Data Visualisation: Theory and Practice

Yujie Chu, Pia Fullaondo, Qinqing Li, Jacko Zhou

January 29, 2024

Contents

Overview

Graphical statistics is an indispensable tool in the arsenal of every contemporary data scientist. Primarily, it enables users to delve into data, explore its nuances, and effectively communicate empirical discoveries through visual means. This project aims to provide an overview of modern data visualisation methods, delve into their theoretical underpinnings, and demonstrate the potency of these methodologies using both synthetic and authentic datasets. Computational implementations via dashboards — using Python Dash or R Shiny — are used to showcase the methodologies in practice. Furthermore, the project is aware of the ongoing evolution of novel data visualisation techniques, recognising it as an active area of research (e.g. Rodu and Kafadar, 2022), and thus aims to cover both textbook methods as well as state-of-the-art approaches and open problems.

This thesis aims to explore the theoretical foundations that underlie various visualisation methodologies, study their mathematical construction, and illustrate their practical application. In particular, Chapters 3, 4, and 5 concentrate on standard implementations of Graphical Statistics for univariate, bivariate, and multivariate datasets respectively. Chapter 3 discusses the Kernel Density Estimate, ROC Curve, and Time Series Analysis, showcasing their use through the Tooth Growth and the Exchange Rate datasets. Chapter 4 is focused on the topic of regression, discussing the theory behind Simple Linear Regression followed by LOESS regression. Chapter 5 introduces the model selection method AIC and Multiple Linear Regression, followed by the theoretical foundation of biplots and t-SNE. Subsequent chapters cover active research areas in Graphical Statistics and implementation of dashboards.

Throughout this thesis, R package names such as *ggplot2* are italicised, while R commands like `faceting` are displayed in a teletype font.The R code chunks for each figure in the report can be found in a separate GitHub code folder.

Literature and Research
Data visualisation is an area of interest in numerous method-specific publications. One notable publication is the Journal of Computational and Graphical Statistics. This journal publishes ongoing research on the latest techniques in computational and graphical methods in statistics, encompassing data analysis and numerical graphical displays. Spatial Statistics publishes articles on the theory and application of spatial and spatio-temporal statistics. The R Journal publishes research articles in statistical computing that are of interest to R users. These together reveal the vast potential for current and future research in this ever-evolving domain, particularly as our digital landscape and data sphere expand at an accelerated pace.

1 Introduction

This chapter provides case examples demonstrating both the influential applications and potential misuses of data visualisations, with the aim of underscoring the enduring importance of data visualisation.These are sourced from BBC Ideas [14]. Following this, an introduction to the R package *ggplot2* is provided, along with a presentation of the datasets used in this thesis.

1.1 Historical Background and Misuses of Data Visualisation

Motivations for Using Data Visualisations — Case 1
Florence Nightingale was not only a social reformer and the founder of modern nursing but also a pioneering statistician. It was her application of data visualisation during the Crimean War that transformed the field of healthcare and pushed for social reform.

During the Crimean War, Nightingale recognised that unsanitary hospital conditions were claiming more lives than the battlefield itself. With the help of William Farr, Nightingale created the coxcomb aimed to illustrate the toll of preventable mortality on soldiers, as shown in Figure 1. The coxcomb, resembling an unconventional pie chart, partitioned mortality by causes. Blue indicates preventable deaths, red indicates deaths by wounds, and black indicates other causes. The blue areas outweighed the red and black sections combined, highlighting the disproportionate impact of unsanitary hospital conditions on the mortality rate.

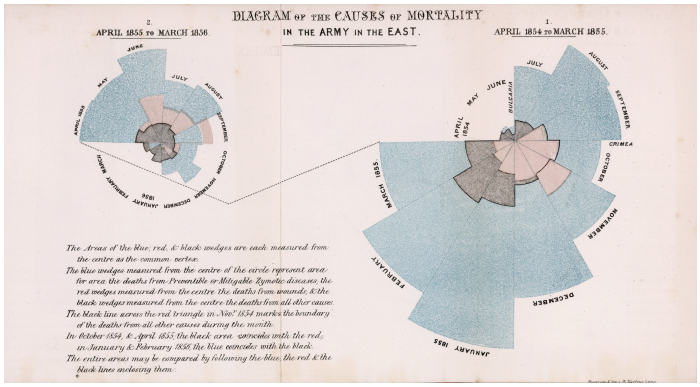

Figure 1: "Diagram of the causes of mortality in the army in the East" (1858) by Florence Nightingale [13]

Nightingale leveraged the compelling visualisations in her advocacy efforts, presenting them to MPs and government officials who otherwise are unlikely to read or understand statistical reports. Nightingale successfully persuaded Queen Victoria, head of the British Army at the time, to allocate funding for the improvement of better conditions in military hospitals.

Motivations for Using Data Visualisations — Case 2
Sometimes, one glance is enough to convey the most powerful idea. Edward Hawkins, a British climate scientist and Professor of climate science at the University of Reading, is renowned for his exceptional datavisualisations of climate change.

In 2018, Edward Hawkins was invited to deliver a lecture on climate change in Wales to an audience with diverse backgrounds. It was important to effectively convey the growing urgency surrounding global warming. To achieve this, he created a chart that used just colours, without any words, titles, or legends, as shown in Figure 2. This seemingly simple yet remarkably powerful chart visually illustrated the Earth's warming trend since 1850.

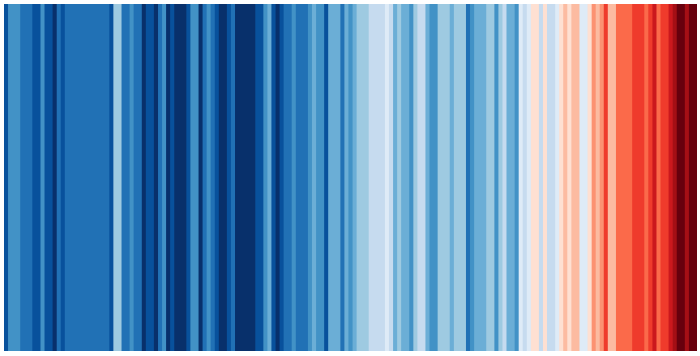

Figure 2: Latest global stripes (1850-2020) by Edward Hawkins [11]

Known as the "warming stripes," this chart cleverly employs blues to indicate cooler-than-average years and reds to signify hotter-than-average years. Its influence reached far and wide, gracing the front pages of major media outlets and featured in news broadcasts worldwide. It became a symbol in climate change demonstrations. Arguably, it stands as one of the most iconic graphics in modern times.

Misuses of Data Visualisation — Case 1
Having observed the remarkable effectiveness of data visualisations, the significance of employing them correctly becomes apparent. The improper use of data visualisations holds the potential to significantly influence the public in misleading ways, resulting in undesired consequences.

Inappropriate data visualisation can conceal trends rather than reveal them. Figure 3 illustrates an instance of this issue. On the left-hand side, an inappropriate scale is used - the y-scale ranging from 0 to 30 million dollars, obscuring the fluctuations in payroll spending. Conversely, on the right-hand side, observe that there's a significant increase of over 500,000 dollars in just two months. This revelation is substantial; considering inflation, 500,000 dollars in 1937 is worth well over 10 million dollars today [1].

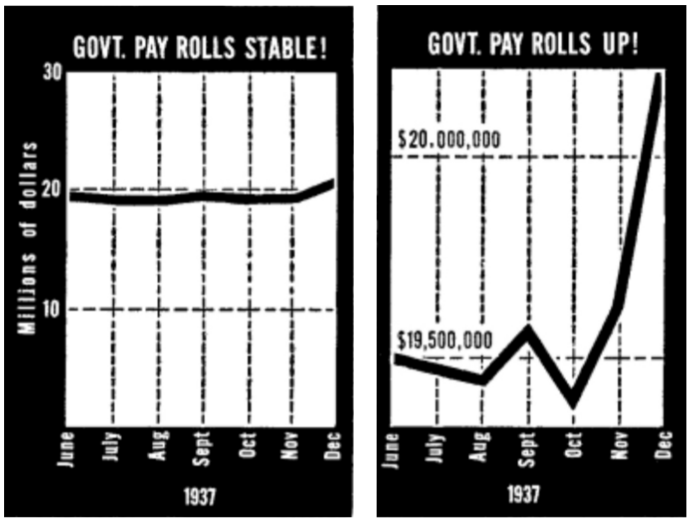

Figure 3: Incomplete Data Analysis, "How to lie with statistics" by Miguel de Carvalho [5]

Misuses of Data Visualisation — Case 2
One striking example of data visualisation misuse is found in the Kallikak Family tree - one of the most prominent eugenic narratives of the 20th century.

The visualisation (as shown in Figure 4) was created by the psychologist Henry Goddard and presented in his 1912 book, "The Kallikak Family: A Study in the Heredity of Feeble-Mindedness." Goddard's narrative centered around Martin Kallikak, a soldier who, in addition to his marriage to a respected citizen, had a one-night stand with a "feeble-minded" maid. Goddard believed that intellectual disabilities were inherited traits. In Goddard's account, the legitimate family was successful, while the children of the "feeble-minded" maid were labeled as "the lowest types of human beings." However, research has since revealed that the entire story was fictitious, as there was no record of the maid's existence [18].

Regrettably, the Kallikak family tree became a central element in the eugenics movement. Figure 4 was featured in the 1935 Nazi propaganda film "Das Erbe" (The Inheritance), which was used to promote public acceptance of Nazi eugenics laws. This propaganda laid the groundwork for the forced sterilization of approximately 400,000 people under Nazi eugenics policies [17].

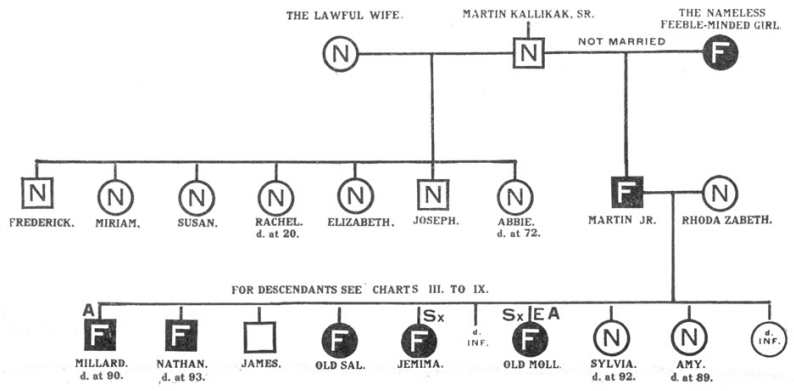

Figure 4: The Kallikak Family tree (1912) by Henry Goddard [3]

1.2 Computing and Data Visualisation

As a powerful tool for statistical analysis and graphic creation, R offers a wide array of functionalities to cater to a range of data visualisation needs. In data visualisation, *ggplot2* is the main tool used to create plots capable of representing various types of datasets. The foundation of *ggplot2* lies in the Grammar of Graphics, allowing the sequential construction of individual elements composing a graph, which are subsequently combined to create a unified graphical representation.

Key R Commands
As shown in subsequent sections, it is possible to produce visualisations rather effortlessly by utilising internal data from R, or even external data, together with *ggplot2*. When using *ggplot2*, certain key tools are used repeatedly to create visual plots from datasets. Some of these include: `geom`, `scale`, `coord`, `theme`, and more.

These particular tools serve the following functions [23]: `geom` refers to geometric objects, pivotal components within *ggplot2* used to define the visual representation of data in a plot. Each `geom` function corresponds to a specific graphical representation type in a chart. Additionally, in *ggplot2*, `scales` functions enable the adjustment of various mapping details, including colour choices, label formatting, legend arrangement, and more.

`coord` provides the axes and gridlines which aid in interpreting the graph. Various coordinate systems such as cartesian, polar, and map projections are available. Furthermore, `faceting` is a robust feature enabling the partitioning of a single plot into multiple plots based on factors present in the dataset. This functionality proves particularly beneficial for exploring and presenting data with multiple groups or categories.

The `theme` function holds significance in customising the non-data elements of plots. The theme system within *ggplot2* allows precise adjustment of aesthetic aspects such as fonts, labels, legends, and background colours. This tool is essential for enhancing plot readability and creating visually captivating graphics tailored to specific audiences or publication requirements.

Key R Packages
The R language is a powerful data analysis tool, and its power lies in the wide variety of R packages that help it achieve a wide variety of functions. Through out this project, different packages, such as *maps*, *dplyr*, *tidyverse*, and *gridExtra* will be used. These packages significantly enhances the analysis and presentation of data.

The *dplyr* packages plays a crucial role in data manipulation and transformation. It helps us filter, sort, summarise, and transform datasets, making data analysis more efficient and manageable. The *maps* package is used for providing map data and tools, allowing one to easily draw and display maps in geospatial analyses. The *tidyverse* package is indeed a collection of R packages designed to provide a cohesive and comprehensive set of tools for data science. It includes several widely used packages such as *ggplot2* and *dplyr*. The *gridExtra* packages offers the capability to combine several graphical objects into a cohesive display.

1.3 Datasets

This section introduces the different datasets analysed and visualised in the subsequent chapters.

mtcars: The built-in R dataset, mtcars was extracted from the 1974 Motor Trend US magazine. It includes the fuel consumption and 10 aspects of automobile design for 32 automobiles (1973–74 models).

iris: The built-in R dataset iris was collected by Edgar Anderson. This dataset was famously used by British statistician Ronald Fisher to demonstrate linear discriminant analysis in 1936. It encompasses 5 flower characteristics for 3 types of iris, each with 50 samples.

ToothGrowth: The built-in R dataset, ToothGrowth, measures the tooth growth of 60 guinea pigs. Each animal was either administered with vitamin C or orange juice.

Fire in Brazil: Open-source fire observation data is provided by NASA. The analysis focuses on Brazil (2013-2022). Note that the dataset contains the variable "confidence", ranging from 0% to 100%. The dataset was filtered with a confidence level of $\geq 95\%$ to ensure an accurate account of fire occurrences [6].

Exchange Rate: The exchange rate data, available at the Bank of England, provides daily spot exchange rates against the pound Sterling (2005-present). This report examines the daily spot rates of the Canadian Dollar, Euro, and US Dollar against the pound Sterling.

Taipei Housing: The historical real estate valuation from Sindian Dist., New Taipei City, Taiwan, available at UC Irvine Machine Learning Repository.

2 Theoretical Foundations of Data Visualisation

This chapter, "Theoretical Foundations of Data Visualisation," delves into the core principles and concepts that serve as the base of this field. We seek to understand not only the "how" but also the "why" behind the creation of visualisations that captivate and inform.

