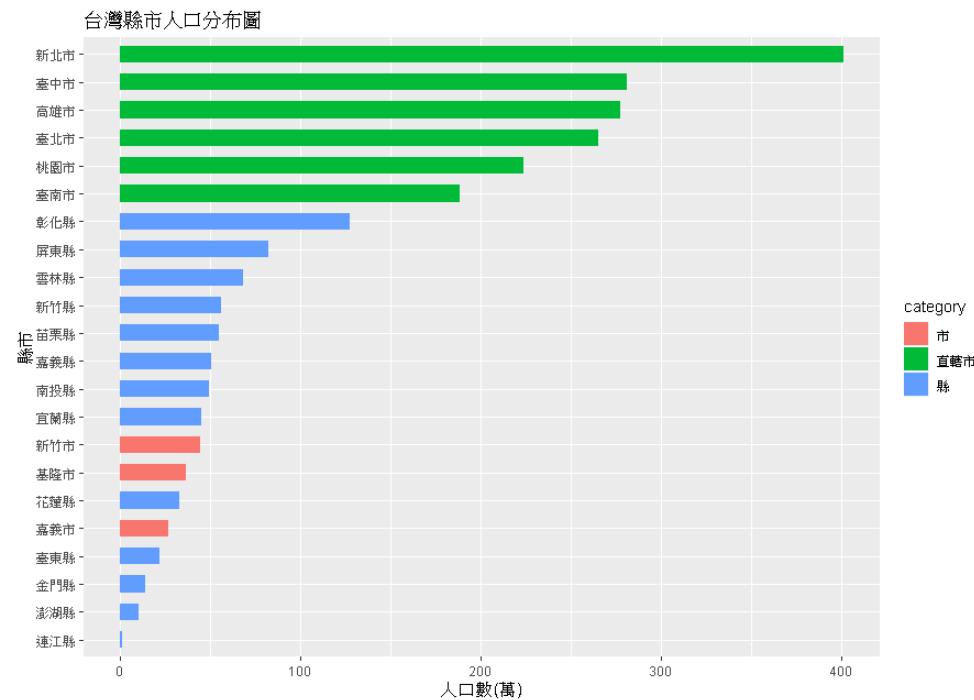
<https://github.com/g0v/twgeojson/tree/master/json>



台灣各縣市人口分布長條圖

30/54

```
TW.Pop2019 <- read.csv("TW_Population2019.csv")
head(TW.Pop2019)
colnames(TW.Pop2019) <- c("rank", "city", "category", "population")
ggplot(TW.Pop2019, aes(x = reorder(city, population), y = population/10000,
fill = category)) +
  geom_bar(stat="identity", width=0.6) +
  coord_flip() +
  labs(title = "台灣縣市人口分布圖", x = "縣市", y = "人口數(萬)")
```



參考: How to plot a Taiwan Map colored with Population by ggplot2, Ozu Shi, 2018/01/13
<http://www.rpubs.com/OzuShi/348822>



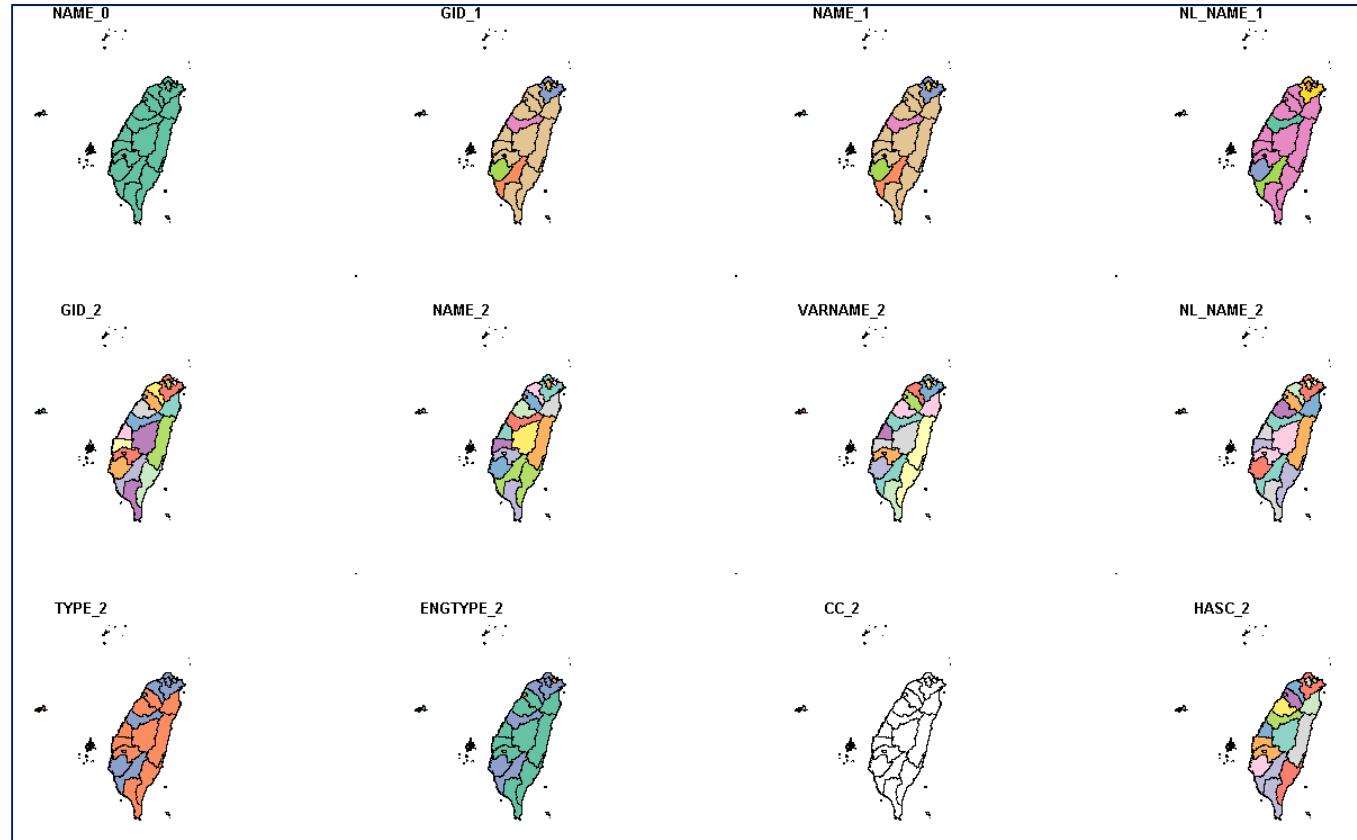
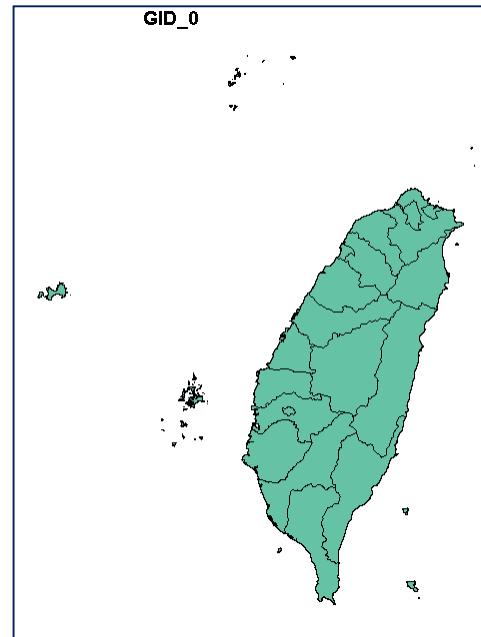
讀取台灣地圖Shapefile檔

31/54

```
> library(sf)
> taiwan.map <- st_read("data/gadm36_TWN_shp/gadm36_TWN_2.shp")
Reading layer `gadm36_TWN_2' from data source
`E:\TaipeiCityMap\data\gadm36_TWN_shp\gadm36_TWN_2.shp' using driver `ESRI Shapefile'
Simple feature collection with 22 features and 13 fields
geometry type:  MULTIPOLYGON
dimension:      XY
bbox:           xmin: 116.71 ymin: 20.6975 xmax: 122.1085 ymax: 26.38542
epsg (SRID):   4326
proj4string:    +proj=longlat +datum=WGS84 +no_defs
> head(taiwan.map)
Simple feature collection with 6 features and 13 fields
...
proj4string:    +proj=longlat +datum=WGS84 +no_defs
  GID_0 NAME_0   GID_1      NAME_1 NL_NAME_1      GID_2      NAME_2
  VARNAME_2 NL_NAME_2
1  TWN Taiwan TWN.1_1     Fujian      福建 TWN.1.1_1     Kinmen
J<U+012B>nmen Xian  金門縣
2  TWN Taiwan TWN.1_1     Fujian      福建 TWN.1.2_1  Lienkiang M<U+01CE>z<U+01D4>
Lied<U+01CE>o|Matsu Islands 馬祖列島
...
6  TWN Taiwan TWN.5_1     Tainan      台南 TWN.5.1_1     Tainan
Tainan City          台南
  TYPE_2          ENGTTYPE_2 CC_2      HASC_2
  1      Xian          County <NA> TW.FK.KM MULTIPOLYGON (((118.3999 24...
  2      Xian          County <NA> TW.FK.LK MULTIPOLYGON (((120.086 26....
...
6 Zhixiashi Special Municipality <NA> TW.TN.TI MULTIPOLYGON (((120.1387 22...
> dim(taiwan.map)
[1] 22 14
> print(taiwan.map, n = 22)
```

