Reading 2: Geospatial Visualization for Transportation Trends using R

Antonio Páez
My Name

12 September, 2021

Introduction

NOTE: This is an R Markdown Notebook. When you execute code within the notebook, the results appear beneath the code.

Transportation is a spatial phenomenon. It is natural then that we would be interested in studying it from a spatial perspective. In recent years, this has become increasingly possible and easy, thanks to 1) the availability of information that is geo-coded, in other words, that has geographical references; and 2) the availability of software to analyze such information.

A key technology fueling this trend is that of Geographical Information Systems (GIS). GIS are, at their simplest, digital mapping technologies for the 21st century. In most cases, however, GIS go beyond cartographic functions to also enable and enhance our ability to analyze data to produce information.

There are many available packages for geographical information analysis. Some are very user friendly, and widely available in many institutional contexts, such as ESRI's Arc software. Others are fairly specialized, such as Caliper's TransCAD, which implements many operations of interest for transportation engineering and planning. Others have the advantage of being more flexible and/or free.

Such is the case of the `R` statistical computing language. `R` has been adopted by many in the geographical analysis community, and a number of specialized packages have been developed to support mapping and spatial data analysis functions.

The objective of this note is to provide an introduction to the use of `R` for geospatial visualization.

To use this note you will need the following:

- The following data objects (available in your course package)
  - `hamilton_taz`
  - `hamilton_taz_travel`

The data used for this tutorial were retrieved from the 2016 Transportation Tomorrow Survey TTS, the periodic travel survey of the Greater Toronto and Hamilton Area.

Preliminaries

The most basic task when doing digital mapping is working with geographical files.

There are many different formats for geographic files, depending on whether they are vectors (i.e., line-based) or raster (i.e., pixel-based), and also by different developers. A library (`rgdal`) has been developed to support reading and working with many different types of geospatial files.

Before starting, it is useful to make sure that you have a clean Environment. The following command will remove all data and values currently in the Environment.

```
rm(list = ls())
```

Next, you must load the libraries needed for analysis (you may need to install them before they are available for loading):

```
library(envsocty3LT3) # Course package
library(sf)
library(tidyverse)
```

We now proceed to see how a geo-spatial file can be read.

For this example, you will use a simple features object that contains the Traffic Analysis Zones for the Hamilton Census Metropolitan Area (CMA) (retrieved from here). This data object is available as part of your course package. To call it you use the function `data()`:

```
data("hamilton_taz")
```

This assignment will create an `sf` table, an `R` object used to store geographical information (in this example spatial polygons; other types of spatial data are points, lines). Objects of the class `sf` use the Simple Features standard for representing geometry.

You can verify that the object is indeed a simple features table:

```
class(hamilton_taz)
```

```
## [1] "sf"         "data.frame"
```

Visualizing a simple features object using `ggplot2`

Traffic Analysis Zones (or TAZ) are a form of zoning system typically used for the analysis of transportation systems. TAZs are typically designed to contain a relatively small number of households, hopefully with relatively homogenous socio-economic and/or demographic characteristics. Currently, the table includes just some identifiers and the geometry of the zones. You can verify this by means of the `head` function:

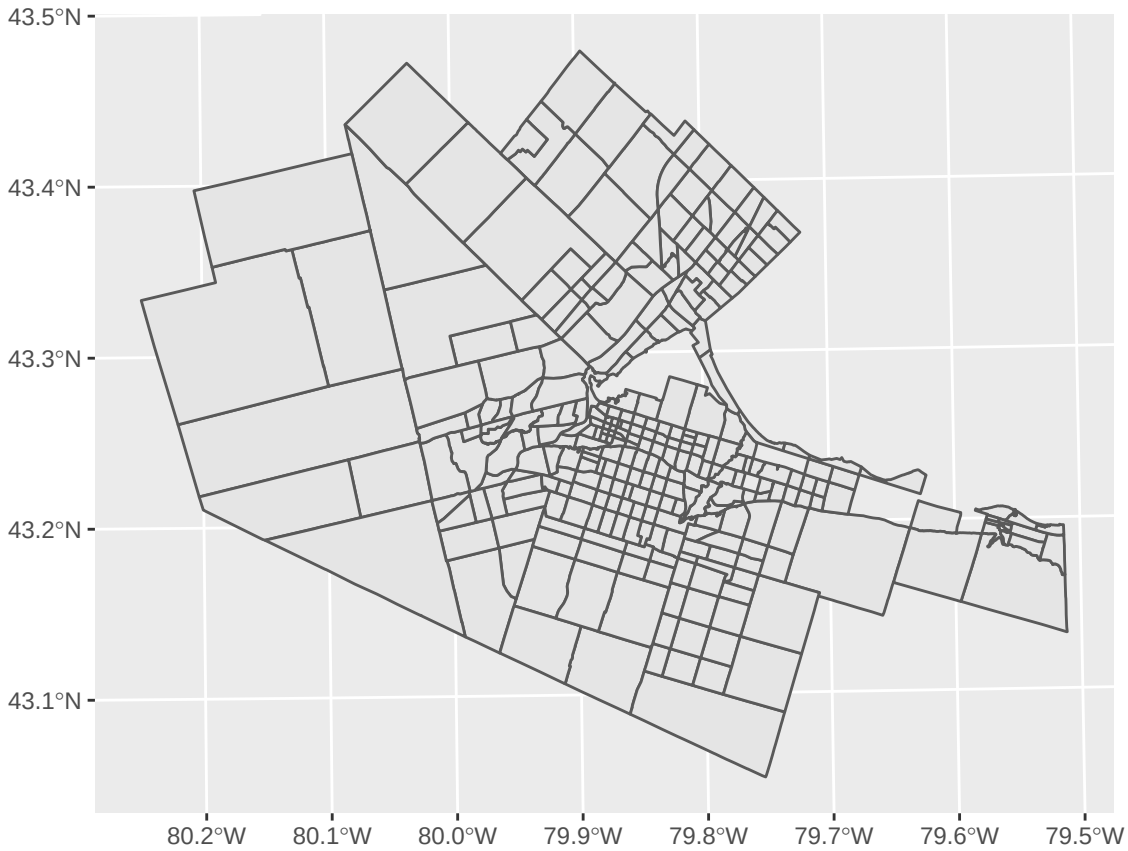
```
head(hamilton_taz)
```

```
## Simple feature collection with 6 features and 3 fields
## Geometry type: MULTIPOLYGON
## Dimension:     XY
## Bounding box:  xmin: 563306.2 ymin: 4777681 xmax: 610844.5 ymax: 4793682
## Projected CRS: WGS 84 / UTM zone 17N
##     ID       Area GTA06                       geometry
## 1 2671 37.7584573  5030 MULTIPOLYGON (((605123.4 47...
## 2 2716  3.1074885  5077 MULTIPOLYGON (((606814 4784...
## 3 2710  0.7508156  5071 MULTIPOLYGON (((605293 4785...
## 4 2745  1.8530103  5108 MULTIPOLYGON (((607542.7 47...
## 5 2810 73.0695950  5177 MULTIPOLYGON (((564681.8 47...
## 6 2740 18.5356169  5103 MULTIPOLYGON (((574373.4 47...
```

The full documentation of the data object is available; run `?hamilton_taz` in your console after loading the course package.