2.1 Introduction to Data Visualisation Theory

Creating effective data visualisations requires a robust theoretical framework underlying every chart, graph, or plot. These theoretical underpinnings not only form the basis of data visualisation but also influence how we represent, perceive, understand, and interpret data.

Guiding Principles for Data Representation
The theoretical framework of data visualisation involves guiding principles dictating visual representation of data. These principles include **accuracy**, emphasising faithful reflection of underlying data to reduce distortion or misinterpretation; **simplicity**, advocating for streamlined visuals to convey information effectively; **clarity**, ensuring visuals are easily understood without unnecessary complexity; **relevance**, presenting information pertinent to the message or question addressed; and **consistency**, maintaining uniform use of visual elements like color coding and labeling throughout a visualisation.

Theoretical Framework and Visual Perception
Understanding how the human brain processes visual information is a fundamental aspect of data visualisation theory. This knowledge plays a crucial role in designing visualisations that effectively connect with viewers. It encompasses several key considerations which will be studied in order: the Gestalt Principles, which encompass proximity, similarity, and continuity, affecting how visual elements are grouped and interpreted; Color Theory, involving the strategic use of color contrasts and harmonies to improve clarity and impact; and the management of Cognitive Load, which emphasises the importance of reducing mental effort needed to process information.

2.2 Data Types and Visualisation Techniques

To have a discussion about data representation, understanding the nature of the data is key. Data comes in various types, and selecting the appropriate visualisation technique is contingent upon recognising these distinctions. In this section, the data types are categorised and matched with their suitable visualisation techniques.

2.2.1 Categorisation of Data Types

Data types can be broadly categorised into four main types:

- **Nominal data**: represents categories or labels without any inherent order. Examples include colours, gender categories, and city names.
- **Ordinal data**: implies a meaningful order or ranking among categories but lacks equal intervals between them. Examples include survey responses (eg. "very satisfied", "satisfied", "neutral", "dissatisfied", "very dissatisfied")
- **Interval data**: possesses ordered categories with equal intervals between them, but it lacks a true zero point. Temperature is measured in Celsius or Fahrenheit as an example.

- **Ratio data**: includes ordered categories with equal intervals and a meaningful zero point. Examples are age, income, and weight.

Matching Data Types with Appropriate Visualisation Techniques
Selecting appropriate visualisation techniques is essential for effective data communication. Various data types demand specific visualisation methods for optimal representation [12]. For **nominal data**, bar charts and stacked bar charts are effective in displaying categorical information and relative proportions. **ordinal data** benefits from ordered bar charts, dot plots, or stacked bar charts, maintaining the ranking and order of categories. **interval data** is best visualised using line charts, histograms, and box plots, showcasing trends and distributions without assuming a true zero point. **ratio data** finds effective representation through scatter plots, histograms, and line charts, enabling precise comparisons and measurements due to the presence of a meaningful zero point. These are represented in Figure 5, where artificial data from a class of 4th-year university students is visualised.

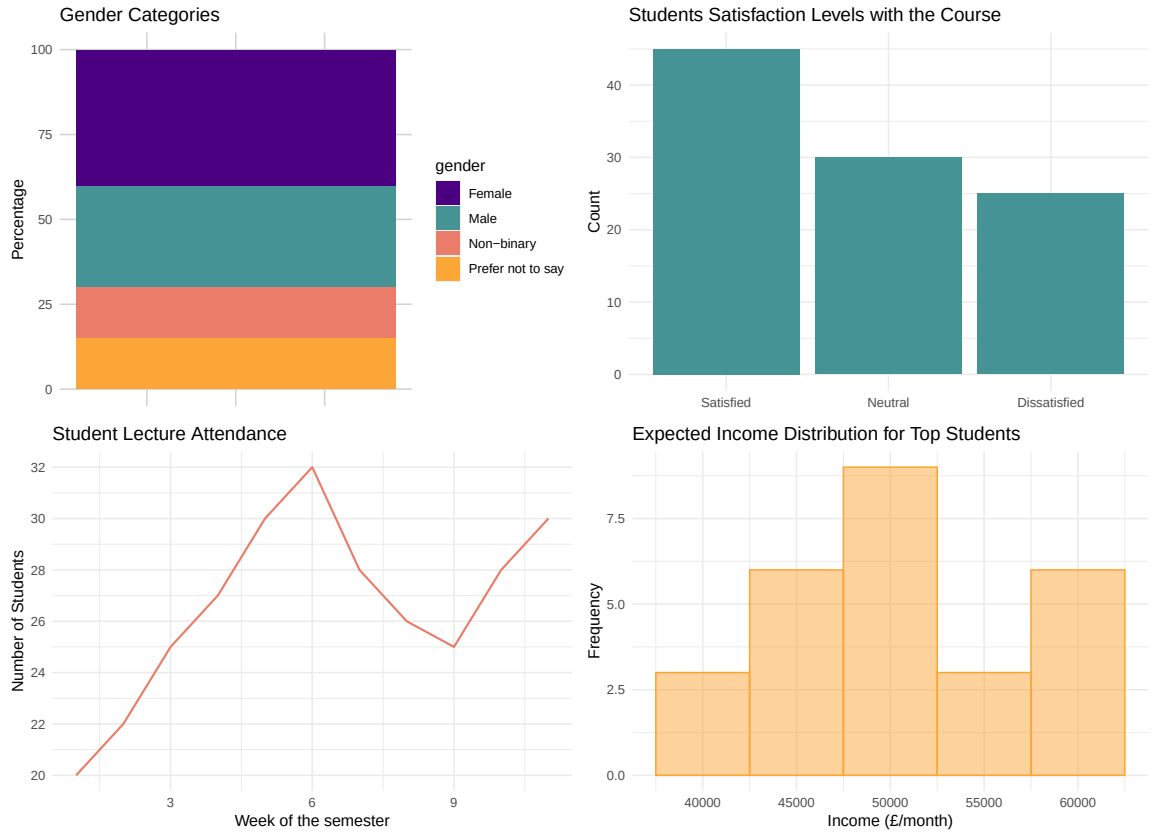

Figure 5: Different types of visualisation techniques according to the data type

2.3 Data Abstraction and Representation

The transformation of raw data into meaningful representations is a pivotal step in data representation. This process, known as data abstraction, involves distilling complex datasets into visual

forms that convey insights. In this section, we explore data abstraction, the hierarchies and levels of abstraction in data visualisation, and the critical trade-offs between abstraction and the potential loss of information.

Data Abstraction: Transforming Raw Data
Data abstraction involves simplifying and structuring raw data into comprehensible and insightful formats. This process serves as the bridge, transforming numbers, text, and variables into visual elements that convey patterns, trends, and relationships, forming the core of informative data visualisations [21].

2.3.1 Hierarchies and Levels of Abstraction

Abstraction operates on multiple levels of granularity. Hierarchies of abstraction allow us to represent data at varying levels of detail:

1. **Low-Level Abstraction**: At the lowest level, raw data is preserved in its most detailed form. This might include individual data points, measurements, or unprocessed text.
2. **Mid-Level Abstraction**: At the mid-level, data is grouped or aggregated to provide a broader overview. For example, hourly data points may be aggregated into daily or weekly averages.
3. **High-Level Abstraction**: At the highest level, data is represented in a condensed and abstracted form, often as summary statistics or key insights. This level provides a big-picture view.

These different levels of abstraction are represented in Figure 6. The first visualisations of the mtcars dataset is a scatter plot that provides detailed information about the relationship between car weight and miles per gallon, with points colored by the number of cylinders. The second is an abstract visualisation using a box-and-whisker plot to provide a high-level summary of the distribution of miles per gallon for different numbers of cylinders. Finally, the third visualisation is a bar plot presenting aggregated information about the average miles per gallon for different numbers of cylinders.

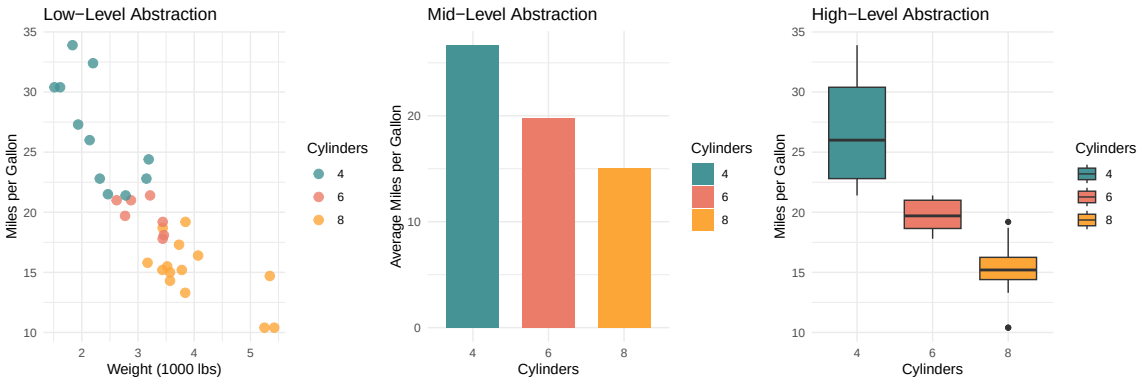

Figure 6: Mtcars dataset visualised on 3 different levels of abstraction

Trade-offs Between Abstraction and Information Loss
While abstraction simplifies complex data, it presents trade-offs. Designers of data visualisation must strike a balance between clarity and detail, generalisation and specificity, and context versus

precision. Abstraction increases clarity but may sacrifice crucial detailed information necessary for some analytical tasks. It offers a more generalised view accessible to a wider audience but might overlook specific nuances essential for experts. While providing valuable context, high-level abstraction may lack the precision required for precise decision-making.

In data visualisation, the art of data abstraction lies in finding the right level of detail that effectively conveys the intended message while minimising the risk of information loss. This balancing act is a critical consideration in the design of informative and meaningful data visualisations.

2.4 Visual Perception and Cognition

In this first section, human visual perception is explored, along with the application of cognitive psychology principles in data visualisation.

Human Visual Perception: Decoding Visual Information
Human visual perception profoundly influences our understanding of the surrounding world. When applied to data visualisation, it shapes the way that individuals engage with and derive meaning from visual data representations.

Significant aspects of human visual perception within data visualisation encompass **pattern recognition**, adept at identifying trends, outliers, and relationships in data representations. Additionally, **perceptual grouping**, where visually similar elements are grouped together, influences the interpretation of data clusters and shapes. Moreover, the **hierarchy of perception** dictates that certain visual attributes are processed more swiftly and effectively than others, such as color being processed faster than text, influencing the viewer's attention hierarchy [4].

The Gestalt Principles
Furthermore, the Gestalt Principles play an important role in the realm of visual perception and design [16]. Key Gestalt principles crucial in shaping visual information perception include proximity, which groups related elements, **similarity** that links similar attributes, **continuity** aiding trend representation, **closure** for implying connections, and **symmetry** for balance and aesthetics in visualisations [20].

By harnessing the principles of human visual perception, applying insights from cognitive psychology, and leveraging pre-attentive attributes, data visualisation designers can create visualisations that are not only aesthetically pleasing but also cognitively efficient.

2.5 Colour Theory in Data Visualisation

In this section, the significance of colour in data visualisation, the principles of colour perception and encoding, and the importance of avoiding misleading visualisations through thoughtful colour choices are explored.

The Importance of Colour in Conveying Information
Colour significantly enhances the impact and comprehension of data visualisations. It serves multiple purposes: distinguishing data points, emphasising trends, and offering contextual information. It is utilised to encode categorical data, differentiating between various groups with distinct colours,

and to represent quantitative data by using colour intensity or gradients to portray values or magnitudes. Additionally, color is instrumental in adding context to visualisations through background elements, labels, or annotations, imparting meaning to the data [12].

Colour Perception and Colour Encoding in Visualisations
Understanding color perception in data visualisation is crucial. Key principles involve considering color discrimination, ensuring accessibility for individuals with color vision deficiencies, as is illustrated by Figure 7. Careful selection of color schemes aligned with the intended message is essential — for instance, using warm colours like red and orange to indicate caution or warmth, and cool colours like blue and green to convey calmness or coldness. Additionally, attention should be paid to how colours interact when combined; certain combinations might create visual vibrations or impact text legibility.

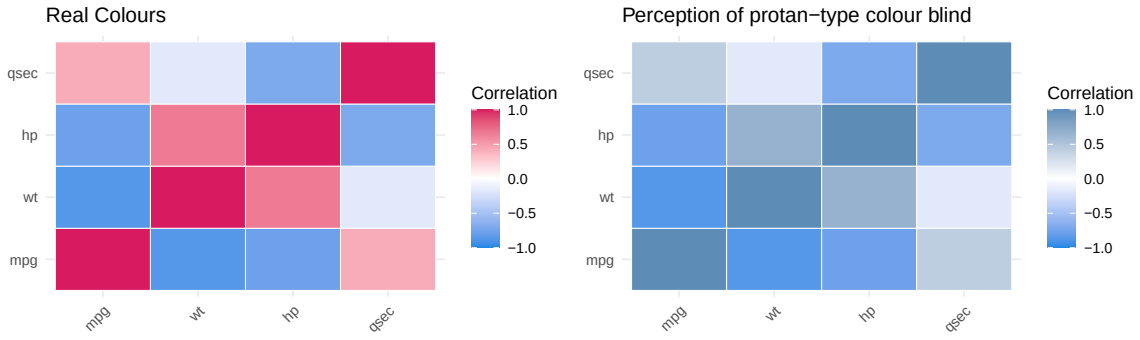

Figure 7: Colour perception of a heatmap for by a colour blind person

Avoiding Misleading Visualisations Due to Colour Choices
Misleading visualisations often stem from inappropriate or deceptive use of colour, requiring precautions to prevent such occurrences. First, maintaining consistency in colour usage throughout the visualisation is essential. Employing a uniform colour scheme for similar data categories or elements helps establish coherence and understanding. Furthermore, it's crucial to avoid colour choices that could distort or exaggerate the data. Overly intense or contrasting colours might mislead interpretations, emphasising the necessity for judicious colour selection.

Additionally, providing a clear and concise legend becomes imperative to explain the meaning of colours, especially when dealing with complex or unfamiliar colour schemes. A comprehensive legend helps viewers decipher the represented data accurately.

2.6 Cognitive Load and Visual Complexity

In data visualisation, achieving a balance between complexity and cognitive load is crucial. Cognitive load significantly influences how viewers engage with and comprehend presented data. Finding a balance is crucial to effectively convey information without overwhelming the viewer's cognitive capacity. This section explores the concept of cognitive load in visualisations, strategies to reduce cognitive load while maintaining complexity, and techniques to combat information overload through

simplification.

