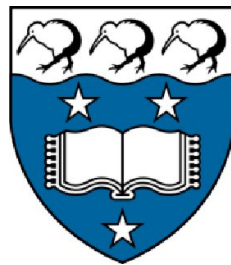


---

# Data Storytelling Dashboard for Exploring Auckland Air Quality



*ZHAOMING SU*

DEPARTMENT OF STATISTICS

THE UNIVERSITY OF AUCKLAND

supervised by

DR. EARO WANG

PROF. CHRIS WILD

A dissertation submitted in partial fulfilment of the requirements for the degree of  
Bachelor of Science (Honours) in Statistics, The University of Auckland, 2021.



# Contents

<b>Abstract</b>	<b>v</b>
<b>Acknowledgements</b>	<b>vii</b>
<b>Copyright notice</b>	<b>ix</b>
<b>Declaration</b>	<b>xi</b>
<b>1 Introduction (Template Demo)</b>	<b>1</b>
1.1 Rmarkdown . . . . .	1
1.2 Data . . . . .	2
1.3 Figures . . . . .	2
1.4 Results from analyses . . . . .	3
1.5 Tables . . . . .	3
<b>2 Background and related works</b>	<b>5</b>
2.1 Tidy time series data-wrangling toolbox . . . . .	5
2.2 Time series graphics toolbox . . . . .	6
2.3 HTML widgets for interactive graphics . . . . .	9
2.4 Visual analysis for temporal air quality data . . . . .	11
<b>3 Auckland air quality data</b>	<b>15</b>
3.1 Introduction . . . . .	15
3.2 Data enrichment . . . . .	16
3.3 Data quality and cleaning . . . . .	18
<b>4 Design layout and philosophy</b>	<b>21</b>
<b>5 Shiny linking and reactivity</b>	<b>27</b>
5.1 Introduction . . . . .	27
5.2 Modularisation of shiny application . . . . .	28
5.3 Reactive caching . . . . .	31
<b>6 Modelling</b>	<b>35</b>
<b>7 Conclusion and future works</b>	<b>37</b>
7.1 Future work . . . . .	37

7.2 Final words . . . . .	37
<b>A Additional stuff</b>	<b>39</b>
<b>Bibliography</b>	<b>41</b>

# **Abstract**

The abstract should outline the main approach and findings of the thesis and must not be more than 500 words.



# Acknowledgements

I would like to thank my pet goldfish for ...





# Copyright notice

© ZHAOMING SU (2021).

I certify that I have made all reasonable efforts to secure copyright permissions for third-party content included in this thesis and have not knowingly added copyright content to my work without the owner's permission.



# Declaration

This dissertation is an original work of my research and contains no material which has been accepted for the award of any other degree or diploma at any university or equivalent institution and that, to the best of my knowledge and belief, this dissertation contains no material previously published or written by another person, except where due reference is made in the text of the dissertation.



# Chapter 1

## Introduction (Template Demo)

This is where you introduce the main ideas of your thesis, and an overview of the context and background.

In a PhD, Chapter 2 would normally contain a literature review. Typically, Chapters 3–5 would contain your own contributions. Think of each of these as potential papers to be submitted to journals. Finally, Chapter 6 provides some concluding remarks, discussion, ideas for future research, and so on. Appendixes can contain additional material that don't fit into any chapters, but that you want to put on record. For example, additional tables, output, etc.

### 1.1 Rmarkdown

In this template, the rest of the chapter shows how to use Rmarkdown. The big advantage of using Rmarkdown is that it allows you to include your R code directly into your thesis, to ensure there are no errors in copying and pasting, and that everything is reproducible. It also helps you stay better organized.

For details on using *R Markdown* see <http://rmarkdown.rstudio.com>.