直接繪製Shapefile地圖檔

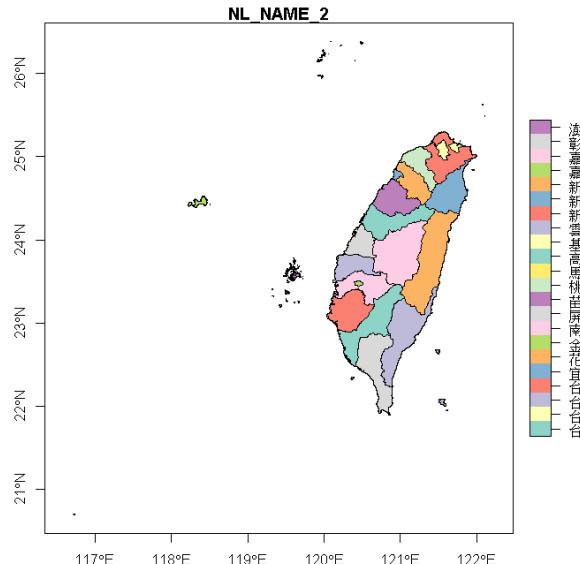
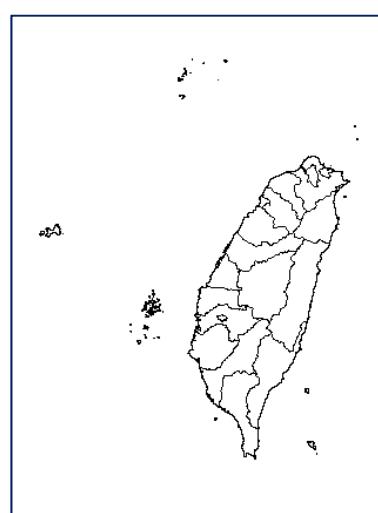
```
> plot(taiwan.map[1])  
> plot(taiwan.map[2:13], max.plot = 12)
```



從Shapefile檔中某一欄位繪製地圖 (plot)

33/54

```
> plot(st_geometry(taiwan.map))
> plot(taiwan.map["NL_NAME_2"], axes = TRUE)
> st_geometry(taiwan.map)
Geometry set for 22 features
geometry type:  MULTIPOLYGON
dimension:      XY
bbox:            xmin: 116.71 ymin: 20.6975 xmax: 122.1085 ymax: 26.38542
epsg (SRID):    4326
proj4string:    +proj=longlat +datum=WGS84 +no_defs
First 5 geometries:
MULTIPOLYGON (((118.3999 24.43569, 118.3979 24.....
MULTIPOLYGON (((120.086 26.38403, 120.086 26.38...
MULTIPOLYGON (((120.3326 22.5532, 120.3326 22.5...
MULTIPOLYGON (((122.0788 25.63431, 122.0788 25....
MULTIPOLYGON (((120.5007 24.26875, 120.5007 24.....
```





以ggplot繪製Shapefile地圖檔

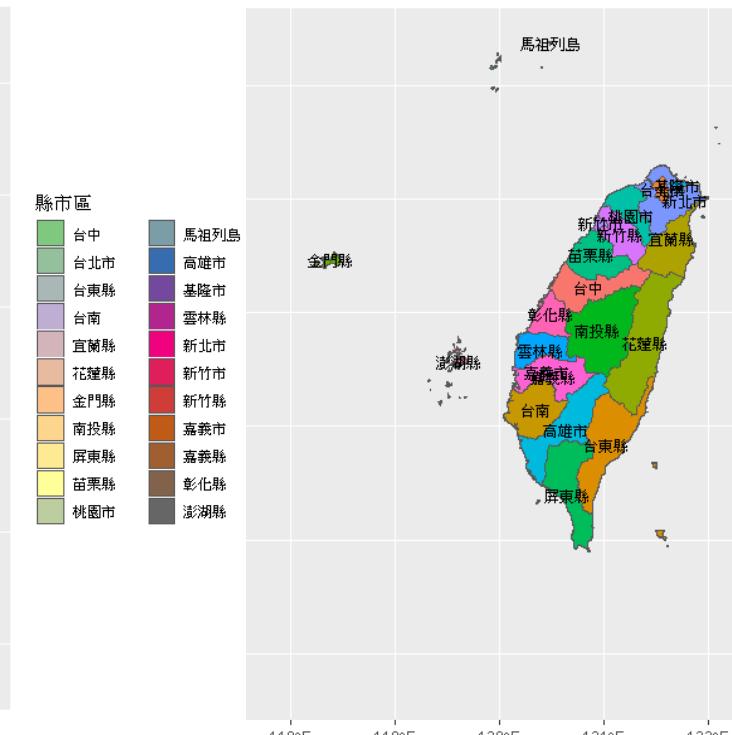
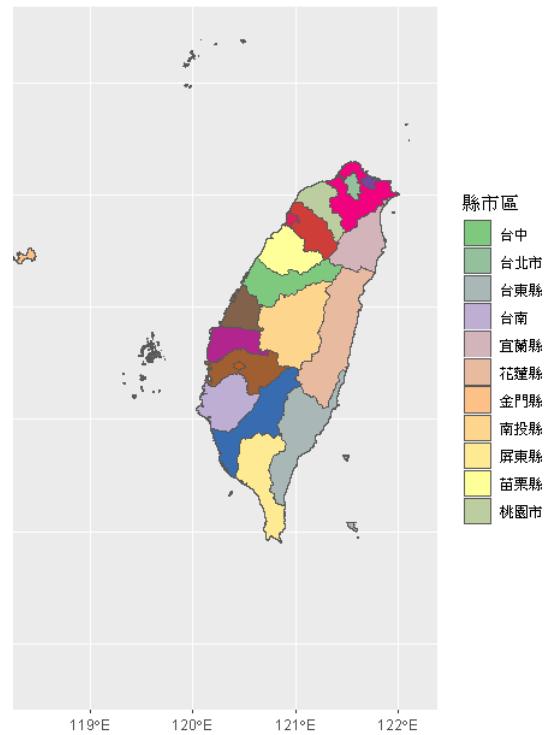
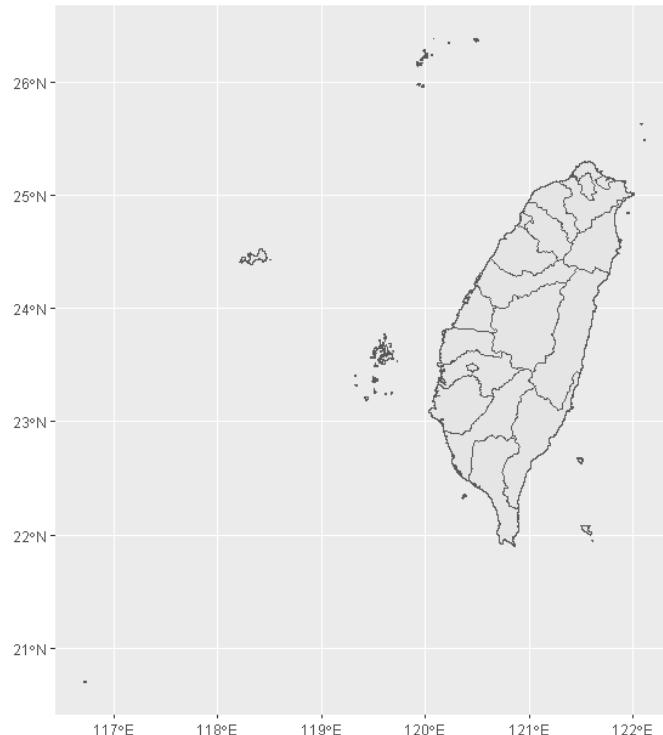
34/54

```
ggplot(data = taiwan.map) +  
  geom_sf() +  
  labs(title = "台灣地圖")
```

```
library(RColorBrewer)  
ggplot(data = taiwan.map) +  
  geom_sf(aes(fill = NL_NAME_2)) +  
  scale_fill_manual(name = "縣市區",  
                     values = colorRampPalette(brewer.pal(8, "Accent"))(22)) +  
  labs(title = "台灣地圖")
```

```
ggplot(data = taiwan.map) +  
  geom_sf(aes(fill = NL_NAME_2), show.legend= F) +  
  geom_sf_text(aes(label = NL_NAME_2), size = 3) +  
  labs(title = "台灣地圖")
```