2.6.1 Strategies to Reduce Cognitive Load While Maintaining Complexity

To reduce cognitive load while maintaining complexity in data visualisation, several strategies can be employed. Firstly, establishing a clear **visual hierarchy** using size, color, and contrast helps direct attention to crucial elements. Additionally, simplifying **labels and text** by avoiding unnecessary complexity ensures information is clear and easily digestible.

Furthermore, employing **interactive features** like tooltips and drill-down functionality assists in providing additional information when required, reducing the density of static visualisations. A final approach involves the use of **progressive disclosure**, presenting complex information gradually, beginning with an overview and allowing users to explore details as needed.

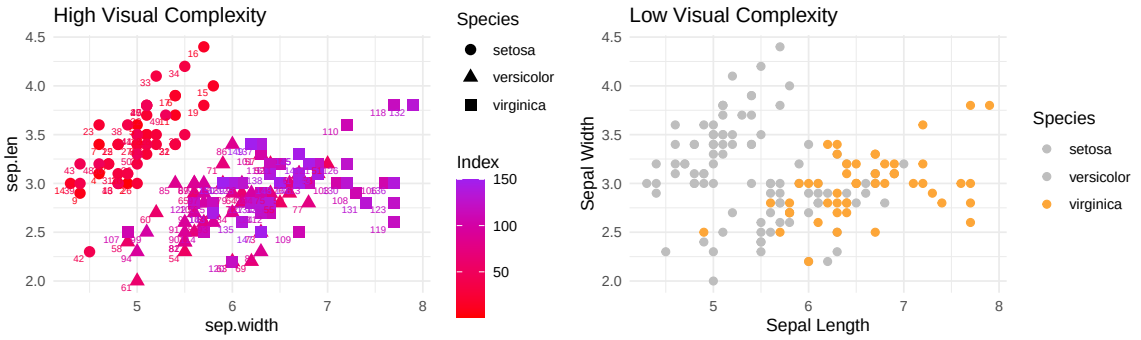

Figure 8: High versus low cognitive load demand through the reduction of visual complexity

2.6.2 Information Overload and Simplification Techniques

Addressing information overload in visualisations necessitates the strategic application of simplification techniques. Filtering enables focused data selection, while data reduction aggregates information to highlight overarching trends. Storyboarding structures data presentation, aiding in contextual comprehension, and prioritisation ensures critical information is prominently displayed, elevating the visualisation's clarity and impact. These strategies collectively combat overwhelming data or excessive visual elements, enhancing comprehension and the effective communication of insights to viewers.

3 Univariate Data Visualisation Methods

This chapter, initiates the discussion about data visualisation methods. This chapter particularly introduces techniques for graphically representing for datasets of the simplest nature - that is, single-variable datasets. It focuses on histograms, bar charts, Kernel Density Estimate, ROC Curves, and Time Series Analysis.

3.1 Histograms

A histogram is a graphical representation of the distribution of a dataset. It consists of bars, each representing a range of data values, and whose height corresponds to the frequency or count of data points within that range.In this way, histograms facilitate the visualisation of differences in frequency or quantity among groups.

On a Cartesian coordinate system, the x-axis shows the endpoints of each group, and the y-axis represents frequency. The height of each rectangle indicates the corresponding frequency, making it a frequency distribution histogram. In order to determine the quantity of each group in the histogram, a multiplication of the frequency by the group interval is necessary. Since every histogram has a fixed group interval, if we use the y-axis to directly show quantity and each rectangle's height indicates the number of data points, we can both retain the distribution and simultaneously see the number in each group at a glance. All examples in this text use the non-standard histogram depiction with the y-axis denoting quantity.

Histograms in Practice
The R language uses the `hist` function to create histograms. This function takes vectors as input, along with other parameters to plot a histogram.

Let $X_1, \ldots, X_n$ be independent and identically distributed random variables with finite mean μ and variance σ^2. Then for any $\epsilon > 0$,

$$P\left(\left|\frac{1}{n}\sum_{i=1}^{n} X_i - \mu\right| < \epsilon\right) \to 1$$

as $n \to \infty$.

Histograms can be also used to visualise the convergence of sample means to the population mean. Let $X_1, \ldots, X_n$ be independent and identically distributed random variables with finite mean μ and variance σ^2. Define the sample mean $\bar{X} = \frac{1}{n}\sum_{i=1}^{n} X_i$. Then, as n becomes large, the variable

$$Z = \frac{\bar{X} - \mu}{\sigma/\sqrt{n}}$$

approaches a standard normal distribution.

By plotting histograms of a large number of sample means, one can observe the shape gradually resembling a normal distribution.

Consider Figure **??** with created on ggplot2 with the use of to the `geom_histogram` function and iris dataset.

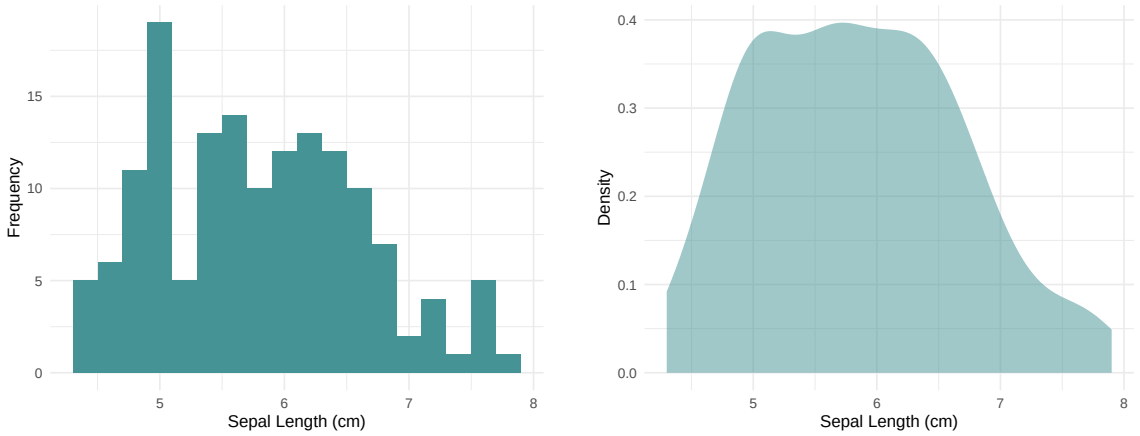

Figure 9: Histogram and Kernel Density Estimation of Sepal Length in Iris Dataset

3.2 Bar Charts

Bar charts are a crucial tool in data presentation, arranging data into vertical or horizontal bars. The varying lengths of these bars directly correspond to the magnitude of the information they represent. Bar charts excel in comparing classified data, especially when values are closely aligned. This stems from the nature of human perception, as our visual acuity for height surpasses that of other visual elements like area or angle.

Bar charts represent a versatile tool for data visualisation, frequently employed to compare distinct categories. The **vertical bar chart**, commonly recognised, exhibits categories along the X-axis and their frequencies or counts along the Y-axis. **Horizontal bar charts**, rotated 90 degrees, prove beneficial for extended category names or numerous categories, displaying categories on the Y-axis and frequencies on the X-axis. **Multi-set or grouped bar charts** facilitate side-by-side comparisons of sub- groups within categories, available in both vertical and horizontal orientations. **Stacked bar charts** illustrate classes of values subdivided into sub-classes, often differentiated by colour, where each segment's size signifies its frequency or count, and the total bar length reflects the cumulative total.

Bar Charts in Practice
The two bar charts on in Figure 10 illustrate the impact of varying vitamin dosages on tooth growth, further categorised by supplement type. On the X-axis, three distinct levels of vitamin dosage are presented, while the Y-axis indicates the average lenth of tooth for each dosage.

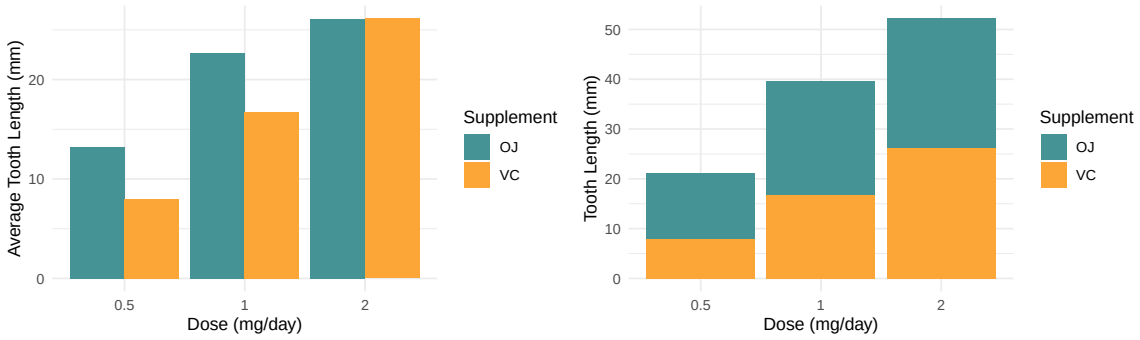

Figure 10: Visualising tooth growth by gosage: A comparison of grouped and stacked bar chart techniques

From the grouped bar, chart found on the left of Figure 10, due to the side-by-side positioning of the bars, it is easy to note that tooth growth varies not only with the dosage but also with the supplement type. In contrast, the stacked bar graph on the right of Figure 10 facilitates the understanding of the combined effects of the two supplements at each dosage level. However, compared to the latter, it becomes more challenging to differentiate the individual contribution of each supplement.

3.3 Kernel Density Estimation

Kernel Density Estimation (KDE) is an very useful tool in statistics. Instead of discrete histograms, it helps create a smooth curve from values in a dataset. KDE is used to infer the distribution of a population based on a limited sample. Thus, the result of the kernel density estimation is an estimate of the sample's probability density function. Based on this estimated probability density function, one can ascertain certain characteristics of the data distribution, such as the regions where data is concentrated.

The KDE algorithm takes a parameter called bandwidth, that affects how smooth the resulting curve is. Changing the bandwidth changes the shape of the kernel: a lower bandwidth implies only points very close to the chosen position are given any weight, which leads to the estimate looking squiggly. In contrast, a higher bandwidth means having a shallow kernel, where distant points can contribute. Thus, leading to a smoother curve.

We can express KDE as follows, where the K represent the kernel function:

$$\hat{f}(x) = \sum_{\text{observations}} K\left(\frac{x - \text{observation}}{\text{bandwidth}}\right).$$

In KDE, the variable x represents the point of density estimation. The total number of data points, denoted as n, constitutes the sample size, and all these points are used for estimating the dataset's probability density function. The bandwidth (h) is a crucial parameter that controls the smoothness of the estimation, with smaller bandwidths leading to closer fits to data and larger ones smoothing out details. The kernel function (K), such as the Gaussian or triangular kernel, assigns mass around each data point, influencing the density estimate, while x_i represents the individual sample points that collectively contribute to the estimated density function, guided by the kernel function and the

bandwidth[22].

$$\hat{f}(x) = \frac{1}{nh} \sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right).$$

The kernel function $K(u)$ is a normalised non-negative function that satisfies:

$$\int K(u)\, du = 1.$$

In Figure 9, the kernel density estimation of the dataset is presented. The distribution of sepal lengths is depicted by the kernel density curve. The peaks observed in the curve correspond to the primary concentration trends of sepal length within the data. A unimodal curve signifies a concentration of sepal lengths for most irises in that specific region. Conversely, a bimodal or multimodal curve suggests the existence of multiple concentration areas.

3.4 ROC Curve

The Receiver Operating Characteristic (ROC) graph is a technique for assessing the performance of classification models, particularly useful for evaluating diagnostic tests and binary classifiers. It displays the performance of a binary classification system as its discrimination threshold is varied.

ROC graph plots two parameters[9]: True Positive Rate (TPR, also known as Sensitivity, is the proportion of positive instances correctly identified) and False Positive Rate (FPR, 1 - Specificity, is the proportion of negative instances incorrectly identified as positive). Also, the Area Under the Curve (AUC) provides a single measure of overall performance of a classifier. The larger the area under the curve, the better the classifier. An AUC of 0.5 suggests no discriminative ability, while an AUC of 1.0 represents perfect classification.

ROC analysis in COVID-19 test
The COVID-19 pandemic underscores the need for accurate diagnostic tests to differentiate between positive and negative cases[10]. An effective mean for evaluating accuracy of various diagnostic tests is the ROC analysis. ROC analysis comprehensively examines test accuracy by integrating TPR, FPR, and AUC. These metrics are calculated by comparing test outcomes against a recognized gold standard, thereby determining the true disease status. This approach is vital for understanding the efficacy of COVID-19 diagnostic tools and guiding public health decisions.

The TPR, or sensitivity, measures the proportion of actual positives correctly identified, reflecting the test's ability to detect COVID-19 cases. The FPR, conversely, indicates the rate of false alarms, where non-infected individuals are wrongly identified as infected. An ideal test minimizes FPR, avoiding unnecessary treatments or quarantine. The ROC curve visualizes the trade-off between TPR and FPR at various thresholds. A curve arching towards the upper left indicates high sensitivity and low FPR — the hallmarks of a reliable test. The AUC consolidates the ROC performance into a single value: 0.5 denotes no better accuracy than random chance, while 1.0 represents perfect accuracy. A higher AUC indicates a better overall ability of the test.