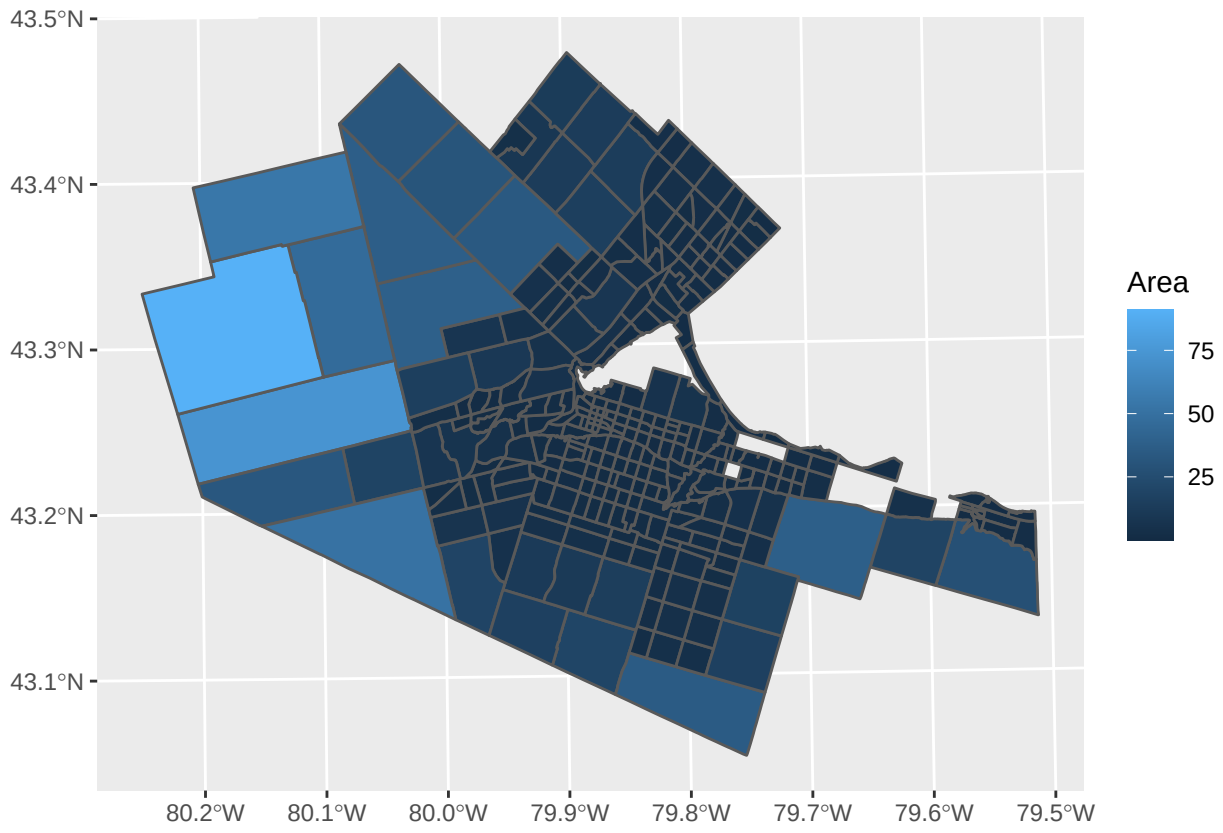
You can quickly plot this object using `ggplot2`. :

```
ggplot() + # Create a blank ggplot object
  geom_sf(data = hamilton_taz) # Overlay a simple features object
```



The command `geom_sf()`, by default, will plot the geometry of the simple features with some default values; for instance the color of the lines is black, and the fill color of the polygons is grey. We can use other variables avaible in the table to modify other features of the map. For example, we can choose the `Area` column and use it for the fill color of the polygons, by means of the `aes` (for aesthetic) value `fill`:

```
ggplot() + # Create a blank ggplot objects
  geom_sf(data = hamilton_taz, # Overlay a simple features object
          aes(fill = Area)) # Use the variable `Area` for the `fill` color of the polygons
```



As you saw above, the `hamilton_taz sf` dataframe contains some identifiers, including "GTA06", which is the unique identifier for the traffic analysis zones. There is not much information in this table, and the table therefore is not particularly interesting. If more data are available, they can be *joined* to the table for further analysis, as you will see next.

Before proceeding, it is important to note there are many packages in `R` that can be used to visualize geospatial information, each with strengths and weaknesses. Such packages are:

- `ggplot2`: used for 2 dimensional plots; you used it before in your readings 0 and 1.
- `tmap`: used to create thematic maps.
- `ggmap`: a package for static maps that works on similar principles as `ggplot2`.
- `leaflet`: used to create dynamic maps.

Joining data to an `sf` object

The geographical file, and map, that we produced above are not terribly interesting, since they contain little more than the geometry of the zones. Fortunately, there is additional information in a separate table that can help us make this map more interesting. The following data object contains information about travel and trips in the Hamilton CMA:

```
data("hamilton_taz_travel")
```

The above loads a data frame called `hamilton_taz_travel`. This data frame includes information drawn from the 2016 Transportation Tomorrow Survey (TTS). The full documentation of this object can be seen by running `?hamilton_taz_travel` in your console. You can quickly inspect the table:

```
summary(hamilton_taz_travel)
```

```
##     GTA06           work_trips_produced work_trips_attracted
##  Length:297         Min.   :   0        Min.   :   0.0
##  Class :character   1st Qu.:   0        1st Qu.:  83.0
##  Mode  :character   Median : 262        Median : 254.0
##                     Mean   : 466        Mean   : 473.9
##                     3rd Qu.: 762        3rd Qu.: 562.0
##                     Max.   :2416        Max.   :7085.0
##  shop_trips_produced shop_trips_attracted      Walk            Cycle
##  Min.   :   0.0      Min.   :   0.0       Min.   :   0.0   Min.   :  0.00
##  1st Qu.:   0.0      1st Qu.:   0.0       1st Qu.:   0.0   1st Qu.:  0.00
##  Median : 174.0      Median :  39.0       Median :   0.0   Median :  0.00
##  Mean   : 319.3      Mean   : 324.6       Mean   : 202.1   Mean   : 46.98
##  3rd Qu.: 521.0      3rd Qu.: 205.0       3rd Qu.: 255.0   3rd Qu.: 27.00
##  Max.   :1892.0      Max.   :8008.0       Max.   :2293.0   Max.   :771.00
##   Auto_driver    Auto_passenger     Motorcycle      Taxi_passenger
##  Min.   :    0   Min.   :   0.0   Min.   :  0.000   Min.   :  0.000
##  1st Qu.:    0   1st Qu.:   0.0   1st Qu.:  0.000   1st Qu.:  0.000
##  Median : 1483   Median : 237.0   Median :  0.000   Median :  0.000
##  Mean   : 2458   Mean   : 511.1   Mean   :  4.401   Mean   :  8.505
##  3rd Qu.: 4306   3rd Qu.: 802.0   3rd Qu.:  0.000   3rd Qu.:  0.000
##  Max.   :11297   Max.   :2915.0   Max.   :402.000   Max.   :180.000
##  Paid_rideshare      School_bus      Transit_excluding_GO_rail
##  Min.   :  0.000   Min.   :   0.00   Min.   :   0
##  1st Qu.:  0.000   1st Qu.:   0.00   1st Qu.:   0
##  Median :  0.000   Median :   0.00   Median :   0
##  Mean   :  5.424   Mean   :  90.38   Mean   : 245
##  3rd Qu.:  0.000   3rd Qu.: 115.00   3rd Qu.: 270
##  Max.   :333.000   Max.   :1483.00   Max.   :3425
##  GO_rail_and_transit  GO_rail_only        Other
##  Min.   :  0.00       Min.   :  0.00   Min.   :  0.000
##  1st Qu.:  0.00       1st Qu.:  0.00   1st Qu.:  0.000
##  Median :  0.00       Median :  0.00   Median :  0.000
##  Mean   : 10.45       Mean   : 15.52   Mean   :  5.471
##  3rd Qu.:  0.00       3rd Qu.:  0.00   3rd Qu.:  0.000
##  Max.   :181.00       Max.   :289.00   Max.   :303.000
```

