Geospatial Data Visualization in R: Mapping New Zealand with Leaflet and sf

Anju Sambasivan

Introduction

This project focuses on using maps to visualize and understand spatial data. We explore **Geographical Information Systems (GIS)** in R with the help of the `leaflet` and `sf` packages. The goal is to create different types of maps, like basic maps, heatmaps, and choropleth maps, to show population data from New Zealand.

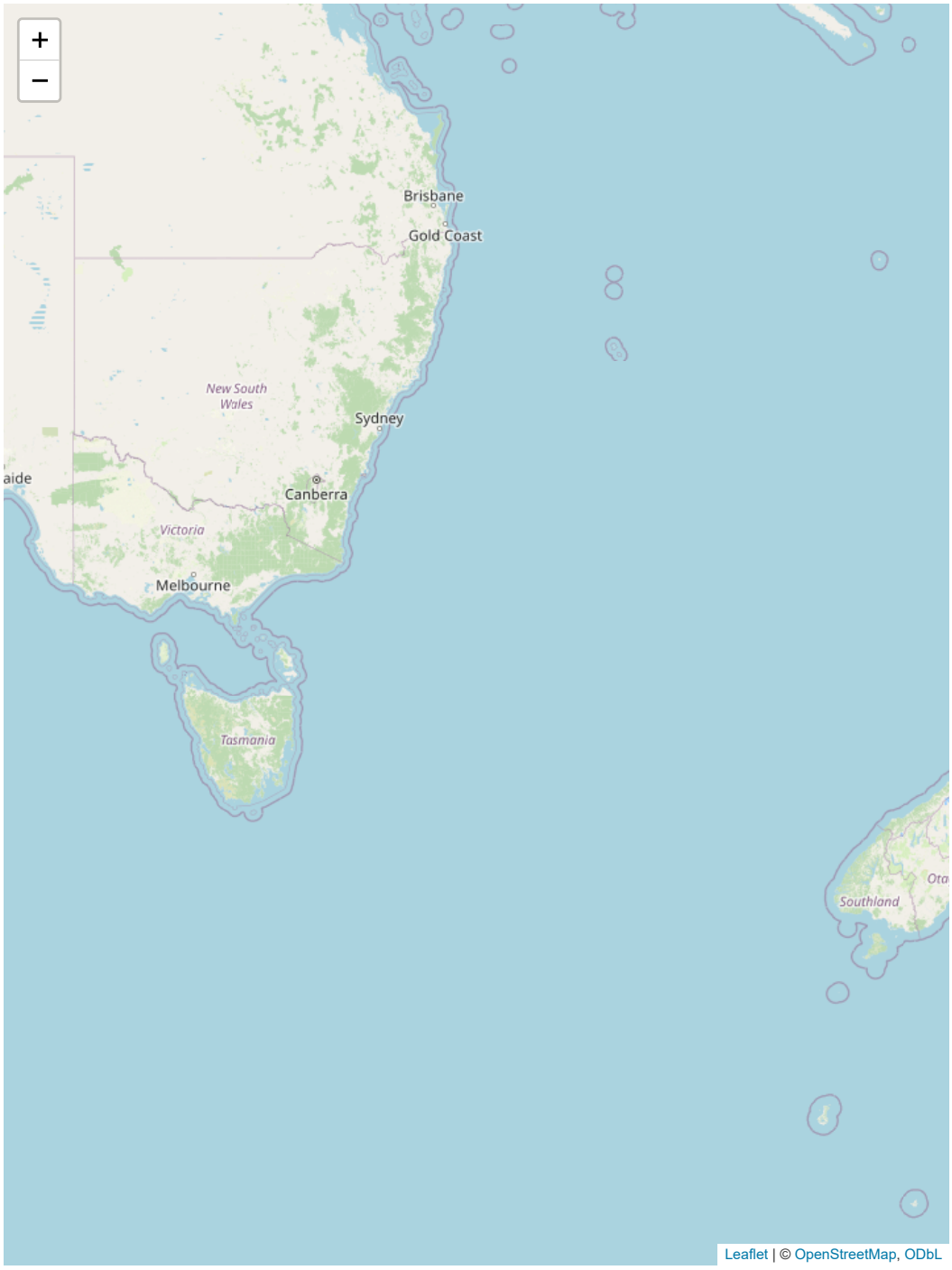
By the end of this project, we combine these techniques to create a customized map that uses multiple layers and styles. This report explains the steps we followed, the tools we used, and how these maps can help analyze and present spatial data in a clear and interactive way.

```
library(leaflet)
library(leaflet.extras)
library(sf)
library(ggplot2)
library(plotly)
library(dplyr)
```

- `leaflet`: The `leaflet` library is used for creating interactive maps. It allows users to plot data on a map, customize markers, and add layers for advanced visualizations. In this project, `leaflet` was used to create the base map of New Zealand and add markers for cities.
- `leaflet.extras`: An extension of the `leaflet` library, this package provides additional features like heatmaps and more advanced tile layers. It was used to generate a heatmap for population visualization in New Zealand cities.
- **`sf` (Simple Features)**: The `sf` package is used for working with geospatial data in R. It allows reading, writing, and analyzing spatial data formats like GeoJSON and shapefiles. In this project, `sf` was used to read and manipulate New Zealand territorial boundary data from a GeoJSON file.