Here, the simulated data will be used to demonstrate the ROC analysis in COVID-19 test. Score_Test1 and Score_Test2 are the simulated test scores from two different diagnostic tests for COVID-19. These scores are continuous variables that typically would represent the likelihood of a positive diagnosis. The scores might come from a lab test result, such as a PCR test or a rapid antigen test,

where higher scores indicate a higher likelihood of infection. The Condition column indicates the true condition of the patient, with 1 representing a positive COVID-19 case and 0 representing a negative case.

```
# Plot of ROC curves
plot(roc_test1, col="#459395", lwd=2)
lines(roc_test2, col="#FDA638", lwd=2, lty=2)
legend("bottomright", legend=c("Diagnostic Test 1", "Diagnostic Test 2"),
       col=c("#459395", "#FDA638"), lwd=2, lty=c(1, 2), cex=0.5)
# Add the perfect lines
abline(h=1, lty=2, col="red")
abline(v=1, lty=2, col="red") # Perfect Diagnostic Test (AUC = 1)
```

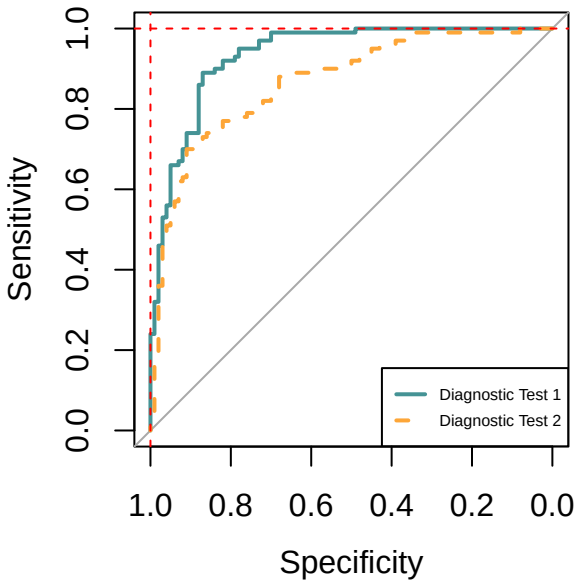

Figure 11: ROC Curves for Two Diagnostic Tests

From the plot 11, the Diagnostic Test 1 has higher AUC. This test is likely to be more accurate in diagnosing COVID-19. It suggests that the test has a higher combined sensitivity and specificity, meaning it can identify positive cases more correctly and has fewer false alarms. Also, the Diagnostic Test 2 has a lower AUC than Test 1, This test is less accurate but still better than random guessing. It may miss more true cases (lower sensitivity) or incorrectly identify healthy individuals as having COVID-19 (higher false positive rate).

In practice, the choice of which test to use may depend on other factors. For instance, if the cost of missing a COVID-19 case is high (for example, in a nursing home setting), a test with higher sensitivity might be preferred. Conversely, if the cost of false positives is high (for example, leading to unnecessary quarantine), a test with higher specificity might be favored.

3.5 Time Series Analysis

Line charts are fundamental tools in data visualisation, particularly useful for displaying time series data. A line chart represents n data points $\{(x_i, y_i)\}_{1 \leq i \leq n}$ on a Cartesian coordinate system, with the x-axis often denoting time intervals or ordered categories and the y-axis representing the measured values.

Basics of Line Charts
In a line chart, consecutive data points are typically connected by straight lines. The line segment between two points (x_i, y_i) and (x_{i+1}, y_{i+1}) can be described by the equation of a line in the slope-intercept form: $y = mx + b$, where m is the slope and b is the y-intercept. Also, a series of linear interpolations between pairs of data points could be used. These interpolations assume that the change between two points is uniform or linear. This linear approach is mathematically represented as:

$$y = y_i + \frac{(y_{i+1} - y_i)}{(x_{i+1} - x_i)} \cdot (x - x_i), \quad x_i \leq x \leq x_{i+1}.$$

This equation highlights that for any point x between x_i and x_{i+1}, the corresponding value of y on the line chart is determined by a linear relation. This method effectively "fills the gaps" between actual observed data points and provides a continuous view of the data.

3.5.1 Time Series

Time series visualisation is particularly suited to line charts. A time series is a collection of observations x_t, where t denotes the time point at which the observation is recorded. An index set T_0 which collects all the time points when observations are available. For instance, we often have $T_0 = \{0, \ldots, n\}$ for $n \in \mathbb{N}$. By plotting these data points over time, line charts help in identifying long-term trends, seasonal patterns, and anomalies.

A time series can be viewed as a realisation of a stochastic process. A stochastic process [2] $X = (X_t)_{t \in T_0}$ is a collection of random variables X_t, where t denotes the time index and T_0 the index set. For a fixed event $\omega \in \Omega$ we obtain the realisation of the stochastic process (sometimes also called a sample path) which is given by $x_t = X_t(\omega)$, $t \in T_0$. In practice, line charts of time series data provide insights into the behavior of such stochastic processes over time.

Time Series Visualisation of Exchange rates
Here, plot daily and 49-day moving average exchange rates of exchange rate data in one figure.

```r
# Plot daily and 49-day moving average exchange rates of CAN, EUR, USD to GBP

# Define the color palette
color_palette <- c("#459395", "#EB7C69", "#FDA638")

# Modify the plots to use the color palette
p1 <- ggplot(plot_dt, aes(x = Date, y = Rate, color = Currency)) +
  geom_line() +
  labs(y = "Exchange Rate to GBP", x = "", color = "Currency") +
  theme_minimal() +
  theme(legend.position = "none") +
  scale_color_manual(values = color_palette)

p2 <- ggplot(plot_data, aes(x = Date, y = Rate, color = Currency)) +
  geom_line() +
  labs(y = "Exchange Rate to GBP", x = "", color = "Currency") +
  theme_minimal() +
  theme(legend.position = "none") +
  scale_color_manual(values = color_palette)

# Extract the legend
p2_legend <- cowplot::get_legend(p2 + theme(legend.position = "right"))

# Combine the plots
combined_plot <- cowplot::plot_grid(p1, p2, rel_widths = c(1, 1), nrow = 2)
cowplot::plot_grid(combined_plot, p2_legend, nrow = 1, rel_widths = c(2, 0.5))
```

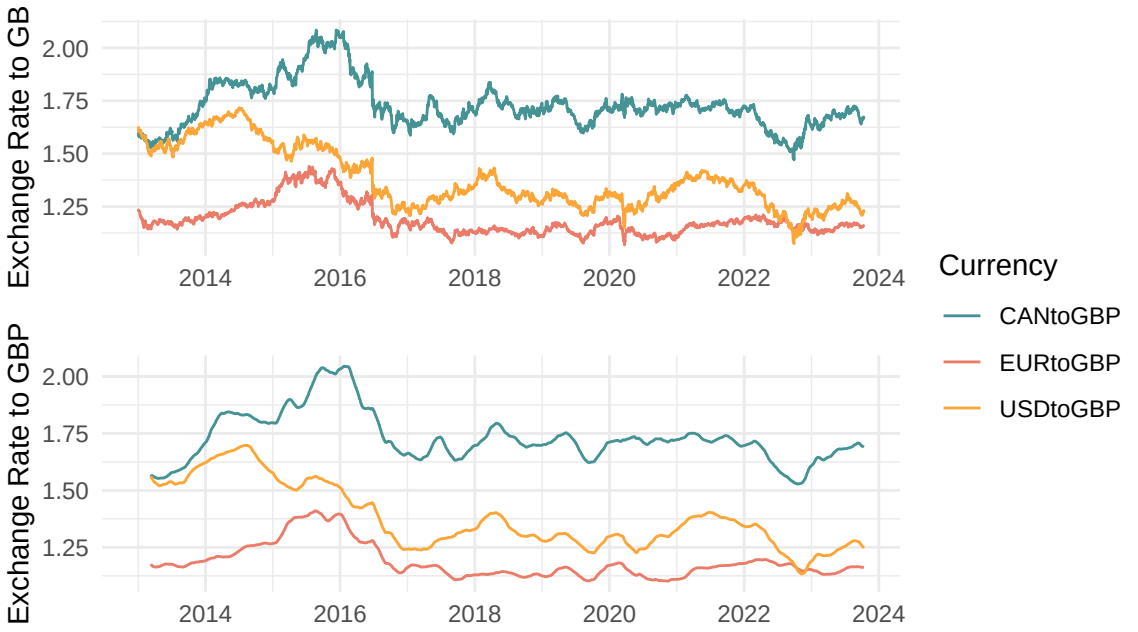

Figure 12: Daily (top) and 49-day moving average (bottom) exchange rates of CAN, EUR, USD to GBP

The first plot in Figure 12 presents a comparative visualisation of daily exchange rates for CAD, EUR, and USD against GBP, offering an overview of their trends and relative performance. This plot enables the identification of overall trends and periods of volatility for each currency pair, allowing for an assessment of their stability and strength relative to GBP. Notice that, there was a simultaneous and significant drop in all three currencies relative to the GBP. The concurrent nature

of these declines across diverse currency pairs suggests that the driving factor is a depreciation of the GBP, rather than independent appreciations of the USD, EUR, and CAN. This notable depreciation of the GBP occurred around 2016, which coincides with the commencement of BREXIT process. Usually, the long term trend attracts financial analyst most. Therefore, filtering fluctuations and anomalies is important. Then, a 49-day moving average (MA49) was employed, shown in the second plot of Figure 12, to elucidate long-term trends while mitigating short-term fluctuations. This is mathematically represented as

$$\mathrm{MA}_{49}(t) = \frac{1}{49} \sum_{k=t-48}^{t} x_k,$$

where x_k denotes the exchange rate on day k.

This method effectively filters out daily noise, allowing a clearer view of overarching trends in currency movements against the GBP. The overlay of these moving averages on the daily exchange rates in visualisations provides both a clear comparative and a quantitative perspective.

Decomposition of Time Series
One of the primary advantages of time series visualisation is the ease with which it allows analysts to identify long-term upward or downward trends in data and patterns that repeat over specific intervals. By decomposing the time series, it would be easy to see those features.

Time series data, X_t, can often be described as a combination of several distinct components: Trend component t_t: The underlying progression in the series, Seasonal component s_t: Periodic fluctuations due to seasonal factor, Residual r_t: The irregular or error component.

The decomposition of a time series can be described in two main models:
Additive Model [2]: In the additive model, the components are added together:

$$X_t = t_t + s_t + r_t.$$

Multiplicative Model [2]: In the multiplicative model, the components are multiplied together:

$$X_t = t_t \times s_t \times r_t \quad \text{or} \quad \log(X_t) = \log(t_t) + \log(s_t) + \log(r_t).$$

In practice, the choice between the additive and multiplicative models often depends on the nature of the time series. If the magnitude of the seasonal fluctuations or the variation around the trend does not vary with the level of the time series, then an additive model is appropriate. If the magnitude of the seasonal fluctuations or the variation around the trend increases or decreases as the time series level changes, then a multiplicative model may be more suitable.

R function `decompose()` is able to decompose the time series by addictive model or multiplicative model. And below is the demonstration of decomposition.

```r
# Plot decomposition of addictive time series model
ggplot(decomposed_df, aes(x = time, y = value)) + geom_line() +
  facet_wrap(~component, scales = "free_y", ncol = 1) + labs(x = "Date",y = "Exchange Rate to GBP") + theme_minimal()
```

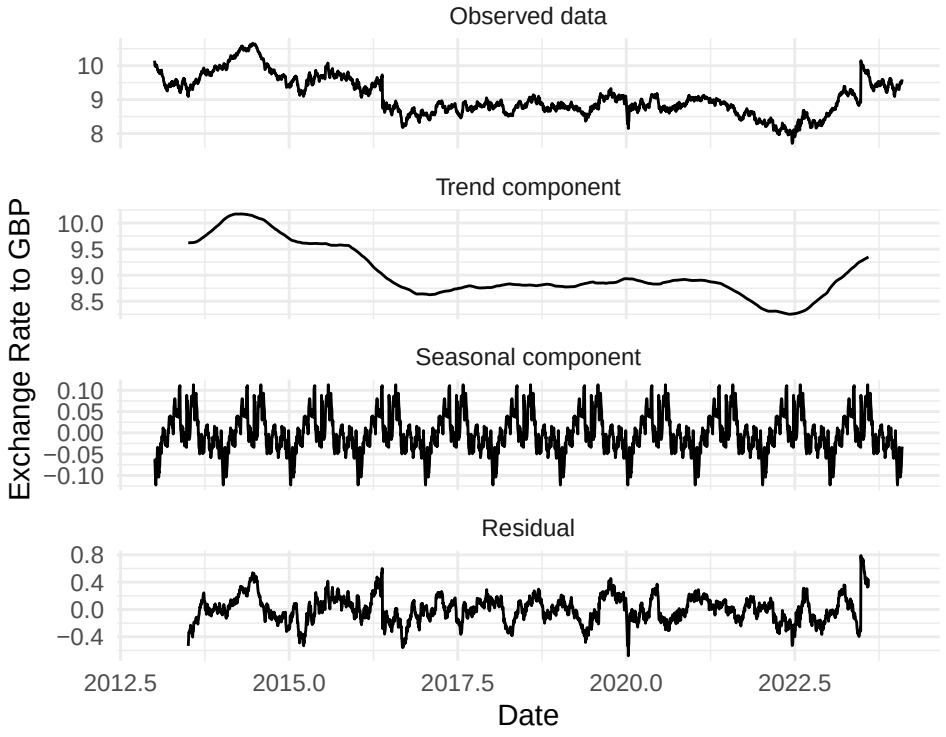

Figure 13: Decomposition of addictive time series model of CNY to GBP exchange rates

From Figure 13, the CNY to GBP exchange rate time series was decomposed into its fundamental components: trend, seasonality, and residual noise by additive model. This additive model, represented mathematically as $X_t = t_t + s_t + r_t$.

As illustrated in Figure 13, the trend component t_t of CNY to GBP exchange rate exhibits a distinct pattern over time: initially, it shows a gradual decrease and reached crest in 2022, followed by a sudden increase afterwards. It coincides with the tax reduction policy issued by UK government in 2022, which leads to a depreciation of GBP. This trend is pivotal for understanding the broader economic relationship between these currencies.

Moreover, the seasonal component s_t of the decomposition highlights cyclical fluctuations, indicative of recurrent patterns within the year. These could be attributed to seasonal economic activities, policy changes, or other cyclical factors influencing the currency market. The clear demarcation of these cyclical trends in the seasonal component helps in isolating such effects from the overarching trend.

Lastly, the residual component r_t encompasses the random, unexplained variations after accounting for the trend and seasonal factors. analysing these residuals is crucial for understanding the unpre-

dictability in the exchange rate and can be pivotal in risk management and forecasting.

Autocorrelation Analysis of CNY to GBP Exchange Rate
Autocorrelation, also referred to as serial correlation, is a crucial concept in time series analysis. It describes the correlation of a time series with its own past and future values. The autocorrelation function (ACF) measures the linear predictability of the series at lag h, which is the time t with its values at a previous time $t-h$. The mathematical fomulation of ACF of time series will be given in the following paragraph.

