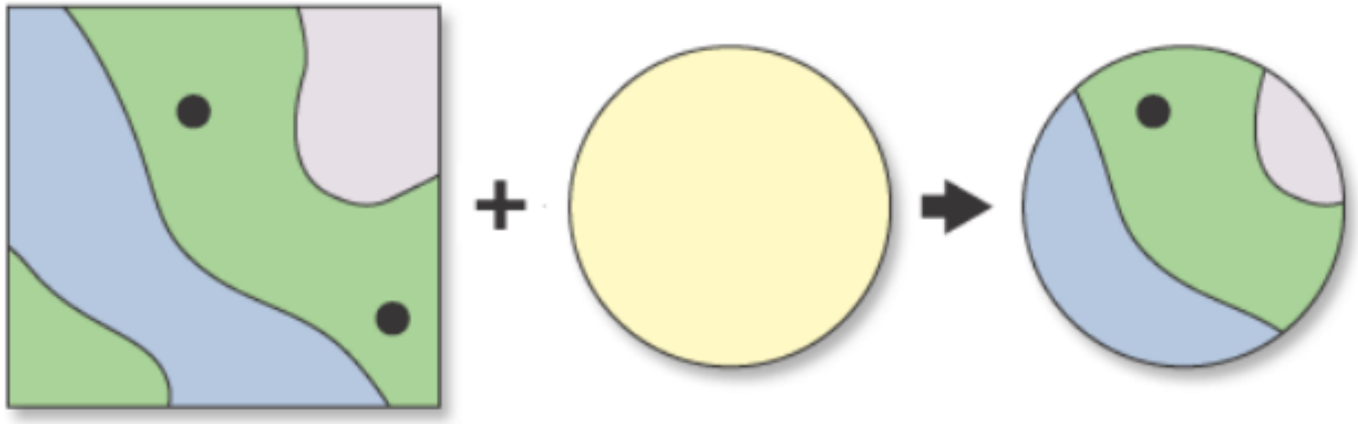
Using R for GIS analysis:

More complex geo-processing

授課教師：溫在弘
E-mail: wenthung@ntu.edu.tw

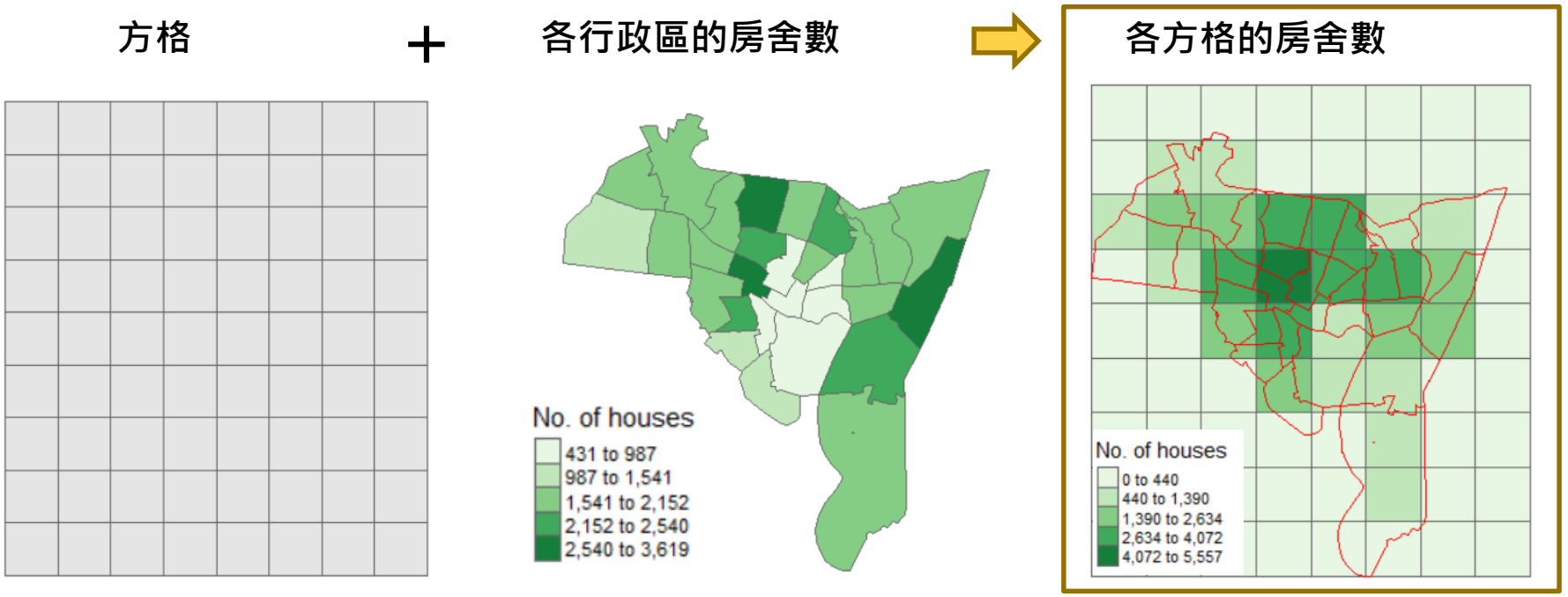
Contents

- Using R as a GIS (2)
  - Spatial intersection of multiple polygon layers



Spatial intersection of multiple polygon layers

- Spatial Intersection: st_intersection()



CRS Transformation

x2 = st_transform(x1, crs)

```
Popn_TWN = st_read("./data/Popn_TWN2.shp",options="ENCODING=BIG5")
st_crs(Popn_TWN)

# EPSG:3826 TWD97-TM2 zone 121
# EPSG:4326 WGS84

Popn_TWN = st_transform(Popn_TWN, 4326)
st_crs(Popn_TWN)
```

R Functions and Procedures

- Step 1. Fishnet: st_make_grid()
- Step 2. Spatial intersection: st_intersection()
- Step 3. Field calculation
- Step 4. Grouping data: group_by() + summarise()
- Step 5. Spatial mapping: tm_shape() + tm_polygons

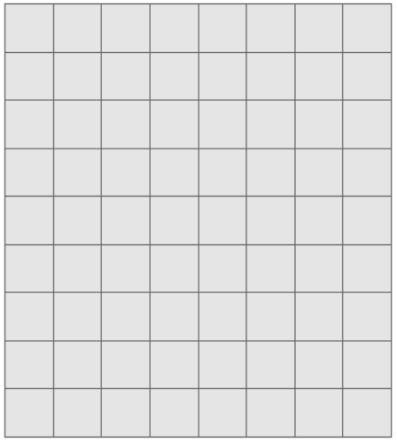
Step 1: Fishnet: st_make_grid()

Description

Create a square or hexagonal grid covering the bounding box of the geometry of an sf or sfc object

Usage

```
st_make_grid(
  x,
  cellsize = c(diff(st_bbox(x)[c(1, 3)]), diff(st_bbox(x)[c(2, 4)]))/n,
  offset = st_bbox(x)[c("xmin", "ymin")],
  n = c(10, 10),
  crs = if (missing(x)) NA_crs_ else st_crs(x),
  what = "polygons",
  square = TRUE,
  flat_topped = FALSE
```

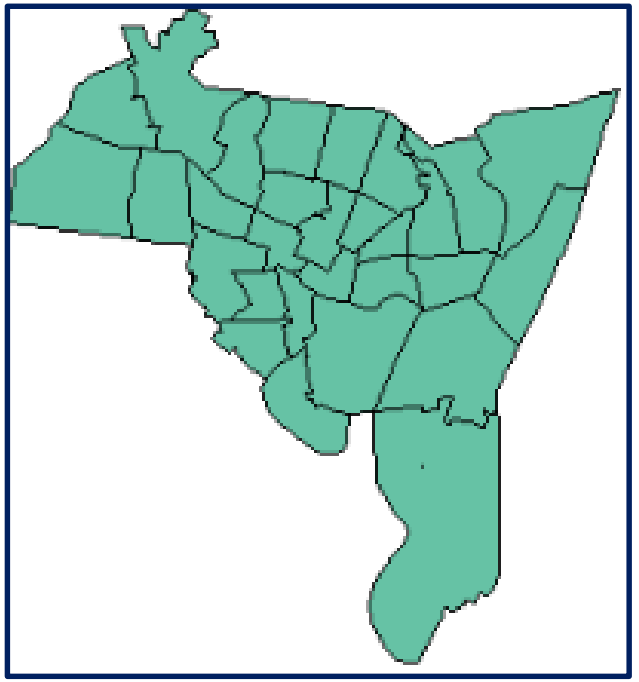


st_make_grid(sf, cellsize, offset, n)