The table has a column with variable called `GTA06`. This variable is identical to the one in the `sf` object `hamilton_taz`, which means that there is a one-to-one correspondence between rows in `hamilton_taz` and rows in `hamilton_taz_travel`. In addition, there are columns with information about trips *produced* (that begin at a zone) and *attracted* (that end at a zone) for two different purposes, work and shopping. The rest of the columns include the number of trips produced by zones for all purposes, but by different modes of transportation (Walk, Cycle, etc.)

Another useful table includes information about persons and households by traffic analysis zone. You can call this table by running the following chunk:

```
data("hamilton_taz_ph")
```

This table also includes the same `GTA06` column, and information about the population and households in the traffic analysis zones (e.g., number of people, number of full-time workers, number of vehicles, etc.) Run a summary of this table to become familiar with its contents.

Information about trips and travel, as well as that about people and households, can be joined to the traffic analysis zones for geospatial visualization. To join two tables, the set of `join_` commands from the

`dplyr` package are useful. In this case, we want to execute a *left_join*. A *left_join* takes all the rows from table x (to the left), and adds to that all the columns of table y for those rows. You can join the data frame `travel_data` to `taz` as follows:

```
hamilton_taz <- left_join(hamilton_taz, # Left table
                          hamilton_taz_travel, # Right table
                          by = "GTA06") # Key for joining the two tables
```

As you can see, *left_join* looks for a matches in the key variable that exists in both files, so that it can join the data of matching rows. In this case, we use the variable `GTA06` as the key.

We can similarly join the person and household information:

```
hamilton_taz <- left_join(hamilton_taz, # Left table
                          hamilton_taz_ph, # Right table
                          by = "GTA06") # Key for joining the two tables
```

Our table `hamilton_taz` now is information-rich! It still includes the geometry, but also a fair amount of information about the characteristics of traffic analysis zones.

Creating thematic maps

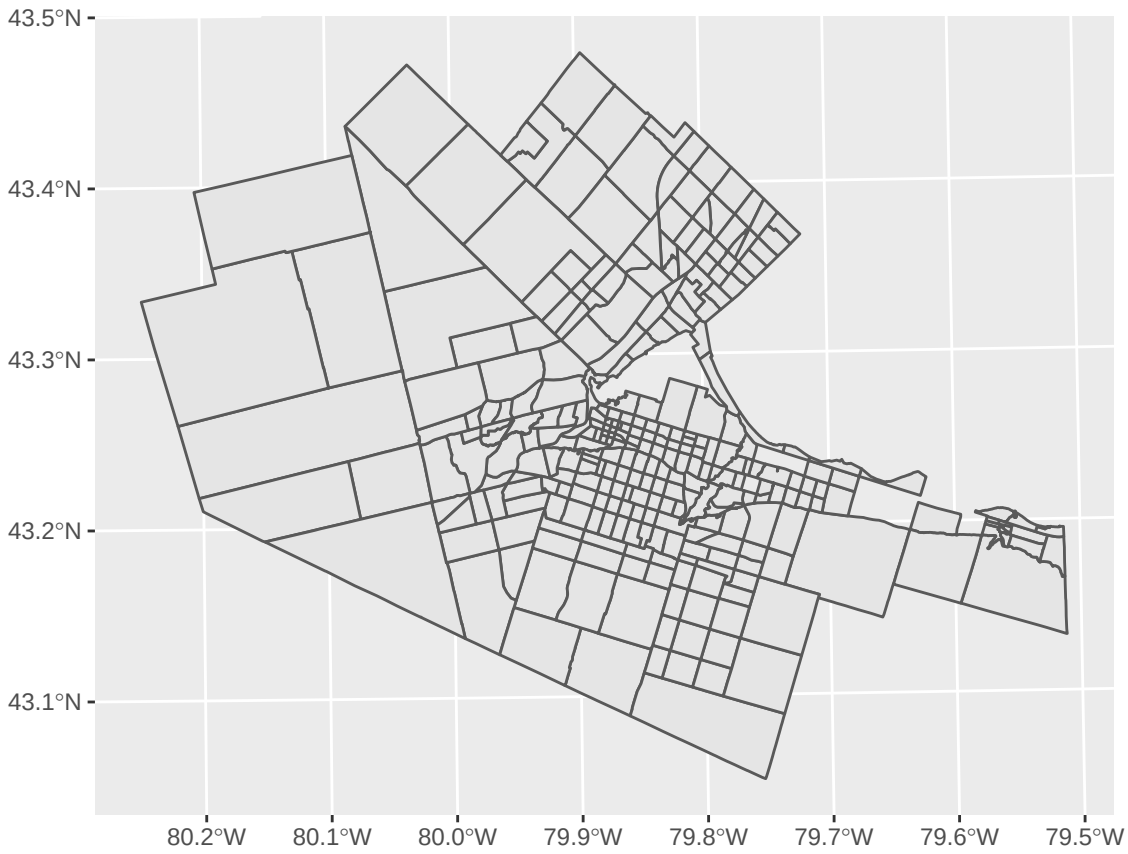
Once a `sf` dataframe has interesting information, it is possible to create thematic maps.