## 1.2 Data

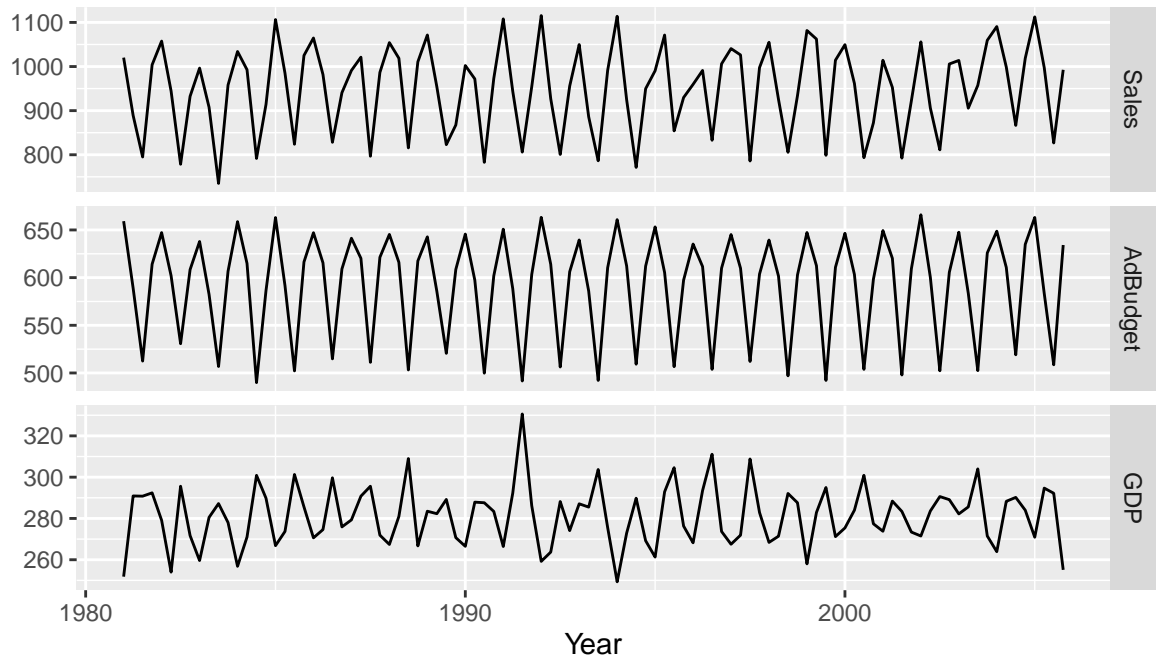
Included in this template is a file called `sales.csv`. This contains quarterly data on Sales and Advertising budget for a small company over the period 1981–2005. It also contains the GDP (gross domestic product) over the same period. All series have been adjusted for inflation. We can load in this data set using the following command:

```
sales <- ts(read.csv("data/sales.csv"),[, -1], start = 1981, frequency = 4)
```

Any data you use in your thesis can go into the data directory. The data should be in exactly the format you obtained it. Do no editing or manipulation of the data outside of R. Any data munging should be scripted in R and form part of your thesis files (possibly hidden in the output).

## 1.3 Figures

Figure 1.1 shows time plots of the data we just loaded. Notice how figure captions and references work. Chunk names can be used as figure labels with `fig:` prefixed. Never manually type figure numbers, as they can change when you add or delete figures. This way, the figure numbering is always correct.



**Figure 1.1:** Quarterly sales, advertising and GDP data.

## 1.4 Results from analyses

We can fit a dynamic regression model to the sales data.

If  $y_t$  denotes the sales in quarter  $t$ ,  $x_t$  denotes the corresponding advertising budget and  $z_t$  denotes the GDP, then the resulting model is:

$$y_t - y_{t-4} = \beta(x_t - x_{t-4}) + \gamma(z_t - z_{t-4}) + \theta_1 \varepsilon_{t-1} + \Theta_1 \varepsilon_{t-4} + \varepsilon_t \quad (1.1)$$

where  $\beta = 2.28$ ,  $\gamma = 0.97$ ,  $\theta_1 = NA$ , and  $\Theta_1 = -0.90$ .

## 1.5 Tables

Let's assume future advertising spend and GDP are at the current levels. Then forecasts for the next year are given in Table 1.1.

Again, notice the use of labels and references to automatically generate Table numbers. In this case, we need to generate the label ourselves.

Point Forecast	Lo 80	Hi 80	Lo 95	Hi 95
1000.2	947.7	1052.7	919.9	1080.5
1013.1	959.3	1066.8	930.9	1095.3
1076.7	1022.9	1130.6	994.4	1159.0
1003.5	949.7	1057.4	921.2	1085.8

**Table 1.1:** *Forecasts for the next year assuming Advertising budget and GDP are unchanged.*

The `knitLatex` package is useful for generating tables from R output. Other packages can do similar things including the `kable` function in `knitr` which is somewhat simpler but you have less control over the result. If you use `knitLatex` to generate tables, don't forget to include `results="asis"` in the chunk settings.