補充 st_bbox()

```
> st_bbox(tracts_sf)
     xmin      ymin      xmax      ymax
531731.9 147854.0 569625.3 188464.6
```

```
box1 <- st_bbox(tracts_sf)
box1 <- unname(box1)
x_range <- box1[3]-box1[1]
y_range <- box1[4]-box1[2]
```



Values	
box1	'bbox' num [1:4] 531732 147854 569625 188465
x_range	37893.4
y_range	40610.6

Step 1: *sfc* format

```
grid <- st_make_grid(tracts_sf, 5000,
                     crs = st_crs(tracts_sf),
                     what = "polygons", square = TRUE)
```

grid	List of 72

```
> class(grid)
[1] "sfc_POLYGON" "sfc"
```

sfc: a list column of containing the geometries

st_sf (): converting *sfc* to *sf* format

```
> n <- length(lengths(grid))
> n
[1] 72
```

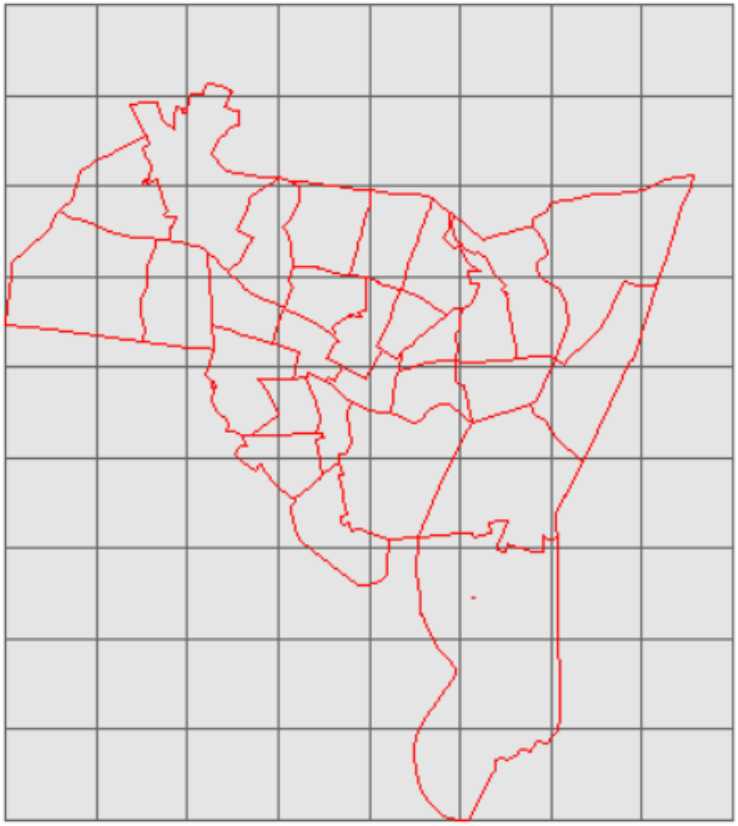
```
> grid_sf <- st_sf(index = 1:n, grid)
> head(grid_sf)
Simple feature collection with 6 features and 1 field
geometry type:  POLYGON
dimension:      XY
bbox:           xmin: 531731.9 ymin: 147854 xmax: 561731.9
CRS:             +proj=lcc +datum=NAD27 +lon_0=-72d45 +lat_
57607315 +y_0=0 +units=us-ft +no_defs +ellps=clrk66 +nadgri
  index                              grid
1     1 POLYGON ((531731.9 147854, ...
2     2 POLYGON ((536731.9 147854, ...
3     3 POLYGON ((541731.9 147854, ...
4     4 POLYGON ((546731.9 147854, ...
5     5 POLYGON ((551731.9 147854, ...
6     6 POLYGON ((556731.9 147854, ...
```

重新命名欄位名稱 grd_id

```
> names(grid_sf) <- c("grd_id","grid")
> head(grid_sf)
Simple feature collection with 6 features and 1 field
Geometry type: POLYGON
Dimension:     XY
Bounding box:  xmin: 531731.9 ymin: 147854 xmax: 561731.9 ymax: 152854
CRS:            +proj=lcc +datum=NAD27 +lon_0=-72d45 +lat_1=41d52 +lat_
2880.3657607315 +y_0=0 +units=us-ft +no_defs +ellps=clrk66 +nadgrids=@c
v1_can.dat
  grd_id                             grid
1      1 POLYGON ((531731.9 147854, ...
2      2 POLYGON ((536731.9 147854, ...
3      3 POLYGON ((541731.9 147854, ...
4      4 POLYGON ((546731.9 147854, ...
5      5 POLYGON ((551731.9 147854, ...
6      6 POLYGON ((556731.9 147854, ...
```