As you probably intuited by now, When working with the package `ggplot2` objects are created by layering. We begin by creating an empty `ggplot2` object as follows:

```
ggplot()
```

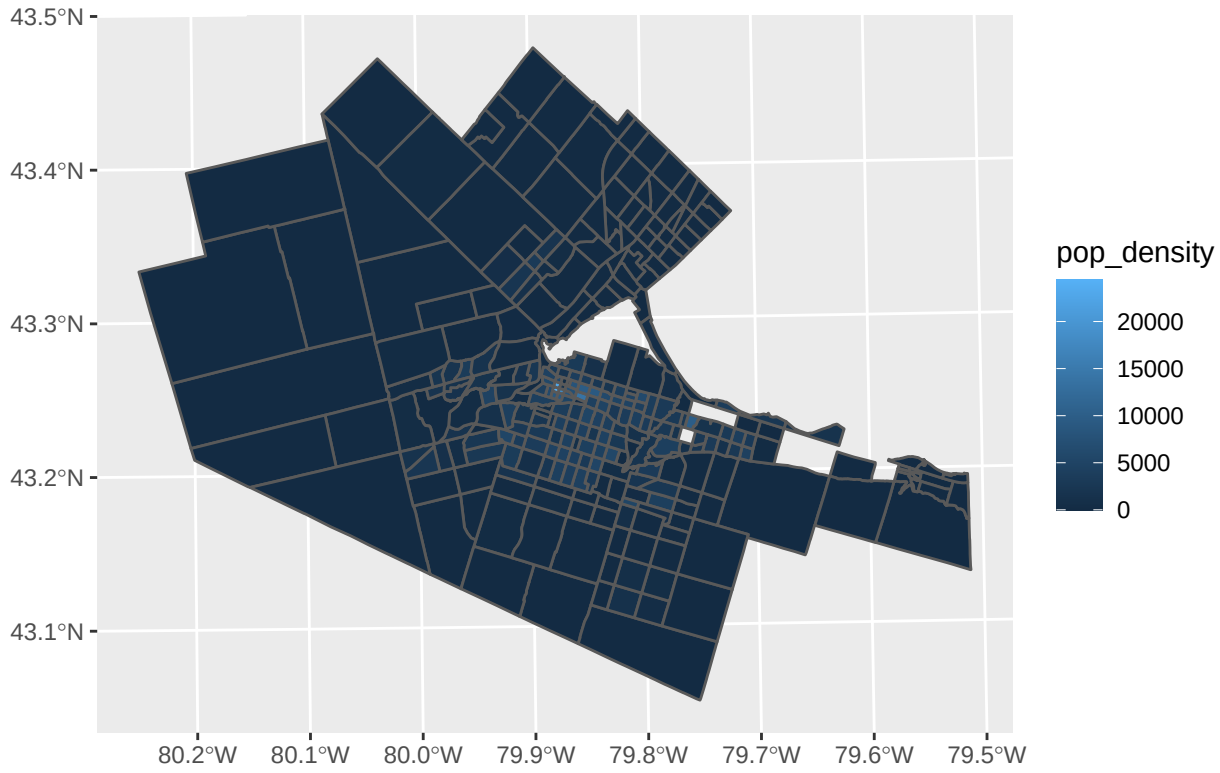
It is blank because we have not yet told the package to actually draw anything. The function for rendering `sf` objects is `geom_sf()`. Again, we can plot the geometry only as follows:

```
ggplot() +
  geom_sf(data = hamilton_taz)
```



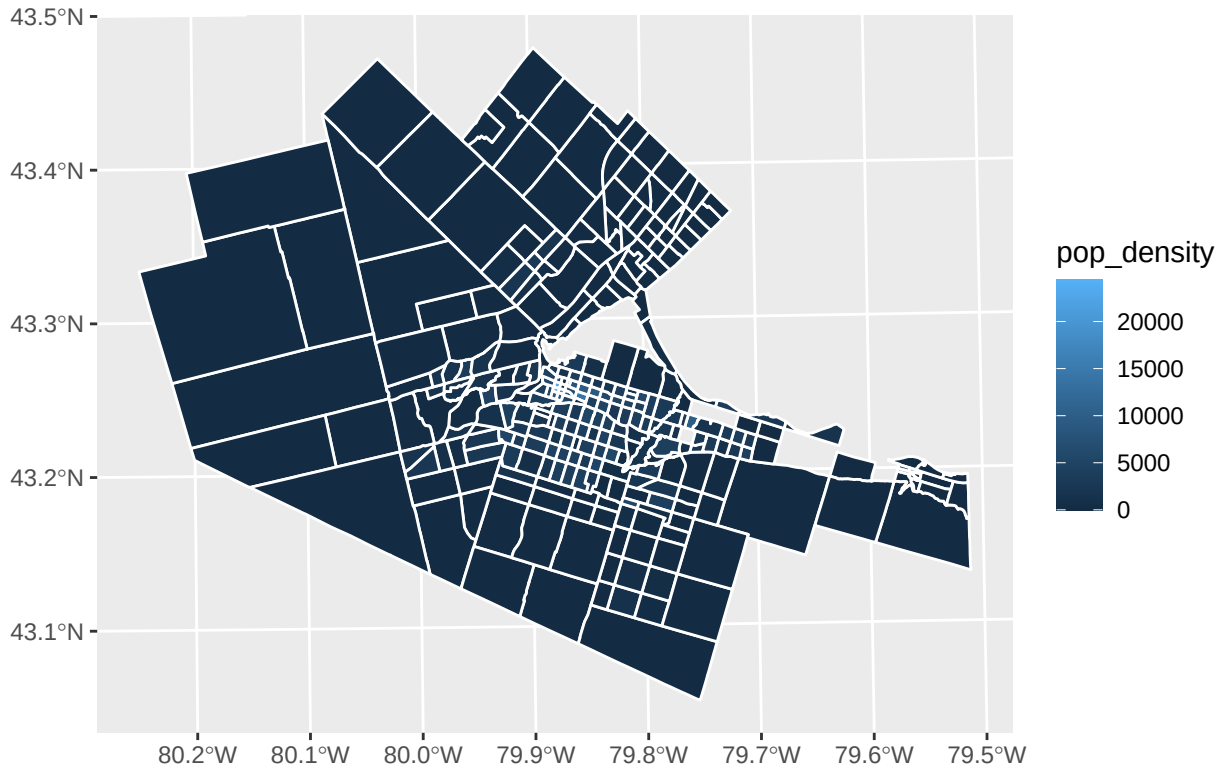
The default settings use a gray style for the polygons and add a frame to the map. Thematic maps are created by mapping some aesthetic value of the plot to a variable. For instance, we can map the aesthetic `fill` to `population_density`:

```
ggplot() +
  geom_sf(data = hamilton_taz,
          aes(fill = pop_density))
```



We can also change other aesthetics without mapping them to a variable (so we put them outside of `aes()`). For example, we can change the color of the lines to "white":

```
ggplot() +
  geom_sf(data = hamilton_taz,
          aes(fill = pop_density),
          color = "white")
```

