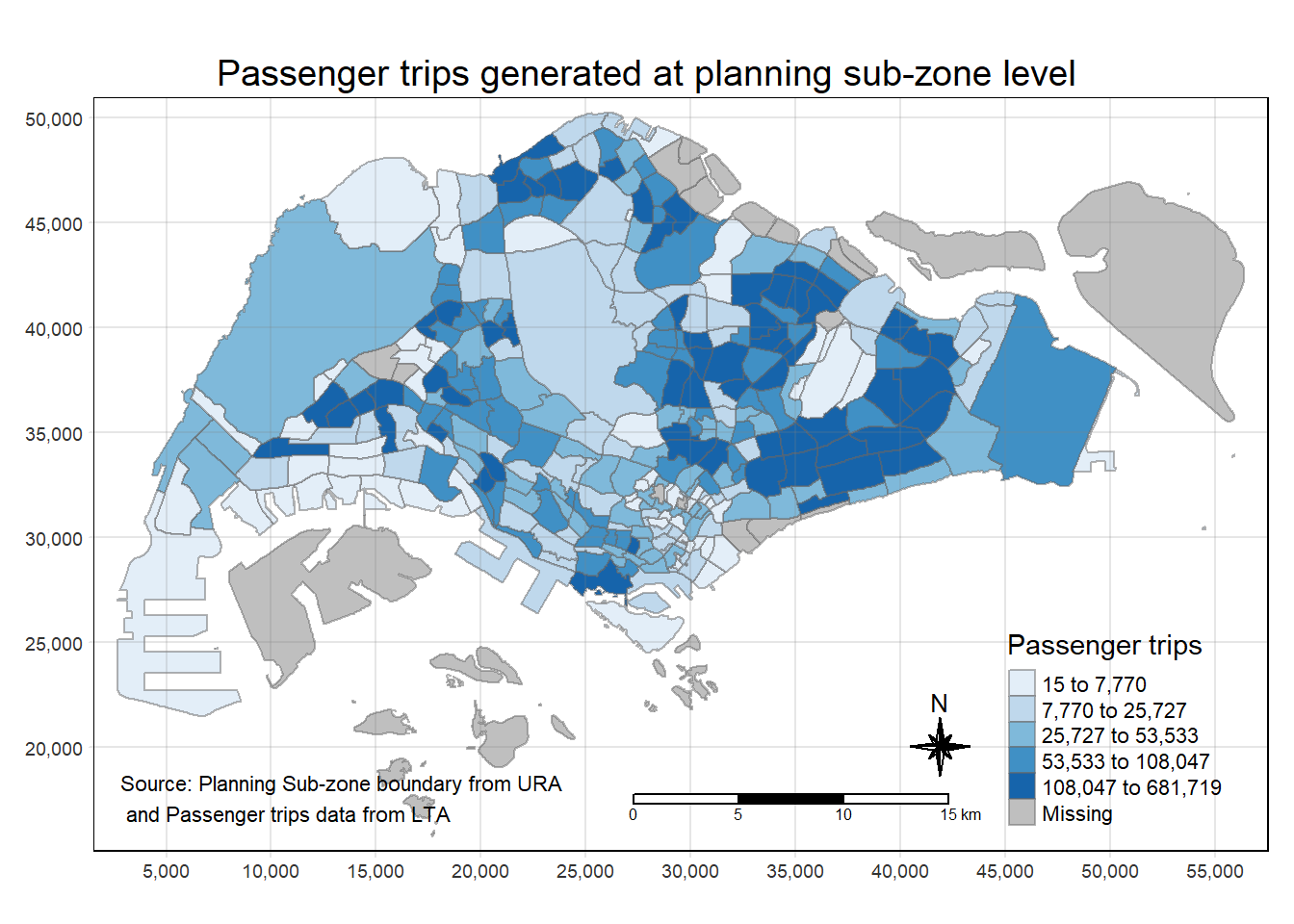
In-class Exercise 1: My First Date with Geospatial Data Science

2023-11-18

## The Task

In this in-class exercise, you are required to prepare a choropleth map showing the distribution of passenger trips at planning sub-zone by integrating *Passenger Volume by Origin Destination Bus Stops* and bus stop data sets downloaded from LTA DataMall and Planning Sub-zone boundary of URA Master Plan 2019 downloaded from data.gov.sg.



The specific task of this in-class exercise are as follows:

* to import *Passenger Volume by Origin Destination Bus Stops* data set downloaded from LTA DataMall in to RStudio environment,
* to import geospatial data in ESRI shapefile format into sf data frame format,
* to perform data wrangling by using appropriate functions from tidyverse and sf pakcges, and
* to visualise the distribution of passenger trip by using tmap methods and functions.