台灣地圖





讀取台灣各縣市之人口統計資料檔

35/54

```
> TW.Population2019 <- read.csv("data/TW_Population2019.csv")
> head(TW.Population2019)
  排名 縣市 類別 人口
1    1 新北市 直轄市 4010657
2    2 臺中市 直轄市 2811729
3    3 高雄市 直轄市 2773786
4    4 臺北市 直轄市 2650154
5    5 桃園市 直轄市 2240328
6    6 臺南市 直轄市 1881730
> TW.Population2019$縣市
[1] 新北市 臺中市 高雄市 臺北市 桃園市 臺南市 彰化縣 屏東縣 雲林縣 新竹縣 苗栗縣 嘉義縣
[13] 南投縣 宜蘭縣 新竹市 基隆市 花蓮縣 嘉義市 臺東縣 金門縣 澎湖縣 連江縣
22 Levels: 宜蘭縣 花蓮縣 金門縣 南投縣 屏東縣 苗栗縣 桃園市 高雄市 基隆市 連江縣 ... 澎湖縣
> TW.Population2019$縣市 <- as.character(TW.Population2019$縣市)
>
> taiwan.map$NL_NAME_2
[1] 金門縣 馬祖列島 高雄市 新北市 台中 台南 台北市 彰化縣 嘉義市 嘉義縣
[11] 新竹市 新竹縣 花蓮縣 基隆市 苗栗縣 南投縣 澎湖縣 屏東縣 台東縣 桃園市
[21] 宜蘭縣 雲林縣
22 Levels: 台中 台北市 台東縣 台南 宜蘭縣 花蓮縣 金門縣 南投縣 屏東縣 苗栗縣 ... 澎湖縣
> levels(taiwan.map$NL_NAME_2)[c(1:4, 12)] <-
+   c("臺中市", "臺北市", "臺東縣", "臺南市", "連江縣")
> taiwan.map$NL_NAME_2
[1] 金門縣 連江縣 高雄市 新北市 臺中市 臺南市 臺北市 彰化縣 嘉義市 嘉義縣 新竹市 新竹縣
[13] 花蓮縣 基隆市 苗栗縣 南投縣 澎湖縣 屏東縣 臺東縣 桃園市 宜蘭縣 雲林縣
22 Levels: 臺中市 台北市 台東縣 台南市 宜蘭縣 花蓮縣 金門縣 南投縣 屏東縣 苗栗縣 ... 澎湖縣
```



將統計資料與地圖資料結合

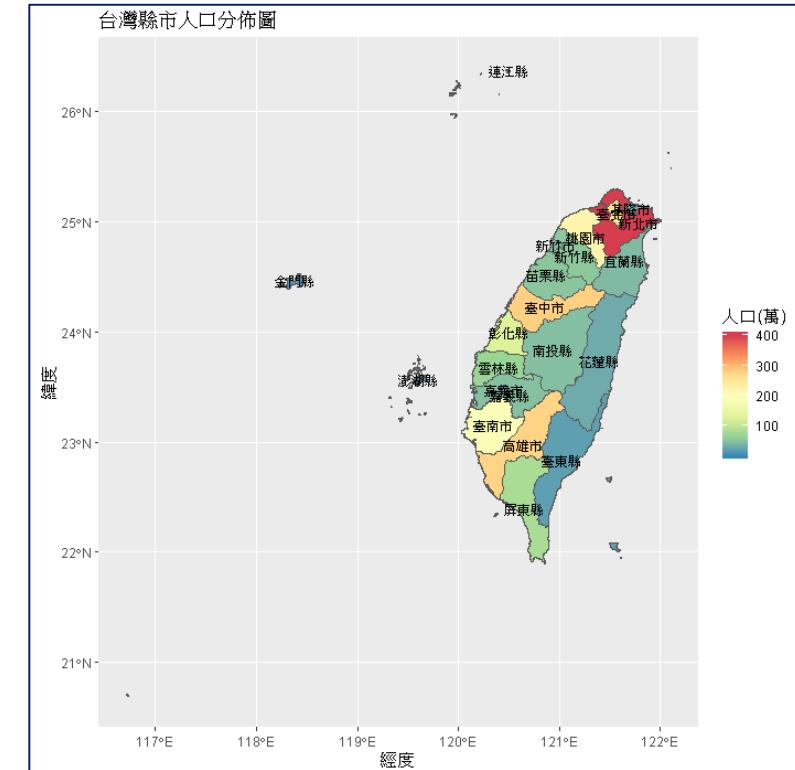
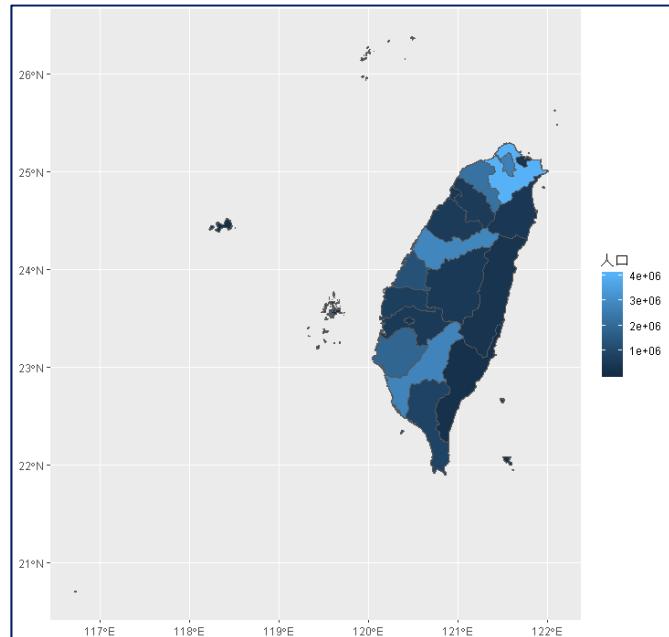
36/54

```
> my.taiwan.map <- taiwan.map[c("NL_NAME_2", "geometry")]
> my.taiwan.map$NL_NAME_2 <- as.character(my.taiwan.map$NL_NAME_2)
> head(my.taiwan.map)
Simple feature collection with 6 features and 1 field
geometry type:  MULTIPOLYGON
dimension:      XY
bbox:           xmin: 118.2113 ymin: 22.47569 xmax: 122.1085 ymax: 26.38542
epsg (SRID):   4326
proj4string:    +proj=longlat +datum=WGS84 +no_defs
  NL_NAME_2                      geometry
1  金門縣 MULTIPOLYGON (((118.3999 24...
2  連江縣 MULTIPOLYGON (((120.086 26....
...
6  臺南市 MULTIPOLYGON (((120.1387 22...
>
> library(dplyr)
> my.taiwan.map.data <- left_join(my.taiwan.map, TW.Population2019,
+                                     by= c("NL_NAME_2" = "縣市"))
> head(my.taiwan.map.data)
Simple feature collection with 6 features and 4 fields
...
proj4string:    +proj=longlat +datum=WGS84 +no_defs
  NL_NAME_2 排名 類別 人口          geometry
1  金門縣    20 縣  139319 MULTIPOLYGON (((118.3999 24...
2  連江縣    22 縣  13073  MULTIPOLYGON (((120.086 26....
...
6  臺南市     6 直轄市 1881730 MULTIPOLYGON (((120.1387 22...
> dim(my.taiwan.map.data)
[1] 22  5
```

繪製統計資訊地圖 (1)

37/54

```
ggplot(data = my.taiwan.map.data) +  
  geom_sf(aes(fill = 人口))
```



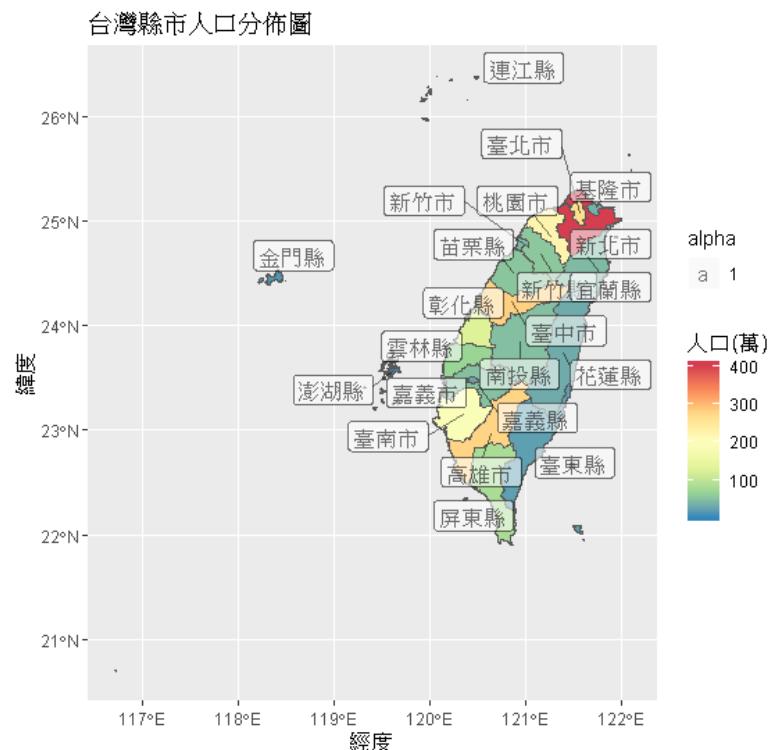
```
ggplot(data = my.taiwan.map.data) +  
  geom_sf(aes(fill = 人口/10000)) +  
  geom_sf_text(aes(label = NL_NAME_2), size = 3) +  
  scale_fill_distiller(palette = "Spectral", name = "人口(萬)") +  
  #scale_fill_gradientn(colours = tim.colors(22), name = "人口(萬)") +  
  #scale_fill_viridis(name = "人口(萬)") +  
  #scale_fill_distiller(palette = "YlOrRd", name = "人口(萬)") +  
  #scale_fill_distiller(palette = "YlOrRd", direction = -1, name = "人口(萬)") +  
  labs(title="台灣縣市人口分佈圖", x ="經度", y = "緯度")
```



繪製統計資訊地圖 (2)