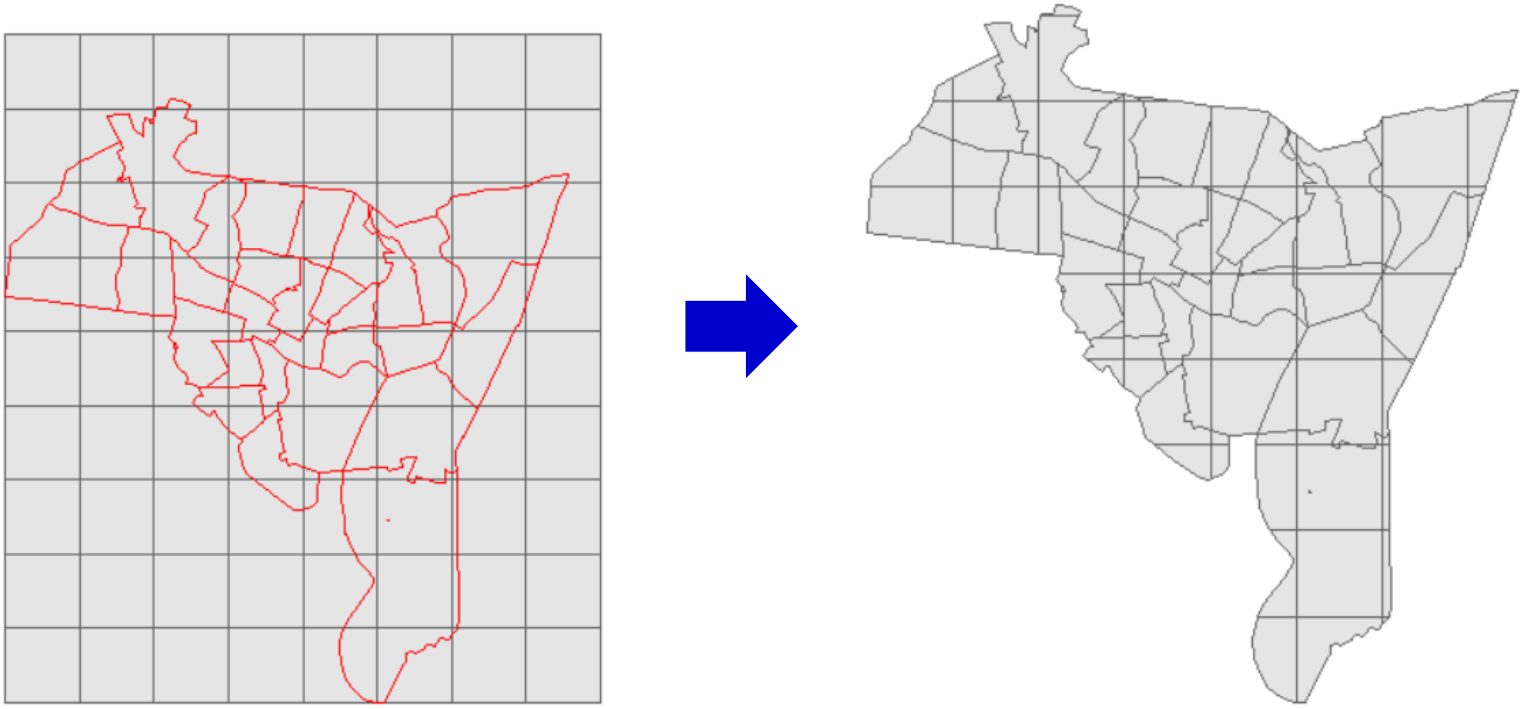
Step 1: Building fishnet

```
> grd_bg <- tm_shape(grid_sf) + tm_polygons("grey90")
> tracts <- tm_shape(tracts_sf) + tm_borders(col = "red")
> grd_bg + tracts
```