- **ggplot2**: The ggplot2 library is a powerful tool for creating static visualizations. It is highly customizable. In this project, ggplot2 was used to create black-and-white maps, basic choropleth maps, and advanced choropleth visualizations.
- **plotly**: plotly is a library for creating interactive plots and visualizations. It allows for converting static ggplot2 maps into interactive maps with hover functionality and zooming. In this project, plotly was used to create an interactive version of the choropleth map.
- **dplyr**: dplyr is a data manipulation package that provides functions for filtering, joining, summarizing, and reshaping data. In this project, dplyr was used to merge geospatial data with population data, ensuring the datasets were ready for visualization.

```
# Creating a Map of New Zealand
nz_map <- leaflet() %>%
  addTiles() %>%
  setView(lng = 174.8860, lat = -40.9006, zoom = 5)
nz_map
```


+
−
Brisbane
Gold Coast
New South Wales
Sydney
Canberra
Victoria
Melbourne
Tasmania
Southland
Leaflet | © OpenStreetMap, ODbL

- **`leaflet()`**: Initializes an empty map canvas for customization.
- **`addTiles()`**: Adds a base map layer from OpenStreetMap, providing the map's background with roads, borders, and terrain details.
- **`setView(lng = 174.8860, lat = -40.9006, zoom = 5)`**: Specifies the initial view of the map
    1. **Longitude (lng)**: `174.8860` — Places the map horizontally on New Zealand's location.
    2. **Latitude (lat)**: `-40.9006` — Positions the map vertically to focus on New Zealand.
    3. **Zoom Level**: `5` — Sets the map to a moderately zoomed-out view that displays the entire country.
- **`nz_map`**: Stores the created map object for display or further customization.

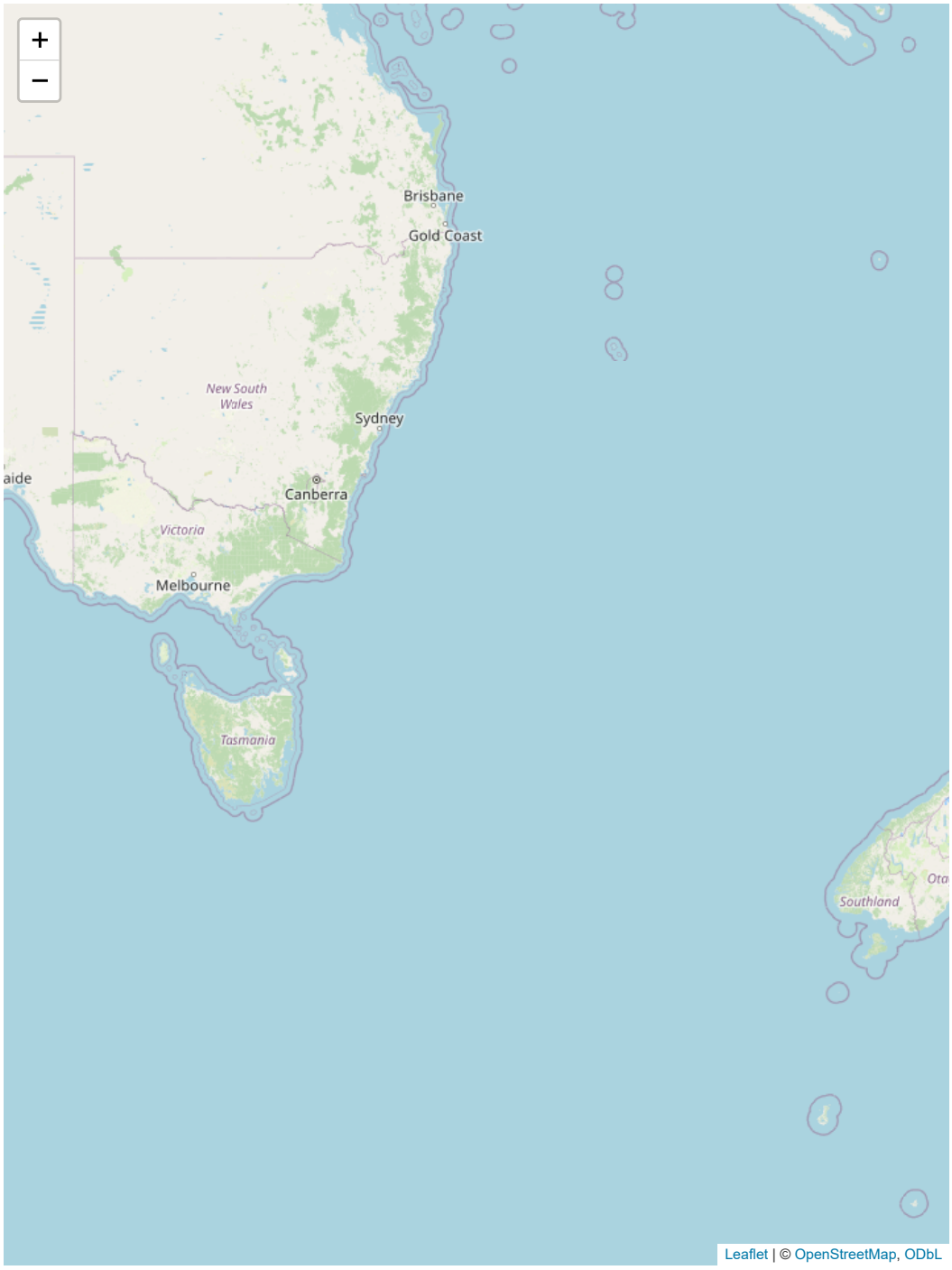
```
# Creating a Data Frame of New Zealand Cities
nz_pop <- data.frame(
  city = c("Auckland", "Wellington", "Christchurch"),
  lat = c(-36.8485, -41.2865, -43.5321),
  lng = c(174.7633, 174.7762, 172.6362),
  population = c(1657200, 216200, 383200)
)
```

This code creates a data frame named **`nz_pop`**, which contains information about three major cities in New Zealand: Auckland, Wellington, and Christchurch. The data frame includes the following columns:

- **`city`**: Lists the names of the cities: Auckland, Wellington, and Christchurch.
- **`lat` (Latitude)**: Provides the latitude coordinates for each city
- **`lng` (Longitude)**: Specifies the longitude coordinates for each city:
- **`population`**: Indicates the population of each city:

This data frame serves as the foundation for adding markers to the map of New Zealand. Each row represents a city with its geographic location and population, which can be visualized interactively.