Suppose we have a time series with observations denoted by $x_1, \ldots, x_n$. Then the sample mean is given by $\bar{x} = \frac{1}{n}\sum_{t=1}^{n} x_t$. And the sample autocovariance function at lag h in days of our time series is

$$\hat{\gamma}(h) := \frac{1}{n}\sum_{t=1}^{n-|h|}(x_{t+|h|} - \bar{x})(x_t - \bar{x}), \quad -n < h < n[2].$$

Hence, the sample autocorrelation function at lag h in days is given by

$$\hat{\rho}(h) := \frac{\hat{\gamma}(h)}{\hat{\gamma}(0)} = \frac{\sum_{t=1}^{n-|h|}(x_{t+|h|} - \bar{x})(x_t - \bar{x})}{\sum_{t=1}^{n}(x_t - \bar{x})^2}, \quad -n < h < n[2].$$

The value of $\hat{\rho}(h)$ lies between -1 and +1. A value close to +1 indicates a strong positive correlation, while a value close to -1 indicates a strong negative correlation. A value near 0 suggests little to no linear correlation. A slow decay in the ACF plot indicates a strong relationship between past and present values, while spikes at specific lags may suggest seasonality. Autocorrelations outside the 95% confidence interval are considered statistically significant.

Next, visualise the ACF for the CNY to GBP exchange rate to understand its time-dependent structure better.

```r
# Plot the Autocorrelation Function (ACF)
acf_data <- acf(MyData$CNYtoGBP, plot = FALSE)
# Plot using base R plotting
plot(acf_data, main="", xlab="Lag h", ylab="ACF")
```

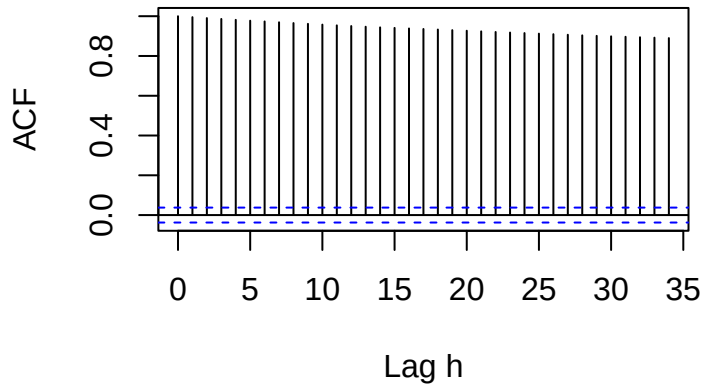

Figure 14: Autocorrelation Function at lag h in days of CNY to GBP Exchange Rate

From Figure 14, the ACF plot for the CNY to GBP exchange rate series reveals that the ACF starts near 1 and decreases gradually. This pattern suggests a strong persistence in the time series, indicating that past values have a significant influence on future values. In time series analysis, such a slow decay in the ACF is indicative of a non-stationary series, where the mean, variance, and autocorrelation structure do not remain constant over time.

This persistent autocorrelation suggests that short-term movements in the CNY to GBP exchange rate are heavily influenced by its recent history. Such a characteristic is crucial for forecasting models, as it implies that recent historical data can be a powerful predictor of near-future trends. Models like ARIMA (Autoregressive Integrated Moving Average), which are well-suited for data with high autocorrelation, may be particularly effective in this context.

4 Bivariate Data Visualisation Methods

4.1 Heatmaps

The heatmap is a data visualisation technique that uses colour coding to represent different intensities. It can be represented as an $m \times n$ matrix $\mathbf{M}$, with m observations for variable 1 and n observations for variable 2:

$$\mathbf{M} = \begin{bmatrix} \mathrm{M}_{11} & \mathrm{M}_{12} & \dots & \mathrm{M}_{1n} \\ \mathrm{M}_{21} & \mathrm{M}_{22} & \dots & \mathrm{M}_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \mathrm{M}_{m1} & \mathrm{M}_{m2} & \dots & \mathrm{M}_{mn} \end{bmatrix},$$

where each entry M_{mn} represent an observation.

In this illustrative example, heatmaps are used to visualise fire occurrences in Brazil. These heatmaps provide a spatially coherent representation, highlighting regions at high risk and seasonal patterns. The data-driven insights could empower policymakers to make informed decisions regarding preventive measures and firefighting strategies.

In Figure 15, it can be observed that significantly higher fire counts are found in certain locations. The presence of two strips with high frequencies of fires are highly unusual. The vertical trend corresponds to the location of BR-230 (Trans-Amazonian Highway) passing through the city of Apuí, State of Amazonas. The horizontal trend corresponds to BR-163 (Brazil highway) passing through Três Pinheiros in Novo Progresso, State of Pará. The western coastal area with a high frequency of fire occurrence corresponds to regions in close proximity to the cities of Vista Alegre do Abunã and Rio Branco. Research has indicated that 95 % of active fires and the most intense ones (FRP $>$ 500 megawatts) occurred at the edges in forests [7].

The seasonal pattern of fire frequency is shown in Figure 16. Observe that more fire occur in the months of August to October compared to the rest of the year.

```
heatmap_plot <- ggplot(pivot_table,
                       aes(x = factor(abb_month, levels = custom_order),
                           y = as.character(year), fill = count)) +
  geom_tile() +
  scale_fill_gradient(low = "#fff7ec", high = "#d7301f") +
  labs(x = " ", y = " ") +
  theme_minimal() +
  theme(axis.text = element_text(size = 9))

print(heatmap_plot)
```

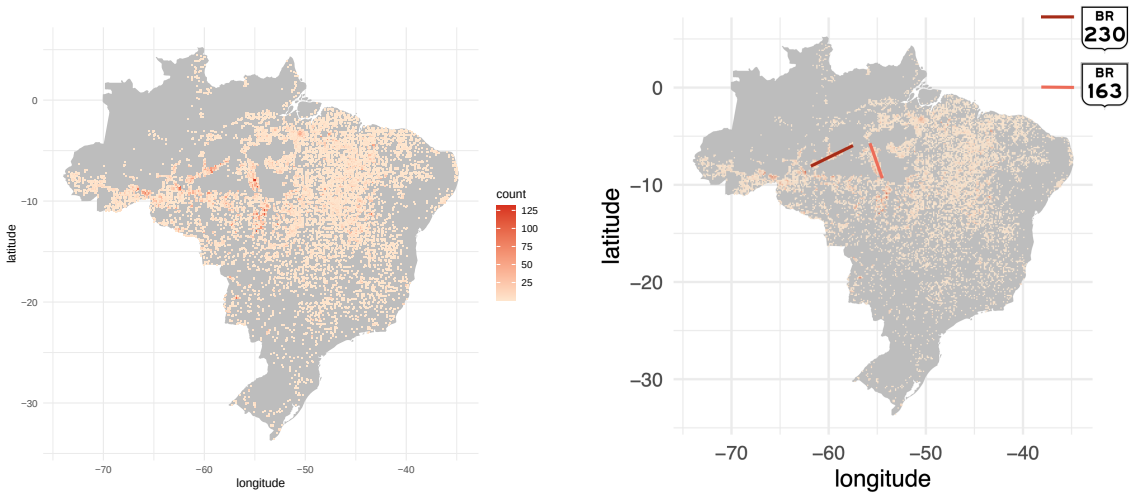

Figure 15: Frequency of fire in Brazil (2022), two strips of high frequencies of fires are highly unusual

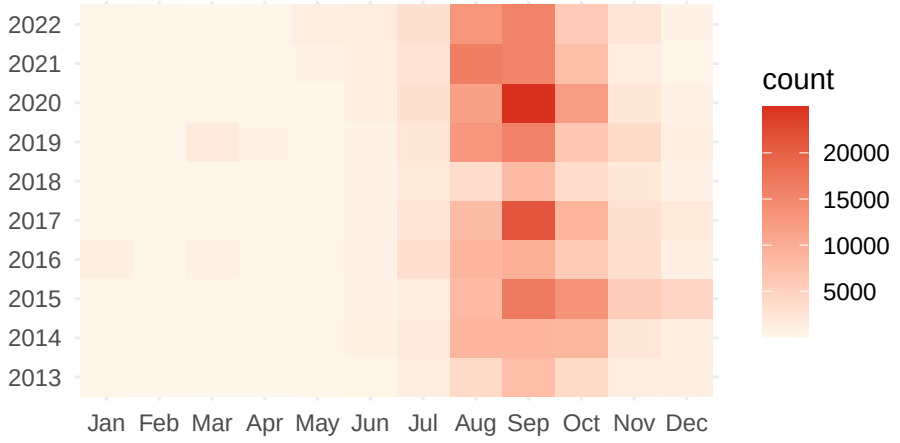

Figure 16: Frequency of fire in Brazil (2013-2022)

4.2 Scatter Plots

A scatter plot is a graphical representation of a set of data points in a two-dimensional coordinate system. Each data point is represented by a dot, and the position of the dot is determined by the values of two variables. Some examples of this type of visualisation can be found in Figure 6, Figure 8, and Figure 17.

In general, the Y-axis denotes the response, or dependent, variable and the X-axis denotes the

explanatory, or independent, variable. Each observation of a dataset is mapped to a dot in the 2-dimentional space. Let (x_i, y_i) represent the coordinates of the i-th data point on the scatter plot produced by mapping a set of data. The scatter plot can be mathematically described as a set of points: $\{(x_1, y_1), \ldots, (x_n, y_n)\}$ where n is the number of observations in the set.

The scatter plots in Figure 17 visualise data from the mtcars dataset. Each of these three graphs represents different information drawn from the dataset. The first plot displays the various car models and their corresponding miles per gallon. The middle plot, once again, shows the car models, but this time their horsepower is represented. Finally, the last scatter plot visualises the relationship between the cars' miles per gallon and their horsepower.

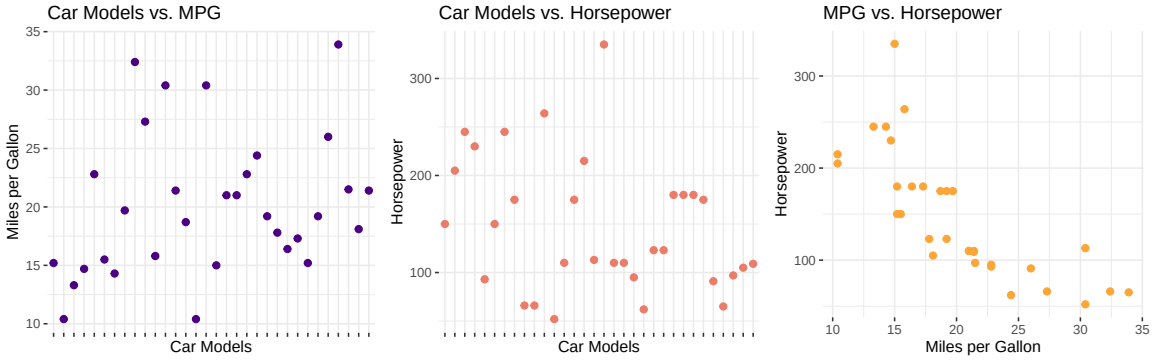

Figure 17: Partial scatter plot deconstruction of the mtcars dataset

4.2.1 Animated Scatter Plots

While static scatter plots are effective in depicting relationships between two variables, animated scatter plot visualisations take this a step further by introducing, for example, a temporal dimension to the data visualisation. Unlike static plots, animated scatter plots enable the depiction of changes in relationships, clusters, or outliers over time. In R, the *gganimate* package, building on the foundation of *ggplot2* package, facilitates the creation of animated plots, including animated scatter plots.

Construction of Animated Scatter Plots

4.3 Bubble Charts

Theory of Bubble Charts Bubble charts are a captivating data visualisation tool that extends beyond the typical two-dimensional scatter plot by introducing an extra dimension. They represent data points as bubbles or circles on a two-dimensional plane, where the size of each bubble encodes a third variable.

The construction of bubble charts, is parallel to that of a scatter plot, but involves the scaling the data values to determine the size of each bubble. While it isn't always the case, the size of these is typically proportional to the variable it represents. The choice of scaling method depends on the

data distribution and the message the chart aims to convey.

Generally, the formula for calculating the bubble radius (R) involves applying the scaling function

$$R = k \cdot V,$$

where R represents the size of the bubble, V the value of the variable being represented, and k a scaling factor to control the bubble size. Selecting an appropriate scaling factor (k) is critical for maintaining the proportionality between the bubble size and the variable being represented.

Hence, bubble chart represents data points in three dimensions: x-coordinate, y-coordinate, and bubble size. Each data point is represented by a set of three values: (x_i, y_i, C_i), where x_i and y_i are the coordinates, and C_i is the equation of the circle for the i-th observation. In the case of a bubble chart, where the radius of the bubble is denoted by $R_i = k \cdot V_i$, we can write the equation for the each circle as:

$$(x - x_i)^2 + (y - y_i)^2 = (k \cdot V)^2,$$

where, x_i and y_i are the coordinates of the center of the circle.

Since each bubble B_i can be mathematically defined by the set of three values: (x_i, y_i, C_i), similarly to the scatter plot, the bubble plot can be mathematically described as a set of points: $\{B_1, \ldots, B_n\}$, that is, $\{(x_1, y_1, C_1), \ldots, (x_n, y_n, C_n)\}$, where n is the number of observations in the set.

Bubble Charts in Practice

The bubble chart shown in Figure 18, as the scatter plots above, visualises data from the mtcars dataset. The single graph deplics the relationship between car models and their fuel efficiency (mpg) while using the size of the bubbles to represent the car's horsepower (hp) and even color-coding the bubbles based on the number of cylinders (cyl).

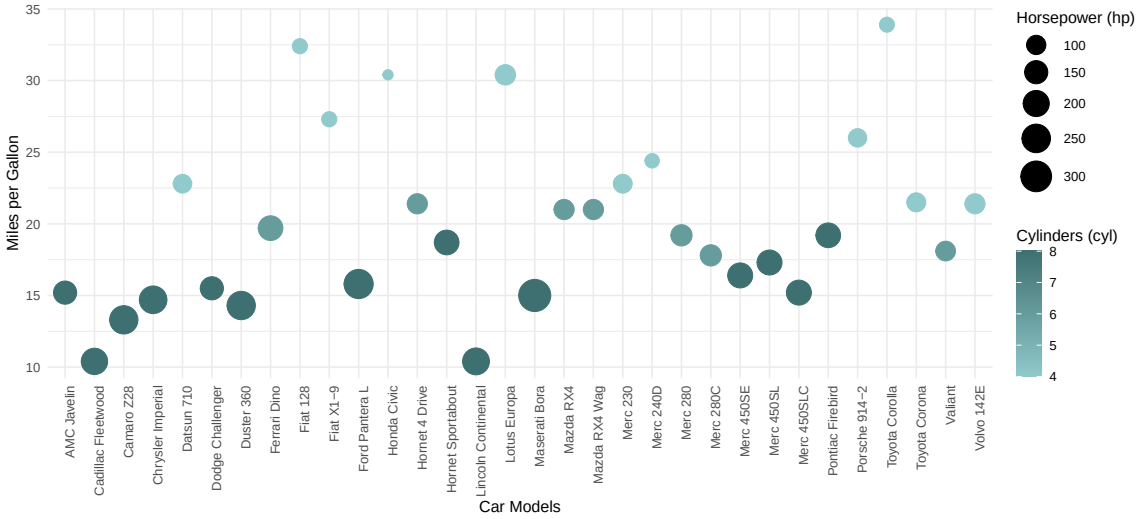

Figure 18: Bubble plot illustrating trends in car types and performance

In Figure 18 a strong correlation between the cars' number of cylinder and their horsepower can be quickly identified. This is due to the fact that as size of the dots (representing horsepower) increases, their colour (representing the number of cylinders) simultaneously becomes darker. Furthermore, a correlation between the miles per gallon consumed by the different types of cars and their number of cylinders and horsepower is easily identifiable in Figure 18. This is shown by the fact that, in the same way that the darker dots cluster towards the bottom of the graph and become lighter as they reach the top of the graph, those with a large diameter also cluster near the lower part and decrease in size as they ascend.