## Chapter 2

# Background and related works

Modern information design has been continuously fostering effective tools of organising and displaying information since Tufte (1983) proposed the landmarking principles of graphical displays in *The Visual Display of Quantitative Information*. Following the disruptive advancement of information and computing technology, infographics designers are faced with unprecedented methods of data display, including interactive and animated graphics.

As such, the prerequisites of a successful design of a storytelling dashboard for visualising the air quality data are the appropriate implementations of suitable data and graphical toolboxes. As the main features of interest in air quality data are temporal, this section will briefly outline the available toolboxes for time series data wrangling and visualisation using **R** (R Core Team, 2021).

### 2.1 Tidy time series data-wrangling toolbox

The **tsibble** package offers a data infrastructure for wrangling time series data (Wang, Cook, and Hyndman, 2020). A time series data set consists of one or more sequences indexed by time, often with a regular interval. As such, data-wrangling processes of time series data need to account for the special requirements of time series data analysis, including the explicit identification of time gaps and a method of handling multiple time series in a single data set for identifying duplicate records.

Analyses of fixed-interval time series require the data to be free from missing value, especially when the series is self-dependent. Whilst the explicit missing values can be easily handled by the substitution with interpolated values, the implicit gaps with missing index values are often neglected. In the case of multiple time series, locations of implicit time gaps may be different in each sequence; filling the gaps with traditional loops can be time-consuming and inefficient. **tsibble** identifies implicit time gaps with the *index* and *key* variables, such that each variable in the **tsibble** object is uniquely identified by the index and the interaction of all keys. As such, each time series is uniquely identified by the keys, allowing efficient identification of implicit time gaps, which is achieved by **tsibble** with a range of wrangling verbs.

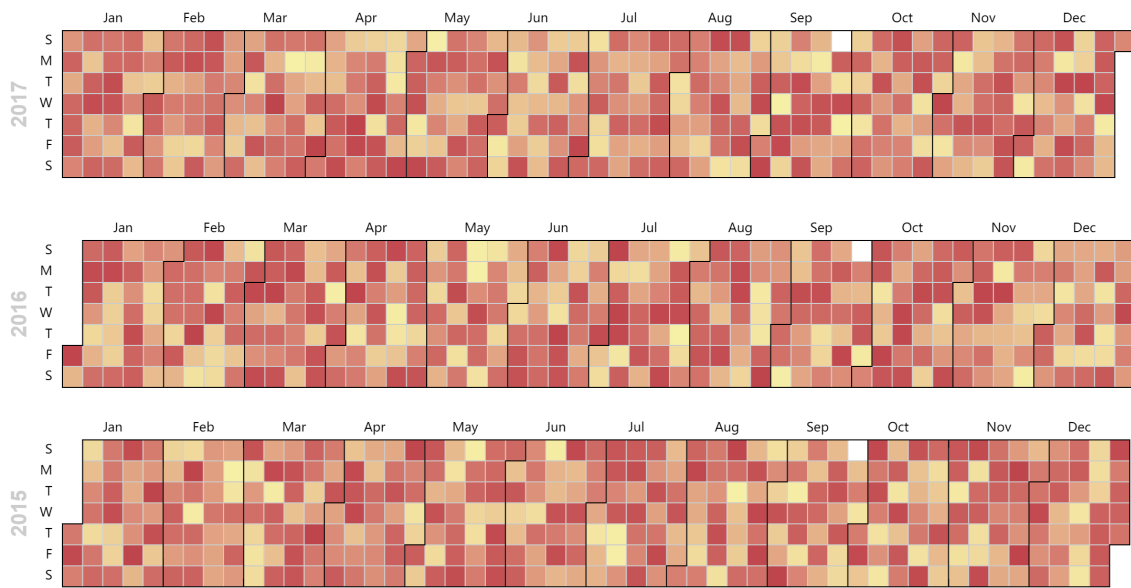
Duplicates exist in different forms in cross-sectional and time series data. Typically, duplicates are identical observations exhibited as rows in a data frame, yet such definition is inadequate in identifying duplicates in time series data. There exists only one true value at any given point in time for each time series, meaning that there may be duplicate values that are non-identical observations with identical key-index pairs yet different in values. Instead of searching merely for duplicate rows, **tsibble** checks for duplicate key-index pairs. To avoid negligence, the creation of **tsibble** will fail upon detected duplicates.