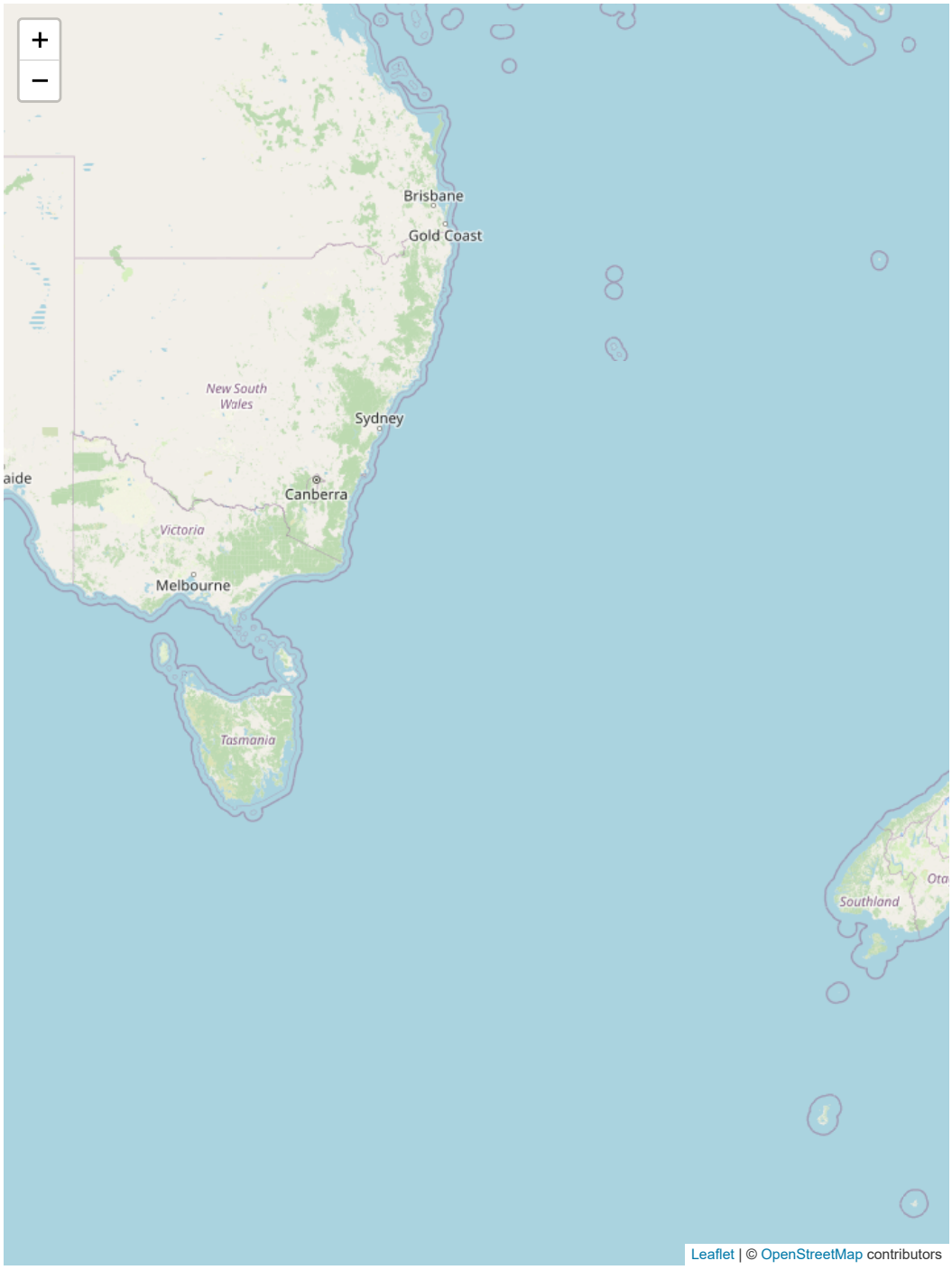
```
# Adding Markers to the Map
nz_map_pop <- leaflet(nz_pop) %>%
  addTiles() %>%
  setView(lng = 174.8860, lat = -40.9006, zoom = 5) %>%
  addMarkers(~lng, ~lat, popup = ~paste(city, "<br>Population:", population))
nz_map_pop
```


+
−
Brisbane
Gold Coast
New South Wales
Sydney
Canberra
Victoria
Melbourne
Tasmania
Southland
Leaflet | © OpenStreetMap, ODbL

This code generates an interactive map of New Zealand with markers representing Auckland, Wellington, and Christchurch. Each marker includes a popup displaying the city's name and population, enhancing the visualization with geographic and demographic information.