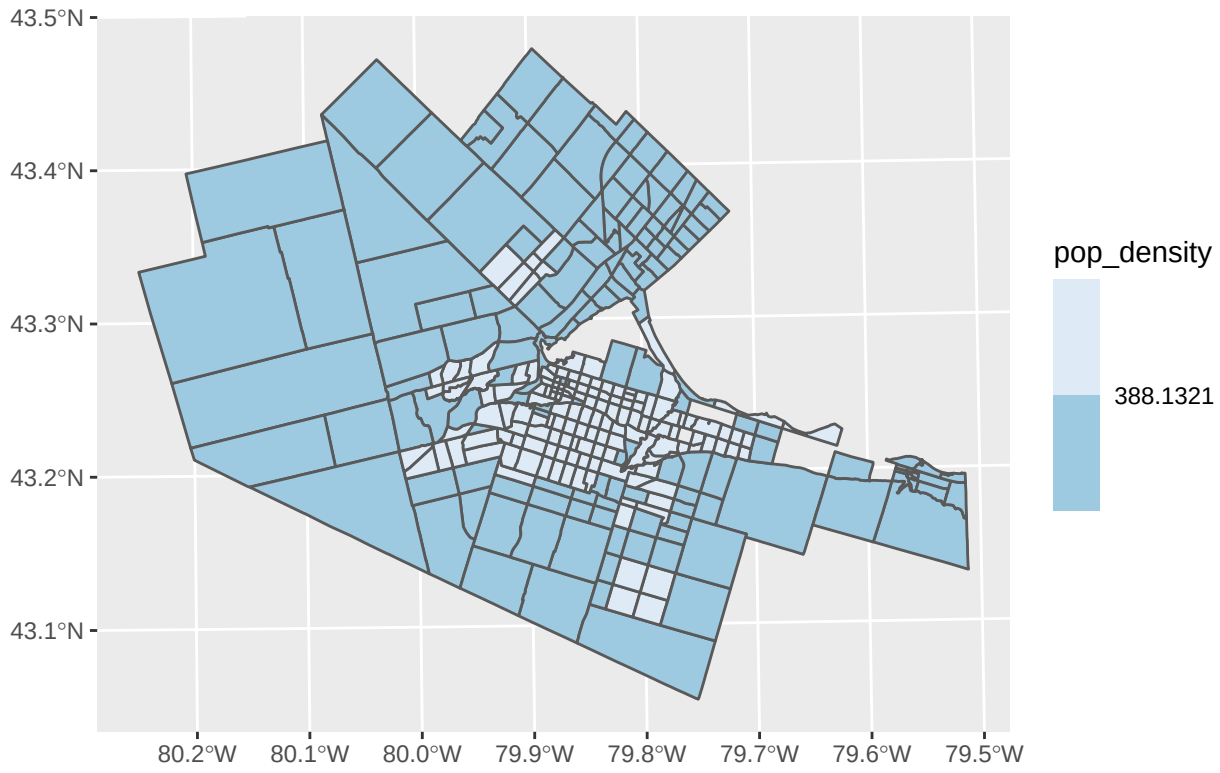


When we use a variable to fill the polygons, like the above population density maps, we are creating a choropleth map (a map where the colors are selected based on the values of an underlying variable). This creates a *statistical map*. The map also includes a legend that can be used to interpret the coloring of the map.

The default coloring scheme was for a continuous variable. As you can see above, the presence of a few zones with a large number of trips obscures somewhat the distribution of values elsewhwere. This reduces the usefulness of the map, and you might want to change this by using a different coloring scheme. For instance, using the `scale_fill_fermenter()` function it is possible to color the choropleth map using the quantiles of the variable. The simplest quantile is when we divide the sample in two. For this we would use the `quantile()` function and indicate that the cuts are done at *breaks* 0, 0.5, and 1, or in other words at 0%, 50%, and 100% of the sample:

```
# Calculate the quantiles of the distribution
pop_density_q <- quantile(hamilton_taz$pop_density, # Obtain the quantiles of the distribution of
                          c(0, 0.5, 1), # Breaks at 0, 50%, and 100%
                          names = FALSE) # Do not keep the names of the quantiles, only the values at w

ggplot() + # Create blank `ggplot2` object
  geom_sf(data = hamilton_taz, # Plot the simple features in `hamilton_taz`
          aes(fill = pop_density)) + # Map the population density to the fill color of the polygons
  scale_fill_fermenter(breaks = pop_density_q) # Change the scale for the fill using the breaks in the
```

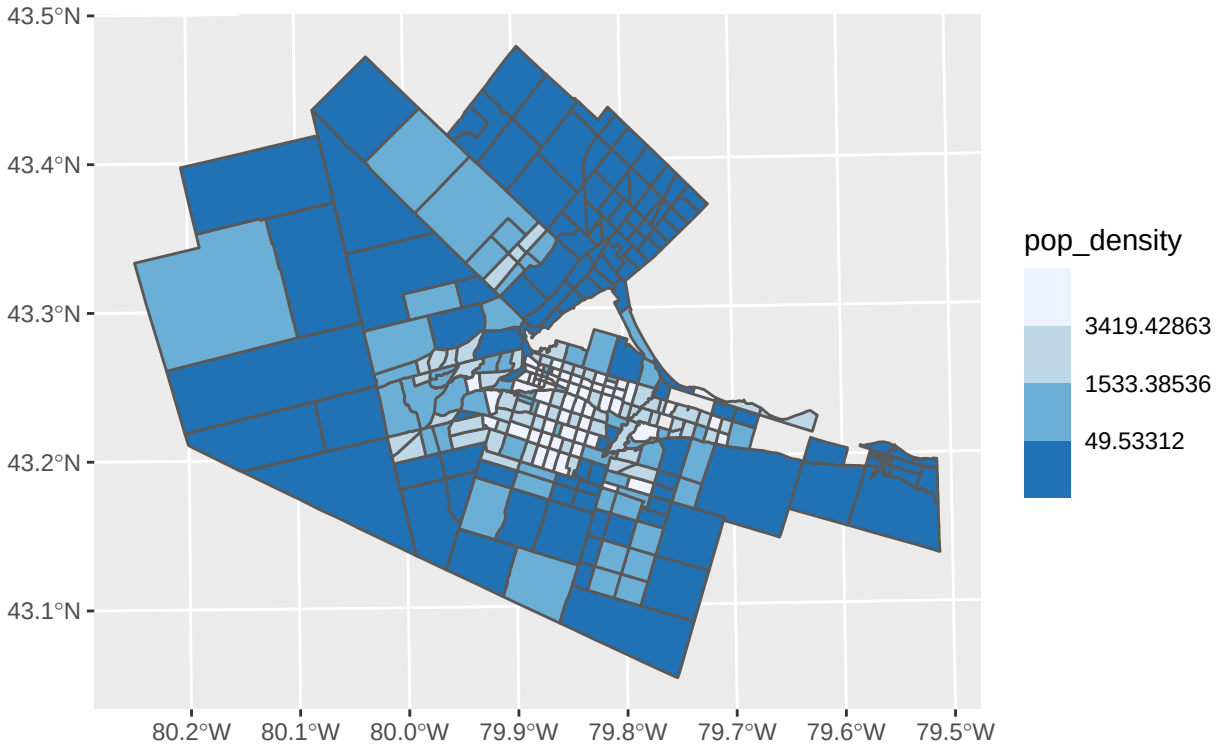


This thematic map shows that half of all traffic analysis zones have a population density lower than 388.13 persons per sq.km, and half higher than that. High population density is generally seen in the urban parts of the Hamilton CMA, and low in the suburban and rural parts of the CMA.

