NYPD Racial Profiling

Philyoung Jeong

24/11/23

Contents

Originality declaration

I, [**Philyoung Jeong**], confirm that the work presented in this assessment is my own. Where information has been derived from other sources, I confirm that this has been indicated in the work.

New York Stop, Question and Frisk

Initial project scope

Research Question

- Does New York City (NYC)'s stop-question-and frisk programme with regards to the suspects' race type show some kinds of spatial pattern across community districts?

Hypothesis

- New York City's stop-question-and frisk programme has been criticised for its controversy over racial-profiling. According to Gelman, Fagan and Kiss (2007), African and Hispanic people were more frequently stopped and inspected more often than white people. Therefore, my research will primarily focus on inspection cases with regard to suspects' racial type.
  - My null and alternative hypotheses are as follows:
    * H0: There is no relationship between densities of police questioning cases and the suspects' racial type
    * H1: There is a spatial autocorrelation between densities of police questioning cases and the suspects' racial type

Data

Stop, Question and Frisk Data:

- This data provides a summary of stop-question-and frisk programme in 2021.
- The data was collected by New York City Police Department (NYPD).
- The dataset contains 83 columns related to the programme: among these columns, my columns of interest are stop_frisk_year, stop_location_x, stop_location_y, stop_location_precinct, and suspect_race_description.
  - stop_location_x and stop_location_y provides local coordinate reference system. There is no missing values in this column, which means we could convert point data to sf object without any issues.
  - Suspect_race_description provides the ethnicity of person who has been asked to stop. There are null values so we have to drop them during the data processing.
  - The stop_frisk_year column offers in which year this happened.
  - The stop_location_precinct will enable us to join it with spatial data.

New York Police Districts:

- This data presents police boundaries in New York.
- The data was compiled by Department of City Planning.
- There are 77 rows and four columns. One of the columns includes precincts.
- When I read in the spatial data, the initial Geodetic CRS is WGS 84 so we need to transform it to local CRS.
- The accompanying metadata of the boundary data says the local CRS is EPSG 2263 but it seems incorrect as the boundaries are not shown with this CRS. A better CRS is EPSG 6538 as it uses meters (https://epsg.io/6538).

Data wrangling

- Dropping NA values and filtering suspect_race_description column
- Reprojecting CRS to local CRS
- Apply st_intersect() on spatial data and attribute data

Analysis

- Map distribution of suspects' race type in New York City (NYC)
- Define neighbours based on one of the criteria - contiguity, distance and KNN
- Create spatial weight matrix and spatial weight list for Moran's I test
- Conduct Global Moran's I and Geary's C to see a global spatial pattern
- Run Local Moran's I test to see if there is local variation that is not captured by the global model

Limitations and Assumptions

- Due to missing values in suspects' ethnicity column, some police questioning cases will not be presented in my analysis
- Another limitation could be related to my columns of interest. As my analysis will be identifying spatial autocorrelation with regard to ethnicity so other variables that might be of importance cannot be fully reflected and could be neglected in the spatial pattern.
- I will assume that the analysed spatial patterns with regard to the suspects' race type are a good representation of the actual police question cases in New York.
- The stop-question-frisk data only has 2021 record, while the New York police boundary was last updated in 2022. Therefore, I will assume that the police questioning cases are well captured within the boundary.

Packages

```
library(dplyr)
library(sf)
library(tidyverse)
library(tmap)
```

```
library(tmaptools)
library(janitor)
library(here)
library(ggplot2)
```

Data loading

Read in boundary and csv data.

Boundary data

```
#install.packages("xfun")
```

```
# NYC Police District boundary
NY_District <- st_read(here::here('Police Precincts.geojson'))
```

```
## Reading layer `Police Precincts' from data source
##   `C:\Users\phily\Desktop\UCL\Term 1\GIS\casa0005-final-exam-22-23-phily5051\Police Precincts.geojson
##   using driver `GeoJSON'
## Simple feature collection with 77 features and 3 fields
## Geometry type: MULTIPOLYGON
## Dimension:     XY
## Bounding box:  xmin: -74.25559 ymin: 40.49613 xmax: -73.70001 ymax: 40.91553
## Geodetic CRS:  WGS 84
```

```
summary(NY_District)
```

```
##    precinct          shape_area         shape_leng                  geometry
##  Length:77          Length:77          Length:77          MULTIPOLYGON :77
##  Class :character   Class :character   Class :character   epsg:4326    : 0
##  Mode  :character   Mode  :character   Mode  :character   +proj=long...: 0
```

There are 77 observations in boundary data.

Attribute data

In the attribute data, there are NA values which I checked in excel so I read in data with specification of 'N/A' values. For convenience, I converted the excel format to csv format.

```
# csv data - contains na values
sqf_2021 <- read_csv("sqf_2021.csv",
                     locale = locale(encoding = "latin1"),
                     na = c("","NA", "n/a"),
                     col_names = T) %>%
  clean_names()
```

```
## Rows: 8947 Columns: 83
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (75): MONTH2, DAY2, STOP_WAS_INITIATED, RECORD_STATUS_CODE, ISSUING_OFF...
## dbl   (6): STOP_ID, YEAR2, OBSERVED_DURATION_MINUTES, STOP_DURATION_MINUTES,...
## date  (1): STOP_FRISK_DATE
## time  (1): STOP_FRISK_TIME
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
# select columns
sqf_2021 <- sqf_2021 %>%
  dplyr::select(year2, suspect_race_description, stop_location_precinct,
                stop_location_x, stop_location_y)
```

There are 8947 observations in the attribute data.

Class type check

Check variable types to make sure that each column contains the most suitable data type.

```
# pivot_longer() is useful for biding all the variables under the same column
# when checking their types

Data_type_list <- sqf_2021 %>%
  summarise_all(class) %>%
  pivot_longer(everything(),
               names_to = 'All_variables',
               values_to = 'Variable_class')

Data_type_list
```

```
## # A tibble: 5 x 2
##   All_variables            Variable_class
##   <chr>                    <chr>
## 1 year2                    numeric
## 2 suspect_race_description character
## 3 stop_location_precinct   character
## 4 stop_location_x          numeric
## 5 stop_location_y          numeric
```

It seems our data type matches with each column.