## 2.2 Time series graphics toolbox

### 2.2.1 Calendar graphics

Calendars are the systematic partition of time from the observed solar-lunar phenomena and cultural custom, which is usable as graphics for temporal representations of societal activities and natural events. Calendar graphics are the method for the aggregated visualisation of time series data at sub-daily intervals, depicting the temporal dimension of time series data as the spatial layout in the calendar grid. The motivation of utilising calendars for data visualisations arises from the convenience of displaying observations in association with exact dates.

Air quality data are conventionally collected at hourly intervals, from which a time series plot becomes overcrowded, impeding the visual detections of abnormalities. Rahman



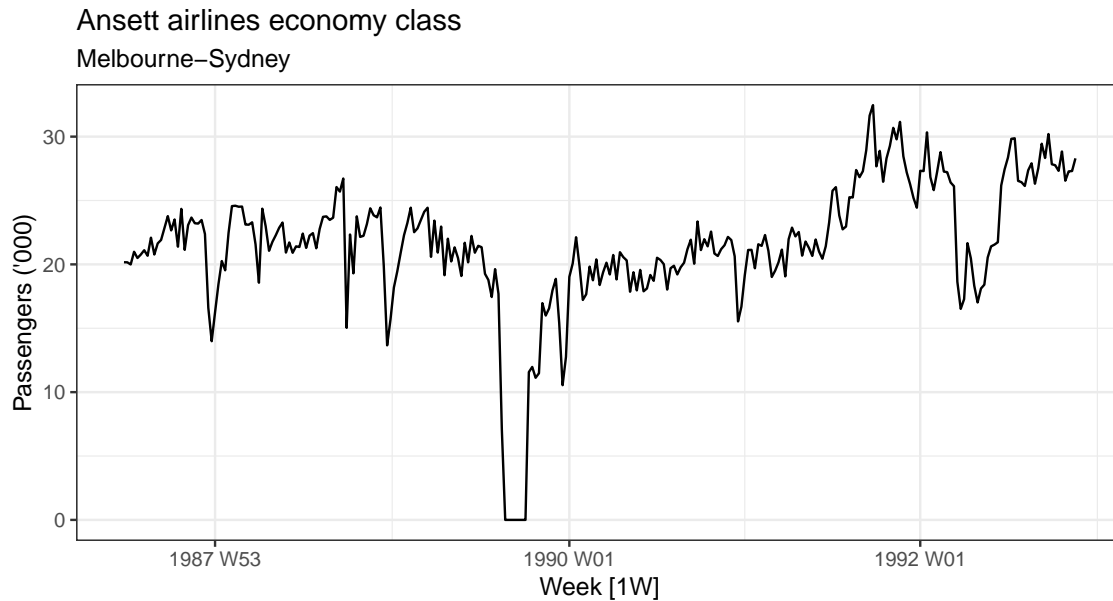
**Figure 2.1:** *Calendar heatmap. Produced with `Echarts.js` (The Apache Software Foundation, 2020) via `echarts4r` (Coene, 2020) faceted (unaligned) by `year ~ month`. Each tile corresponds to the value of an observed day or aggregated sub-daily observations, whose exact date and day of the week are easily identifiable.*

and Lee (2020) depicted a method of non-cartesian heatmaps on a calendar coordinate for visualising air pollution data. Prior to the paper, Liu, Li, and Li (2016) used calendar heatmaps to analyse the correlation of particulate matter temporally. Calendar graphics highlight the abnormalities and allow the association of the detected abnormalities to events with dates, providing insights and directions for analyses.

Calendar graphics is an application of trellis displays (Becker, Cleveland, and Shyu, 1996), which spatially modularise the temporal dimension into conditional groups of small multiples (Tufte, 1983) using calendar period (i.e., month and year) as an integrated and aligned plot. The expanded layout of the temporal dimension on a plane eases the cognitive load (Tufte, 1983) in temporally locating the date of the events and extracting the date components, contrary to conventional time series plots.