Of course, we could use a finer set of breaks, say at 0%, 20%, 40%, 60%, 80%, 100% (these are called *quintiles*, they divide the sample in five equal parts):

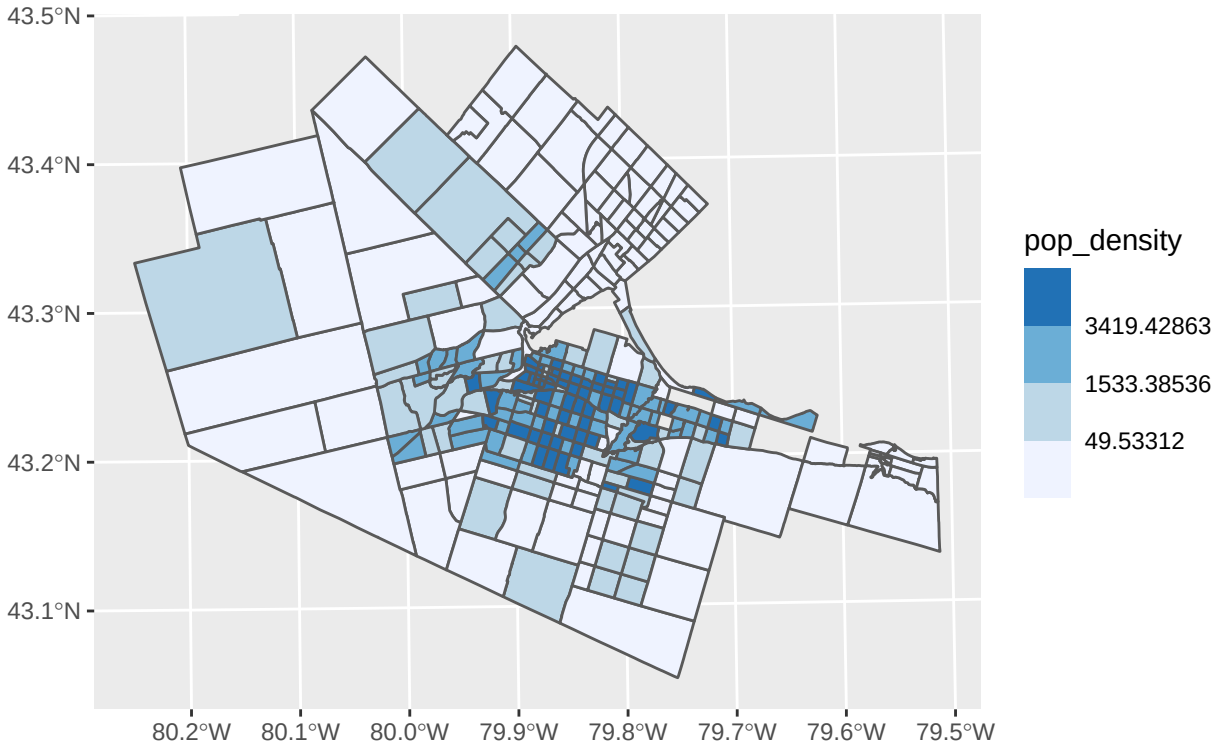
```
# Calculate the quantiles of the distribution
pop_density_q <- quantile(hamilton_taz$pop_density, # Obtain the quantiles of the distribution of
                          c(0, 0.2, 0.4, 0.6, 0.8, 1), # Breaks at 0, 20%, 40%, 60%, 80%, and 100%
                          names = FALSE) # Do not keep the names of the quantiles, only the values at w

ggplot() + # Create blank `ggplot2` object
  geom_sf(data = hamilton_taz, # Plot the simple features in `hamilton_taz`
          aes(fill = pop_density)) + # Map the population density to the fill color of the polygons
  scale_fill_fermenter(breaks = pop_density_q) # Change the scale for the fill using the breaks in the
```



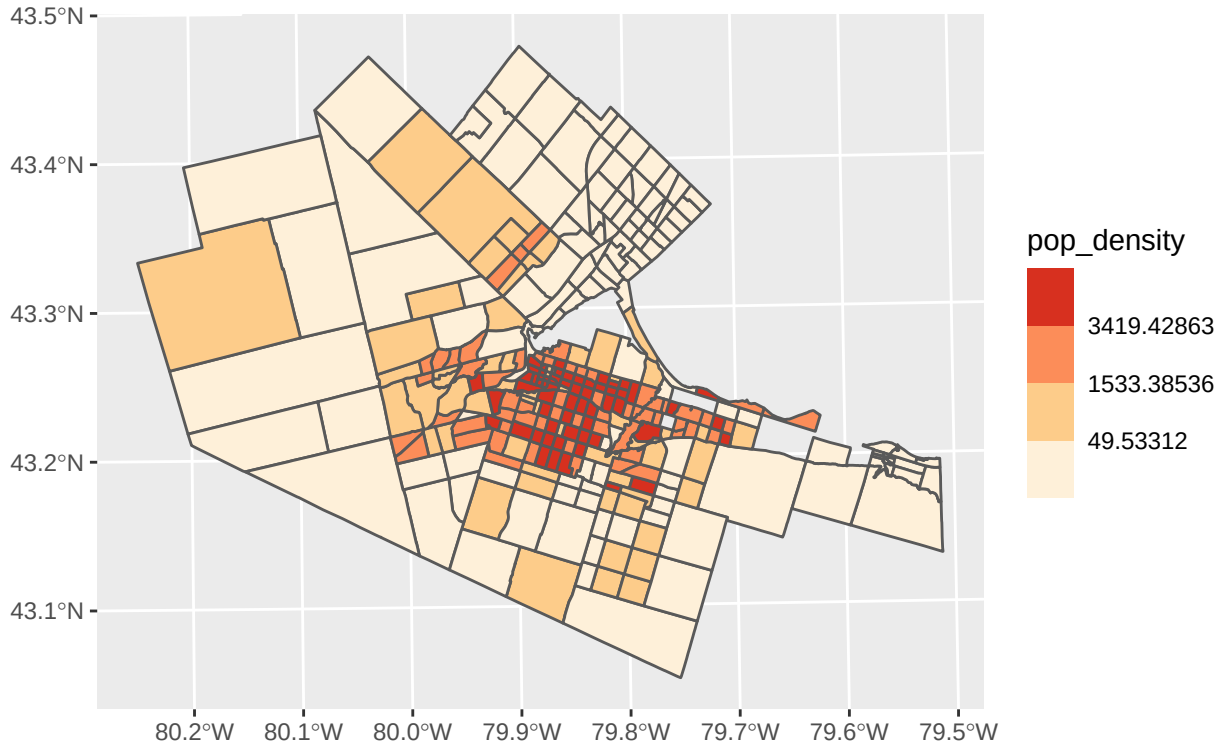
We can improve the map above in many ways, but for the moment, it is useful to change the coloring scheme, so that higher values of population density are colored darker. We do this by changing the `direction` of the fill scale:

```
ggplot() + # Create blank `ggplot2` object
  geom_sf(data = hamilton_taz, # Plot the simple features in `hamilton_taz`
          aes(fill = pop_density)) + # Map the population density to the fill color of the polygons
  scale_fill_fermenter(direction = 1, # Change the direction of the fill scale
                       breaks = pop_density_q) # Change the scale for the fill using the breaks in the
```



We can also change the *color palette*, for instance to oranges and reds (the names of some available palettes are here):

```
ggplot() + # Create blank `ggplot2` object
  geom_sf(data = hamilton_taz, # Plot the simple features in `hamilton_taz`
          aes(fill = pop_density)) + # Map the population density to the fill color of the polygons
  scale_fill_fermenter(palette = "OrRd",
                       direction = 1,
                       breaks = pop_density_q) # Change the scale for the fill using the breaks in the
```



The maps with quantiles are easier to interpret from a statistical standpoint, because they show the places where, say, the traffic analysis zones in the bottom 20% of population density are.