Data wrangling

Remove null & NA values

Let's check whether there are NA values in our columns.

```
# Check na values
sum(is.na(sqf_2021$stop_location_precinct))
```

```
## [1] 0
```

```
sum(is.na(sqf_2021$stop_location_x))
```

```
## [1] 0
```

```
sum(is.na(sqf_2021$stop_location_y))
```

```
## [1] 0
```

```
sum(is.na(sqf_2021$suspect_race_description))
```

```
## [1] 0
```

```
sum(is.na(sqf_2021$year2))
```

```
## [1] 0
```

There are no NA values in the columns.

Null value check

```
# Check drop null values
sqf_2021[!sqf_2021$stop_location_precinct == ''|
           sqf_2021$stop_location_x == '' |
           sqf_2021$stop_location_y == '' |
           sqf_2021$suspect_race_description == '',]
```

```
## # A tibble: 8,947 x 5
##    year2 suspect_race_description stop_location_precinct stop_location_x
##    <dbl> <chr>                    <chr>                             <dbl>
##  1  2021 BLACK                    042                             1013737
##  2  2021 BLACK                    105                             1052511
##  3  2021 ASIAN / PACIFIC ISLANDER 106                             1025662
##  4  2021 BLACK                    073                             1010122
##  5  2021 BLACK                    026                              996623
##  6  2021 WHITE HISPANIC           042                             1013048
##  7  2021 BLACK                    076                              984160
##  8  2021 WHITE HISPANIC           042                             1016413
##  9  2021 BLACK                    043                             1025585
## 10  2021 WHITE HISPANIC           076                              983789
## # i 8,937 more rows
## # i 1 more variable: stop_location_y <dbl>
```

```
sqf_2021
```

```
## # A tibble: 8,947 x 5
##    year2 suspect_race_description stop_location_precinct stop_location_x
##    <dbl> <chr>                    <chr>                             <dbl>
##  1  2021 BLACK                    042                             1013737
##  2  2021 BLACK                    105                             1052511
##  3  2021 ASIAN / PACIFIC ISLANDER 106                             1025662
##  4  2021 BLACK                    073                             1010122
##  5  2021 BLACK                    026                              996623
##  6  2021 WHITE HISPANIC           042                             1013048
##  7  2021 BLACK                    076                              984160
##  8  2021 WHITE HISPANIC           042                             1016413
##  9  2021 BLACK                    043                             1025585
## 10  2021 WHITE HISPANIC           076                              983789
## # i 8,937 more rows
## # i 1 more variable: stop_location_y <dbl>
```

With the code chunk above, we have dropped null values.

Suspects' racial type distribution

Let's check racial distribution of stopped persons, which might give us a general idea of racial profiling.

```
library(stringr)

# group data by race types
race_types <- sqf_2021 %>%
  filter(., suspect_race_description == 'WHITE HISPANIC' |
           suspect_race_description == 'WHITE' |
```

```
           suspect_race_description == 'BLACK' |
           suspect_race_description == 'ASIAN / PACIFIC ISLANDER' |
           suspect_race_description == 'MIDDLE EASTERN/SOUTWEST ASIAN' |
           suspect_race_description == 'BLACK HISPANIC' |
           suspect_race_description == 'AMERICAN INDIAN/ALASKAN NATIVE'
           )


# plot histogram
p <- ggplot(race_types, aes(x = suspect_race_description, color = suspect_race_description)) +

  geom_histogram(stat = 'count', position = 'dodge', bindwidth = 50, alpha = 0.5) +
  labs(title = 'NY SQF Race Profiling',
       subtitle = 'By Ethnicity',
       x = 'Race',
       y = 'Frequency') +
  theme_classic()+
  theme(plot.title = element_text(hjust = 0.5),
        plot.subtitle = element_text(hjust = 0.5)) +

  # x labels are long so I came up with this
  # credits to https://datavizpyr.com/how-to-dodge-overlapping-text-on-x-axis-labels-in-ggplot2/
  scale_x_discrete(guide = guide_axis(n.dodge=3))
```

```
## Warning in geom_histogram(stat = "count", position = "dodge", bindwidth = 50, :
## Ignoring unknown parameters: `binwidth`, `bins`, `pad`, and `bindwidth`
```

```
# add aesthetics:
# adjusted from ggplot2 website(https://ggplot2.tidyverse.org/reference/element.html)

# x, y axis label
p + theme(axis.title.x = element_text(size = 8,
                                      color = 'black',
                                      face = 'bold',
                                      angle = 0)) +

  theme(axis.title.y = element_text(size = 8,
                                    color = 'black',
                                    face = 'bold',
                                    angle = 0)) +

  # x, y axis tick mark labels
  theme(axis.text.x = element_text(family = 'mono',
                                     face = 'bold',
                                     colour = 'black',
                                     size = 6)) +

  theme(axis.text.y = element_text(family = 'mono',
                                     face = 'bold',
                                     colour = 'black',
                                     size = 6)) +

  # legend title text
  theme(legend.title = element_text(color = 'blue',
```