38/54

```
library(devtools)
devtools::install_github("yutannihilation/ggsflabel")
library(ggsflabel)
ggplot(data = my.taiwan.map.data) +
  geom_sf(aes(fill = 人口/10000)) +
  scale_fill_distiller(palette = "Spectral", name = "人口(萬)") +
  geom_sf_label_repel(aes(label = NL_NAME_2, alpha = 1)) +
  labs(title="台灣縣市人口分佈圖", x ="經度", y = "緯度")
```

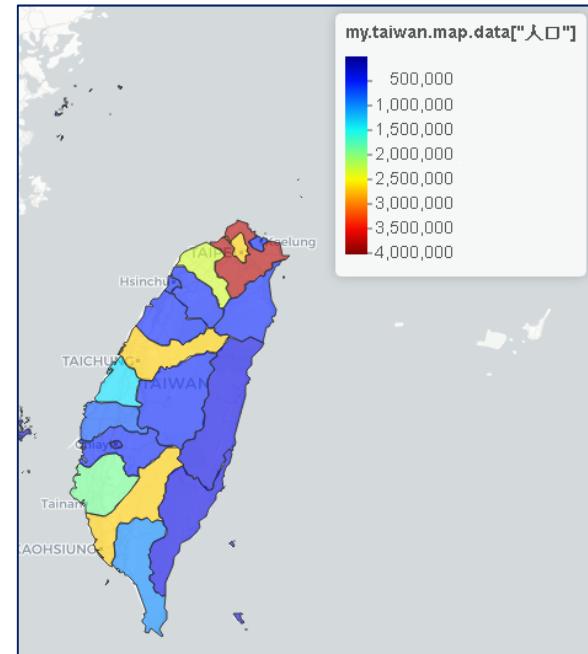
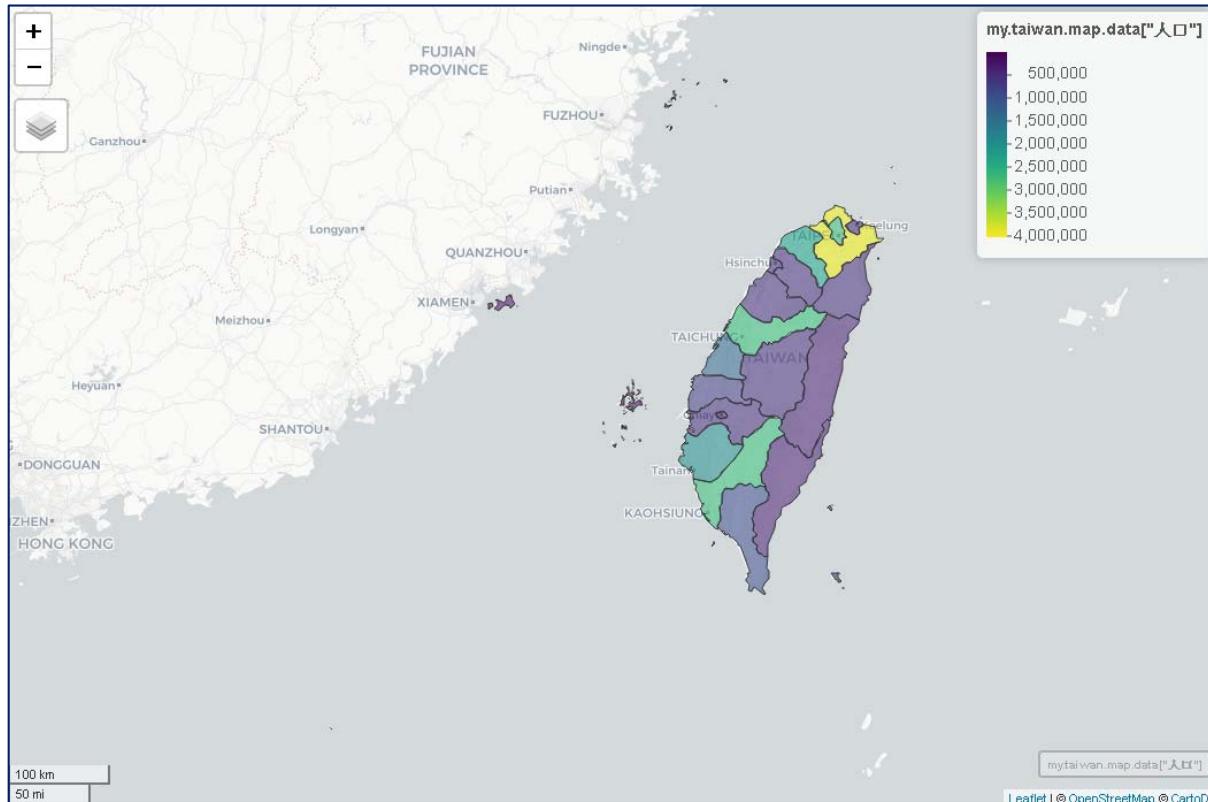




繪製統計資訊地圖 (3)

39/54

```
library(mapview)  
mapview(my.taiwan.map.data["人口"])  
mapview(my.taiwan.map.data["人口"], col.regions = tim.colors(100))
```





下載台北市行政區重要統計指標資料檔

40/54

臺北市統計資料庫查詢系統

- 統計資料庫查詢系統
- 重要統計資料庫
- 重要統計指標資料庫
- 近年來重要統計指標
- 行政區重要統計指標
 - 土地人口
 - 土地面積
 - 公告地價
 - 公告土地現值
 - 都市地價指數
 - 人口概況
 - 人口年齡分配
 - 人口消長
 - 原住民概況
 - 婚育概況
 - 婚姻狀況
 - 結婚情形
- 勞動

統計專區 | 主計處



地址：臺北市信義區市府路1號5、6樓中央區（市政大樓5、6樓）

表格： 人口概況

標示您的選項並選擇螢幕看表與檔案格式 指示提示
已選變數 ♦ 您至少必須選擇一項

年底別 ♦ 全選 不全選	行政區 ♦ 全選 不全選	項目 ♦ 全選 不全選
總數: 15. 已選: 15	總數: 13. 已選: 13	總數: 7. 已選: 7
101年 102年 103年 104年 105年 106年 107年	大同區 萬華區 文山區 南港區 內湖區 士林區 北投區	人口數/總計(人) 人口數/男(人) 人口數/女(人) 戶數(戶) 戶量(人/戶) 人口密度(人/平方公里) 性比例(男/百女)
搜尋 <input type="text"/> 文件開始	搜尋 <input type="text"/> 文件開始	搜尋 <input type="text"/> 文件開始

請點選 繼續

螢幕顯示限制為 16000 列與 350 列

已選資料列數 195 已選資料欄數 7

輸出格式：
選號分隔符(CSV)

A	B	C	D	E	F	G	H	I
1	人口概況-年 依 年底別 行政區 與 項目							
2								
3		人口數/總計(人)	人口數/男(人)	人口數/女(人)	戶數(戶)	戶量(人/戶)	人口密度(人/平方公里)	性比例(男/百女)
4	93年							
5	總計	2622472	1286303	1336169	923325	2.84	9649	96.27
6	松山區	205962	98839	107123	73768	2.79	22176	92.27
7	信義區	232506	114648	117858	83875	2.77	20745	97.28
8	大安區	312554	149602	162952	112428	2.78	27510	91.81
9	中山區	216868	104299	112569	84172	2.58	15850	92.65
10	中正區	158486	77676	80810	58841	2.69	20834	96.12
11	大同區	128512	64291	64221	44580	2.88	22619	100.11
12	萬華區	197445	100338	97107	71281	2.77	22305	103.33
13	文山區	258046	127225	130821	90972	2.84	8190	97.25
14	南港區	112982	56618	56364	37928	2.98	5173	100.45
15	內湖區	261201	127887	133314	87335	2.99	8271	95.93
16	士林區	288921	142259	146662	94924	3.04	4633	97
17	北投區	248989	122621	126368	83221	2.99	4382	97.03
18	94年							
19	總計	2616375	1279513	1336862	933110	2.8	9626	95.71
20	松山區	208101	99575	108526	75009	2.77	22406	91.75
21	信義區	230780	113351	117429	84605	2.73	20591	96.53
22	大安區	312166	148926	163240	113108	2.76	27476	91.23

人口概況:

年底別、行政區、人口數/總計(人)、人口數/男(人)、人口數/女(人)、戶數(戶)、戶量(人/戶)、人口密度(人/平方公里)、性比例(男/百女)。

"1. 戶量 = 人口數 ÷ 戶數。"

"2. 人口密度 = 人口數 ÷ 土地面積。"

"3. 性比例 = 男性人口數 ÷ 女性人口數 × 100
。"