Working with data in tables

The thematic maps above show how maps can help us understand the spatial distribution of variables that are relevant for transportation phenomena.

It is possible to do more sophisticated analysis by generating information based on the variables available. For instance, suppose that we would like to calculate a per capita measure of mobility, say trips per capita, for a specific purpose. This can be done using the number of work trips and the number of full time workers, and the function `mutate`:

```
hamilton_taz <- mutate(hamilton_taz, # Mutate the `hamilton_taz` table and store the results in `hamilt
                       work.pc = work_trips_produced / ft_workers) # The mutation will add a new column
```

More complex formulas can be used too. For instance, for each zone we can find the proportion of trips by mode relative to the total number of trips by all modes produced by each zone:

```
hamilton_taz <- mutate(hamilton_taz,
                       Auto_driver.prop = Auto_driver/(Walk + Cycle + Auto_driver + Auto_passenger + Mo
```

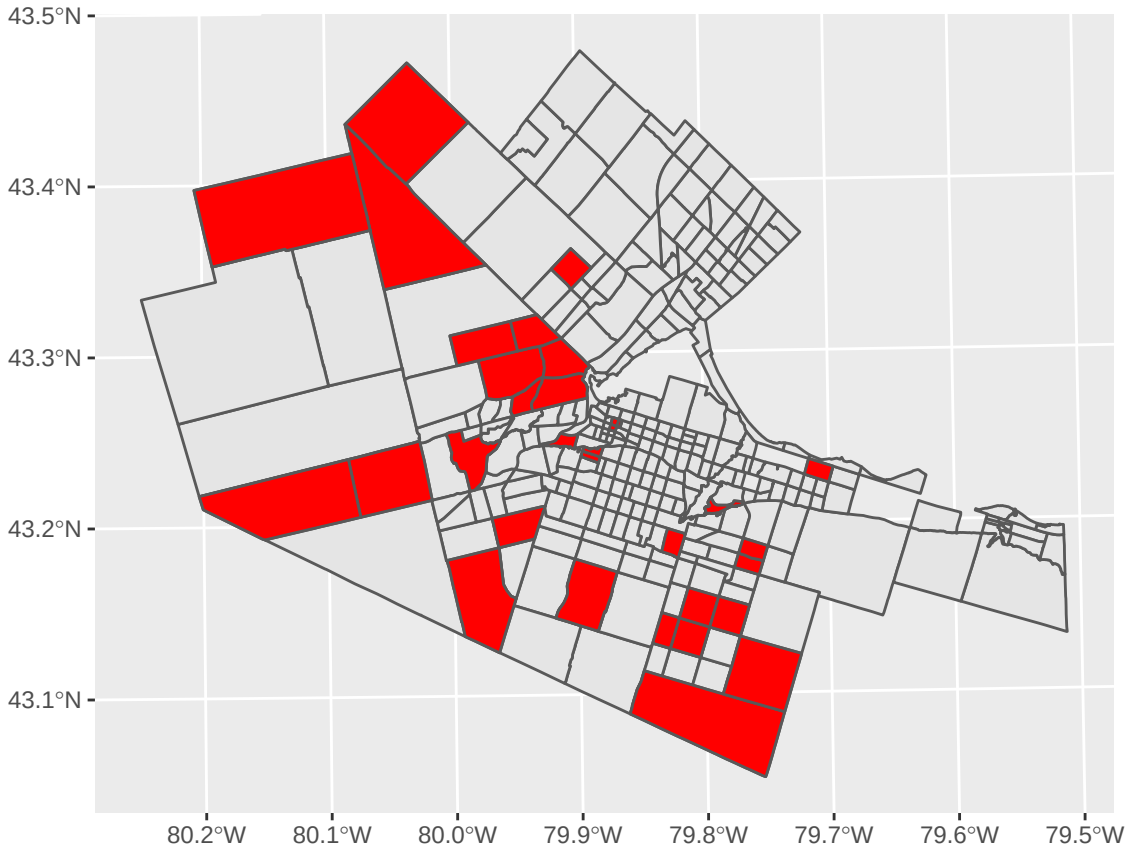
And we can query this table. As an example, we can find how many zones had a large proportion of all their trips made by a person driving an auto. We can do this by using the `filter()` function from package `dplyer` (in `tidyverse`). The function extracts the rows in the table that meet a logical condition (i.e., `Auto_driver.prop >= 0.85`. Let us save the result of this operation in a separate table that we will call `auto_dependent`:

```
auto_dependent <- filter(hamilton_taz, # Filter `hamilton_taz`
                         Auto_driver.prop >= 0.85) # Logical condition for filtering
```

How many zones are there with a very high proportion of trips made by auto drivers?

Another interesting question is, where are those traffic analysis zones that are probably auto-dependent (i.e., more than 85% of all trips made by person driving a car)? We can map this:

```
ggplot() + # Create a blank `ggplot2` object
  geom_sf(data = hamilton_taz) + # Overlay the traffic analysis zones
  geom_sf(data = auto_dependent, # Overlay the auto-dependent traffic analysis zones
          fill = "red") # Use red to fill the polygons of the auto-dependent zones
```