### 2.2.2 Time series plots

In most scenarios, visualising time series relies on connecting points of observations with lines, curves or splines (Wilke, 2019), such that the sequences are plotted against the time index by positions along common xy-scales on the Cartesian coordinate (Hyndman,

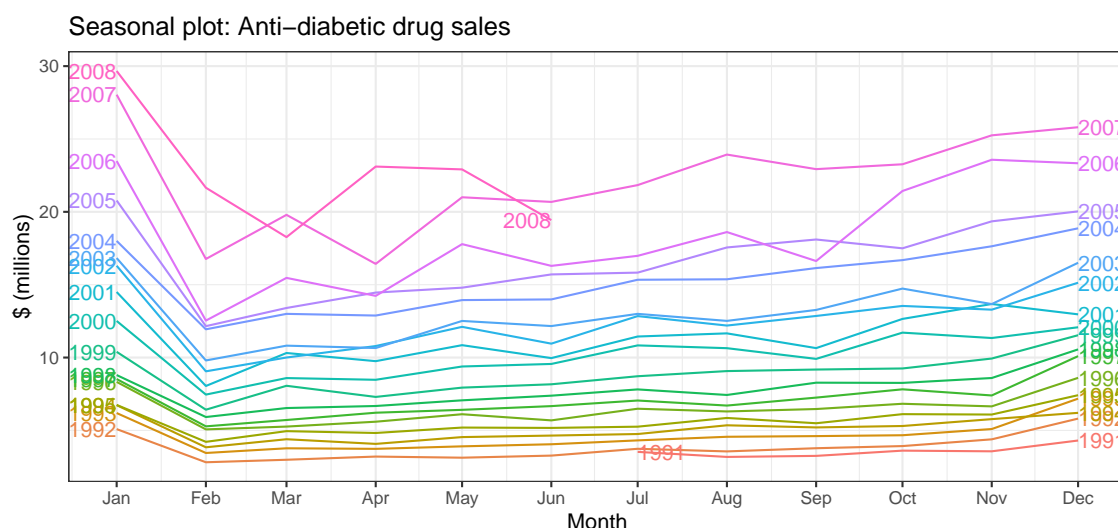


**Figure 2.2:** Line plot. Produced with the *ggplot2* package (Wickham, 2016) for the weekly economy passenger load on Ansett Airlines (Hyndman, 2021b). Visual analysis shows weak trend and cycle and abnormal zero values to be investigated. The presence of clustering also indicates a positive temporal dependence in the time series. It is nonetheless uneasy to align any observation to a date accurately.

2021a). Connected time plots are the most elementary visual method for spotting extrinsic features in time series, including trends, seasons, cycles, clustering and oscillations.

Based on connected time plots, O’Hara-Wild, Hyndman, and Wang (2021) proposed the seasonal plot as a method of visualising seasonal patterns in the **feasts** package. The method conditionally subsets the complete time series into partitions of seasonal periods, each to be plotted in a homogeneous time plot and distinguished using a gradient colour scale for longitudinal comparison between periods.

Nevertheless, interpreting connected time plots relies on the visual alignment of positions to the  $xy$ -scales in the Cartesian coordinates. Such visual alignments can be challenging upon the absence of explicit gridlines and axes, such as when the plot is fitted as a part of trellis displays (Becker, Cleveland, and Shyu, 1996), in which the trend and scale of variations become ambiguous. As such, it is common to fill the area under the curve to emphasise the temporal variation in the plot (Wilke, 2019).



**Figure 2.3:** *ggplot2*-based seasonal time series plot. Produced with the *feasts* package for the monthly anti-diabetic drug sales in Australia (Hyndman, 2021b). The plot allows for both visualisation of intra-seasonal patterns and inter-seasonal variations shown as a positive trend in drug sales from year to year.

## 2.3 HTML widgets for interactive graphics

### 2.3.1 Development of web applications with R

The **shiny** package (Chang et al., 2021) provides a framework for developing web applications with **R** code (R Core Team, 2021), both user and developer-friendly. It enables **R** users with no prior knowledge of HTML, CSS and JavaScript to create custom web applications with sophisticated functionality with template UI components and a server powered with reactive programming.

Reactive programming focuses only on the evaluation of the changes of values over time, which is the core computation logic behind **shiny** (Wickham, 2021), significantly simplifying the workflow design. Each change in reactive values is observed as an event by pre-defined callback functions, and workflows are executed as responses to events observed. The lazy nature of reactivity avoids repeated evaluations of expressions leading to wastage in computational resources. Reactive programming also allows users to define abstract workflows without conceiving the low-level data and programming logic by restricting evaluations to merely reactions to events, including user actions and internal value changes. A single reactive value can be observed and called by several callback

functions, which can be shared across different functionalities. As the reactive value always keeps the previous evaluated result, conflicts among functionalities are avoided.