讀取「台灣鄉鎮市區界線」shapefile檔 並選取「台北市」之地圖資料

41/54

```
> taiwan.town.map <- st_read("data/mapdata201907310953/TOWN_MOI_1080726.shp")
Reading layer `TOWN_MOI_1080726' from data source
`E:\TaipeiCityMap_R\data\mapdata201907310953\TOWN_MOI_1080726.shp' using driver `ESRI Shapefile'
Simple feature collection with 368 features and 7 fields
geometry type:  MULTIPOLYGON
dimension:      XY
bbox:           xmin: 114.3593 ymin: 10.37135 xmax: 124.5611 ymax: 26.38528
epsg (SRID):   NA
proj4string:    +proj=longlat +ellps=GRS80 +no_defs
> head(taiwan.town.map)
Simple feature collection with 6 features and 7 fields
...
proj4string:    +proj=longlat +ellps=GRS80 +no_defs
  TOWNID TOWNCODE COUNTYNAME TOWNNAME          TOWNENG COUNTYID COUNTYCODE             geometry
1     V02  10014020    臺東縣  成功鎮 Chenggong Township      V      10014  MULTIPOLYGON (((121.4098 23...
2     T21  10013210    屏東縣  佳冬鄉 Jiadong Township     T      10013  MULTIPOLYGON (((120.5485 22...
...
6     N07  10007120    彰化縣  田中鎮 Tianzhong Township     N      10007  MULTIPOLYGON (((120.5825 23...
> dim(taiwan.town.map)
[1] 368   8
>
> taipei.map <- taiwan.town.map[taiwan.town.map$COUNTYNAME == "臺北市",]
> head(taipei.map)
Simple feature collection with 6 features and 7 fields
...
proj4string:    +proj=longlat +ellps=GRS80 +no_defs
  TOWNID TOWNCODE COUNTYNAME TOWNNAME          TOWNENG COUNTYID COUNTYCODE             geometry
152     A02  63000030    臺北市  大安區 Daan District      A      63000  MULTIPOLYGON (((121.5438 25...
153     A11  63000080    臺北市  文山區 Wenshan District     A      63000  MULTIPOLYGON (((121.5973 25...
...
346     A13  63000090    臺北市  南港區 Nangang District     A      63000  MULTIPOLYGON (((121.6143 25...
> dim(taipei.map)
[1] 12   8
```



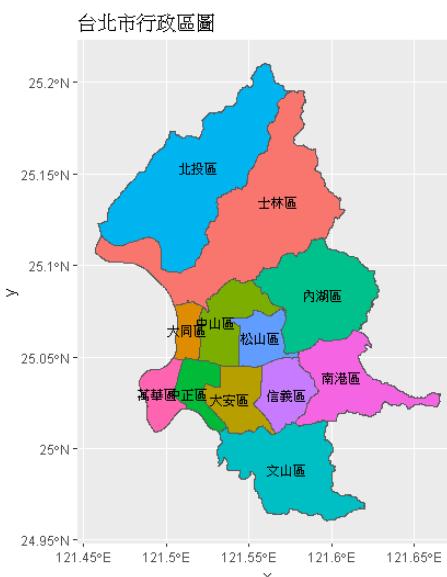
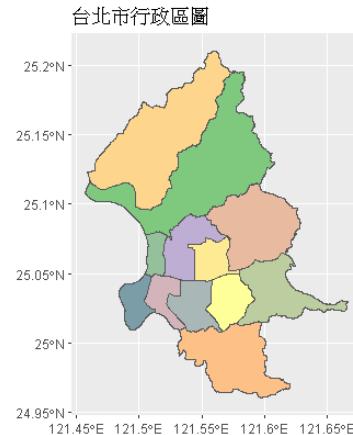
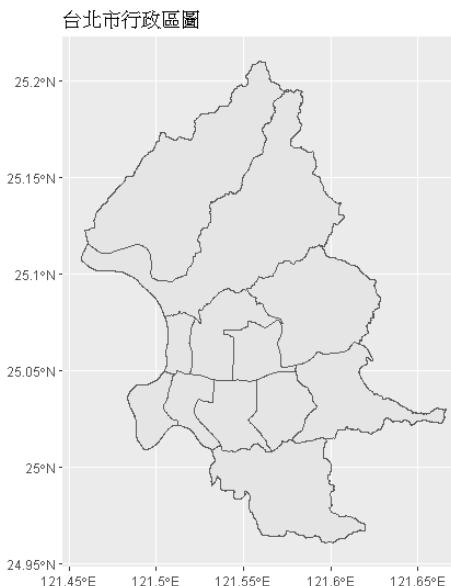
繪製台北市行政區圖

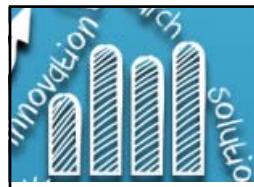
```
library(gridExtra)
g1 <- ggplot(data = taipei.map) + geom_sf() + labs(title = "台北市行政區圖")

g2 <- ggplot(data = taipei.map) +
  geom_sf(aes(fill = TOWNNAME)) +
  scale_fill_manual(name = "行政區",
                    values = colorRampPalette(brewer.pal(8, "Accent"))(22)) +
  labs(title = "台北市行政區圖")

g3 <- ggplot(data = taipei.map) +
  geom_sf(aes(fill = TOWNNAME), show.legend= F) +
  geom_sf_text(aes(label = TOWNNAME), size = 3) +
  labs(title = "台北市行政區圖")

grid.arrange(g1, g2, g3, nrow=1, ncol=3)
```





讀取北市之歷年人口統計資料檔

43/54

```
> TPE.Population <- read.csv("data/Taipei_Population_93-107.csv")
> head(TPE.Population)
  Year Admin.Area Pop.All Pop.Male Pop.Female Household AvePop.H Pop.Density Gender.Ratio
1  93    松山區  205962   98839   107123    73768     2.79      22176      92.27
2  93    信義區  232506   114648   117858    83875     2.77      20745      97.28
...
6  93    大同區  128512   64291    64221    44580     2.88      22619     100.11
>
> TPE.Population107 <- TPE.Population[TPE.Population$Year == 107,]
> TPE.Population107$Admin.Area
[1] 松山區 信義區 大安區 中山區 中正區 大同區 萬華區 文山區 南港區 內湖區 士林區 北投區
Levels: 士林區 大同區 大安區 中山區 中正區 內湖區 文山區 北投區 松山區 信義區 南港區 萬華區
> TPE.Population107$Admin.Area <- as.character(TPE.Population107$Admin.Area)
>
> my.taipei.map <- taipei.map[c("TOWNNAME", "geometry")]
> my.taipei.map$TOWNNAME <- as.character(my.taipei.map$TOWNNAME)
> head(my.taipei.map)
Simple feature collection with 6 features and 1 field
geometry type:  MULTIPOLYGON
dimension:      XY
bbox:           xmin: 121.4836 ymin: 24.9605 xmax: 121.6659 ymax: 25.06455
epsg (SRID):   NA
proj4string:   +proj=longlat +ellps=GRS80 +no_defs
  TOWNNAME                      geometry
152  大安區 MULTIPOLYGON (((121.5438 25...
153  文山區 MULTIPOLYGON (((121.5973 25...
343  信義區 MULTIPOLYGON (((121.58 25.0...
...
...
```



將統計資料與地圖資料結合

44/54

```
> library(dplyr)
> my.taipei.map.data <- left_join(my.taipei.map, TPE.Population107,
+                                   by= c("TOWNNAME"= "Admin.Area"))
> head(my.taipei.map.data)
Simple feature collection with 6 features and 9 fields
geometry type:  MULTIPOLYGON
dimension:      XY
bbox:           xmin: 121.4836 ymin: 24.9605 xmax: 121.6659 ymax: 25.06455
epsg (SRID):    NA
proj4string:   +proj=longlat +ellps=GRS80 +no_defs
  TOWNNAME Year Pop.All Pop.Male Pop.Female Household AvePop.H Pop.Density Gender.Ratio      geometry
1  大安區  107  308843  144007   164836   121344    2.55     27184     87.36 MULTIPOLYGON (((121.5438 25...
2  文山區  107  273762  131145   142617   106640    2.57     8688     91.96 MULTIPOLYGON (((121.5973 25...
3  信義區  107  223406  106457   116949   89574    2.49     19933     91.03 MULTIPOLYGON (((121.58 25.0...
4  萬華區  107  189603   92969    96634   78770    2.41     21419     96.21 MULTIPOLYGON (((121.5052 25...
5  中正區  107  159000   75898    83102   65474    2.43     20902     91.33 MULTIPOLYGON (((121.5134 25...
6  南港區  107  121670   59439    62231   47386    2.57     5570     95.51 MULTIPOLYGON (((121.6143 25...
> dim(my.taipei.map.data)
[1] 12 10
```