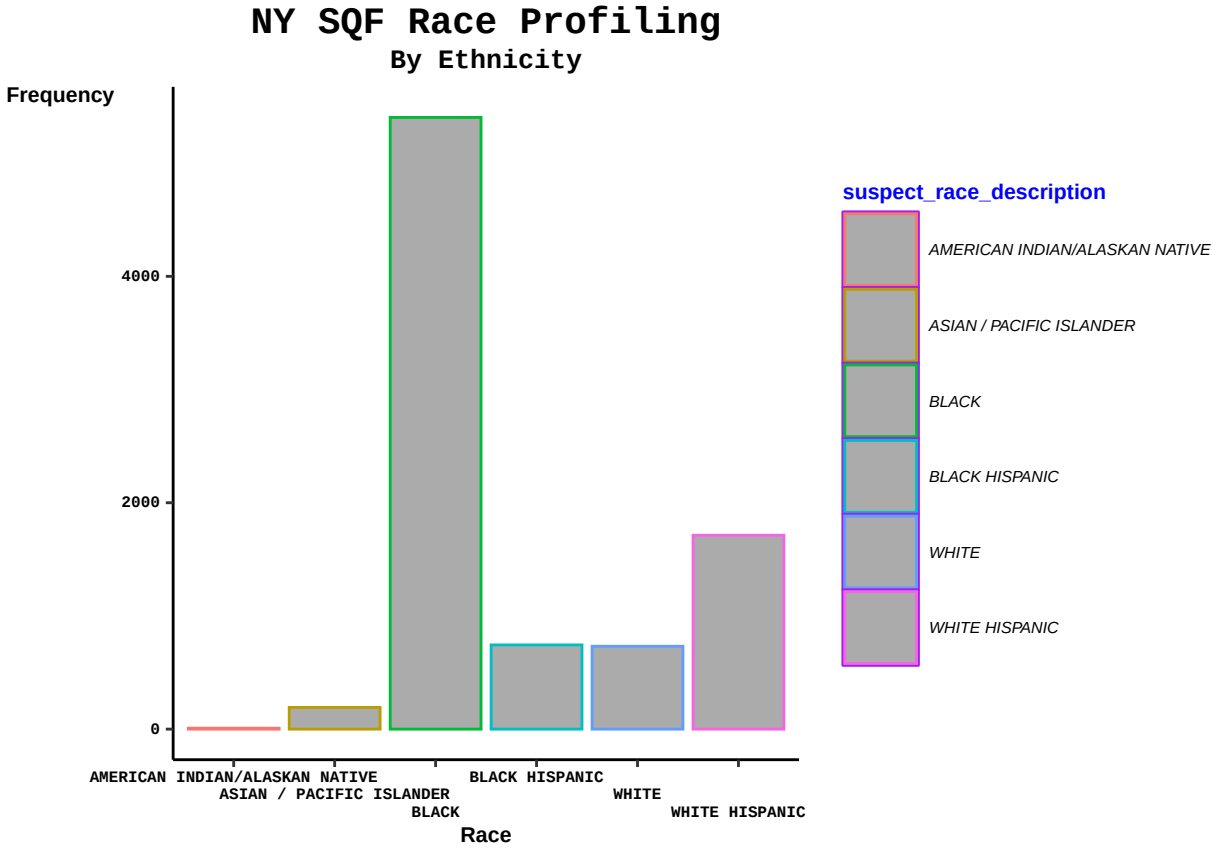
```
                                    face = 'bold',
                                    size = 8)) +

  # legend text
  theme(legend.text = element_text(face = 'italic',
                                   color = 'black',
                                   size = 6)) +

  # legend bounding box
  theme(legend.key = element_rect(color = "purple",
                                  fill = "white"),
        legend.key.size = unit(1, "cm")) +

  # ggplot title
  theme(plot.title = element_text(size = 14,
                                  colour = 'black',
                                  face = 'bold',
                                  family = 'mono')) +

  # ggplot subtitle
  theme(plot.subtitle = element_text(size = 10,
                                     color = 'black',
                                     face = 'bold',
                                     family = 'mono'))
```



We can see that police questioning was predominantly occurred with Black people.

Therefore, I will refine my hypothesis as follows:

- **Hypothesis**
  - H0: There is no relationship between densities of police questioning cases and Black people.
  - H1: There is a spatial autocorrelation between densities of police questioning cases and Black people.

Filtering data

Based on the histogram, I will filter suspect's race type.

```
# filter black suspects
sqf_2021 <- sqf_2021 %>%
  filter(suspect_race_description == 'BLACK')
```

Our data has been reduced to 5404 observations.

Convert CSV to sf object

To convert CSV to sf object, I employed st_as_sf().

```
# csv to sf object
sqf_2021_spatial <- sqf_2021 %>%

  st_as_sf(., coords = c('stop_location_x', 'stop_location_y'),

           # normally you have to set your crs 4326 when you convert your point data
           # to sf object but it didn't work so I set the crs to crs.
           # adjusted from https://mattherman.info/blog/point-in-poly/
           crs = 2908)


# and then I converted it to EPSG 6538 because it is meter-based
```

CRS reprojection to local CRS

```
sqf_2021_spatial <- sqf_2021_spatial %>% st_transform(6538)
NY_District <- NY_District %>% st_transform(6538)
```