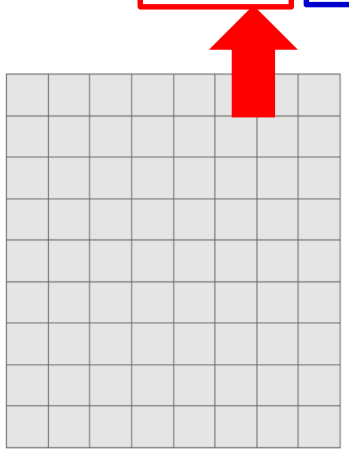
Step 2: Spatial intersection: st_intersection()

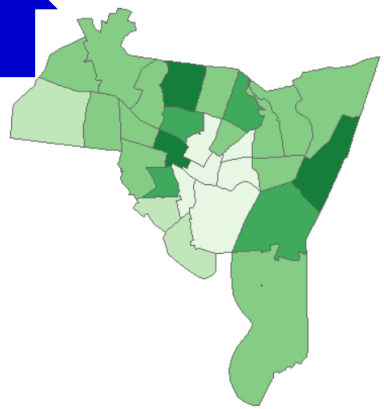
```
new_sf <- st_intersection(grid_sf, tracts_sf)
new_lyr <- tm_shape(new_sf) + tm_polygons("grey90")
new_lyr
```

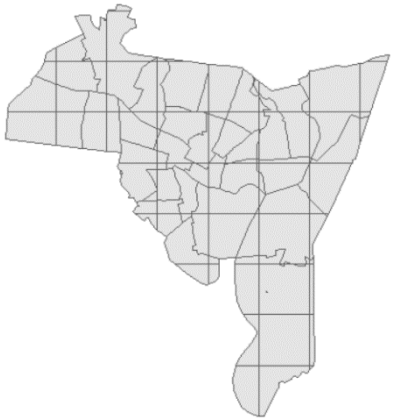


Checking the attributes of new *sf* data

```
> head(new_sf)
Simple feature collection with 6 features and 78 fields
geometry type:  POLYGON
dimension:      XY
bbox:           xmin: 538629.8 ymin: 178187.5 xmax: 546803.4
CRS:             +proj=lcc +datum=NAD27 +lon_0=-72d45 +lat_1=
57607315 +y_0=0 +units=us-ft +no_defs +ellps=clrk66 +nadgrids
   grd_id     AREA PERIMETER T009075H_ T009075H_I ARCINFOFPS
50     50 38821430  39255.55          2         554  090091413
51     51 38821430  39255.55          2         554  090091413
58     58 38821430  39255.55          2         554  090091413
59     59 38821430  39255.55          2         554  090091413
60     60 38821430  39255.55          2         554  090091413
67     67 38821430  39255.55          2         554  090091413
```







Step 3: Field calculation

```
head(new_sf)
new_sf$new_area<-st_area(new_sf)
new_sf$houses<- (new_sf$new_area / new_sf$AREA) * new_sf$HSE_UNITS
```

```
   PERS_UNIT SPLIT                                grid
50      2.42     0 POLYGON ((540203.5 182854, ...
51      2.42     0 POLYGON ((541731.9 179671.7...
58      2.42     0 POLYGON ((541731.9 187318.2...
59      2.42     0 POLYGON ((546106.7 182854, ...
60      2.42     0 POLYGON ((546731.9 183238.1...
67      2.42     0 POLYGON ((544065.5 187854, ...
                                houses
50 175.19847263 [US_survey_foot^2]
51 661.54956400 [US_survey_foot^2]
58 502.01366295 [US_survey_foot^2]
59 616.32037917 [US_survey_foot^2]
60   0.09510318 [US_survey_foot^2]
67  41.82397472 [US_survey_foot^2]
```

```
    new_area
3405836.375 [US_survey_foot^2]
12860440.706 [US_survey_foot^2]
9759082.762 [US_survey_foot^2]
11981191.015 [US_survey_foot^2]
1848.794 [US_survey_foot^2]
813052.833 [US_survey_foot^2]
```

Using group_by() and summarise()

範例:

Popn_TWN = st_read("Popn_TWN2.shp",options="ENCODING=BIG5")

```
> head(Popn_TWN)
Simple feature collection with 6 features and 14 fields
Geometry type: MULTIPOLYGON
Dimension:     XY
Bounding box:  xmin: -26119.97 ymin: 2700346 xmax: 201273.2 ymax: 2919551
Projected CRS: TWD97 / TM2 zone 121
   TOWN_ID   TOWN COUNTY_ID COUNTY A0A14_CNT A0A14_M A0A14_F A15A64_CNT A15A64_M
1 09007010 南竿鄉     09007 連江縣       971     499     472       5893     3391
2 09007020 北竿鄉     09007 連江縣       249     136     113       1839     1035
3 09007030 莒光鄉     09007 連江縣       126      73      53       1296      815
4 09007040 東引鄉     09007 連江縣       179     107      72       1064      644
5 09020010 金城鎮     09020 金門縣      4501    2358    2143      33324    16606
6 09020020 金沙鎮     09020 金門縣      1749     945     804      15916     7860
```

Popn_County = group_by(Popn_TWN, COUNTY_ID)

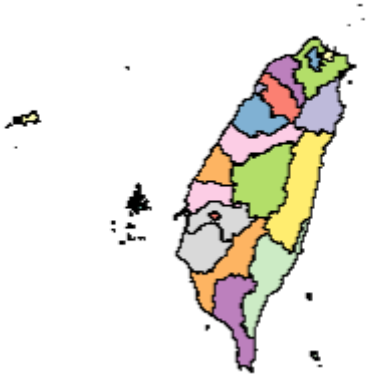
Popn_County = summarise(Popn_County, OLD = sum(A65UP_CNT))

plot(Popn_County)

補充：tidyverse的 pipe語法

網路上的中文說明：https://bookdown.org/tonykuoyj/eloquentr/data-workflow.html

```
# A tibble: 6 x 3
  COUNTY_ID    OLD
  <chr>      <dbl>
1 09007       1427
2 09020      17276
3 10002      71217
4 10004      67348
5 10005      86465
6 10007     192540
```



```
Popn_County = group_by(Popn_TWN, COUNTY_ID)
Popn_County = summarise(Popn_County, OLD = sum(A65UP_CNT))
```

改寫成一行：

```
Popn_County = summarise(group_by(Popn_TWN, COUNTY_ID), OLD = sum(A65UP_CNT))
```

改寫成pipe的寫法：

```
Popn_County = Popn_TWN %>% group_by(COUNTY_ID) %>%
                summarise (OLD = sum(A65UP_CNT))
```

補充：tidyverse的 pipe語法

Use %>% to emphasize a sequence of actions, rather than the object that the actions are being performed on.

範例：new_sf$new_area <- st_area(new_sf)

可改寫成：
new_sf$new_area <- new_sf %>% st_area

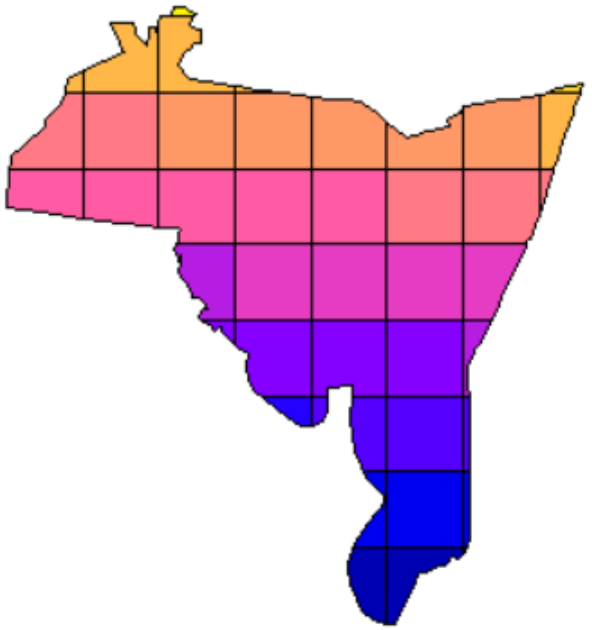
Step 4: Grouping data: summarise()

```
new_sf <- summarise(group_by(new_sf, grd_id), count = sum(houses))

new_sf <- new_sf %>% group_by(grd_id) %>% summarise(count = sum(houses))
```

```
head(new_sf)
```

```
> head(new_sf)
Simple feature collection
geometry type:  POLYGON
dimension:      XY
bbox:           xmin: 5541
CRS:             +proj=lcc
57607315 +y_0=0 +units=us-
# A tibble: 6 x 3
  grd_id                count
   <int> [US_survey_foot^
1      5           224.70602
2      6           243.68082
3      7             2.08143
4     13           115.92200
5     14           536.60648
6     15            47.44232
```