https://spatialanalysis.github.io/lab_tutorials/4_R_Mapping.html

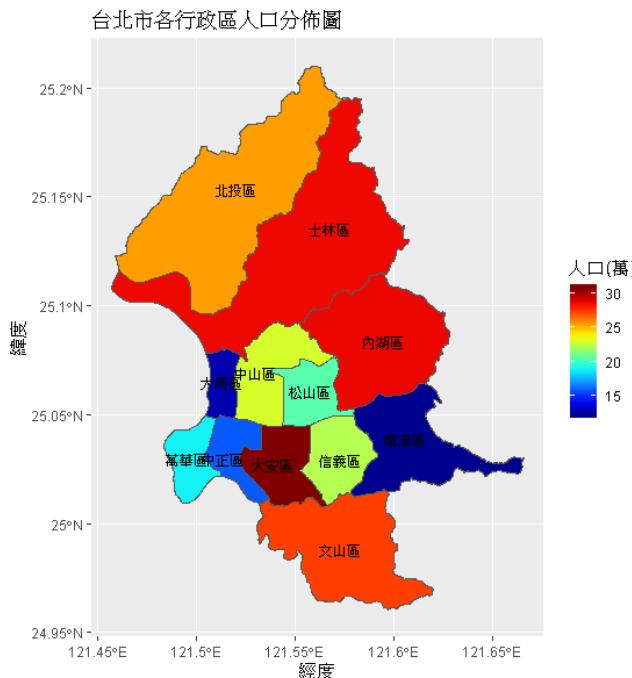
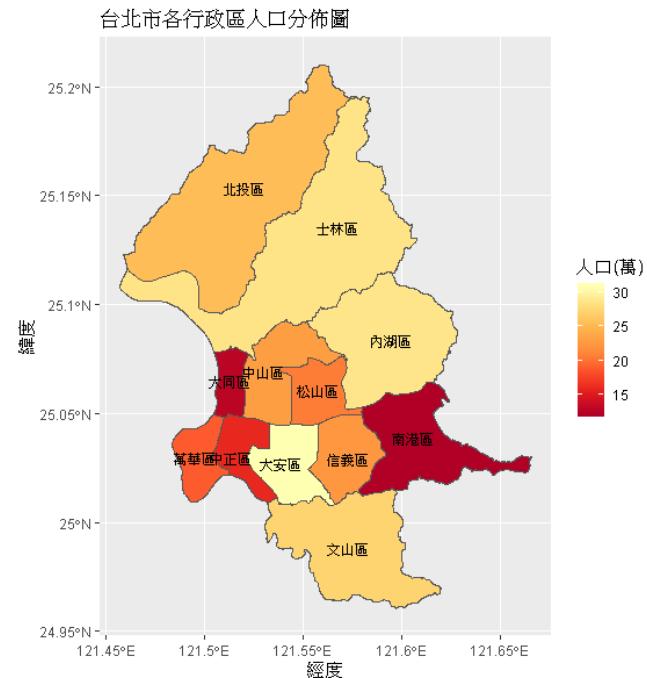
<https://hmwu.idv.tw>



繪製台北市各行政區人口統計地圖 (1)

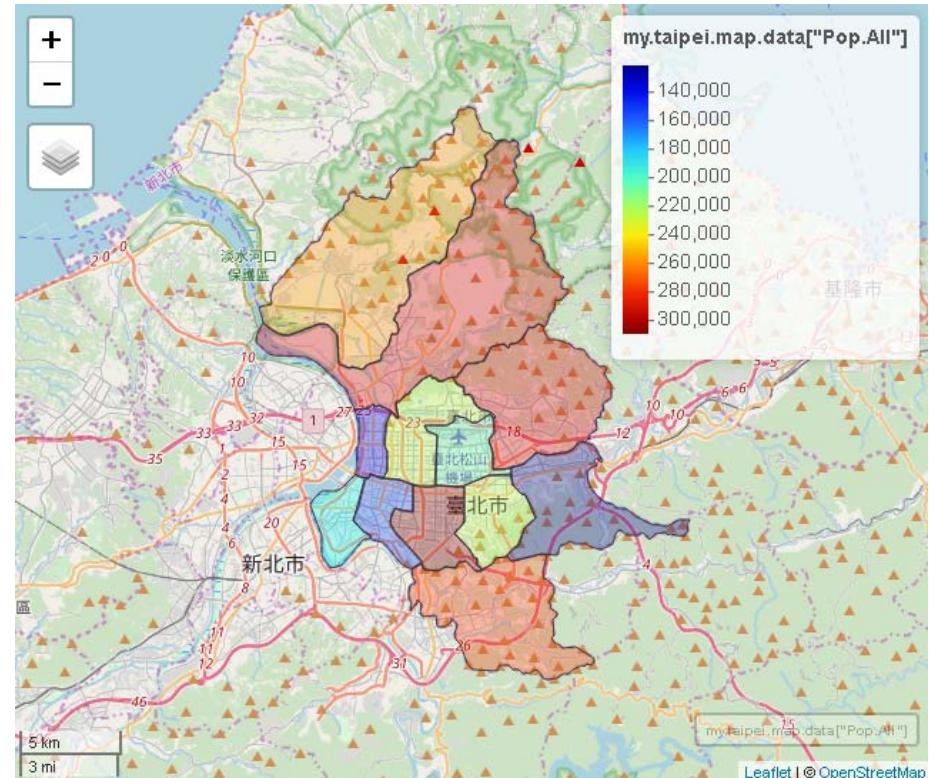
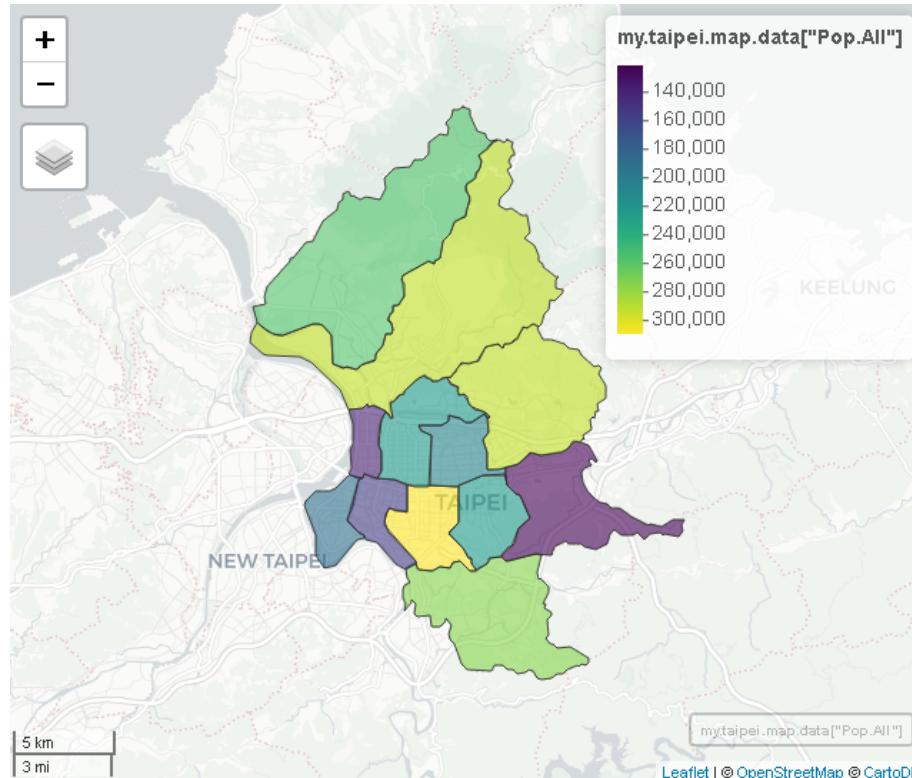
45/54

```
library(viridis)
library(fields)
ggplot(data = my.taipei.map.data) +
  geom_sf(aes(fill = Pop.All/10000)) +
  geom_sf_text(aes(label = TOWNNAME), size = 3) +
  #scale_fill_distiller(palette = "Spectral", name = "人口(萬)") +
  #scale_fill_gradientn(colours = tim.colors(22), name = "人口(萬)") +
  #scale_fill_viridis(name = "人口(萬)") +
  #scale_fill_distiller(palette = "YlOrRd", name = "人口(萬)") +
  scale_fill_distiller(palette = "YlOrRd", direction = -1, name = "人口(萬)") +
  labs(title="台北市各行政區人口分佈圖", x ="經度", y = "緯度")
```



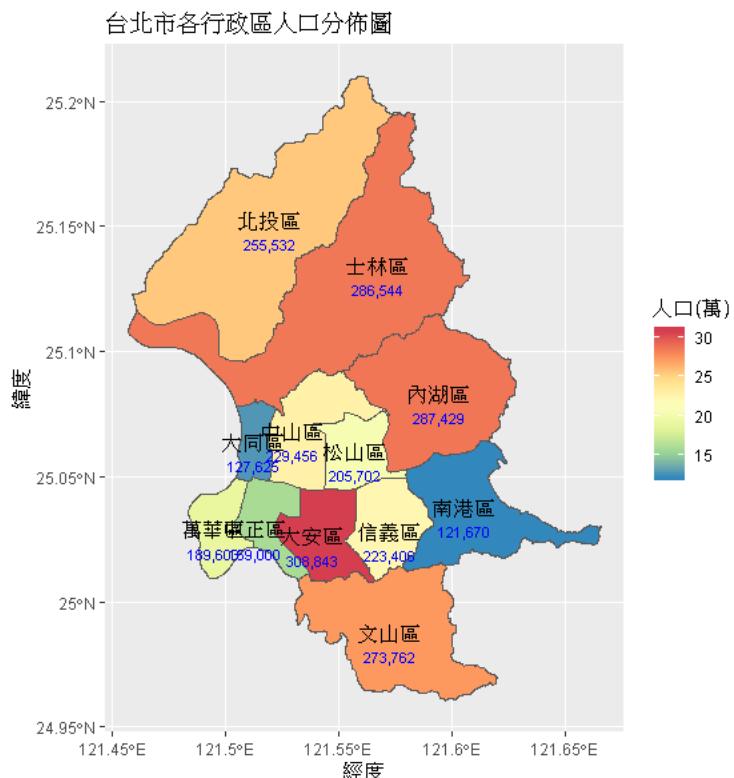
繪製台北市各行政區人口統計地圖 (2)

```
library(mapview)  
mapview(my.taipei.map.data["Pop.All"])  
mapview(my.taipei.map.data["Pop.All"], map.types = "OpenStreetMap",  
        col.regions = tim.colors(100), alpha.regions = 0.3)
```