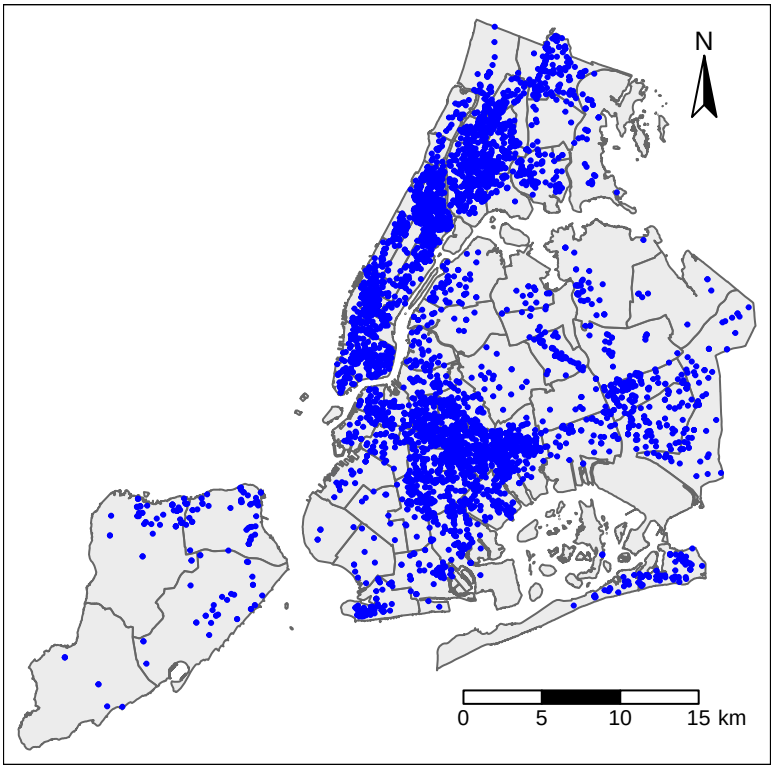
Make a map

To check whether points are within the NYC boundaries, we can draw a map to check visually.

```
# map
tmap_mode('plot')
```

```
## tmap mode set to plotting
```

```
tm_shape(NY_District) +
  tm_polygons(col = NA, alpha = 0.5) +
tm_shape(sqf_2021_spatial) +
  tm_dots(col = 'blue') +
  tm_layout(main.title = 'NYC Police Inspection on Black',
            main.title.position = 'center') +
  tm_compass(type = "arrow", position = c("right", "top") , size = 2) +
    tm_scale_bar(text.size = 0.7, position =  c("right", "bottom"))
```



It seems like that all the police inspection occurred within the boundary.

Spatial subsetting & plot the police inspection cases within NYC

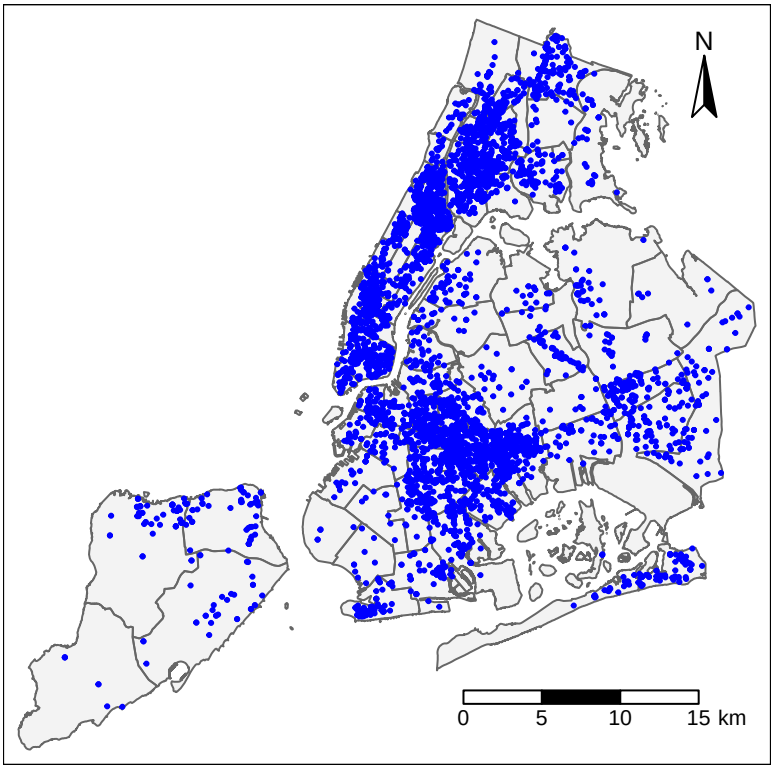
```
# spatial subsetting
# points within the boundary
sqf_2021_spatial_sub <- sqf_2021_spatial[NY_District,]


# Map again
tmap_mode('plot')
```

```
## tmap mode set to plotting
```

```
tm_shape(NY_District) +
  tm_polygons(col = NA, alpha = 0.3) +

tm_shape(sqf_2021_spatial_sub) +
  tm_dots(col = 'blue') +
  tm_layout(main.title = 'NYC Police Inspection on Black') +
  tm_compass(type = "arrow", position = c("right", "top") , size = 2) +
    tm_scale_bar(text.size = 0.7, position =  c("right", "bottom"))
```



After conducting a spatial-subset on our attribute data, we dropped 5 observations. Therefore we now have 5399 observations. We can be sure that our police inspection on Black people are now within the boundary.

Data Aanlysis

In this section, I am going to use Global spatial autocorrelation methods as well as Local Moran's I and use these measures to test whether densities of police inspection cases on Black show spatial autocorrelation or not.

According to Getis (2010), spatial autocorrelation is the relationship between neighbouring spatial units. Spatial autocorrelation requires continuous observations to be spatially referenced. The reason we are using spatially continuous observations in lieu of count data for spatial autocorrelation measures is that summing the counts generates results that are dependent on the size of spatial unit.

Therefore, to create continuous observations of police inspection cases on Black people, we need to count all the police questioning on Black people that fall within each borough in the City and normalise it based on some sort of underlying data such as population and area.

st_join() is a left join so it will retain all the data in the left side, which indicates if a borough is empty, there will still be a row for that borough. So we don't use it here. Instead we will use st_intersects().

Create continuous observations

st_intersects() returns a list of points in each spatial unit.

```
# st_inersects()
points_sf_joined <- NY_District %>%

  # police inspection cases on Black people that fall within each police precinct in NYC
```

```
  mutate(n = lengths(st_intersects(., sqf_2021_spatial_sub))) %>%
  janitor::clean_names() %>%

  # calculate area
  mutate(area = st_area(.)) %>%
  # density of the points per police precinct
  mutate(density = n/area)
```

We now have stop-question-and-frisk cases on Black people per each police precinct.

```
tmap_mode('plot')
```