The efficiency of bubble charts is highlighted by the fact that capturing just some of the information provided by this single bubble chat - that is, the information regarding the car models, the miles covered per gallon, and the horsepower of each of these, requires three different scatter plots. These are displayed in Figure 17. The contrast between the simplicity and readability of Figure 18, and the density and complexity of Figure 17 emphasises the advantages of bubble charts in representing datasets with a higher number of variables.

4.4 Simple Linear Regression

Regression models are statistical tools that provide functions to estimate the relationship between the response variable and one or more explanatory variables. Regression analysis is widely adopted by data scientists, who use large datasets to build predictive models for trend forecasting. The following paragraphs will introduce simple linear regression models and demonstrate their usage using the mtcars dataset.

4.4.1 Theory of Simple Linear Regression

Let $\mathbf{x} = (x_1 \dots x_n)^T$ denote n explanatory variables and let $\mathbf{Y} = (Y_1 \dots Y_n)^T$ denote n corresponding response variables.

In a simple linear model, it is assumed that the response variables $Y_1 \dots Y_n$ are uncorrelated with a common variance σ^2, and their expectations are given by $E(Y_i|x_i) = \beta_0 + \beta_1 x_i$. The expectations generated by β_0 and β_1 given x_i can be expressed as:

$$\mathrm{E}(\mathbf{Y}|\mathbf{x}) = \begin{pmatrix} \beta_0 + \beta_1 x_1 \\ \vdots \\ \beta_0 + \beta_1 x_n \end{pmatrix} = \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix} \beta_0 + \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} \beta_1 = \mathbf{1}_n \beta_0 + \mathbf{x}\beta_1, \tag{4.1}$$

where $\mathbf{1}_n$ is an n-vector of 1's.

Given design matrix $\mathbf{X}$ where $X_i = (1, x_i)$ and $\beta = (\beta_0, \beta_1)^T$, then $E(Y_i|x_i) = X_i\beta$. These assumptions can be equivalently written in the vector form:

$$\mathrm{E}(\mathbf{Y}|\mathbf{x}) = \begin{pmatrix} 1 & x_1 \\ \vdots & \vdots \\ 1 & x_n \end{pmatrix} \begin{pmatrix} \beta_0 \\ \beta_1 \end{pmatrix} = \mathbf{X}\beta, \quad \mathrm{var}(\mathbf{Y}|\mathbf{x}) = \begin{pmatrix} \sigma^2 & 0 & \cdots & 0 \\ 0 & \sigma^2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sigma^2 \end{pmatrix} = \sigma^2 \mathbf{I}_n. \tag{4.2}$$

4.4.2 Least Squares Estimation

The residual sum of squares (RSS) is a measure of the goodness of fit in a regression model, where residuals are the differences between the response variables y_i and responses generated by the regression model $\mathrm{E}(\mathbf{Y}_i|\mathbf{x})$. In least squares estimation, the goal is to find values of parameters $\beta = (\beta_0, \beta_1)^T$ to minimise the RSS, denoted by Q:

$$\mathrm{Q} = \sum_{i=1}^{n} [y_i - \mathrm{E}(Y_i|\mathbf{x})]^2 = [\mathbf{y} - \mathrm{E}(\mathbf{Y}|\mathbf{x})]^T [\mathbf{y} - \mathrm{E}(\mathbf{Y}|\mathbf{x})] = [\mathbf{y} - \mathbf{X}\beta]^T [\mathbf{y} - \mathbf{X}\beta], \tag{4.3}$$

where $\mathbf{y}$ is n-vector of response variables and $\mathbf{X}$ is the $n \times 2$ design matrix. The partial derivative of Q with respect to vector β is:

$$\frac{\partial Q}{\partial \beta} = 2(\mathbf{X}^T\mathbf{X}\beta - \mathbf{X}^T\mathbf{y}), \tag{4.4}$$

Equating $\frac{\partial Q}{\partial \beta} = \mathbf{0}$, the vector $\hat{\beta}$, the least squares estimate of β, can be written as:

$$\mathbf{X}^T(\mathbf{y} - \mathbf{X}\hat{\beta}) = \mathbf{0}. \tag{4.5}$$

The least squares estimate of β is given by:

$$\hat{\beta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}. \tag{4.6}$$

Case example: 1970s automobiles
In this section, we will study the performance of 1970s automobiles using the mtcars dataset, employing the method of linear regression. Performance is measured in Miles per Gallon (mpg); the higher the mileage, the more efficient the automobile. We will start with the visualisation of a simple linear regression model, followed by the discussion of linear regression models and the model selection method.

In the preliminary stages of data exploration, calculating the correlation matrix is a crucial step before engaging in regression modeling, as shown in Figure 19. In real-world scenarios, variables are often correlated, and entirely independent relationships are seldom encountered. Therefore, analysing pairwise correlations becomes essential. This helps in understanding multicollinearity issues within the model. Multicollinearity occurs when one covariate within the model can be accurately predicted from another covariate. When this happens, the coefficient estimates of the model can change unpredictably due to minor changes in the data.

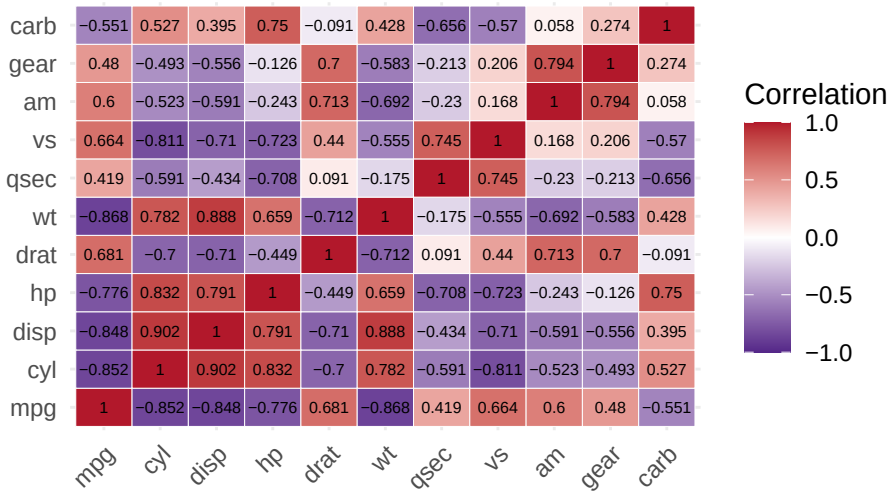

Figure 19: Correlation matrix of all variables in mtcars dataset

For the simple linear regression model, the response variable is Miles per Gallon (mpg), and we select weight (wt) as the explanatory variable. Note that mpg and wt are highly correlated, with a correlation coefficient of -0.868. This suggests that wt may have strong predictive power for mpg. Use the R function `lm()` to calculate the linear regression model, with the summary displayed below.

Observe that the t-test yields a p-value of 1.29×10^{-10}, which is less than 0.001. This indicates that the variable wt holds high statistical significance in this model. For the fitted model, the slope is $\beta_1 = -5.3445$, meaning that for every increase of 1000 lbs, the car efficiency decreases by 5 miles per gallon. The RSS is 3.046. The simple linear regression line is displayed in Figure 20.

```
Modelwt <- lm(formula = mpg ~ wt, data = mtcars)
summary(Modelwt)

##
## Call:
## lm(formula = mpg ~ wt, data = mtcars)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.5432 -2.3647 -0.1252  1.4096  6.8727
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  37.2851     1.8776  19.858  < 2e-16 ***
## wt           -5.3445     0.5591  -9.559 1.29e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.046 on 30 degrees of freedom
## Multiple R-squared:  0.7528,Adjusted R-squared:  0.7446
## F-statistic: 91.38 on 1 and 30 DF,  p-value: 1.294e-10
```

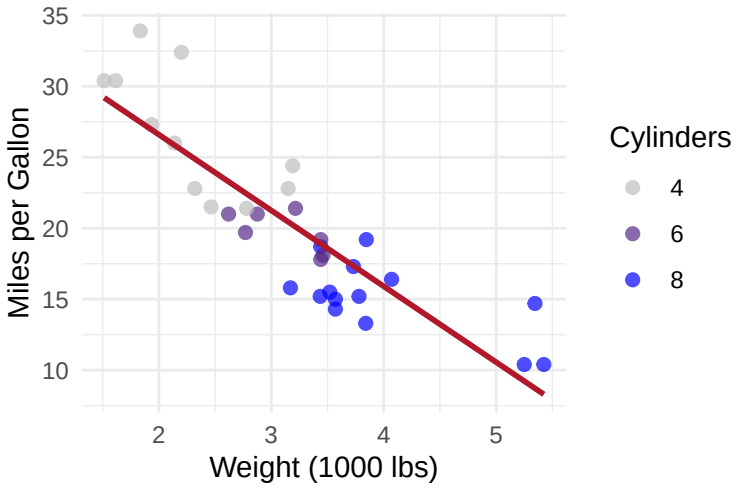

Figure 20: Scatter plot of car weights vs MPG

4.5 LOESS Regression

Locally weighted scatterplot smoothing (LOWESS) is a local regression method for a response variable Y on a single predictor X. LOWESS was proposed by William S. Cleveland in 1979, it is the special case to higher dimention locally estimated scatterplot smoothing (LOESS) regression method. They are non-parametric regression methods that fits a linear regression model for each neighbourhood of data. Unlike GLM, the LOESS model does not provide a global function to fit the data; rather, it fits neighborhoods of the data.

4.5.1 Theory of LOESS regression

Suppose we have n ordered observations $(x_1, y_1) \dots (x_n, y_n)$ with predictors x_i and response variables y_i. Assume a model of the form

$$y_i = g(x_i) + \varepsilon_i,$$

where $g(\cdot)$ is an unknown smooth function and ε_i is i.i.d. Gaussian error terms with mean 0 and variance σ^2.

Let $\Delta_i(x) = |x - x_i|$ represent the horizontal distance between x and x_i. Let $\Delta_{(i)}(x)$ denote the ordered distances to x, arranged from smallest to largest.

For each point (x, y) consider the neighbourhood to fit a line segment around it. The size of the neighbourhood indicates the number of data points used to fit the line segments and is determined by the smoothing parameter α. For $\alpha \leq 1$, each neighbourhood consists of $\frac{n}{\alpha}$ number of points [15]. Each point (x_i, y_i) in the neighbourhood of point (x, y) is assigned with a weight, "importance" of the data point to the line segment [19]. The weight of (x_i, y_i) is denoted as $w_i(x)$, which is defined as:

$$w_i(x) = T(\Delta_i(x); \Delta_{(q)}(x)). \tag{4.7}$$

Tricube weight function $T(u, t)$ is defined as:

$$T(u;t) = \begin{cases} (1-(u/t)^3)^3 & \text{for } 0 \le u < t \\ 0 & \text{for } u \ge t \end{cases}, \tag{4.8}$$

where u is the horizontal distance between neighbourhood point of interest (x_i, y_i) to point (x, y) and t is the maximum distance threshold. For points in the neighbourhood with horizontal distance exceeding threshold t, they are assigned with weight 0 by the tricube weight function.

For $\alpha > 1$, for each point (x, y) the neighbourhood include all points. The maximum distance threshold is assumed to be $\alpha^{\frac{1}{p}}$, for p explanatory variables. The weight is given by:

$$w_i(x) = T(\Delta_i(x); \Delta_{(n)}(x))\alpha. \tag{4.9}$$

The line segment is fitted by weighted least square, and this is the preliminary estimation $\hat{g}(x)$ for every point (x, y).

Both LOWESS and LOESS penalise the outliers in the preliminary estimation by assigning points further away from $\hat{g}(x)$ with less weight than before. The robustness weight (the adjusted weight) is given by:

$$r_i = B(\hat{\varepsilon}_i, 6\mathrm{m}),$$

where bisquare weight function is defined as:

$$B(u;b) = \begin{cases} (1-(u/b)^2)^2 & \text{for } 0 \le |u| < b \\ 0 & \text{for } |u| \ge b \end{cases}, \tag{4.10}$$

and the median absolute residual m is defined as $\mathrm{m} = \mathrm{median}(|\hat{\varepsilon}_i|)$, the residual $\hat{\varepsilon}_i$ is defined as $\hat{\varepsilon}_i = y_i - \hat{y}(x_i)$.

Note the process of weight adjustment using previous estimation is iterated several times until a smooth curve is obtained.

The updated model estimate $\hat{g}(x)$, is computed using the local fitting method, but with the neighborhood weights $w_i(x)$ replaced by $r_i w_i(x)$ [8].

Case example: Taiwan Housing dataset

Property valuation can be modelled as a regression problem, in this analysis, we delve into the pricing of properties based on their age. Using the Taipei Housing dataset, we investigate the relationship between the age of houses (measured in years) and their prices (measured in New Taiwan dollars per unit area). Linear regression model and the LOESS regression model are used, as shown in Figure **??**.