- **`leaflet(nz_pop)`**: Initializes the leaflet map and links it to the **`nz_pop`** data frame, which contains information about the cities (name, latitude, longitude, and population).
- **`addTiles()`**: Adds a base map layer using OpenStreetMap to provide the map's visual background.
- **`setView(lng = 174.8860, lat = -40.9006, zoom = 5)`**: Sets the initial view of the map:
    1. **Longitude (lng)**: `174.8860` positions the map horizontally to focus on New Zealand.
    2. **Latitude (lat)**: `-40.9006` centers the map vertically.
    3. **Zoom Level**: `5` provides a moderately zoomed-out view to display the entire country.
- **`addMarkers(~lng, ~lat, popup = ~paste(city, "<br>Population:", population))`**: Adds markers at the latitude (`~lat`) and longitude (`~lng`) coordinates for each city. Includes interactive popups for the markers that display the city's name and population. The `~paste()` function is used to create a formatted string combining the city name and population.
- **`nz_map_pop`**: Stores the created map object with the markers, making it ready for display or further customization.

```
# Creating a Heatmap with City Markers
nz_map_pop <- leaflet(nz_pop) %>%
  addProviderTiles(providers$OpenStreetMap) %>%
  setView(lng = 174.8860, lat = -40.9006, zoom = 5) %>%
  addMarkers(~lng, ~lat, popup = ~paste(city, "<br>Population:", population)) %>%
  addHeatmap(lng = ~lng, lat = ~lat, intensity = ~population, blur = 20, max = 1, radius = 15
nz_map_pop
```


+
−
Brisbane
Gold Coast
New South Wales
Sydney
Canberra
Victoria
Melbourne
Tasmania
Southland
Leaflet | © OpenStreetMap contributors

This code generates an interactive map of New Zealand that includes both city markers and a heatmap to visualize population density.

- `leaflet(nz_pop)`: Initializes the leaflet map using the `nz_pop` data frame, which contains city names, coordinates (latitude and longitude), and population data.
- `addProviderTiles(providers$OpenStreetMap)`: Adds an OpenStreetMap layer as the base map to display geographic features and locations.
- `setView(lng = 174.8860, lat = -40.9006, zoom = 5)`: Sets the initial view of the map:
    1. **Longitude (`lng`)**: `174.8860` centers the map horizontally over New Zealand.
    2. **Latitude (`lat`)**: `-40.9006` centers the map vertically.
    3. **Zoom Level**: `5` provides a view of the entire country.
- `addMarkers(~lng, ~lat, popup = ~paste(city, "<br>Population:", population))`: Adds markers to the map at the locations specified by the latitude (`~lat`) and longitude (`~lng`) coordinates from the `nz_pop` data frame. Includes interactive popups that show the city name and population for each marker.
- `addHeatmap(lng = ~lng, lat = ~lat, intensity = ~population, blur = 20, max = 1, radius = 15)`: Adds a heatmap layer to visualize population density.
    1. `lng and lat`: Use the longitude and latitude to position the heatmap points.
    2. `intensity`: Indicates the population size to determine the strength of the heatmap at each location.
    3. `blur`: Controls the blurriness of the heatmap spots (higher values create a smoother appearance).
    4. `max`: Sets the maximum intensity for the heatmap (value `1` normalizes the population data).
    5. `radius`: Specifies the size of the heatmap spots (value `15` determines the spot size).
- `nz_map_pop`: Stores the created map object, which includes both the city markers and the heatmap.

A **heatmap** is a type of map that uses colors to show where something is more or less intense. In this project, the heatmap shows areas in New Zealand with more people using brighter colors and less populated areas with darker colors

```
nz_boundaries <- st_read("nz_ta.geojson")
```

```
Reading layer `nz_ta' from data source
  `C:\Desktop\3MonthWorks\DataWranglingProjects\GIS-with-leaflet-sf-in-R\nz_ta.geojson'
  using driver `GeoJSON'
Simple feature collection with 66 features and 5 fields
Geometry type: MULTIPOLYGON
Dimension:     XY
Bounding box:  xmin: 166.4263 ymin: -47.29 xmax: 178.5768 ymax: -34.3926
Geodetic CRS:  WGS 84
```

```
nz_territory_2016_population <- read.csv("nz_territory_2016_population.csv")
```

This section of code loads two datasets, one for geographic information and the other for population statistics:

1. **`nz_boundaries <- st_read("nz_ta.geojson")`**: This line reads a **GeoJSON file** (`nz_ta.geojson`) into R using the `sf` (Simple Features) package. The file contains **geographic boundary data** for New Zealand's territories. Each territory is represented as a **polygon** that outlines its shape on a map. This data is crucial for creating maps that display the boundaries of New Zealand's regions.
2. **`nz_territory_2016_population <- read.csv("nz_territory_2016_population.csv")`**: This line loads a **CSV file** containing population statistics for New Zealand's territories in 2016. The dataset includes columns such as territory names, population totals, and changes in population (both in numbers and percentages).

A **GeoJSON** dataset is a file format used to store and share geographic data, like points, lines, and areas (e.g., cities, roads, or regions). It is based on a simple text format called JSON, making it easy to read and work with. Each geographic feature in the file can include additional information, like names, population, or area size. In this project, the **"nz_ta.geojson"** file contains the shapes and boundaries of New Zealand's regions, which are used to create maps and analyze the data visually.

```
head(nz_boundaries)
```