Mapping to get a general understanding of our data

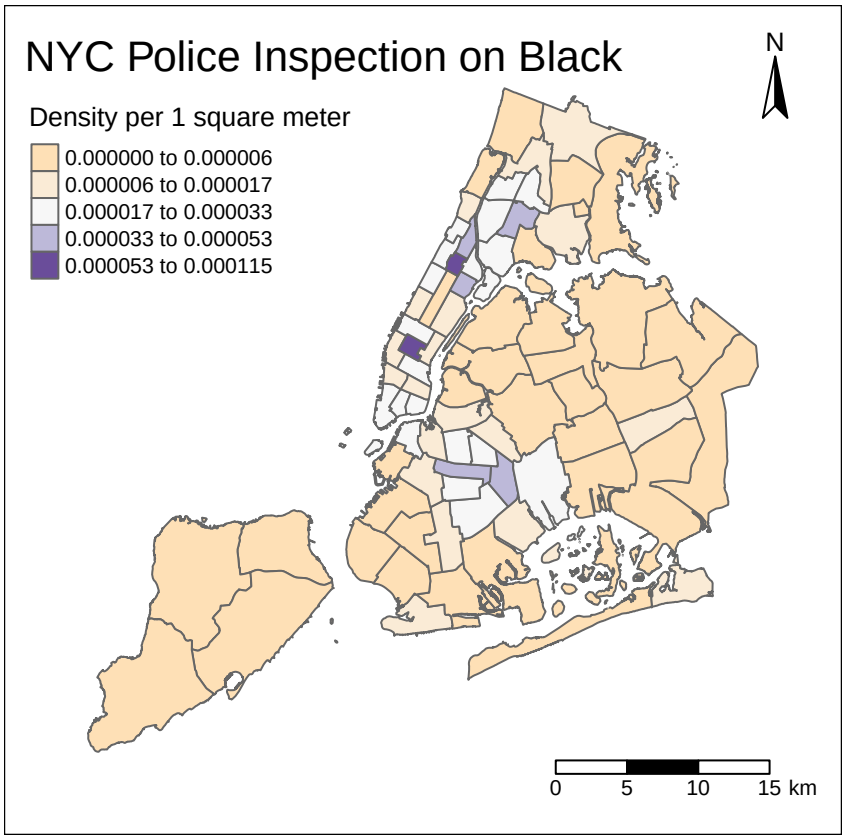
```
## tmap mode set to plotting
```

```
points_sf_joined <- points_sf_joined %>%

  # group by police precinct
  group_by(precinct) %>%

  summarise(density = first(density),
            inspection_count = first(n))


tm_shape(points_sf_joined) +
  tm_polygons('density',
              style = 'jenks',
              palette = 'PuOr',
              midpoint = NA,
              popup.vars = 'density',
              title = 'Density per 1 square meter') +
tm_compass(type = "arrow", position = c("right", "top") , size = 2) +
    tm_scale_bar(text.size = 0.7, position =  c("right", "bottom")) +
  tm_layout(legend.position = c("left", "top"),
            title= "NYC Police Inspection on Black",
            title.position = c('left', 'top'),
            inner.margins = 0.1)
```



From the map, it seems like we might have some clustering of police inspection cases on Black people in NYC which are coloured in purple and light purple. We can check this clustering with the use of global and local spatial autocorrelation test. We have two prerequisite for Moran's I test which are as follows: Firstly, we have to set the rule. This means that we have to define our neighbours based on some criteria, such as distance, k-nearest neighbours and contiguity. Secondly, we have to create a spatial weight matrix, which gives numeric values to neighbours (B - binary, W - Row standardisation, C - Globally standardisation)

Create Centroids

```
library(spdep)
```

```
## Warning:    'spdep'  R   4.2.2
```

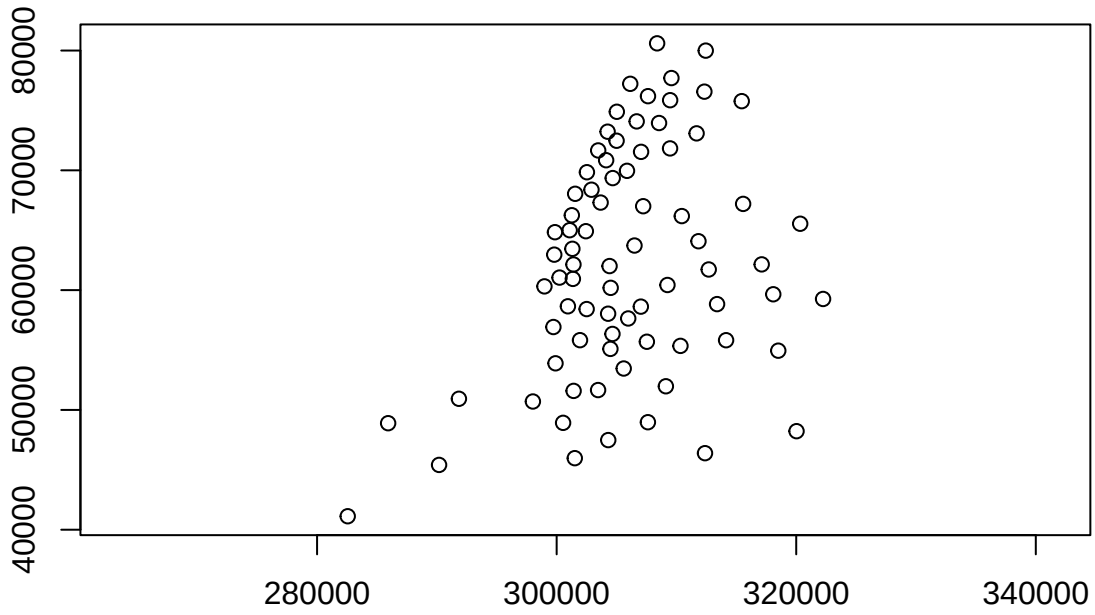
```
##          : sp
```

```
##          : spData
```

```
# create the centroids
coordsW <- points_sf_joined %>%
  st_centroid() %>%
  st_geometry()
```

```
## Warning in st_centroid.sf(.): st_centroid assumes attributes are constant over
## geometries of x
```

```
plot(coordsW, axes = TRUE)
```