The linear model has a residual standard error of 13.32, while the LOESS model has a residual standard error of 12.16. Cross-validation is a method to compare model goodness of fit. LOESS, with its ability to capture local variations may outperform linear regression, which minimises the least squares estimate $\hat{\beta}$.

```r
ggplot(data = estate, aes(x = house_age, y = price_per_area)) +
  geom_point(size =0.5) +
  theme_minimal()+
  scale_x_continuous(labels = scales::number_format(scale = 1)) +
  scale_y_continuous(labels = scales::number_format(scale = 1)) +
  geom_smooth(method = "loess", se = FALSE, color = "#bd0026", span = 0.5) +
  geom_smooth(method = "lm", se = FALSE, color = "#0868ac", formula = y ~ x) +
  labs(title = "House Age vs. House Price",
       x = "house age (year)",
       y = "price (unit area)") +
  theme_minimal()
```

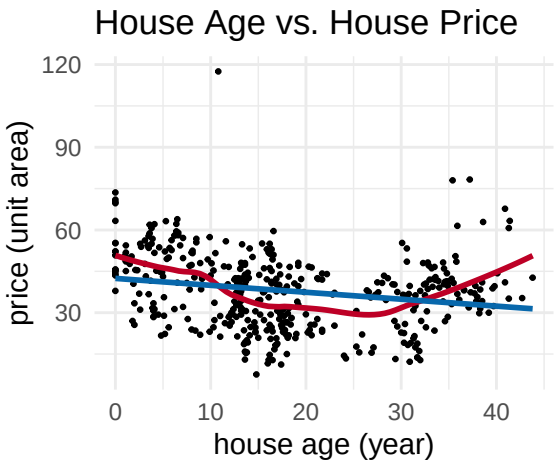

Figure 21: Estate valuation using house age, linear regression and smooth regression methods

5 Visualising Beyond Two Dimensions

5.1 Multiple Linear Regression

Linear Regression Model and Model selection method AIC

Model selection methods, such as the Akaike Information Criterion (AIC), play a crucial role in statistical modeling. AIC provides an estimate of the information lost when a given model is used to represent the process that generated the data, it is calculated as follows:

$$\text{AIC} = -2l(\hat{\boldsymbol{\theta}}) + 2\dim(\boldsymbol{\theta})[24],$$

where $l(\hat{\boldsymbol{\theta}})$ is the log-likelihood function, let $l(\hat{\boldsymbol{\theta}}) = 0$ to find the Maximum Likelihood Estimator $\hat{\boldsymbol{\theta}}$ of a distribution.

The R algorithm `step()`, which minimises the AIC score, aids in selecting the most appropriate regression model from a set of explanatory variables. This algorithm uses backward selection method. Starting with all explanatory variables, the algorithm iteratively excludes one covariate at each step, until statistically significant reduction in AIC is achieved. This algorithm aims to find the best-fitting model by balancing goodness of fit and simplicity. Note that `step` provides comparison between competing models rather than an absolute measure of model quality.

```
ModelAll <- lm(formula = mpg ~ ., data = mtcars)
ModelBest <- step(ModelAll)
```

The procedure for selecting the best linear model for the mtcars dataset is shown in Appendix A, where our best-fitted model is `formula = mpg ~wt + qsec + am`. Given 10 explanatory variables, Miles per Gallon is best predicted by weight (wt), quarter-mile time (qsec), and Transmission (am), where 0 represents automatic and 1 represents manual.

5.2 PCA

Visualizing and analyzing data with many variables can be challenging due to the complexity and high dimensionality. Dimensionality reduction techniques like Principal Component Analysis (PCA) simplify this by transforming the data into a lower-dimensional space (typically 2D or 3D), making it more accessible and manageable. PCA, a linear method, is especially effective for datasets with linear relationships, identifying the principal components that maximize variance and preserve significant features of the original data. This not only aids in visualization but also enhances computational efficiency and helps mitigate the curse of dimensionality.

For a collection of N high-dimensional objects $\mathbf{X_1}, \ldots, \mathbf{X_N}$, where each $\mathbf{X_i}$ is a vector in a D-dimensional space. The entire dataset can be represented as a matrix $X \in \mathbb{R}^{N \times D}$, where each row corresponds to one of the N objects.

Before applying PCA, it is crucial to standardize the dataset because it ensures that each feature contributes equally to the analysis. Without standardization, features with larger scales dominate

the variance, potentially skewing the PCA results. For $1 \leq i \leq N$ and $1 \leq j \leq D$, the standardized value Z_{ij} for each data point X_{ij} is calculated as:

$$Z_{ij} = \frac{X_{ij} - \mu_j}{\sigma_j},$$

where X_{ij} is the original value, μ_j is the mean of feature j, and σ_j is the standard deviation of feature j. The result is a standardized dataset $\mathbf{Z} = \{\mathbf{Z_1}, \ldots, \mathbf{Z_N}\}$ with N objects, each represented in a D-dimensional standardized space. Then, the covariance matrix is $\mathbf{\Sigma}$ for the standardized dataset $\mathbf{Z}$. The covariance matrix is a $D \times D$ matrix, calculated as:

$$\mathbf{\Sigma} = \frac{1}{N-1}\mathbf{Z}^T\mathbf{Z}.$$

Since $\mathbf{Z}$ is standardized, this matrix reflects the covariance between each pair of features. Therefore, the eigen decomposition of the covariance matrix Σ is given by:

$$\mathbf{\Sigma v} = \lambda \mathbf{v},$$

where each eigenvector $\mathbf{v}$ and its corresponding eigenvalue λ satisfy this equation. The eigen decomposition of the covariance matrix $\mathbf{\Sigma}$ reveals the principal components of the data. Each eigenvector represents a direction in the feature space along which the data is variably distributed, and the corresponding eigenvalue indicates the variance of its eigenvector. Importantly, these eigenvectors are orthogonal to each other, underscoring the identification of uncorrelated principal components in the data. Then, order the eigenvectors by their corresponding eigenvalues in descending order and select the top k eigenvectors $\{\mathbf{v_1}, \ldots, \mathbf{v_k}\}$, which correspond to the largest k eigenvalues. These form the columns of the matrix $\mathbf{W} \in \mathbb{R}^{D \times k}$.

Finally, project the standardized data $\mathbf{Z}$ onto the subspace defined by the principal components:

$$\mathbf{Y} = \mathbf{ZW},$$

where $\mathbf{Y} \in \mathbb{R}^{N \times k}$ is the transformed data matrix in the new subspace. Hence, in the reduced space, the variance along each principal component $\mathbf{v_i}$ is maximized, preserving the most significant variances in the data. PCA also attempts to preserve pairwise Euclidean distances between data points in the lower-dimensional representation.

5.3 Biplots

Biplots are designed visualise high-dimensional data by projecting it onto a lower-dimensional space while retaining essential information. They are commonly understood to be a generalisation of the simple two-variable scatterplot.

At their core, biplots offer a graphical representation of a standardised data matrix, whose rows n are the samples, and whose columns p are the variables, by projecting these onto a two-dimensional plane. To do this, mathematical computations derived from techniques like Singular Value Decomposition (SVD) or Principal Component Analysis (PCA) are required. SVD is particularly essential in the generation of biplots as it breaks down the data matrix into three constituent matrices, allowing for the extraction of singular values and singular vectors — which form the basis for plotting the data in the reduced-dimensional space.

5.3.1 Construction of Biplots

As we will see in detail subsequently, the steps to generate a biplot involve:

- **Data Standardisation**: Standardising the data by centering and scaling it to ensure each variable contributes equally.
- **Calculating Singular Value Decomposition (SVD)**: Applying SVD to the standardised data matrix to obtain the singular values and vectors.
- **Projection**: Projecting both the samples and variables onto a two-dimensional space using the singular vectors obtained from SVD.

Data Standardisation
Depending on the initial data matrix $\mathbf{M}$, various transformations may be necessary to arrived at the standardised data matrix $\mathbf{H}$. These may include functional transformations like logarithmic, sine, power, etc., as well as matrix transformations such as observed/expected frequency ratio, odds ratio, log ratio, etc. Additionally, transformations may involve applying weights to rows and/or columns, (weighted) centering of rows and columns, double-centering, and normalization of rows and columns.

Singular Value Decomposition (SVD)
(Weighted) SVD is performed via the eigendecomposition of a cross-product matrix whose order depends on the number of columns of $\mathbf{H}$, since the number of columns is usually less than the number of rows.

The most general situation is that in which there are weights r for the rows and weights c for the columns, with associated diagonal matrices $\mathbf{D}_r$ and $\mathbf{D}_c$. Defaults are $\mathbf{D}_r = (1/n)\mathbf{I}$ (each of the n rows is weighted equally by $1/n$) and $\mathbf{D}_c = \mathbf{I}$.

The weighted SVD is obtained by decomposing the matrix

$$\mathbf{D}_r^{1/2}\mathbf{H}\mathbf{D}_c^{1/2} = \mathbf{U}\Gamma\mathbf{V}^{\mathbf{T}}$$

, where Γ is the diagonal matrix of eigenvalues. Alternatively, if proceeding via the weighted eigendecomposition,

$$\mathbf{D}_c^{1/2}\mathbf{H}^{\mathbf{T}}\mathbf{D}_r\mathbf{H}\mathbf{D}_c^{1/2} = \mathbf{V}\Gamma^2\mathbf{D}^{\mathbf{T}}$$

where Γ_2 is the diagonal matrix of eigenvalues, followed by the transformation to the left vectors:

$$\mathbf{U} = \mathbf{D}_r^{1/2}\mathbf{H}\mathbf{D}_c^{1/2}\mathbf{V}\Gamma^{-1}$$

.

Projection: Computation of Coordinates
There are three types of coordinates. Firstly, there's the standard coordinates whose rows are given by $\mathbf{F}_s = \mathbf{D}_r^{-1/2}\mathbf{U}$, and the columns by $\mathbf{G}_s = \mathbf{D}_c^{-1/2}\mathbf{V}$. There's also the principal coordinates which have rows given by $\mathbf{F}_p = \mathbf{D}_s\Gamma$ and columns given by $\mathbf{G}_p = \mathbf{G}_s\Gamma$. Finally, there's the "canonical" coordinates which have rows given by $\mathbf{F}_c = \mathbf{F}_s\Gamma^{1/2}$ and columns $\mathbf{G}_c = \mathbf{G}_s\Gamma^{1/2}$.

PCA-based Biplots

In the context of PCA-based biplots, the primary steps involve determining principal components from the covariance or correlation matrix of the data. The first two principal components, capturing the most significant variation, form the basis of the biplot. In this graphical representation, observations are depicted as points, while variables are represented as vectors emanating from the origin. The angles and lengths of these vectors signify relationships between variables and their contributions to the principal components, while the proximity of points reflects similarity or dissimilarity between observations.

Mathematically, the coordinates of observations on the first two principal components can be calculated using formulas derived from PCA:

$$x_{i1} = s_{i1} \cdot u_1$$

$$x_{i2} = s_{i2} \cdot u_2,$$

where x_{i1} and x_{i2} are the coordinates of observation i on the first and second principal components respectively, s_{i1} and s_{i2} are the scores of observation i on these components, and u_1 and u_2 are the unit eigenvectors corresponding to the first and second principal components.

Similarly, the coordinates of the variable vectors are computed using the following formulas:

$$v_{j1} = \sqrt{\lambda_1} \cdot c_{j1}$$

$$v_{j2} = \sqrt{\lambda_2} \cdot c_{j2}.$$

Here, v_{j1} and v_{j2} represent the coordinates of variable j on the first and second principal components respectively, λ_1 and λ_2 are the eigenvalues corresponding to the first and second principal components, and c_{j1} and c_{j2} are the loadings of variable j on these components.

```r
data(mtcars)
#Generation of a PCA Biplot

# Standardise the data
scaled_data <- scale(mtcars)

# Perform Principal Component Analysis (PCA)
pca_result <- prcomp(scaled_data, scale. = TRUE)

# Create a biplot
biplot(pca_result, cex = 0.7)
```

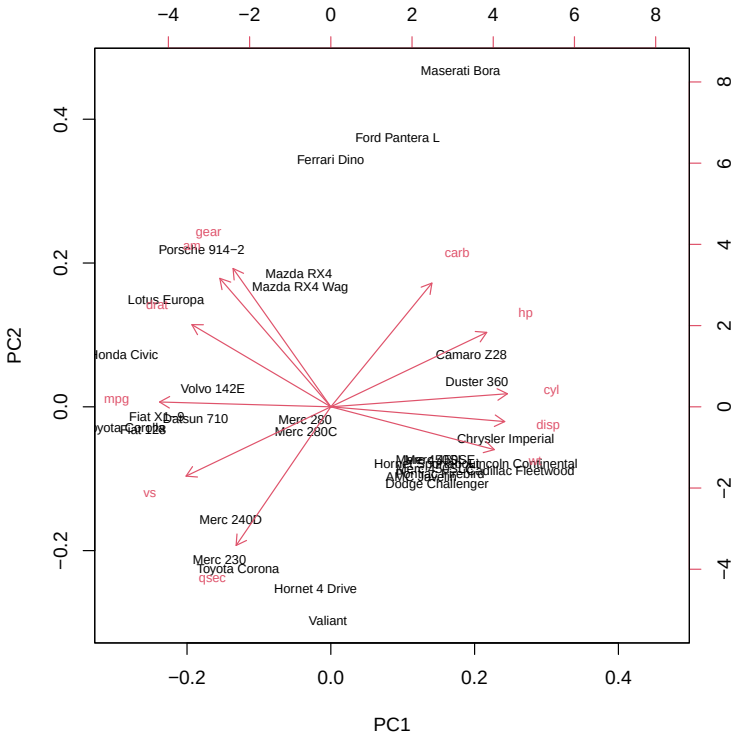

Figure 22: PCA Biplot of mtcars Dataset: Principal Component Analysis

5.4 t-SNE

https://jmlr.org/papers/volume9/vandermaaten08a/vandermaaten08a.pdf
Since PCA is a linear method and may not perform well with non-linear data structures, often missing complex non-linear relationships. Although effective in capturing the global structure of data, PCA focuses on maintaining large pairwise distances to maximize variance and overlook important local patterns and structures.

Unlike PCA, t-Distributed Stochastic Neighbor Embedding (t-SNE) is a nonlinear technique for embedding high-dimensional data for visualization in a low-dimensional space of two or three dimensions, focusing on preserving the small pairwise similarities between data points thus retain the local structure of the dataset in a lower-dimensional space. It starts by converting high-dimensional Euclidean distances into probabilities that reflect the similarity between points, then maps these points to a lower-dimensional space in a way that tries to preserve these similarities.