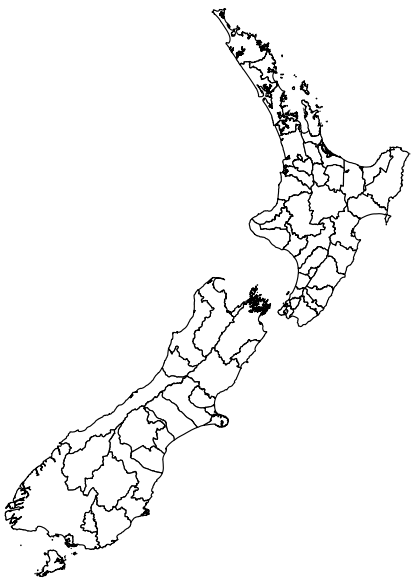
```
Simple feature collection with 6 features and 5 fields
Geometry type: MULTIPOLYGON
Dimension:     XY
Bounding box:  xmin: 172.6353 ymin: -38.0154 xmax: 176.098 ymax: -34.3926
Geodetic CRS:  WGS 84
  TA2016                TA2016_NAM AREA_SQ_KM LAND_SQ_KM rmapshaperid
1    001         Far North District   6689.840   6677.410            0
2    002         Whangarei District   2711.798   2711.798            1
```

```
3    003           Kaipara District   3108.712   3108.712            2
4    011 Thames-Coromandel District   2207.012   2207.012            3
5    012           Hauraki District   1270.049   1270.049            4
6    013           Waikato District   4450.592   4403.242            5
                         geometry
1 MULTIPOLYGON (((173.5798 -3...
2 MULTIPOLYGON (((174.7021 -3...
3 MULTIPOLYGON (((173.7648 -3...
4 MULTIPOLYGON (((175.9267 -3...
5 MULTIPOLYGON (((175.54 -37....
6 MULTIPOLYGON (((175.2678 -3...
```

```r
head(nz_territory_2016_population)
```

```
               nz_territory X2016_population X2016_population_change_number
1         Far North District            62000                           700
2         Whangarei District            87700                          1800
3           Kaipara District            21700                           600
4                   Auckland          1614300                         44400
5 Thames-Coromandel District            28400                           600
6           Hauraki District            19550                           450
  X2016_population_change_percent
1                             1.2
2                             2.0
3                             2.8
4                             2.8
5                             2.1
6                             2.2
```

```r
ggplot(data = nz_boundaries) +
  geom_sf(color = "black", fill = "white") +
  theme_void() +
  labs(title = "New Zealand Territories Map")
```



This code creates a simple black-and-white map of New Zealand's territorial boundaries using the **ggplot2** library and spatial data (**sf** format).

- `ggplot(data = nz_boundaries)`: Initializes a plot using the **nz_boundaries** dataset, which contains spatial data about New Zealand's regions.
- `geom_sf(color = "black", fill = "white")`: Adds the geographical shapes (boundaries) to the map using the **geom_sf** function. `color = "black"`: The boundaries of the regions are drawn in black. `fill = "white"`: The inside of the regions is filled with white.
- `theme_void()`: Removes all unnecessary elements like axes, grids, and background, leaving only the map.
- `labs(title = "New Zealand Territories Map")`: Adds a title to the map, which is displayed as **"New Zealand Territories Map"**.

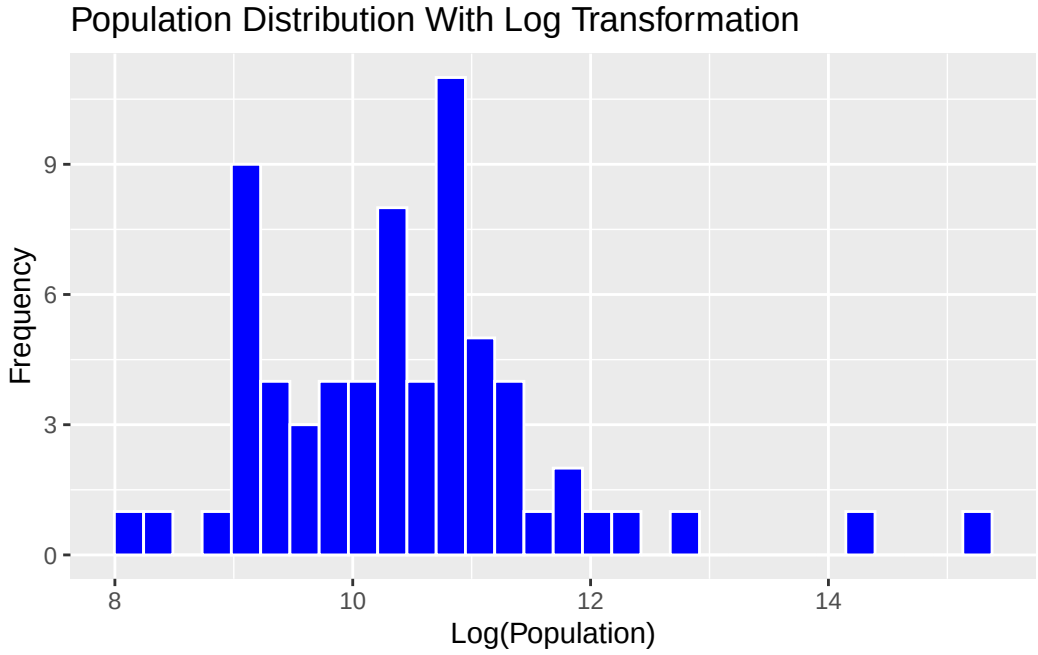
```
nz_territory_2016_population$log_population <-
  log1p(nz_territory_2016_population$X2016_population)
```

This code creates a new column called **log_population** in the dataset **nz_territory_2016_population** by applying a **logarithmic transformation** to the population values.

- `log1p()`: This function calculates the natural logarithm of a value plus 1. It is used here to handle cases where population values might be zero, as the logarithm of zero is undefined. Adding 1 ensures the transformation works for all values.
- `$X2016_population`: Refers to the column in the dataset that contains the population numbers for 2016.
- `nz_territory_2016_population$log_population`: This creates a new column named **log_population** in the dataset, which stores the log-transformed population values.

Logarithmic transformation is used to make large numbers easier to understand and compare. For example, population numbers can vary a lot between areas—some are very small, while others are extremely large. Taking the logarithm of the population reduces the size of large values, making the data more balanced and easier to visualize. This is especially helpful for maps because it ensures both small and large populations are clearly represented, without the larger values overshadowing the smaller ones. In this project, the `log1p()` function was used, which calculates the logarithm after adding 1 to avoid issues with very small or zero values.