## Getting Started

Three R packages will be used in this in-class exercise, they are:

* tidyverse for non-spatial data handling,
* sf for geospatial data handling,
* tmap for thematic mapping, and
* knitr for creating html table.

## The task

Using the steps you learned from Hands-on Exercise 1, load these three R packages into RStudio.

## The solution

pacman::p\_load(tmap, sf, tidyverse,   
 knitr)

## Importing the OD data

Firstly, we will import the *Passenger Volume by Origin Destination Bus Stops* data set downloaded from LTA DataMall by using read\_csv() of **readr** package.

## The task

Using the steps you learned from Hands-on Exercise 1, import *origin\_destination\_bus\_202308.csv* downloaded from LTA DataMall into RStudio and save it as a tibble data frame called odbus.

## The solution

odbus <- read\_csv("data/aspatial/origin\_destination\_bus\_202308.csv")

A quick check of odbus tibble data frame shows that the values in OROGIN\_PT\_CODE and DESTINATON\_PT\_CODE are in numeric data type.

glimpse(odbus)

Rows: 5,709,512  
Columns: 7  
$ YEAR\_MONTH <chr> "2023-08", "2023-08", "2023-08", "2023-08", "2023-…  
$ DAY\_TYPE <chr> "WEEKDAY", "WEEKENDS/HOLIDAY", "WEEKENDS/HOLIDAY",…  
$ TIME\_PER\_HOUR <dbl> 16, 16, 14, 14, 17, 17, 17, 17, 7, 17, 14, 10, 10,…  
$ PT\_TYPE <chr> "BUS", "BUS", "BUS", "BUS", "BUS", "BUS", "BUS", "…  
$ ORIGIN\_PT\_CODE <chr> "04168", "04168", "80119", "80119", "44069", "4406…  
$ DESTINATION\_PT\_CODE <chr> "10051", "10051", "90079", "90079", "17229", "1722…  
$ TOTAL\_TRIPS <dbl> 7, 2, 3, 10, 5, 4, 3, 22, 3, 3, 7, 1, 3, 1, 3, 1, …

## The task

Using appropriate tidyverse functions to convert these data values into factor data type.

## The solution

odbus$ORIGIN\_PT\_CODE <- as.factor(odbus$ORIGIN\_PT\_CODE)  
odbus$DESTINATION\_PT\_CODE <- as.factor(odbus$DESTINATION\_PT\_CODE)

Notice that both of them are in factor data type now.

glimpse(odbus)

Rows: 5,709,512  
Columns: 7  
$ YEAR\_MONTH <chr> "2023-08", "2023-08", "2023-08", "2023-08", "2023-…  
$ DAY\_TYPE <chr> "WEEKDAY", "WEEKENDS/HOLIDAY", "WEEKENDS/HOLIDAY",…  
$ TIME\_PER\_HOUR <dbl> 16, 16, 14, 14, 17, 17, 17, 17, 7, 17, 14, 10, 10,…  
$ PT\_TYPE <chr> "BUS", "BUS", "BUS", "BUS", "BUS", "BUS", "BUS", "…  
$ ORIGIN\_PT\_CODE <fct> 04168, 04168, 80119, 80119, 44069, 44069, 20281, 2…  
$ DESTINATION\_PT\_CODE <fct> 10051, 10051, 90079, 90079, 17229, 17229, 20141, 2…  
$ TOTAL\_TRIPS <dbl> 7, 2, 3, 10, 5, 4, 3, 22, 3, 3, 7, 1, 3, 1, 3, 1, …

### Extracting the study data

## The task

For the purpose of this exercise, we will extract commuting flows during the weekday morning peak. Call the output tibble data table as origin7\_9.

## The solution

origin7\_9 <- odbus %>%  
 filter(DAY\_TYPE == "WEEKDAY") %>%  
 filter(TIME\_PER\_HOUR >= 7 &  
 TIME\_PER\_HOUR <= 9) %>%  
 group\_by(ORIGIN\_PT\_CODE) %>%  
 summarise(TRIPS = sum(TOTAL\_TRIPS))

It should look similar to the data table below.

kable(head(origin7\_9))

| ORIGIN\_PT\_CODE | TRIPS |
| --- | --- |
| 01012 | 1617 |
| 01013 | 813 |
| 01019 | 1620 |
| 01029 | 2383 |
| 01039 | 2727 |
| 01059 | 1415 |

We will save the output in rds format for future used.

write\_rds(origin7\_9, "data/rds/origin7\_9.rds")

The code chunk below will be used to import the save origin7\_9.rds into R environment.

origin7\_9 <- read\_rds("data/rds/origin7\_9.rds")

## Working with Geospatial Data

In this section, you are required to import two shapefile into RStudio, they are:

* BusStop: This data provides the location of bus stop as at last quarter of 2022.
* MPSZ-2019: This data provides the sub-zone boundary of URA Master Plan 2019.