List of neighbours

In this example, I will define my neighbours based on K-nearest neighbours. Conventionally, we set k's value as 4 because we think it's a good way to start with and which reflects our neighbours.

```
# 1) K-nearest neighbours - KNN
knn_precinct <-coordsW %>%
  knearneigh(., k=4)

NY_knn_nb <- knn_precinct %>%
  knn2nb()

# summary of our neighbour's list
summary(NY_knn_nb)
```

```
## Neighbour list object:
## Number of regions: 77
## Number of nonzero links: 308
## Percentage nonzero weights: 5.194805
## Average number of links: 4
## Non-symmetric neighbours list
## Link number distribution:
##
##  4
## 77
## 77 least connected regions:
## 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 3
```

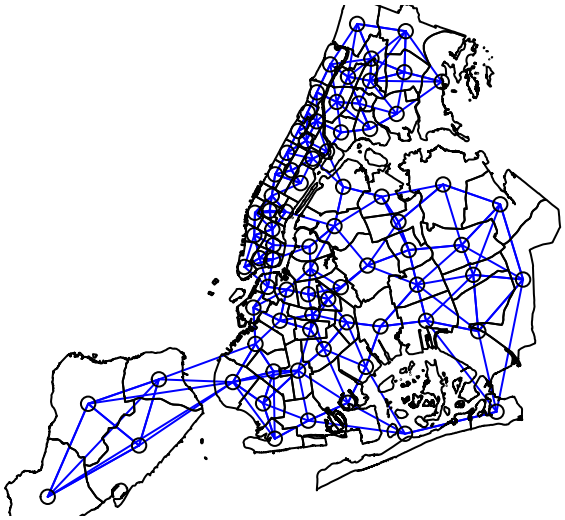
```
## 77 most connected regions:
## 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 3
```

The result indicates that we have the average number of 4 neighbours.

We can plot the neighbours as follows.

```
# plot the KNN neighbours
plot(NY_knn_nb, st_geometry(coordsW), col = 'blue')

# add the map underneath
plot(points_sf_joined$geometry, add = T)
```



We can have a general picture of how our neighbours are linked to eachother.

Spatial weight matrix

Spatial weight matrix represents the spatial elements of our data. It is a n by n matrix composed of ones and zeros. According to Solymosi and Medina (2022), spatial weight matrix can be thought of a data table that we are creating with our definition of neighbours that we have set previously.

From the neighbour's list, we can now make a spatial weight matrix. The reason that we standardise our matrix is that standardisation permits comparable spatial parameters. But we should also take into account geographical features in mind such as a river or lake between neighbours. Generally speaking, **'Row Standardisation'** is the most common way of standardising.

Row standardisation of a matrix makes sure that the sum total values in each row adds up to 1, and then we divide 1 by the number of neighbours in that row. For instance, if there are 4 neighbours in a row, you need to divide 1 by 4, and it returns 0.25.

```
#create a spatial weights matrix from these weights
NY_knn_weight_matrix <- NY_knn_nb %>%
  nb2mat(., style="W")

sum(NY_knn_weight_matrix)
```

```
## [1] 77
```

We just created row standardised spatial weight matrix.

Spatial weight list

Global and local model require spatial weight list type as object so we have to convert the neighbour's list we have created to a spatial weight list.

```
# Row standardised spatial weight list
NY_knn_weights_list <- NY_knn_nb %>%
  nb2listw(style = "W")
```

We just created a KNN-spatial weight list. We are now ready to do our tests.

Global Model

Global Moran's I Values close to 1 means that we have a clustering values. Values close to -1 means our data are dispersed. If the value is 0, it means there is no autocorrelation. We will use densities as it is spatially continuous observations.

```
Morans_I_NY_Global_Density <- points_sf_joined %>%
  pull(density) %>%
  as.vector() %>%
  moran.test(., NY_knn_weights_list)

Morans_I_NY_Global_Density
```

```
##
##  Moran I test under randomisation
##
## data:  .
## weights: NY_knn_weights_list
##
## Moran I statistic standard deviate = 5.1528, p-value = 1.283e-07
## alternative hypothesis: greater
## sample estimates:
## Moran I statistic       Expectation          Variance
##       0.333134230      -0.013157895       0.004516527
```

The Moran's I statistic = 0.33 indicates that we have some clustering (1 = clustered, 0 = no pattern, -1 = dispersed). The p-value is less than 0.05, indicating that the results are statistically significant at the 95% level of significance. The standard deviate is positive, which implies that the spatial distribution of high values and/or low values of residuals is more spatially clustered than it was expected.

Geary's C To further check whether similar values or dissimilar values are clustering.

```
Geary_C_NY_Global_Density <-
  points_sf_joined %>%
  pull(density) %>%
  as.vector() %>%
```

```
  geary.test(., NY_knn_weights_list)

Geary_C_NY_Global_Density
```

```
##
##  Geary C test under randomisation
##
## data:  .
## weights: NY_knn_weights_list
##
## Geary C statistic standard deviate = 2.5298, p-value = 0.005706
## alternative hypothesis: Expectation greater than statistic
## sample estimates:
## Geary C statistic       Expectation          Variance
##        0.74214702         1.00000000        0.01038885
```

The Geary's C statistic ranges between 0 and 2 (1 = no pattern, less than 1 = positive autocorrelation, more than 1 = negative autocorrelation). Geary's C statistic is 0.74 which indicates positive spatial autocorrelation (similar values are clustering).

Getis Ord If G is bigger than the expected values, it indicates high values are clustering. On the other hand, if G is smaller than the expected value, it implies low values are clustering.

```
G_NY_Global_Density <- points_sf_joined %>%
  pull(density) %>%
  as.vector() %>%
  globalG.test(., NY_knn_weights_list)
```

```
## Warning in globalG.test(., NY_knn_weights_list): Binary weights recommended
## (sepecially for distance bands)
```

```
G_NY_Global_Density
```

```
##
##  Getis-Ord global G statistic
##
## data:  .
## weights: NY_knn_weights_list
##
## standard deviate = 5.9736, p-value = 1.16e-09
## alternative hypothesis: greater
## sample estimates:
## Global G statistic        Expectation           Variance
##       2.261615e-02       1.315789e-02       2.506983e-06
```

The statistics indicates high values clustering as G is bigger than expected value.