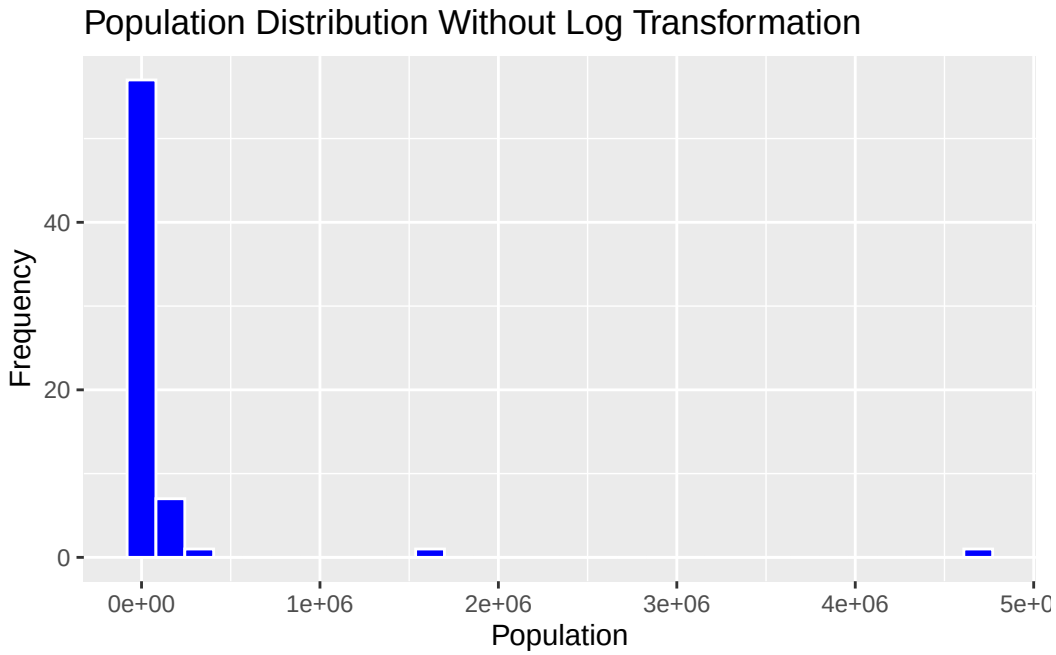
```
ggplot(nz_territory_2016_population, aes(x = log_population)) +
  geom_histogram(bins = 30, fill = "blue", color = "white") +
  labs(title = "Population Distribution With Log Transformation", x = "Log(Population)", y =
```



This code generates a bar chart that shows how the population values are distributed after applying the logarithmic transformation, making it easier to observe patterns in the data.

- **`ggplot(nz_territory_2016_population, aes(x = log_population))`**: Sets up the plot using the dataset **`nz_territory_2016_population`**, with the log-transformed population (**`log_population`**) as the x-axis.
- **`geom_histogram(bins = 30, fill = "blue", color = "white")`**: Creates a histogram with 30 bins (bars), where the bars are filled with blue and have white borders. The bins divide the population data into intervals to show how frequently values occur in each range.
- **`labs(title = "Population Distribution", x = "Log(Population)", y = "Frequency")`**: Adds a title to the plot ("Population Distribution") and labels the x-axis as "Log(Population)" and the y-axis as "Frequency".

```
ggplot(nz_territory_2016_population, aes(x = X2016_population)) +
  geom_histogram(bins = 30, fill = "blue", color = "white") +
  labs(title = "Population Distribution Without Log Transformation", x = "Population", y = "
```



This graph is a histogram showing the distribution of population data without applying a logarithmic transformation. Most of the population values are concentrated in the smaller intervals (close to zero), making it hard to see details for regions with larger populations. This is why a logarithmic transformation is often applied to spread out the data and make patterns easier to observe.

```r
nz_choropleth_data <- left_join(nz_boundaries, nz_territory_2016_population, by = c("TA2016_
```

This code merges two datasets, `nz_boundaries` and `nz_territory_2016_population`, based on a common column. Here's what it does in simple terms:

1. **`nz_boundaries`**: Contains geographical data, like the shapes of New Zealand's territorial boundaries.
2. **`nz_territory_2016_population`**: Contains population data for each territory in 2016.

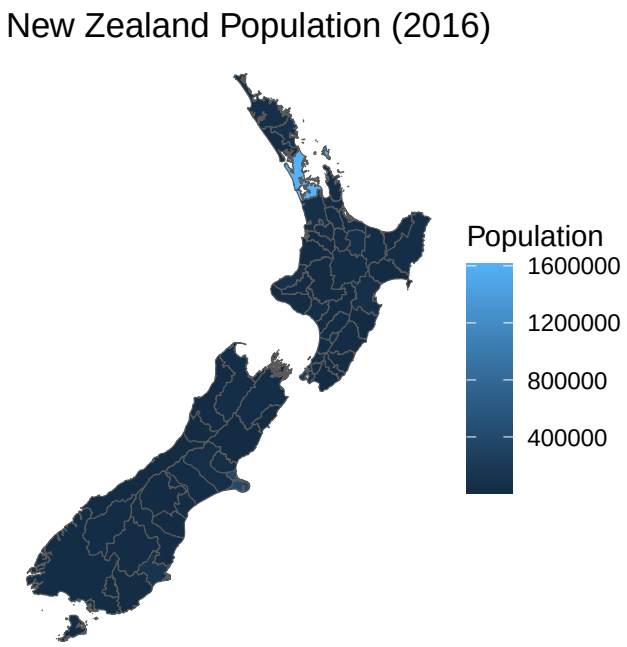
How the Code Works:

- **`left_join`**: Combines the two datasets so that each territory in `nz_boundaries` gets its corresponding population data from `nz_territory_2016_population`.
- **`by = c("TA2016_NAM" = "nz_territory")`**: Specifies which columns to match:
  - `TA2016_NAM` in `nz_boundaries` (territory names) is matched with `nz_territory` in the population dataset.