於圖上標記文字、數字

```
p <- ggplot(data = my.taipei.map.data) +  
  geom_sf(aes(fill = Pop.All/10000)) +  
  geom_sf_text(aes(label = TOWNNAME), size = 4) +  
  scale_fill_distiller(palette = "Spectral", name = "人口(萬)") +  
  labs(title="台北市各行政區人口分佈圖", x ="經度", y = "緯度")  
  
p + geom_sf_text(aes(label = format(Pop.All, big.mark = ",", scientific = FALSE)),  
                 colour = "blue", size = 3,  
                 position = position_nudge(y = -0.01))
```





讀取地址資料，轉換成經緯度

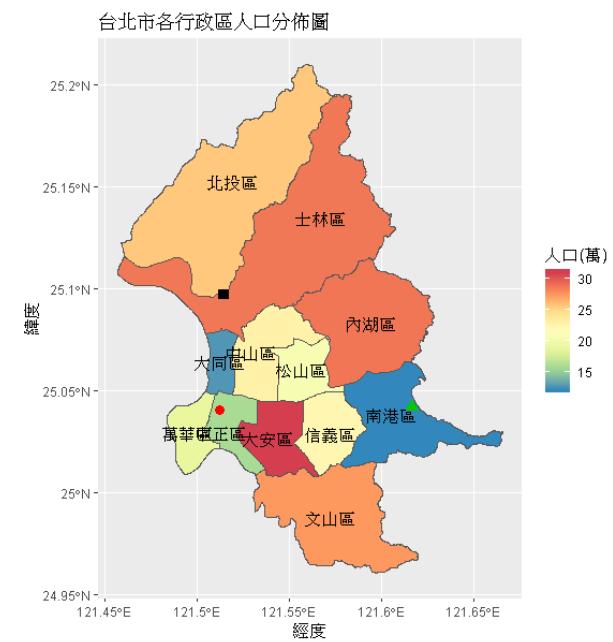
48/54

```
> address.df <- read.csv("data/address.csv", stringsAsFactors=FALSE, header=FALSE)
> has_google_key()
[1] TRUE
> address.df[1,2]
[1] " 11169台北市士林區承德路五段55號"
> geocode(address.df[1,2])
Source :
https://maps.googleapis.com/maps/api/geocode/json?address=11169%E5%8F%B0%E5%8C%97%E5%B8%82%E5%A
3%AB%E6%9E%97%E5%8D%80%E6%89%BF%E5%BE%B7%E8%B7%AF%E4%BA%94%E6%AE%B555%E8%99%9F&key=xxx
# A tibble: 1 x 2
  lon     lat
  <dbl> <dbl>
1 122.   25.1
> address.df
      V1          V2
1 兒童新樂園 11169台北市士林區承德路五段55號
2 中華民國總統府 100台北市中正區重慶南路一段122號
3 中央研究院 11529台北市南港區研究院路二段128號
> address.xy <- data.frame(name = address.df$V1, mutate_geocode(address.df["V2"], V2))
Source : https://maps.googleapis.com/maps/api/geocode/json?address=11169%E5...8%99%9F&key=xxx
Source : https://maps.googleapis.com/maps/api/geocode/json?address=100%E5%8F...8%99%9F&key=xxx
Source : https://maps.googleapis.com/maps/api/geocode/json?address=11529%E5...8%99%9F&key=xxx
> address.xy
      name          V2     lon     lat
1 兒童新樂園 11169台北市士林區承德路五段55號 121.5147 25.09709
2 中華民國總統府 100台北市中正區重慶南路一段122號 121.5123 25.04036
3 中央研究院 11529台北市南港區研究院路二段128號 121.6167 25.04208
```



給定經緯度，於圖上標記文字、數字及符號

```
p + geom_text(data = address.xy, aes(x = lon, y = lat, label = name),  
               colour = "purple", size = 3)  
  
p + geom_label(data = address.xy, aes(x = lon, y = lat, label = name),  
               colour = 1:nrow(address.xy), size = 3) +  
  annotate("text", label = "圖例說明", x = 121.63, y = 25.2, size = 5,  
          fontface = "bold")  
  
p + geom_point(data = address.xy, aes(x = lon, y = lat, label = name),  
               colour = 1:nrow(address.xy), size = 3,  
               shape = 15:17)
```





choroplethr: Simplify the Creation of Choropleth Maps in R

50/54

Choropleths are thematic maps where geographic regions, such as states, are colored according to some metric, such as the number of people who live in that state. This package simplifies this process by

1. Providing ready-made functions for creating choropleths of common maps.
2. Providing data and API connections to interesting data sources for making choropleths.
3. Providing a framework for creating choropleths from arbitrary shapefiles.
4. Overlaying those maps over reference maps from Google Maps.

NOTE:

choroplethrMaps:

Contains Maps Used by the 'choroplethr' Package
Contains 3 maps: US States, US Counties, Countries of the world.

The US Mexican American Population data: the percentage of Mexican Americans by US state from the 2010 Census.

state_choropleth function: the function requires that the **data frame** to be plotted has a column named region to represent state, and a column named value.

Additionally, the entries in the **region** column must exactly match how the entries are named in the **region** column of the dataset **state.map** from the **choroplethrMaps** package.

```
> library(choroplethrMaps)
> data(state.map)
> head(state.map)
  long      lat order  hole piece group id      GEO_ID STATE region LSAD CENSUSAREA
1 -112.5386 37.00067    1 FALSE     1  0.1  0 0400000US04    04 arizona <NA> 113594.1
2 -112.5345 37.00068    2 FALSE     1  0.1  0 0400000US04    04 arizona <NA> 113594.1
...
5 -111.4128 37.00148    5 FALSE     1  0.1  0 0400000US04    04 arizona <NA> 113594.1
6 -111.4059 37.00148    6 FALSE     1  0.1  0 0400000US04    04 arizona <NA> 113594.1
> dim(state.map)
[1] 50763    12
```



The US Mexican American Population data

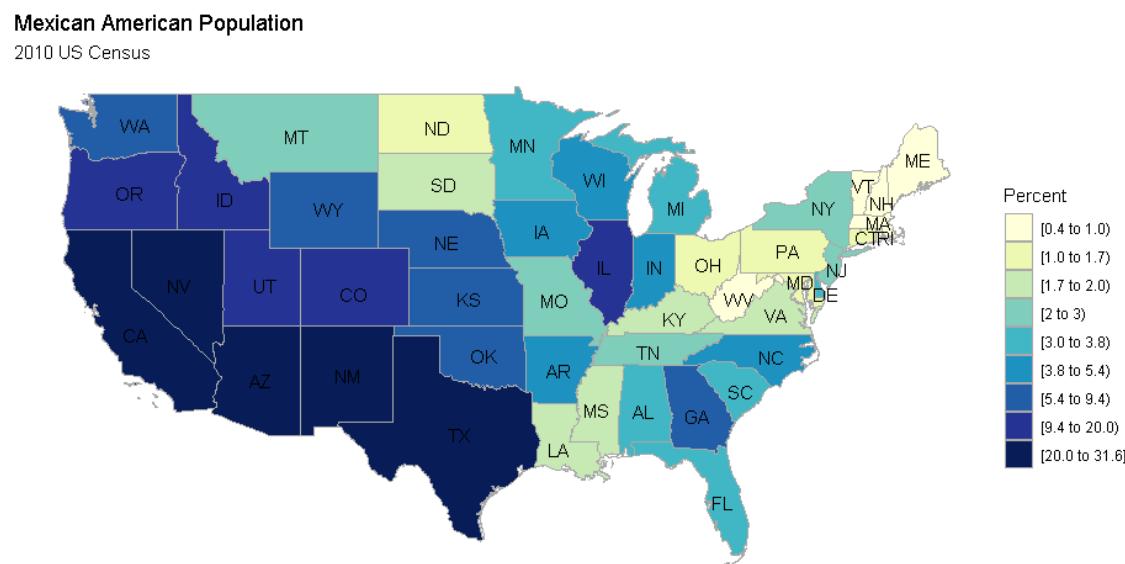
```
> library(ggplot2)
> library(choroplethr)
> data(continental_us_states)
> head(continental_us_states)
[1] "alabama"      "arkansas"      "arizona"       "california"    "colorado"
[6] "connecticut"
>
> mex.am <- read.csv("data/mexican_american.csv")
> head(mex.am)
  state population percent
1  Alabama     185602     3.7
2  Alaska      214642     3.0
3  Arizona     1957668    25.9
4  Arkansas     138164     4.7
5 California   11423146    30.7
6 Colorado      757181    15.1
> dim(mex.am)
[1] 51  3
>
> mex.am$region <- tolower(mex.am$state)
> mex.am$value <- mex.am$percent
> head(mex.am)
  state population percent      region value
1  Alabama     185602     3.7    alabama  3.7
2  Alaska      214642     3.0    alaska   3.0
3  Arizona     1957668    25.9   arizona  25.9
4  Arkansas     138164     4.7  arkansas  4.7
5 California   11423146    30.7  califonia 30.7
6 Colorado      757181    15.1  colorado 15.1
```



