Take-Home Exercise 1: Data Visualisation Makeover

Wang Anqi

# **1 Overview**

In this part, I will select one exercise provided by my classmate, critic three good design principles and three areas for further improvement.

Then with reference to the comment, I will prepare the makeover version of the data visualisation.

## 1.1 **Original visualization**

The original visualization is from my classmate Wang Shensi. Please click on this [Link](https://wshensi.netlify.app/take-home_ex/take-home_ex01/take-home_ex01) to view the original visualization report or obtain more information.

In this section, I will make comments and revisions on visualization 2 and 3.

# **2 Getting Started**

## 2.1 **Installing and loading the required libraries**

The following R packages were utilized for data pre-processing and visualization:

pacman::p\_load(plotly, tidyverse, DT, ggiraph, patchwork, ggstatsplot, GGally, corrplot)

install.packages("webshot", repos = "https://cloud.r-project.org")

package 'webshot' successfully unpacked and MD5 sums checked  
  
The downloaded binary packages are in  
 C:\Users\Wang\_Anqi\AppData\Local\Temp\RtmpyeOSfd\downloaded\_packages

webshot::install\_phantomjs()

## 2.2 **Importing data**

The code chunk below imports *exam\_data.csv* into R environment by using [*read\_csv()*](https://readr.tidyverse.org/reference/read_delim.html) function of [**readr**](https://readr.tidyverse.org/) package. **readr** is one of the tidyverse package.

respop <- read\_csv("data/respopagesex2024.csv", show\_col\_types = FALSE)

# **3 Data Wrangling**

To maintain data consistency, I will replicate exactly the data processing steps performed in the selected exercise.

## **3.1 Remove Duplicates**

Remove any duplicated rows to avoid double-counting:

## The code chunk

respop <- distinct(respop)  
DT::datatable(head(respop), options = list(scrollX = TRUE), caption = "After Removing Duplicates")

## Data table

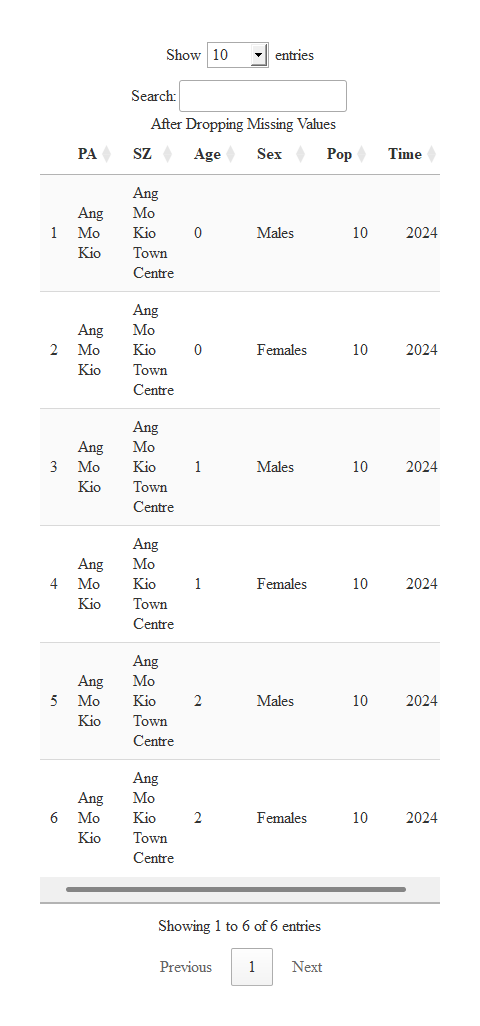
## **3.2 Handle Missing Values**

Ensure there are no missing or malformed entries in critical columns:

## The code chunk

respop <- drop\_na(respop)  
DT::datatable(head(respop), options = list(scrollX = TRUE),   
 caption = "After Dropping Missing Values")

## Data table



## **3.3 Standardize Age Format**

The Age column includes a non-numeric category "90\_and\_Over" which cannot be coerced into an integer. We convert this category into a numeric age of **90**, representing a minimum estimate for analysis:

## The code chunk

respop <- respop %>%  
 mutate(Age = ifelse(Age == "90\_and\_Over", "90", Age),  
 Age = as.integer(Age))  
DT::datatable(head(respop), options = list(scrollX = TRUE), caption = "After Converting Age Format")

## Data table



## **3.4 Convert Column Types**

Ensure categorical and numeric columns are properly formatted:

## The code chunk

respop <- respop %>%  
 mutate(  
 Sex = as.factor(Sex),  
 PA = as.factor(PA),  
 SZ = as.factor(SZ)  
 )  
DT::datatable(head(respop), options = list(scrollX = TRUE), caption = "After Converting Column Types")

## Data table



# **4 Evaluation and Improvement**

## 4.1 **Gender Ratio by Planning Area**

This section displays the **male-to-female population ratio** for each Singapore planning area. It uses color to indicate whether an area has more males, more females, or is balanced, and includes interactive tooltips for details.

### 4.1.1 **Original visualization**

## The plot



## The code

# Load Required Libraries  
library(tidyverse)  
library(ggiraph)  
  
#Calculate Gender Ratio and Prepare Tooltip  
gender\_ratio <- respop %>%  
 group\_by(PA, Sex) %>%  
 summarise(Total = sum(Pop), .groups = "drop") %>%  
 pivot\_wider(names\_from = Sex, values\_from = Total) %>%  
 mutate(Total\_Pop = Males + Females) %>%  
 filter(Total\_Pop > 100) %>%   
 mutate(  
 Ratio = round(Males / Females, 2),  
 Status = case\_when(  
 Ratio > 1.05 ~ "More Males",  
 Ratio < 0.95 ~ "More Females",  
 TRUE ~ "Balanced"  
 ),  
 Tooltip = paste0("Planning Area: ", PA,  
 "\nMales: ", format(Males, big.mark = ","),  
 "\nFemales: ", format(Females, big.mark = ","),  
 "\nRatio (M/F): ", Ratio),  
 Label = as.character(Ratio)  
 )  
  
#Create the ggplot Object with Interactive Bars  
gg <- ggplot(gender\_ratio,  
 aes(x = reorder(PA, Ratio), y = Ratio,  
 fill = Status,  
 tooltip = Tooltip, data\_id = PA)) +  
 geom\_col\_interactive(width = 0.7) +  
 geom\_text(aes(label = Label),  
 hjust = -0.1, color = "black", size = 3) +   
 scale\_fill\_manual(values = c("More Males" = "#1f77b4",  
 "More Females" = "#ff7f0e",  
 "Balanced" = "#66c2a5")) +  
 coord\_flip() +  
 theme\_minimal() +  
 theme(  
 axis.text.y = element\_text(size = 7),  
 plot.margin = margin(r = 20)   
 ) +  
 labs(title = "Interactive Gender Ratio by Planning Area (M/F)",  
 x = "Planning Area", y = "Male-to-Female Ratio")  
  
# Render Interactive Widget  
girafe(ggobj = gg,  
 width\_svg = 8, height\_svg = 6,  
 options = list(  
 opts\_tooltip(css = "background-color:white;color:black;border:1px solid gray;padding:5px;"),  
 opts\_hover(css = "fill-opacity:0.8;cursor:pointer;")  
 ))

### **4.1.2 Comments**

**Good design principles:**

* ☒ **Clear title and axis labels**
* The chart has a well-defined title and clearly labeled axes, making it easy for viewers to understand that it visualizes the male-to-female ratio by planning area.
* ☒ **Effective color encoding of categories**
* The use of three distinct colors to represent “More Males”, “Balanced”, and “More Females” helps convey categorical differences clearly, supported by a well-positioned legend.
* ☒ **Sorted bars enhance interpretability**
* The bars are ordered from highest to lowest gender ratio, which allows for quick comparison across planning areas and easy identification of outliers.

**Areas for further improvement:**

* **Insufficient color contrast**
* The colors for “Balanced” and “More Females” are relatively muted and close in tone, making them harder to distinguish—especially for viewers with color vision deficiencies.
* **Inconsistent use of data labels**
* Only bars on the right have numeric labels, while shorter bars (on the left) do not, making it harder to interpret exact values for those areas.
* **Lack of a reference line or benchmark**  
  The chart does not include a reference line (e.g., a ratio of 1.0) or an average, which would help viewers better understand deviations from gender balance.