The new dataset `nz_choropleth_data` contains both the geographical boundary information and the population data, making it ready for creating a map.

A **choropleth map** is a map that uses colors or shades to represent data for specific geographic regions, such as districts or territories. Darker or lighter shades indicate higher or lower values, making it easy to compare data across regions visually. In this project, the choropleth map was used to show the population of New Zealand territories in 2016. Each region was shaded based on its population, providing a clear way to identify areas with higher or lower populations.

```r
ggplot(data = nz_choropleth_data) +
  geom_sf(aes(fill = X2016_population)) +
  theme_void() +
  labs(title = "New Zealand Population (2016)", fill = "Population")
```



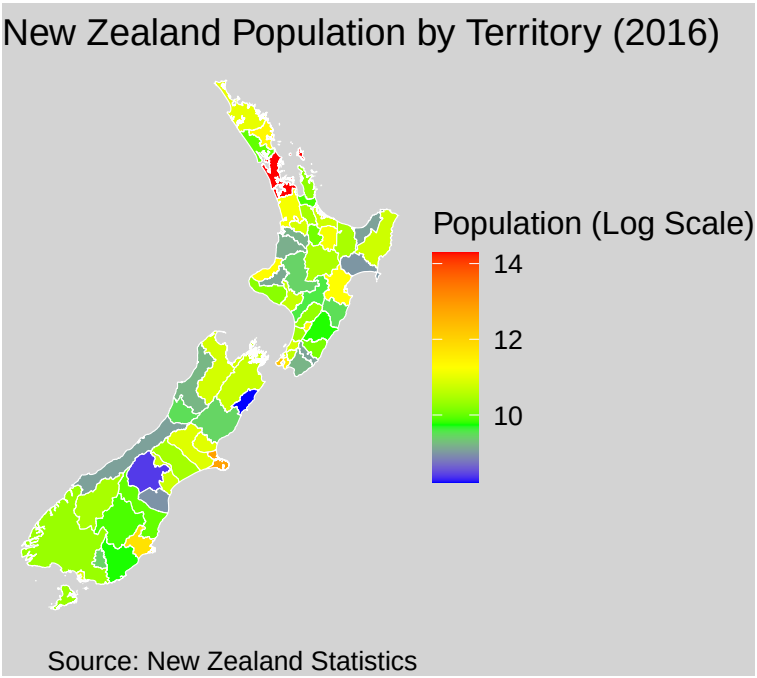
This code creates a **choropleth map** using the `ggplot2` library to visualize the population of New Zealand's territories in 2016. Here's what each part does:

- **`ggplot(data = nz_choropleth_data)`**: Specifies the data (`nz_choropleth_data`) to be used for the map. This dataset combines geographic boundaries and population data for each territory.
- **`geom_sf(aes(fill = X2016_population))`**: Adds the map layer using `geom_sf`, which works with spatial data. The `fill` aesthetic is used to color the regions based on the `X2016_population` column, representing the population for each territory.
- **`theme_void()`**: Removes unnecessary elements like axes, gridlines, and backgrounds to make the map cleaner and focus on the data.
- **`labs(title = "New Zealand Population (2016)", fill = "Population")`**: Adds a title to the map and labels the legend to indicate that the shading represents the population.

The result is a clean, color-coded map that visually shows population distribution across New Zealand's territories in 2016.

```
nz_plot <- ggplot(data = nz_choropleth_data) +
  geom_sf(aes(fill = log1p(X2016_population)), color = "white", size = 0.2) +        scale_f
    colors = c("blue", "green", "yellow", "orange", "red"),
    name = "Population (Log Scale)"
```

```
  ) +
  theme_void() +
  labs(
    title = "New Zealand Population by Territory (2016)",
    caption = "Source: New Zealand Statistics"
  ) +
  theme(
    text = element_text(family = "sans", size = 12, color = "black"),
    plot.background = element_rect(fill = "lightgray", color = NA),
    legend.position = "right"
  )
print(nz_plot)
```




```
ggsave("nz_plot.png", plot = nz_plot, width = 10, height = 7, dpi = 300)
```

This code generates an **enhanced choropleth map** of New Zealand's population distribution by territory for the year 2016, applying a log transformation for better visualization.

- **`ggplot(data = nz_choropleth_data)`**: Specifies the dataset (`nz_choropleth_data`) for the map. This dataset combines geographic boundaries and population data for each territory.

- **`geom_sf(aes(fill = log1p(X2016_population)), color = "white", size = 0.2)`**: Adds the geographic shapes (territories) to the map. The `fill` aesthetic applies a log-transformed population value (`log1p`) to assign colors based on population size. The `color = "white"` outlines each region in white, and `size = 0.2` defines the border thickness.
- **`scale_fill_gradientn(...)`**: Defines a custom color gradient for the map, ranging from **blue** (low population) to **red** (high population), making differences easier to interpret. The legend is labeled as "Population (Log Scale)" to indicate the applied transformation.
- **`theme_void()`**: Removes unnecessary elements such as axes and gridlines, ensuring the focus is on the map.
- **`labs(...)`**: Adds a title: "New Zealand Population by Territory (2016)". Adds a caption as "Source: New Zealand Statistics" to credit the data source.
- **`theme(...)`**: Adjusts the appearance of text and the map's background:

1. `text = element_text(family = "sans", size = 12, color = "black")` sets the font and size for text elements.
2. `plot.background = element_rect(fill = "lightgray", color = NA)` adds a light gray background with no boader.
3. `legend.position = "right"` places the legend on the right.