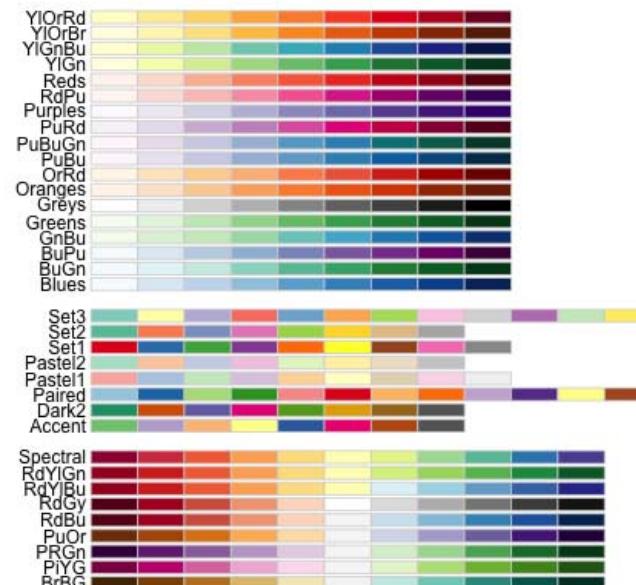
state_choropleth {choroplethr}: 52/54

Create a choropleth of US States

```
> state_choropleth(mex.am,
+                     num_colors = 9,
+                     zoom = continental_us_states) +
+   scale_fill_brewer(palette="YlGnBu") +
+   labs(title = "Mexican American Population",
+        subtitle = "2010 US Census",
+        caption = "source: https://en.wikipedia.org/wiki/List_of_U.S._states_by_Hispanic_and_Latino_population",
+        fill = "Percent")
```



RColorBrewer package palette



Top R Color Palettes to Know for Great Data Visualization
<https://www.datanovia.com/en/blog/top-r-color-palettes-to-know-for-great-data-visualization/>



county_choropleth {choroplethr} : 53/54

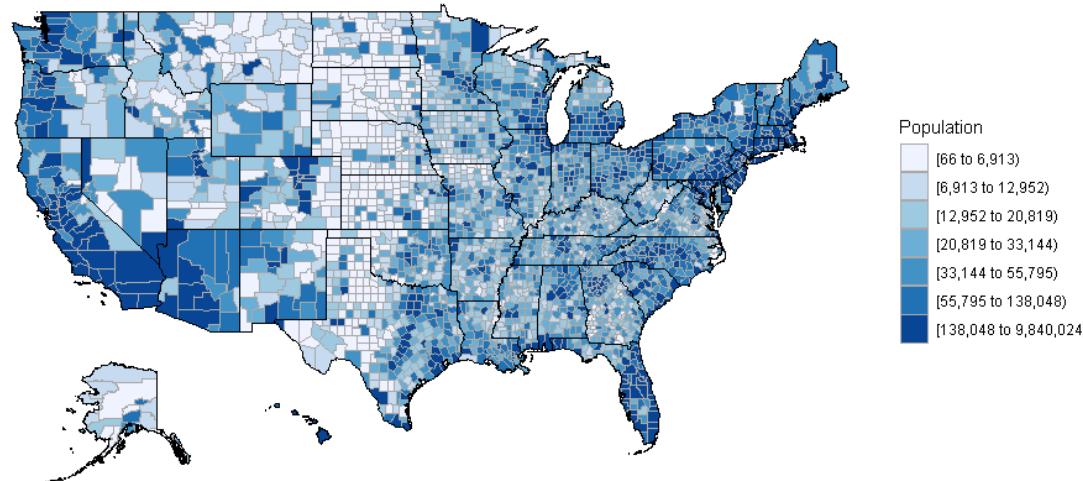
Create a choropleth of US Counties

```
> data(df_pop_county)
> head(df_pop_county)
  region  value
1 1001 54590
2 1003 183226
3 1005 27469
4 1007 22769
5 1009 57466
6 1011 10779
> dim(df_pop_county)
[1] 3143     2
```

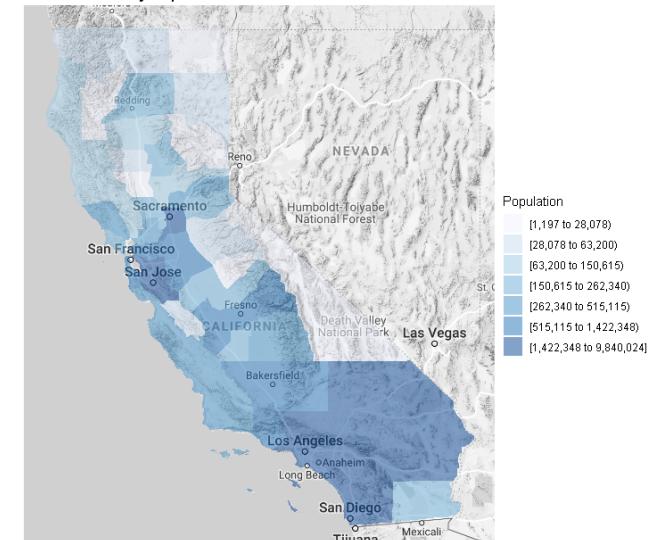
```
> county_choropleth(df_pop_county,
+                     title = "US 2012 County Population Estimates",
+                     legend = "Population")
> county_choropleth(df_pop_county,
+                     title = "California County Population Estimates",
+                     legend = "Population",
+                     state_zoom = "california",
+                     reference_map = TRUE)
```

Source : [https://maps.googleapis.com/maps/api/staticmap?center=37.010625,-120.716466&zoo](https://maps.googleapis.com/maps/api/staticmap?center=37.010625,-120.716466&zoom=6&size=640x640&scale=2&maptype=terrain&language=en-EN&key=xxx)m=6&size=640x640&scale=2&maptype=terrain&language=en-EN&key=xxx
Scale for 'x' is already present. Adding another scale for 'x', which will replace the existing scale.
Scale for 'y' is already present. Adding another scale for 'y', which will replace the existing scale.
Coordinate system already present. Adding new coordinate system, which will replace the existing one.
Warning message:
Removed 1 rows containing missing values (geom_rect).

US 2012 County Population Estimates



California County Population Estimates



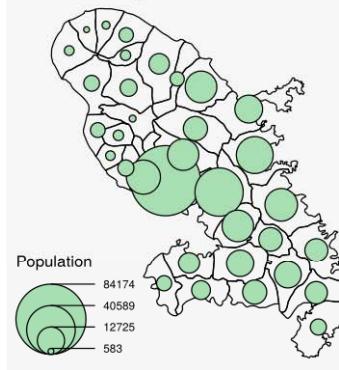
cartography: Thematic Cartography

<https://cran.r-project.org/web/packages/cartography/vignettes/cartography.html>

Thematic maps with cartography :: CHEAT SHEET

Use cartography with spatial objects from sf or sp packages to create thematic maps.

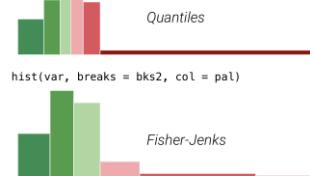
```
library(cartography)
library(sf)
mtq <- st_read("martinique.shp")
plot(st_geometry(mtq))
propSymbolsLayer(x = mtq, var = "P13_POP",
  legend.title.txt = "Population",
  col = "#a7dfb4")
```



Classification

Available methods are: quantile, equal, q6, fisher-jenks, mean-sd, sd, geometric progression...

```
bks1 <- getBreaks(v = var, nclass = 6,
  method = "quantile")
bks2 <- getBreaks(v = var, nclass = 6,
  method = "fisher-jenks")
pal <- carto.pal("green.pal", 3, "wine.pal", 3)
hist(var, breaks = bks1, col = pal)
```



Symbology

In most functions the x argument should be an sf object. sp objects are handled through spdf and df arguments.

Choropleth
choroLayer(x = mtq, var = "myvar",
 method = "quantile", nclass = 8)

Typology
typoLayer(x = mtq, var = "myvar")

Proportional Symbols
propSymbolsLayer(x = mtq, var = "myvar",
 inches = 0.1, symbols = "circle")

Colorized Proportional Symbols (relative data)
propSymbolsChoroLayer(x = mtq, var = "myvar",
 var2 = "myvar2")

Colorized Proportional Symbols (qualitative data)
propSymbolsTypoLayer(x = mtq, var = "myvar",
 var2 = "myvar2")

Double Proportional Symbols
propTrianglesLayer(x = mtq, var1 = "myvar",
 var2 = "myvar2")

OpenStreetMap Basemap (see rosm package)
tiles <- getTiles(x = mtq, type = "osm")
 tilesLayer(tiles)

Isopleth (see SpatialPosition package)
smoothLayer(x = mtq, var = "myvar",
 typeefct = "exponential", span = 500,
 beta = 2)

Discontinuities
discLayer(x = mtq.borders, df = mtq,
 var = "myvar", threshold = 0.5)

Flows
propLinkLayer(x = mtq_link, df = mtq_df,
 var = "fij")

Dot Density
dotDensityLayer(x = mtq, var = "myvar")

Labels
labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Lorenz

Ipsum

Sil

Amet

Transformations

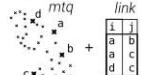
Polygons to Grid
mtq_grid <- getGridLayer(x = mtq, cellsize = 3.6e+07,
 type = "hexagonal", var = "myvar")

Grids layers can be used by choroLayer() or propSymbolsLayer().



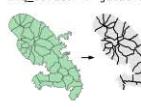
Points to Links

mtq_link <- getLinkLayer(x = mtq, df = link)



Polygons to Borders

mtq_border <- getBorders(x = mtq)



Borders layers can be used by disclayer() function

Polygons to Pencil Lines

mtq_pen <- getPencilLayer(x = mtq)



Pencil lines layers can be used by disclayer() function

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x = mtq, txt = "myvar",
 halo = TRUE, overlap = FALSE)

Labels

labelLayer(x