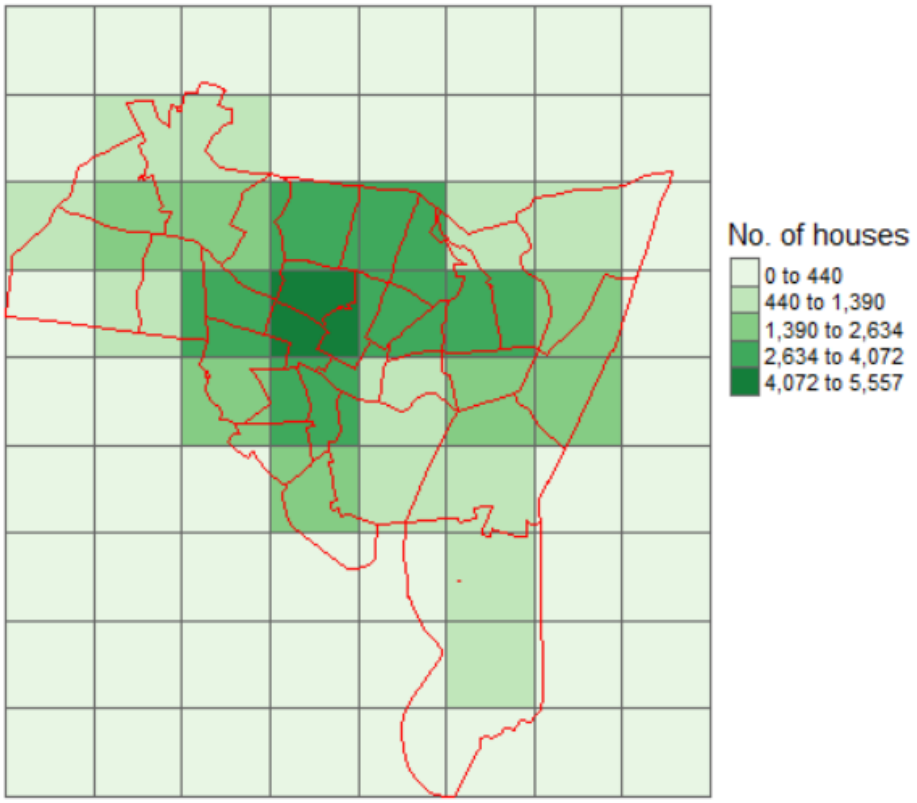
Link to *grid_sf* data

```
grid_sf$houses <- 0
grid_sf$houses[new_sf$grd_id] <- new_sf$count  # using [grd_id] as the index
```

```
> head(grid_sf)
Simple feature collection with 6 features and 2 fields
geometry type:  POLYGON
dimension:      XY
bbox:           xmin: 531731.9 ymin: 147854 xmax: 56173
CRS:             +proj=lcc +datum=NAD27 +lon_0=-72d45 +
57607315 +y_0=0 +units=us-ft +no_defs +ellps=clrk66 +na
  grd_id                              grid   houses
1      1 POLYGON ((531731.9 147854, ...   0.0000
2      2 POLYGON ((536731.9 147854, ...   0.0000
3      3 POLYGON ((541731.9 147854, ...   0.0000
4      4 POLYGON ((546731.9 147854, ...   0.0000
5      5 POLYGON ((551731.9 147854, ... 224.7060
6      6 POLYGON ((556731.9 147854, ... 243.6808
```

Step 5: Spatial mapping

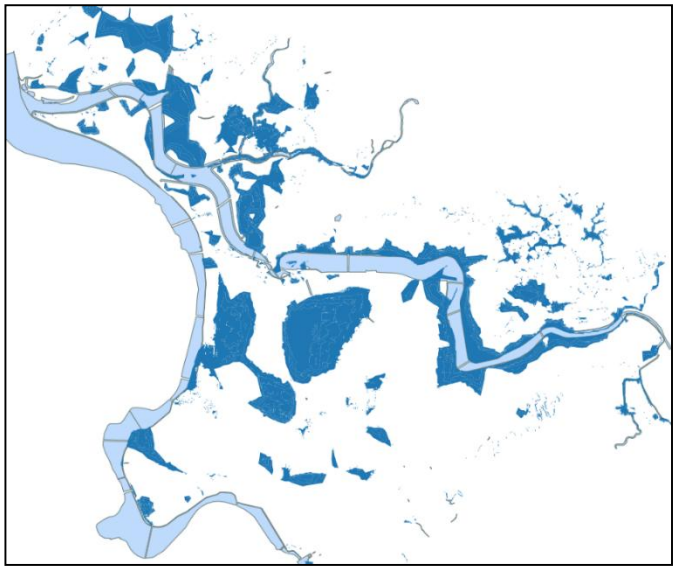
```
tm_shape(grid_sf) +
  tm_polygons("houses", palette = "Greens", style = "jenks", title = "No. of houses") +
  tm_layout(frame = F, legend.position = c(1,0.5)) +
  tm_shape(tracts_sf) + tm_borders(col = "red")
```