To sum up, the three different global statistics are suggesting that there is a clear global spatial autocorrelation with regards to the police inspection on Black people in NYC. However, global measures might mask local trends. Therefore, it would be worth looking into local measures as well, which could capture local trends (hot/cold spots) to an extent.

Local Moran's I

Local Moran's I is a measure of degrees to which value at a target site is similar to values at adjacent sites.

```
# Using the localmoran() to generate I for each districts

I_NY_Local_count <- points_sf_joined %>%
  pull(inspection_count) %>%
  as.vector() %>%
  localmoran(., NY_knn_weights_list) %>%
  as_tibble()

I_NY_Local_Density <- points_sf_joined %>%
  pull(density) %>%
  as.vector() %>%
  localmoran(., NY_knn_weights_list) %>%
  as_tibble()


# have a glance at it
slice_head(I_NY_Local_Density, n=5)
```

```
## # A tibble: 5 x 5
##   Ii          E.Ii          Var.Ii      Z.Ii       `Pr(z != E(Ii))`
##   <localmrn>  <localmrn>    <localmrn>  <localmrn> <localmrn>
## 1  0.06445661 -0.0003983597 0.007358755  0.7560336 0.449629078
## 2 -0.39796903 -0.0011002943 0.020311066 -2.7847135 0.005357504
## 3  0.31133514 -0.0071354581 0.130922361  0.8801616 0.378771766
## 4  0.21990745 -0.0014066109 0.025957606  1.3736519 0.169549774
## 5  0.25100189 -0.0069968251 0.128396630  0.7200137 0.471516589
```

We can see that there are 5 columns of data but we will only use 'I score' column and 'z-score standard deviation' column.

```
# values to NY spatial polygons Dataframe
points_sf_joined <- points_sf_joined %>%
  mutate(inspection_count_I = as.numeric(I_NY_Local_count$Ii)) %>%
  mutate(inspection_count_Iz = as.numeric(I_NY_Local_count$Z.Ii)) %>%
  mutate(density_I = as.numeric(I_NY_Local_Density$Ii)) %>%
  mutate(density_Iz = as.numeric(I_NY_Local_Density$Z.Ii))
```

We have now added these values to our original data.

Mapping First, We set thresholds. +-2.58 indicates standard deviations away from the mean at a 99% confidence level. +-1.96 indicates standard deviations away from the mean at a 95% confidence level. And +-1.65 means standard deviations away from the mean at a 90% confidence level.

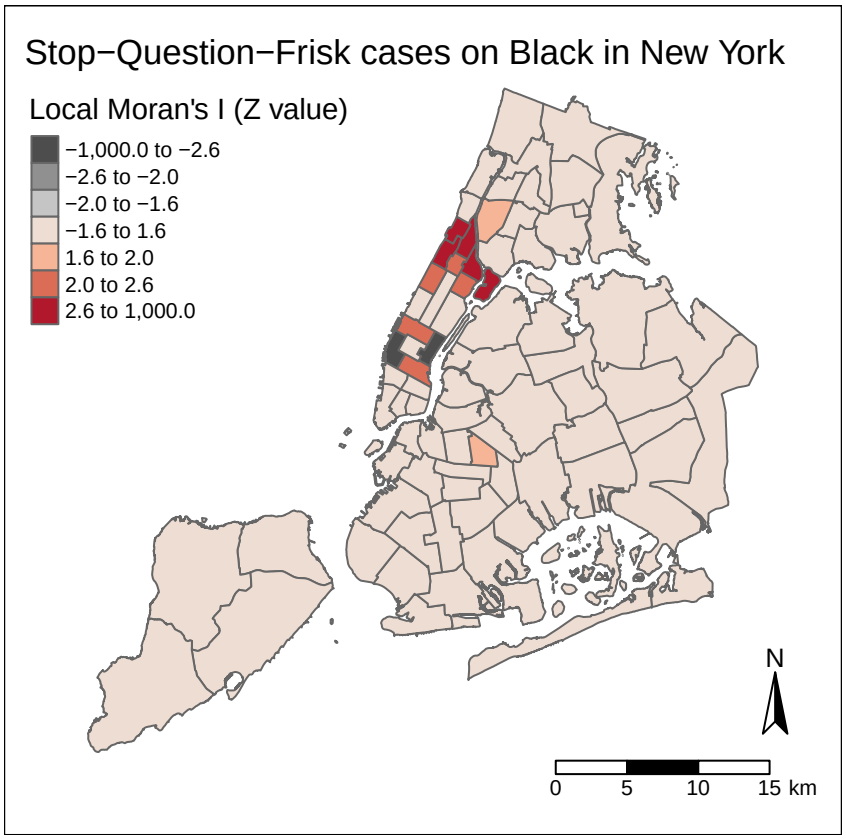
```
# setting thresholds
breaks1 <- c(-1000, -2.58, -1.96, -1.65, 1.65, 1.96, 2.58, 1000)

# colour palette
library(RColorBrewer)
MoranColours <- rev(brewer.pal(8, 'RdGy'))
```

As you can see, I also created a colour palette where higher values correspond to red colour.

```
# Set the breaks manually based on the rule that data points
breaks1<-c(-1000,-2.58,-1.96,-1.65,1.65,1.96,2.58,1000)
library(RColorBrewer)
MoranColours<- rev(brewer.pal(8, "RdGy"))
```

```
tm_shape(points_sf_joined) +
    tm_polygons("density_Iz",
        style="fixed",
        breaks=breaks1,
        palette=MoranColours,
        midpoint=NA,
        title="Local Moran's I (Z value)") +
tm_compass(type = "arrow", position = c('right', 'bottom') , size = 2) +
    tm_scale_bar(text.size = 0.7, position =  c("right", "bottom")) +
  tm_layout(legend.position = c("left", "top"),
            title= "Stop-Question-Frisk cases on Black in New York",
            title.position = c('left', 'top'),
            inner.margins = 0.1)
```



The map indicates police inspection cases on Black citizens were reported relatively high in those coloured in red and orange. Furthermore, this indicates that areas with a lot of police questioning cases were neighbouring other areas with a lot of police questioning cases.