### **4.1.3 Makeover Version of the Visualization**

## The plot



## The code

# Calculate Gender Ratio and Prepare Tooltip  
gender\_ratio <- respop %>%  
 group\_by(PA, Sex) %>%  
 summarise(Total = sum(Pop), .groups = "drop") %>%  
 pivot\_wider(names\_from = Sex, values\_from = Total) %>%  
 mutate(Total\_Pop = Males + Females) %>%  
 filter(Total\_Pop > 100) %>%  
 mutate(  
 Ratio = round(Males / Females, 2),  
 Status = case\_when(  
 Ratio > 1.05 ~ "More Males",  
 Ratio < 0.95 ~ "More Females",  
 TRUE ~ "Balanced"  
 ),  
 Tooltip = paste0("Planning Area: ", PA,  
 "\nMales: ", format(Males, big.mark = ","),  
 "\nFemales: ", format(Females, big.mark = ","),  
 "\nRatio (M/F): ", Ratio),  
 Label = as.character(Ratio)  
 )  
  
gg <- ggplot(gender\_ratio,  
 aes(x = reorder(PA, Ratio), y = Ratio,  
 fill = Status, tooltip = Tooltip, data\_id = PA)) +  
 geom\_col\_interactive(width = 0.7) +  
  
 # Reference line at gender balance (M/F = 1)  
 geom\_hline(yintercept = 1, linetype = "dashed", color = "black", size = 0.6) +  
  
  
 # Improved data labels for all bars  
 geom\_text(  
 aes(label = Label),  
 hjust = 1.1,   
 color = "white",  
 size = 3  
) +  
  
  
 # Enhanced color contrast  
 scale\_fill\_manual(values = c(  
 "More Males" = "#1f77b4", # Blue  
 "More Females" = "#d62728", # Red  
 "Balanced" = "#2ca02c" # Green  
 )) +  
  
 coord\_flip() +  
 theme\_minimal() +  
 theme(  
 axis.text.y = element\_text(size = 7),  
 plot.margin = margin(r = 20)  
 ) +  
 labs(title = "Interactive Gender Ratio by Planning Area (M/F)",  
 x = "Planning Area", y = "Male-to-Female Ratio")  
  
# Render the interactive widget  
girafe(ggobj = gg,  
 width\_svg = 8, height\_svg = 6,  
 options = list(  
 opts\_tooltip(css = "background-color:white;color:black;border:1px solid gray;padding:5px;"),  
 opts\_hover(css = "fill-opacity:0.8;cursor:pointer;")  
 ))

## 4.2 **Total Population by Planning Area**

This section presents the total 2024 population for each planning area in Singapore using an interactive bar chart. Tooltips reveal the exact population figures on hover, and low-population areas are excluded to improve readability.

### 4.2.1 **Original visualization**

## The plot



## The code

# Aggregate and Filter Population Data  
total\_pop <- respop %>%  
 group\_by(PA) %>%  
 summarise(Total\_Pop = sum(Pop), .groups = "drop") %>%  
 filter(Total\_Pop > 100) %>%  
 mutate(  
 Tooltip = paste0("Planning Area: ", PA,  
 "\nPopulation: ", format(Total\_Pop, big.mark = ",")),  
 Label = paste0(round(Total\_Pop / 1000), "k")  
 )  
  
# Build Interactive ggplot Object  
max\_val <- max(total\_pop$Total\_Pop)  
  
gg <- ggplot(total\_pop, aes(x = reorder(PA, Total\_Pop), y = Total\_Pop,  
 tooltip = Tooltip, data\_id = PA)) +  
 geom\_col\_interactive(fill = "#2ca02c", width = 0.7) +  
 geom\_text(aes(label = Label),  
 hjust = -0.1, color = "black", size = 3) +   
 coord\_flip() +  
 scale\_y\_continuous(  
 labels = ~ paste0(.x / 1000, "k"),  
 limits = c(0, max\_val \* 1.1)   
 ) +  
 theme\_minimal() +  
 theme(axis.text.y = element\_text(size = 7),  
 plot.margin = margin(r = 30)) +   
 labs(title = "Interactive Total Population by Planning Area (2024)",  
 x = "Planning Area", y = "Total Population")  
  