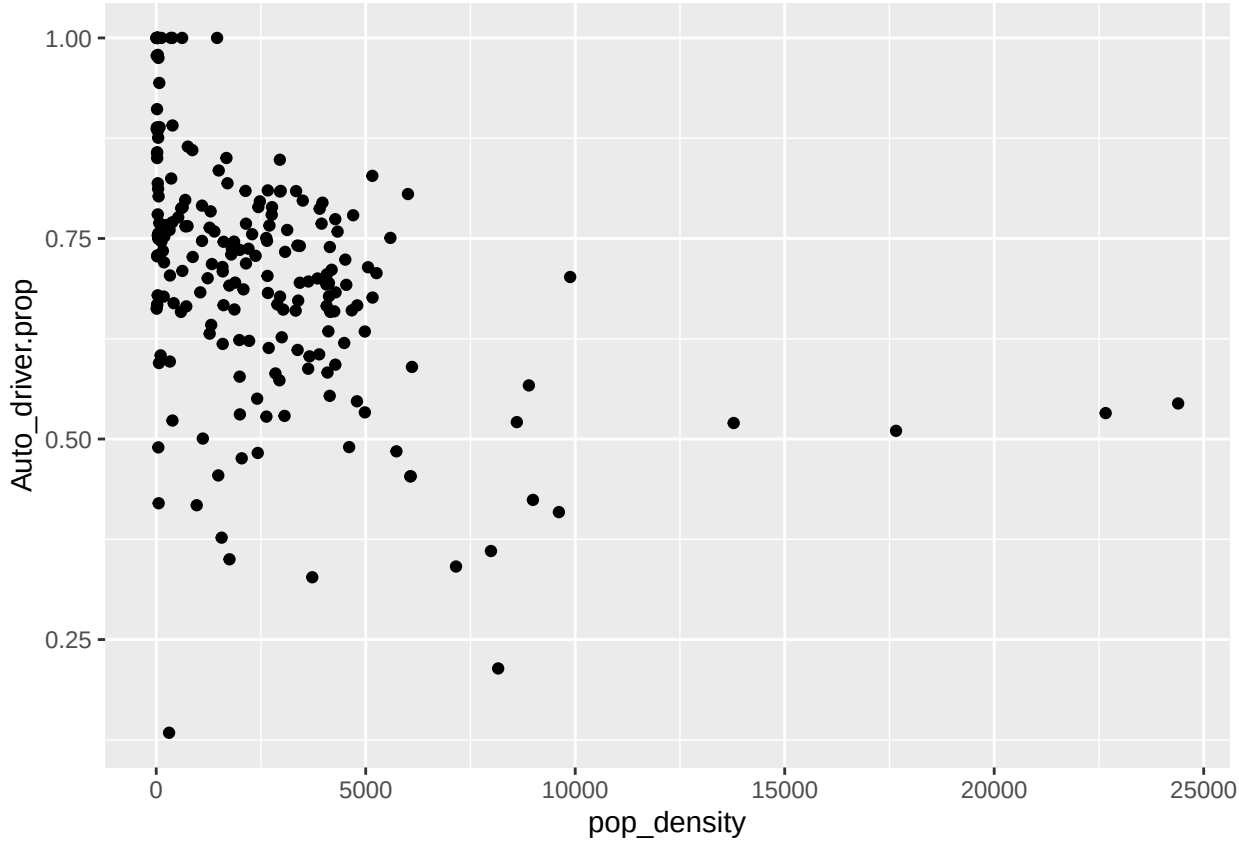
Do you find this pattern surprising? What do you think could explain it?

More on regressions

You can use a chart to see the distribution of a variable across different categories. For example, here is how to plot a density chart, with multiple lines for multiple characteristics:

```
# make a density plot showing the difference in income distribution
# for different unemployment rates
ggplot(hamilton_taz, # Create a `ggplot2` object using data in the table `hamilton_taz`
       aes(x = pop_density, # Map the population density of traffic analysis zones to the x-axis
           y = Auto_driver.prop)) + # Map the proportion of trips by auto driver to the y-axis
  geom_point() # Add points to the plot
```

```
## Warning: Removed 89 rows containing missing values (geom_point).
```



The above chart is a scatterplot, a visualization typically used to explore whether there is a statistical relationship between two variables. In this case, it plots the population density for every TAZ in the dataset on the x-axis, and the proportion of trips made by auto driver on the y-axis. The result is a chart that shows how traffic analysis zones with low population density tend to have more trips made by auto drivers.

When you see a chart like this, it suggests a relationship between population density and travel by car. You could study this relationship by meand of regression analysis:

```
model1 <- lm(formula = Auto_driver.prop ~ pop_density,
             data = hamilton_taz)
summary(model1)
```

```
##
## Call:
## lm(formula = Auto_driver.prop ~ pop_density, data = hamilton_taz)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.62260 -0.08159  0.00855  0.08858  0.28557
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  7.628e-01  1.251e-02  60.988  < 2e-16 ***
## pop_density -2.066e-05  3.010e-06  -6.866 7.64e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 0.141 on 206 degrees of freedom
##   (89 observations deleted due to missingness)
## Multiple R-squared:  0.1862, Adjusted R-squared:  0.1823
## F-statistic: 47.14 on 1 and 206 DF,  p-value: 7.636e-11
```

The model suggests that there is a negative relationship between `pop_density` and `Auto_driver.prop` (see the negative coefficient of `pop_density`). Furthermore, the p-value of the coefficient is very small, indicating that this relationship is unlikely to be random. The coefficient is interpreted as the rate of change: for every increase in population density of 1 person per sq.km, the proportion of trips by auto driver goes down *on average* by 0.00206%.

The R-squared of 0.18 indicates that there is still much variation left unexplained. The R-squared coefficient multiplied by 100 is interpreted as the percentage of the variance explained by the model: in this case approximately 18.6%, which means the model does *not* explain 71.4% of the variance.

We can do a regression of a dependent variable on two or more independent variables, not just on one. For example, we may be curious about the relationship between travel by auto and the age of the population. The following regression adds the median age of the population in the traffic analysis zones to the regression:

```
model2 <- lm(formula = Auto_driver.prop ~ pop_density + median_age,
             data = hamilton_taz)
summary(model2)
```

```
##
## Call:
## lm(formula = Auto_driver.prop ~ pop_density + median_age, data = hamilton_taz)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.59120 -0.07823  0.01685  0.08522  0.30034
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  5.265e-01  4.283e-02  12.295  < 2e-16 ***
## pop_density -1.622e-05  2.906e-06  -5.582 7.48e-08 ***
## median_age   5.087e-03  8.875e-04   5.732 3.51e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1312 on 205 degrees of freedom
##   (89 observations deleted due to missingness)
## Multiple R-squared:  0.2986, Adjusted R-squared:  0.2918
## F-statistic: 43.64 on 2 and 205 DF,  p-value: < 2.2e-16
```

Both coefficients for population density and median age of population in the traffic analysis zone have very small p-values, which means the relationships are unlikely random. The coefficient for `pop_density` is negative but the effect is a little smaller after introducing `median_age`. The coefficient for `median_age` is positive, which suggests that as the population trends older, the proportion of trips by auto driver tends to increase. In fact, since the coefficient is 0.00508, this means that as the median age of the population goes up by one year, the proportion of trips by auto driver goes up by 0.508%.

This illustrates how you can do a regression on multiple variables, not just on one.

Bonus Material

After completing this reading you are equipped to do some basic thematic mapping using `R`.

The advantage of using a programmatic approach to visualization (i.e., programming your maps) is that everything is documented, it makes it easy to reproduce the maps, and it also gives you very fine control over the aspect of your maps. An advanced example is below (do not worry about understanding the code below, it is just to illustrate the possibilities):

```
ggplot() +
  geom_sf(data = hamilton_taz,
          aes(fill = Walk)) +
  scale_fill_fermenter(type = "seq",
                       direction = 1) +
  theme_minimal() +
  labs(fill = "Cycle Trips (Quintiles)") +
  theme(legend.position = "bottom",
        legend.text = element_text(size = 7))
```