We can find out where these red/orange areas are from by filtering z-values.

```
# add a hotspot column
points_sf_joined <- points_sf_joined %>%
  mutate(hot_spots = case_when(density_Iz > 1.96 ~ 'high_inspection',
                               TRUE ~ 'not_high'))

# show hotspots
hotspots <- points_sf_joined %>%
```

```
  filter(hot_spots == 'high_inspection') %>% print()
```

```
## Simple feature collection with 9 features and 8 fields
## Geometry type: MULTIPOLYGON
## Dimension:     XY
## Bounding box:  xmin: 299727.9 ymin: 62192.36 xmax: 307276.3 ymax: 74346.68
## Projected CRS: NAD83(2011) / New York Long Island
## # A tibble: 9 x 9
##   precinct density inspection_count                  geometry inspection_count_I
## * <chr>    [1/m^2]            <int>        <MULTIPOLYGON [m]>              <dbl>
## 1 13       2.01e-5               55 (((301557.4 64097.56, 30~          -0.0264
## 2 18       1.87e-5               56 (((300511.9 67350.91, 30~          -0.0228
## 3 23       3.96e-5               79 (((305719 69187.09, 3057~           0.0284
## 4 24       2.96e-5               70 (((303405.3 70458.46, 30~           0.000304
## 5 25       2.45e-5              102 (((305229.7 68202.24, 30~           0.425
## 6 26       2.03e-5               45 (((304548.9 72691.93, 30~          -0.168
## 7 28       9.92e-5              141 (((304951.6 71187.48, 30~           0.124
## 8 30       3.31e-5               58 (((305049.1 73703.78, 30~          -0.0299
## 9 32       5.28e-5              113 (((305528.9 74328.49, 30~           0.320
## # i 4 more variables: inspection_count_Iz <dbl>, density_I <dbl>,
## #   density_Iz <dbl>, hot_spots <chr>
```

We can identify police precincts where high police inspection on Black people is statistically high and these precincts are in Manhattan.

Interpretation

In order to identify spatial autocorrelation, global and local measures have been conducted. From the outputs of global measures, all three measures indicate that there is a global spatial autocorrelation with regards to the police inspection on Black people in NYC. I could have solely done Moran's I test but global Moran's I only tells whether clustering exists or not. Thus, I went further with other global measures, which tell us that similar values are clustering globally.

However, the global measures might cast some local trends so I also went on conducting local Moran's I with the hope of finding local trends. Local Moran's I gives values for each spatial unit in relation to neighbours.

If we look at the map of local trends, high police inspections on Black people were carried out predominantly in Manhattan in 2021. There could be two plausible reasons for this occurrence: Black demography and suspicion on Black.

Firstly, predominant cases were occurred in Harlem areas (precinct number: 25 - 32) or close to Harlem (precinct number: 13 to 24). These areas are known to be populated with Black people. Statistical Atlas's demographic proportion (n.d) shows that over 60% of inhabitants are Black. Therefore, one might say that police officers stop a passenger on the street and the person being questioned was Black by accident.

Another reason could be related to people's perception on Black people. According to Oliver (2003), there is a clear streotype, associating Black men with a potential threat. Hester and Gray (2018) also found out that being a tall and healthy black man could increase their chances of being stopped by a police officer.

These evidences could allude to us why the densities of police questioning on Black are high in certain areas and why they show spatial patterns.

Limitation and Reflection

- My research question whether densities of police inspection on Black people in NYC show spatial autocorrelation has been answered by using both global and local measures.

- These results might inform the New York Police Department to detect areas where police inspection were relatively too excessive for Black people. This might help the department prevent inappropriate or unnecessary police questioning.
- I have set 4 as my K-value as a rule of thumb. However, we could use cross validation, or plot the k values which reduces errors the most and test how different K-value affects our results. This can be a good step for a further research.
- Finally, we have to remember that these results are meaningful at this police precinct level. As Gilmond (2022) noted the results of something are only meaningful at that particular aggregation level. Therefore, the densities of stop-question-frisk cases on Black showing spatial autocorrelation are only meaningful at this particular aggregation level (police precinct level). In other aggregation level, the result could turn out to be different.

Reference

Gelman, A., Fagan, J. and Kiss, A. (2007) 'An analysis of the New York City police department's "stop-and-frisk" policy in the context of claims of racial bias'. Journal of the American statistical association, 102(479), pp.813-823.

Getis, A. (2010) 'Spatial autocorrelation. In Handbook of applied spatial analysis (pp. 255-278)'. Springer, Berlin, Heidelberg.

Gimoond, M. (2022) 'Intro to GIS and Spatial Analysis'[Online]. Available at: https://mgimond.github.io/Spatial/pitfalls-to-avoid.html#ecological-fallacy. (Accessed: 16 December 2022).

Hester, N. and Gray, K. (2018) 'For Black men, being tall increases threat stereotyping and police stops'. Proceedings of the National Academy of Sciences, 115(11), pp.2711-2715.

Oliver, M.B. (2003) 'African American men as "criminal and dangerous": Implications of media portrayals of crime on the "criminalization" of African American men'. Journal of African American Studies, pp.3-18.

Solymosi, R., and Medina, J. (2022) 'Crime Mapping in R'[Online]. Available at: https://maczokni.github.io/crime_mapping_textbook/. (Accessed: 16 December 2022)

Statistical Atlas. (n.d) 'Race and Ethnicity in Harlem, New York, New York'[Online]. Available at: https://statisticalatlas.com/neighborhood/New-York/New-York/Harlem/Race-and-Ethnicity. (Accessed: 16 December 2022)