First, for each pair of high-dimensional points x_i and x_j, the similarity between data points is quantified using conditional probabilities $p_{j|i}$, which is calculated using a Gaussian distribution

centered at x_i:

$$p_{j|i} = \frac{\exp(-||\mathbf{x}_i - \mathbf{x}_j||^2/2\sigma_i^2)}{\sum_{k \neq i} \exp(-||\mathbf{x}_i - \mathbf{x}_k||^2/2\sigma_i^2)} \tag{5.1}$$

Then, the joint probability distribution P in the high-dimensional space is symmetrized as follows:

$$p_{ij} = \frac{p_{j|i} + p_{i|j}}{2N} \tag{5.2}$$

In the low-dimensional space, t-SNE alleviates the crowding problem, which is a common issue in methods like PCA (Principal Component Analysis). It does so by using a t-distribution in the low-dimensional space, which has heavier tails than a Gaussian distribution. t-SNE calculates similar probabilities using a Student's t-distribution:

$$q_{ij} = \frac{(1 + ||\mathbf{y}_i - \mathbf{y}_j||^2)^{-1}}{\sum_{k \neq l}(1 + ||\mathbf{y}_k - \mathbf{y}_l||^2)^{-1}} \tag{5.3}$$

The objective is to minimize the Kullback-Leibler divergence between the joint probability distributions P and Q:

$$C = \mathrm{KL}(P||Q) = \sum_{i \neq j} p_{ij} \log \frac{p_{ij}}{q_{ij}} \tag{5.4}$$

This process is achieved through gradient descent. The gradient of symmetric SNE is fairly similar to that of asymmetric SNE, and is given

$$\partial C = \mathrm{KL}(P||Q) = \sum_{i \neq j} p_{ij} \log \frac{p_{ij}}{q_{ij}} \tag{5.5}$$

Perplexity influences the variance of the Gaussians in the high-dimensional space and is a crucial hyperparameter in t-SNE.

R provides us with a dedicated package for t-SNE analysis, *Rtsne*, which allows us to conduct the analysis conveniently using the *Rtsne* function. In the graph presented below, we observe a t-SNE visualization of the penguin dataset.

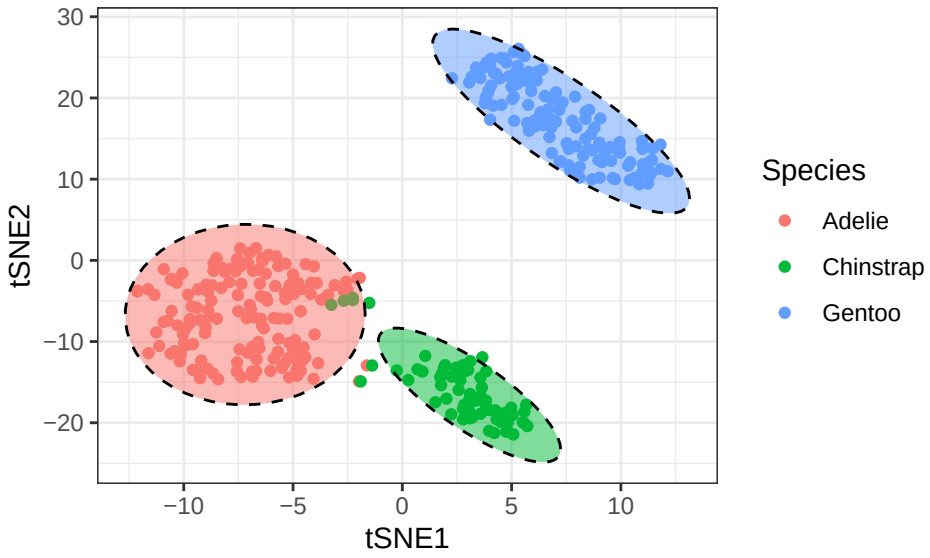

Figure 23: t-SNE of Penguins Dataset

The provided plot in Figure 23 is a t-SNE visualization of the Penguins dataset, rendered using the *ggplot*2 package in R. In this case, using the T-sne algorithm helps us to reduce the multidimensional numerical data of the penguin dataset to a 2-dimensional space,and we also using scatter plots and elliptical areas to visualize the distribution of different penguin. The data points are color-coded to distinguish between three penguin species: Adelie in red, Chinstrap in green, and Gentoo in blue. This color coding aids in the visual differentiation of the species clusters. The axes are labeled $tSNE1$ and $tSNE2$, corresponding to the dimensions reduced by the t-SNE algorithm. Dashed ellipses overlaid on the scatter plot demarcate the clusters of each species, providing a visual guide to the density and separation of the species within the transformed feature space. The plot effectively uses the t-SNE technique to illustrate the grouping of species, highlighting the algorithm's utility in discerning inherent data patterns in a lower-dimensional representation.

While t-SNE is an exceptionally potent tool for data visualization, it does come with its own set of constraints. Firstly, it has a significant memory footprint and can be time-consuming to run, which may pose challenges when dealing with large datasets. Secondly, it is tailored specifically for visualization, which constrains the embedding space to two or three dimensions. This limitation means that t-SNE is optimized for human interpretability rather than for capturing higher-dimensional relationships. Additionally, it requires experimentation with different initializations to mitigate the risk of local suboptimal solutions affecting the results. Therefore, it's crucial to carefully consider and select the most suitable method based on the specific requirements and nature of the data at hand.

6 Index

- **Accuracy**: Emphasising faithful reflection of data to reduce distortion or misinterpretation.
- **Clarity**: Ensuring visuals are easily understood without unnecessary complexity.
- **Cognitive Load**: The mental effort required to process information presented visually.
- **Cognitive psychology**: Understanding how mental processes affect perception and understanding of visual data.
- **Color Theory**: Principles guiding the strategic use of color in visualisations for clarity and impact.
- **Colour encoding**: Utilising colors to represent data categories or values in visualisations.
- **Colour perception**: Understanding how humans perceive and interpret colors in visualisations.
- **Consistency**: Maintaining uniform use of visual elements throughout a visualisation.
- **Data abstraction**: Simplifying and structuring raw data into comprehensible visual forms.
- **Gestalt Principles**: Rules affecting how visual elements are grouped and interpreted in perception.
- **Hierarchies of abstraction**: Representing data at various levels of detail in visualisation.
- **Information overload**: Occurs when excessive data or visual elements overwhelm understanding.
- **Interval data**: Ordered categories with equal intervals but lacking a true zero point.
- **Nominal data**: Represents categories or labels without any inherent order.
- **Ordinal data**: Implies a meaningful order among categories but lacks equal intervals.
- **Ratio data**: Includes ordered categories with equal intervals and a meaningful zero point.
- **Relevance**: Presenting information pertinent to the addressed message or question.

References

[1] $500,000 in 1937 is how much today? https://www.calculateme.com/inflation/500000-dollars/from-1937/to-now, n.d. Accessed 9 Nov. 2023.

[2] Peter J. Brockwell and Richard A. Davis. *Introduction to Time Series and Forecasting*. Springer, 2016. Accessed 13 Nov. 2023.

[3] K. Dakwa. The kallikak family – historical influences, current controversies, teaching resources. https://intelltheory.com/intelli/the-kallikak-family/, 2001. Accessed 9 Nov. 2023.

[4] Mehdi Dastani. The role of visual perception in data visualization. *Journal of Visual Languages & Computing*, 13(6):601–622, 2002.

[5] M. de Carvalho. Incomplete data analysis. University of Edinburgh, Lecture 2, 9-10, 2023.

[6] Earth Science Data Systems. Firms frequently asked questions — earthdata. https://www.earthdata.nasa.gov/faq/firms-faq#ed-confidence, 2023. Accessed 12 Nov. 2023.

[7] Celso Silva et. al. Deforestation-induced fragmentation increases forest fire occurrence in central brazilian amazonia. *Forest*, 9(6):305, 2018.

[8] William S. Cleveland et. al. *Local regression models. Chapter 8 of Statistical Models in S*. Wadsworth Brooks/Cole, 1992.

[9] Tom Fawcett. Introduction to roc analysis. *Pattern Recognition Letters*, 27:861–874, 06 2006.

[10] JP García, JC Ferreira, and CM Patino. Receiver operating characteristic analysis: an ally in the pandemic. *J Bras Pneumol*, 47(2):e20210139, Apr 30 2021.

[11] E. Hawkins. Warming stripes — climate lab book. https://www.climate-lab-book.ac.uk/warming-stripes/, n.d. Accessed 9 Nov. 2023.

[12] Kieran Healy. *Data visualization: a practical introduction*. Princeton University Press, 2018.

[13] F. Nightingale. Diagram of the causes of mortality in the army in the east. https://www.davidrumsey.com/luna/servlet/detail/RUMSEY~8~1~327826~90096398:Diagram-of-the-Causes-of-Mortality-;JSESSIONID=802dd785-32fc-4b43-a53d-72ed57e15cc8?qvq=q%3Aauthor%3D%22Nightingale%2C%20Florence%22%3Blc%3ARUMSEY%7E8%7E1&mi=1&trs=10, 1859. Accessed 20 Oct. 2023.

[14] A. Proctor. Five charts that changed the world - bbc ideas. https://www.bbc.co.uk/ideas/videos/five-charts-that-changed-the-world/p0fb69c1, 2023. Accessed 20 Oct. 2023.

[15] B. D. Ripley. loess: Local polynomial regression fitting. https://www.rdocumentation.org/packages/stats/versions/3.6.2/topics/loess, 1998. Accessed 20 Jan. 2024.

[16] Muhammad Hafiz Wan Rosli and Andres Cabrera. Gestalt principles in multimodal data representation. *IEEE computer graphics and applications*, 35(2):80–87, 2015.

[17] A. Rutherford. Where science meets fiction: the dark history of eugenics. https://www.theguardian.com/science/2022/jun/19/where-science-meets-fiction-the-dark-history-of-eugenics, 2022. Accessed 9 Nov. 2023.

[18] J.D. Smith and M.L. Wehmeyer. Who was deborah kallikak? *Intellectual and Developmental Disabilities*, 50(2):169–178, 2012.

[19] Josh Starmer. Lowess and loess, clearly explained!!! `https://www.youtube.com/watch?v=Vf7oJ6z2LCc`, 2018. Accessed 3 Jan. 2024.

[20] Dejan Todorovic. Gestalt principles. *Scholarpedia*, 3(12):5345, 2008.

[21] Edward R Tufte. *The visual display of quantitative information*, volume 2. Graphics press Cheshire, CT, 2001.

[22] Matt P Wand and M Chris Jones. *Kernel smoothing*. CRC press, 1994.

[23] Hadley Wickham. *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York, 2016.

[24] Simon N. Wood. *Core Statistics*. Cambridge University Press, 2015.

A Appendix A

```
Start:  AIC=70.9
mpg ~ cyl + disp + hp + drat + wt + qsec + vs + am + gear + carb

       Df Sum of Sq    RSS    AIC
- cyl   1    0.0799 147.57 68.915
- vs    1    0.1601 147.66 68.932
- carb  1    0.4067 147.90 68.986
- gear  1    1.3531 148.85 69.190
- drat  1    1.6270 149.12 69.249
- disp  1    3.9167 151.41 69.736
- hp    1    6.8399 154.33 70.348
- qsec  1    8.8641 156.36 70.765
<none>              147.49 70.898
- am    1   10.5467 158.04 71.108
- wt    1   27.0144 174.51 74.280

Step:  AIC=68.92
mpg ~ disp + hp + drat + wt + qsec + vs + am + gear + carb

       Df Sum of Sq    RSS    AIC
- vs    1    0.2685 147.84 66.973
- carb  1    0.5201 148.09 67.028
- gear  1    1.8211 149.40 67.308
- drat  1    1.9826 149.56 67.342
- disp  1    3.9009 151.47 67.750
- hp    1    7.3632 154.94 68.473
<none>              147.57 68.915
- qsec  1   10.0933 157.67 69.032
- am    1   11.8359 159.41 69.384
- wt    1   27.0280 174.60 72.297

Step:  AIC=66.97
mpg ~ disp + hp + drat + wt + qsec + am + gear + carb

       Df Sum of Sq    RSS    AIC
- carb  1    0.6855 148.53 65.121
- gear  1    2.1437 149.99 65.434
- drat  1    2.2139 150.06 65.449
- disp  1    3.6467 151.49 65.753
- hp    1    7.1060 154.95 66.475
<none>              147.84 66.973
- am    1   11.5694 159.41 67.384
- qsec  1   15.6830 163.53 68.200
- wt    1   27.3799 175.22 70.410

Step:  AIC=65.12
```

```
mpg ~ disp + hp + drat + wt + qsec + am + gear

       Df Sum of Sq    RSS    AIC
- gear  1     1.565 150.09 63.457
- drat  1     1.932 150.46 63.535
<none>              148.53 65.121
- disp  1    10.110 158.64 65.229
- am    1    12.323 160.85 65.672
- hp    1    14.826 163.35 66.166
- qsec  1    26.408 174.94 68.358
- wt    1    69.127 217.66 75.350

Step:  AIC=63.46
mpg ~ disp + hp + drat + wt + qsec + am

       Df Sum of Sq    RSS    AIC
- drat  1     3.345 153.44 62.162
- disp  1     8.545 158.64 63.229
<none>              150.09 63.457
- hp    1    13.285 163.38 64.171
- am    1    20.036 170.13 65.466
- qsec  1    25.574 175.67 66.491
- wt    1    67.572 217.66 73.351

Step:  AIC=62.16
mpg ~ disp + hp + wt + qsec + am

       Df Sum of Sq    RSS    AIC
- disp  1     6.629 160.07 61.515
<none>              153.44 62.162
- hp    1    12.572 166.01 62.682
- qsec  1    26.470 179.91 65.255
- am    1    32.198 185.63 66.258
- wt    1    69.043 222.48 72.051

Step:  AIC=61.52
mpg ~ hp + wt + qsec + am

       Df Sum of Sq    RSS    AIC
- hp    1     9.219 169.29 61.307
<none>              160.07 61.515
- qsec  1    20.225 180.29 63.323
- am    1    25.993 186.06 64.331
- wt    1    78.494 238.56 72.284

Step:  AIC=61.31
mpg ~ wt + qsec + am
```

```
       Df Sum of Sq    RSS    AIC
<none>              169.29 61.307
- am    1    26.178 195.46 63.908
- qsec  1   109.034 278.32 75.217
- wt    1   183.347 352.63 82.790

Call:
lm(formula = mpg ~ wt + qsec + am, data = mtcars)

Residuals:
    Min      1Q  Median      3Q     Max
-3.4811 -1.5555 -0.7257  1.4110  4.6610

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   9.6178     6.9596   1.382 0.177915
wt           -3.9165     0.7112  -5.507 6.95e-06 ***
qsec          1.2259     0.2887   4.247 0.000216 ***
am            2.9358     1.4109   2.081 0.046716 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 2.459 on 28 degrees of freedom
Multiple R-squared:  0.8497,Adjusted R-squared:  0.8336
F-statistic: 52.75 on 3 and 28 DF,  p-value: 1.21e-11
```