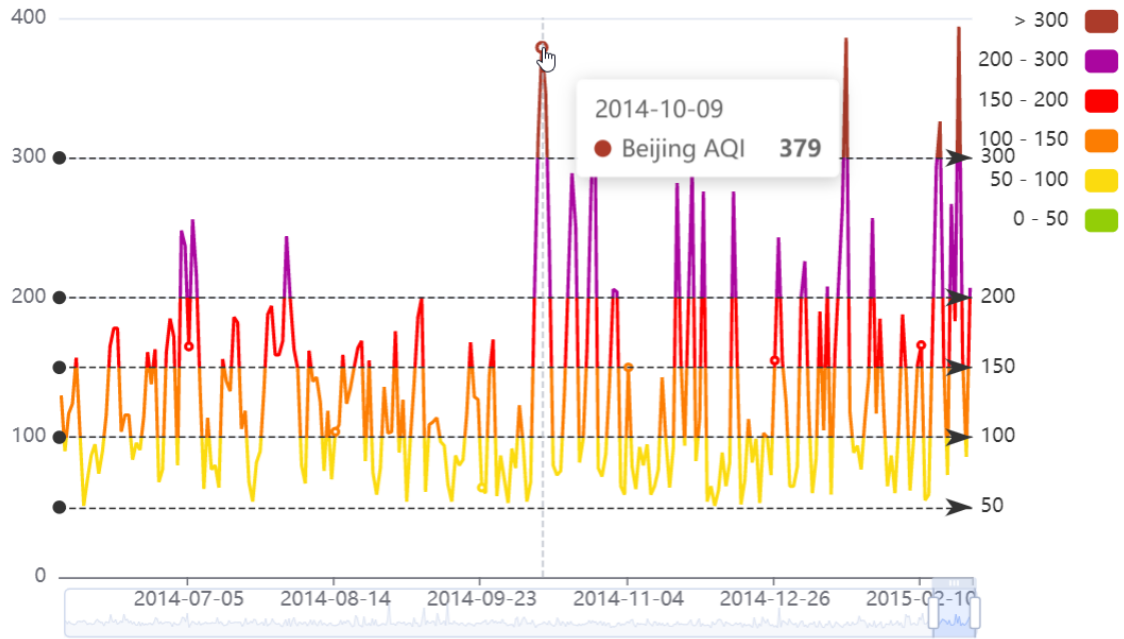
The user interface of **shiny** applications provides the front-end inputs and output display from the back end server logic. The collection of user input is the primary source of change of reactive values, events that trigger the evaluation of expressions in the back end server logic. The results of the evaluated expressions are rendered as the outputs, which may be as simple as prints of R objects or as sophisticated as HTML interactive graphics.

### 2.3.2 Interactive graphics

In the technological age flooded with data, it is usually tempting to fit all the information in a single display upon data visualisation. To avoid excessive cognitive load to viewers, numerous methods, such as trellis displays (Becker, Cleveland, and Shyu, 1996), are proposed to maximise the information density in a dashboard of static graphics. As the available dimensions of a single display are exhausted, the need to further increase the information density motivated the introduction of interactive graphics (Cook and Swayne, 2007). The primary purpose of utilising graphical interactivity is to provide a coarse-to-granular orientation for data exploration, such that only the user-chosen details of a broad summary are dynamically shown, reducing the wastage of limited spatial recourses on display. Another use case of interactive graphics is to establish dynamic linkings between plots (Cook and Swayne, 2007).

The selective presentation of information in interactive graphics is mainly achieved by hover-over tooltips and drill-down. Tooltips, initially introduced by Microsoft Corporation (1995) in *Windows NT 3.51*, refer to tags of brief descriptive messages upon hovering over graphical elements in the context of interactive graphics, temporarily hiding granular details of a coarser summary until user input which allows a top-down approach to preliminary data exploration. Nevertheless, tooltips are dynamic displays that vanish as the cursor moves away. As such, drill-down (Sievert, 2020) and click triggered popups are often used as supplements to tooltips in need of stable auxiliary graphics.