### Importing geospatial data

## The task

Using the steps you learned from Hands-on Exercise 1, import *BusStop* downloaded from LTA DataMall into RStudio and save it as a sf data frame called busstop.

## The solution

busstop <- st\_read(dsn = "data/geospatial",  
 layer = "BusStop") %>%  
 st\_transform(crs = 3414)

Reading layer `BusStop' from data source   
 `D:\tskam\ISSS624\In-class\_Ex\In-class\_Ex1\data\geospatial'   
 using driver `ESRI Shapefile'  
Simple feature collection with 5161 features and 3 fields  
Geometry type: POINT  
Dimension: XY  
Bounding box: xmin: 3970.122 ymin: 26482.1 xmax: 48284.56 ymax: 52983.82  
Projected CRS: SVY21

The structure of busstop sf tibble data frame should look as below.

glimpse(busstop)

Rows: 5,161  
Columns: 4  
$ BUS\_STOP\_N <chr> "22069", "32071", "44331", "96081", "11561", "66191", "2338…  
$ BUS\_ROOF\_N <chr> "B06", "B23", "B01", "B05", "B05", "B03", "B02A", "B02", "B…  
$ LOC\_DESC <chr> "OPP CEVA LOGISTICS", "AFT TRACK 13", "BLK 239", "GRACE IND…  
$ geometry <POINT [m]> POINT (13576.31 32883.65), POINT (13228.59 44206.38),…

## The task

Using the steps you learned from Hands-on Exercise 1, import *MPSZ-2019* downloaded from eLearn into RStudio and save it as a sf data frame called mpsz.

## The solution

mpsz <- st\_read(dsn = "data/geospatial",  
 layer = "MPSZ-2019") %>%  
 st\_transform(crs = 3414)

Reading layer `MPSZ-2019' from data source   
 `D:\tskam\ISSS624\In-class\_Ex\In-class\_Ex1\data\geospatial'   
 using driver `ESRI Shapefile'  
Simple feature collection with 332 features and 6 fields  
Geometry type: MULTIPOLYGON  
Dimension: XY  
Bounding box: xmin: 103.6057 ymin: 1.158699 xmax: 104.0885 ymax: 1.470775  
Geodetic CRS: WGS 84

The structure of mpsz sf tibble data frame should look as below.

glimpse(mpsz)

Rows: 332  
Columns: 7  
$ SUBZONE\_N <chr> "MARINA EAST", "INSTITUTION HILL", "ROBERTSON QUAY", "JURON…  
$ SUBZONE\_C <chr> "MESZ01", "RVSZ05", "SRSZ01", "WISZ01", "MUSZ02", "MPSZ05",…  
$ PLN\_AREA\_N <chr> "MARINA EAST", "RIVER VALLEY", "SINGAPORE RIVER", "WESTERN …  
$ PLN\_AREA\_C <chr> "ME", "RV", "SR", "WI", "MU", "MP", "WI", "WI", "SI", "SI",…  
$ REGION\_N <chr> "CENTRAL REGION", "CENTRAL REGION", "CENTRAL REGION", "WEST…  
$ REGION\_C <chr> "CR", "CR", "CR", "WR", "CR", "CR", "WR", "WR", "CR", "CR",…  
$ geometry <MULTIPOLYGON [m]> MULTIPOLYGON (((33222.98 29..., MULTIPOLYGON (…

|  |
| --- |
| Note |
| * st\_read() function of sf package is used to import the shapefile into R as sf data frame. * st\_transform() function of sf package is used to transform the projection to crs 3414. |

## Geospatial data wrangling

### Combining Busstop and mpsz

Code chunk below populates the planning subzone code (i.e. SUBZONE\_C) of mpsz sf data frame into busstop sf data frame.

busstop\_mpsz <- st\_intersection(busstop, mpsz) %>%  
 select(BUS\_STOP\_N, SUBZONE\_C) %>%  
 st\_drop\_geometry()

|  |
| --- |
| Note |
| * st\_intersection() is used to perform point and polygon overly and the output will be in point sf object. * select() of dplyr package is then use to retain only BUS\_STOP\_N and SUBZONE\_C in the busstop\_mpsz sf data frame. * five bus stops are excluded in the resultant data frame because they are outside of Singapore bpundary. |

Before moving to the next step, it is wise to save the output into rds format.

write\_rds(busstop\_mpsz, "data/rds/busstop\_mpsz.csv")

## The task

Next, we are going to append the planning subzone code from busstop\_mpsz data frame onto odbus7\_9 data frame.