本週實習

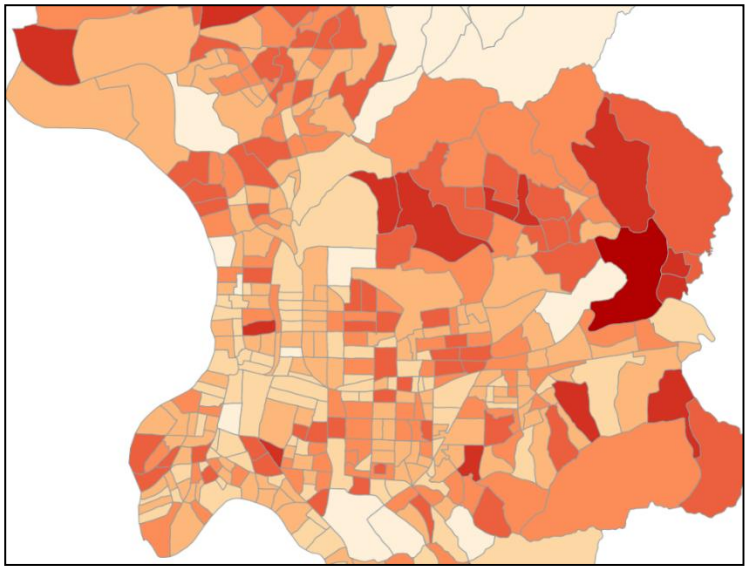
Flood50.shp

淡水河流域 洪災範圍



Taipei_Vill.shp

台北市村里人口數 (census欄位)



(1) 利用村里淹水面積比例計算，估計**洪災影響人數**。

(2) 依照「行政區（大安區、中正區、…）」彙總統計，

列表**各行政區**的洪災影響人數。

實習的學習資源

https://wenlab501.github.io/GEOG2017/

本週作業：計算環域涵蓋人數的自訂函數

*** 圖資** Data_MRT.zip

MRT.shp：台北市捷運站點資料（TWD97-TM2）

- MRT_NAME (MRT_ID)：捷運站名稱（捷運站編號）

TPE_LI.shp：台北市村里面資料（WGS84-經緯度）

- VILLAGE (V_ID)：村里（村里編號）
- CENSUS：人口數（單位：人）

建立自訂函數，回傳使用者設定某捷運站在特定距離方圓內涵蓋的人數。

建立自訂函數**STN_POP**(id,dist)，其中id代表捷運站的編號，dist代表離捷運站的距離。該函數能回傳「編號id車站」在方圓距離「dist公尺」內涵蓋的人數（回傳整數格式），**，以涵蓋村里的面積加權計算人口數**。

例如：STN_POP (38,500) 表示該函數回傳**編號38**的捷運站在**500公尺方圓內**所涵蓋的人口數。

參考答案

```
STN_POP (38,500)
```

```
## [1] 6729
```

下週(3/24) R as a GIS 綜合演練

- 測驗分數：滿分 30分 （額外加入期中考分數）
- 時間：下午 2:30 – 5:00
- 電腦上機實作 (可自行使用筆電)
  - 繳交格式：(R Notebook產生的) html檔，並上傳NTU COOL。
  - 可參考或使用任何工具協助作答，唯以個人方式作答，

    不允許：任何形式的相互交談、訊息傳遞與資訊交換。

    違規者，當次測驗不計分。
  - 屬於額外加分性質，因故缺席者，不另行補考。

類似題目：參考109-1 第一次期中考題
https://wenlab501.github.io/GEOG2017/EXAM/1092/1092Mid1.pdf