- **`print(nz_plot)`**: Displays the map in the R console or viewer.
- **`ggsave("nz_plot.png", plot = nz_plot, width = 10, height = 7, dpi = 300)`**: Saves the map as a PNG file named **`nz_plot.png`** with high quality (300 dpi) for further use in reports or presentations.

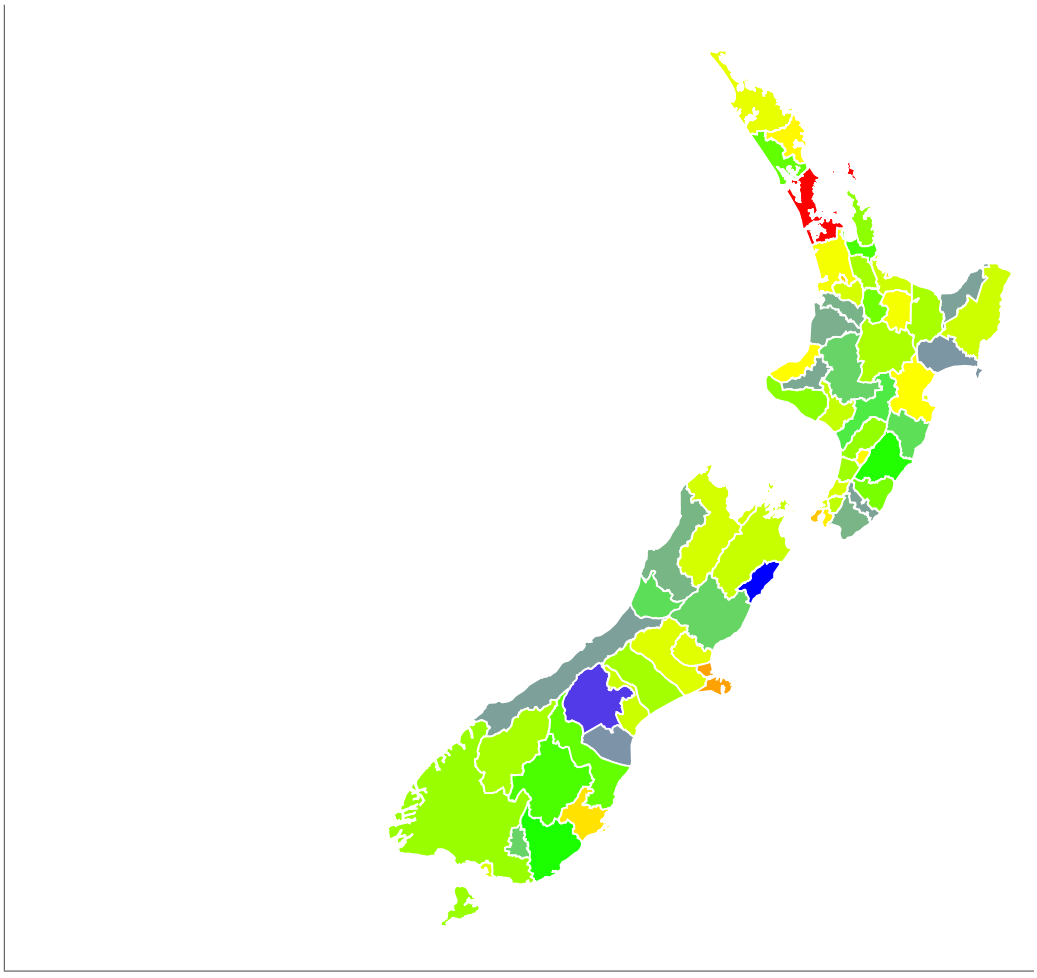
```
interactive_choropleth <- ggplotly(nz_plot)
interactive_choropleth <- interactive_choropleth %>%
  style(textposition = "top center") %>%
  layout(
    title = list(text = "New Zealand Population by Territory (2016)", font = list(size = 16))
  )

htmlwidgets::saveWidget(interactive_choropleth, "interactive_map.html")


ggsave("interactive_map.png", plot = nz_plot, width = 5, height = 7, dpi = 300)

interactive_choropleth
```

New Zealand Population by Territory (2016)



This code creates both an **interactive map** (saved as an HTML file) and a **static image** of the map (saved as a PNG file). The interactive map is more suitable for web-based exploration, while the PNG file can be used for offline reports.

- `interactive_choropleth <- ggplotly(nz_plot) :` Converts a static ggplot2 object (`nz_plot`) into an interactive map using `ggplotly`. The resulting interactive map allows zooming, panning, and displaying additional information on hover.
- `interactive_choropleth <- interactive_choropleth %>% ... layout(...):` Customizes the appearance of the interactive map. Adds a title, "New Zealand Population by Territory (2016)," with a font size of 16. Removes unnecessary gridlines from the X and Y axes.
- `htmlwidgets::saveWidget(interactive_choropleth, "interactive_map.html"):` Saves the interactive map as an HTML file named `interactive_map.html`. This file can be opened in any web browser to view and explore the interactive map.
- `ggsave("interactive_map.png", plot = nz_plot, width = 5, height = 7, dpi = 300):` Saves the original static version of the map (`nz_plot`) as a PNG file named `interactive_map.png`. The PNG file is useful for inclusion in reports or documents that do not support interactivity.
- `interactive_choropleth:` Displays the interactive map directly in the R environment or a supported viewer, allowing further exploration.

Conclusion

This project explored the use of R and its libraries to create various types of maps, including interactive maps and choropleth visualizations. The process involved working with geospatial data to represent New Zealand's population by territory. Through tools like `leaflet` for interactive maps and `ggplot2` for detailed static visualizations, the project successfully demonstrated how data can be effectively presented to highlight population patterns. These maps provide valuable insights and are useful for better understanding spatial data in a clear and engaging way.