## The solution

origin\_SZ <- left\_join(origin7\_9 , busstop\_mpsz,  
 by = c("ORIGIN\_PT\_CODE" = "BUS\_STOP\_N")) %>%  
 rename(ORIGIN\_BS = ORIGIN\_PT\_CODE,  
 ORIGIN\_SZ = SUBZONE\_C) %>%  
 group\_by(ORIGIN\_SZ) %>%  
 summarise(TOT\_TRIPS = sum(TRIPS))

Before continue, it is a good practice for us to check for duplicating records.

duplicate <- origin\_SZ %>%  
 group\_by\_all() %>%  
 filter(n()>1) %>%  
 ungroup()

If duplicated records are found, the code chunk below will be used to retain the unique records.

origin\_data <- unique(origin\_SZ)

It will be a good practice to confirm if the duplicating records issue has been addressed fully.

## The task

Next, write a code chunk to update od\_data data frame with the planning subzone codes.

## The solution

origintrip\_SZ <- left\_join(mpsz,   
 origin\_SZ,  
 by = c("SUBZONE\_C" = "ORIGIN\_SZ"))

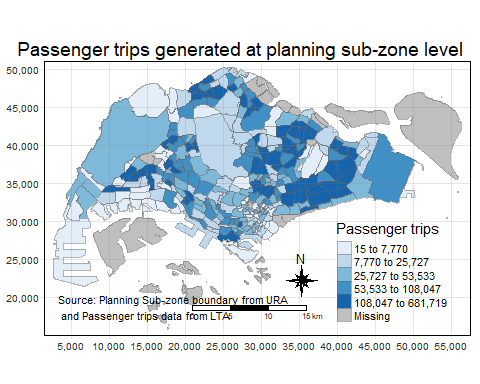
## Choropleth Visualisation

## The task

Using the steps you had learned, prepare a choropleth map showing the distribution of passenger trips at planning sub-zone level.

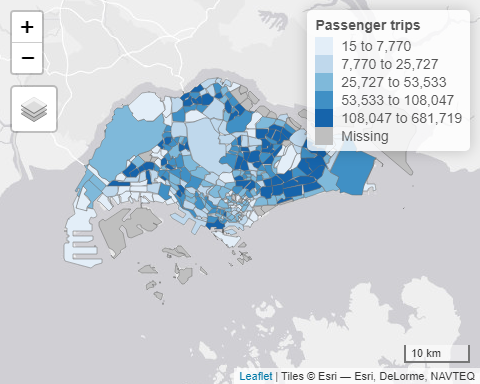
## The solution

tm\_shape(origintrip\_SZ)+  
 tm\_fill("TOT\_TRIPS",   
 style = "quantile",   
 palette = "Blues",  
 title = "Passenger trips") +  
 tm\_layout(main.title = "Passenger trips generated at planning sub-zone level",  
 main.title.position = "center",  
 main.title.size = 1.2,  
 legend.height = 0.45,   
 legend.width = 0.35,  
 frame = TRUE) +  
 tm\_borders(alpha = 0.5) +  
 tm\_compass(type="8star", size = 2) +  
 tm\_scale\_bar() +  
 tm\_grid(alpha =0.2) +  
 tm\_credits("Source: Planning Sub-zone boundary from URA\n and Passenger trips data from LTA",   
 position = c("left", "bottom"))



### Creating interactive map

tmap\_mode("view")  
tmap\_options(check.and.fix = TRUE)  
tm\_shape(origintrip\_SZ)+  
 tm\_fill("TOT\_TRIPS",   
 style = "quantile",   
 palette = "Blues",  
 title = "Passenger trips") +  
 tm\_layout(main.title = "Passenger trips generated at planning sub-zone level",  
 main.title.position = "center",  
 main.title.size = 1.2,  
 legend.height = 0.45,   
 legend.width = 0.35,  
 frame = TRUE) +  
 tm\_borders(alpha = 0.5) +  
 tm\_compass(type="8star", size = 2) +  
 tm\_scale\_bar() +  
 tm\_grid(alpha =0.2) +  
 tm\_credits("Source: Planning Sub-zone boundary from URA\n and Passenger trips data from LTA",   
 position = c("left", "bottom"))



:::