# Render Interactive Widget  
girafe(ggobj = gg,  
 width\_svg = 8, height\_svg = 6,  
 options = list(  
 opts\_tooltip(css = "background-color:white;color:black;border:1px solid gray;padding:5px;"),  
 opts\_hover(css = "fill-opacity:0.8;cursor:pointer;")

### **4.2.2 Comments**

**Good design principles:**

* ☒ **Clear ranking and labeling** The bar chart is sorted from highest to lowest population, and all bars are labeled with clear population values (in ‘k’), allowing viewers to quickly grasp the distribution.
* ☒ **Effective use of color and consistency** The uniform green color keeps the design clean and avoids unnecessary visual distraction, which is appropriate for a single-variable chart.
* ☒ **No benchmark or comparative line** There’s no indication of average population, median, or any thresholds (e.g., “high density”), which could help interpret what counts as a large or small population.

**Areas for further improvement:**

* **Overcrowded y-axis labels** With over 30 planning areas displayed, the vertical axis labels appear densely packed, making it difficult for viewers to quickly scan and compare area names.
* **Lack of reference line or benchmark** The chart does not include an average, median, or any threshold line to indicate what constitutes a high or low population, reducing the interpretability of individual values.
* **No color encoding for additional meaning** The use of a single green color keeps the chart clean but fails to convey additional insights, such as population tiers or regional grouping, which could enhance visual impact and analytical depth.

### **4.2.3 Makeover Version of the Visualization**

## The plot



## The code

# Aggregate and Filter Population Data  
total\_pop <- respop %>%  
 group\_by(PA) %>%  
 summarise(Total\_Pop = sum(Pop), .groups = "drop") %>%  
 filter(Total\_Pop > 100) %>%  
 mutate(  
 ToolTip = paste0("Planning Area: ", PA,  
 "\nPopulation: ", format(Total\_Pop, big.mark = ",")),  
 Label = paste0(round(Total\_Pop / 1000), "k")  
 )  
  
max\_val <- max(total\_pop$Total\_Pop)  
avg\_val <- mean(total\_pop$Total\_Pop)  
  
# Build the Interactive ggplot Object  
gg <- ggplot(total\_pop,  
 aes(x = reorder(PA, Total\_Pop), y = Total\_Pop,  
 fill = Total\_Pop, tooltip = ToolTip, data\_id = PA)) + #   
  
 geom\_col\_interactive(width = 0.7) +  
  
 geom\_hline(yintercept = avg\_val, linetype = "dashed", color = "black", size = 0.6) +  
  
 geom\_text(aes(label = Label),  
 hjust = -0.1, color = "black", size = 3) +  
  
 scale\_fill\_gradient(  
 low = "#b7e3b1", high = "#1b7837",  
 labels = function(x) paste0(round(x / 1000), "k")  
 ) +  
  
 coord\_flip() +  
  
 scale\_y\_continuous(  
 labels = ~ paste0(.x / 1000, "k"),  
 limits = c(0, max\_val \* 1.1)  
 ) +  
  
 theme\_minimal() +  
 theme(  
 axis.text.y = element\_text(size = 6.5),  
 plot.margin = margin(r = 40),  
 legend.title = element\_blank()  
 ) +  
  
 labs(  
 title = "Interactive Total Population by Planning Area (2024)",  
 x = "Planning Area", y = "Total Population"  
 )  
  
# Render Interactive Widget  
girafe(ggobj = gg,  
 width\_svg = 8, height\_svg = 6,  
 options = list(  
 opts\_tooltip(css = "background-color:white;color:black;border:1px solid gray;padding:5px;"),  
 opts\_hover(css = "fill-opacity:0.8;cursor:pointer;")  
 ))

|  |
| --- |
| Note |
| The dashed vertical line in the chart represents the **average total population** across all planning areas. It serves as a benchmark to help identify which areas have significantly higher or lower population levels relative to the national average. |