Linked views of multiple graphics are a powerful method for deconstructing high-dimensional data, achieved via either client or server-side linking (Sievert, 2020). The



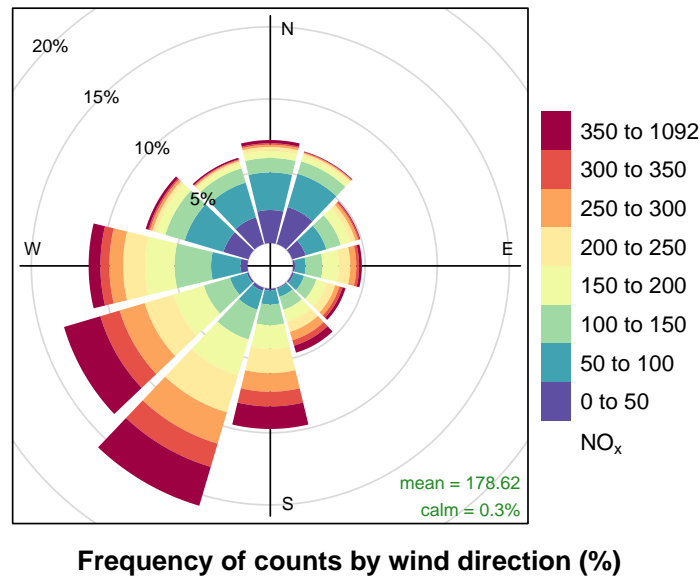
**Figure 2.4:** A demonstration of tooltips with *Echarts.js* via *echarts4r*. A tag with informative messages including the date and observed AQI value is shown upon cursor hover-over the data point.

client-side linking employs internal graphical queries without callbacks to the application server, with examples including internal connections (e.g., brush and linked filters) in **plotly** (Sievert, 2020) and **echarts4r** (Coene, 2020). Alternatively, server-side linkings involve queries to the server environment via callbacks (Section 2.3.1), mainly for inter-modular linking of different types of plots.

A special application of server-side linking is to facilitate selection control in interactive maps. In addition to the positioning and navigational functions of interactive maps, they are a native geographical user interface for click or hover-and-show inputs and outputs.

## 2.4 Visual analysis for temporal air quality data

Temporal air quality data requires domain-specific graphic tools for exploratory visual analysis. This section covers examples of visualising air quality in meteorological and temporal aspects.



**Figure 2.5:** *Pollution rose. Produced with **openair** (Carslaw and Ropkins, 2012), which depicts the proportion of time (radius of the segmental arches) the wind is bounded from each direction (angle) observing the levels of NO<sub>x</sub> (colour).*

### 2.4.1 Wind roses

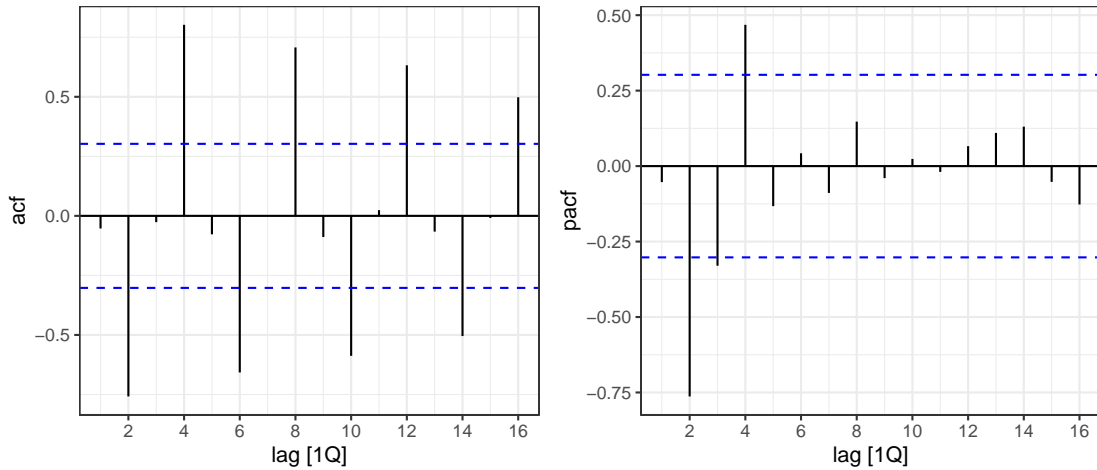
Wind speed and direction are critical meteorological parameters that affect the concentration of ambient air pollutants by altering their transportation, diffusion and accumulation (Zhang et al., 2018). The visual evaluation of wind as vectors requires specialised graphics capable of depicting the directional nature of wind. In conjunction with the requirement for visual association analysis between wind and air pollution, a temporal-proportion-based wind rose plot is proposed and implemented on the **openair** package (Carslaw and Ropkins, 2012), which is an application of the stacked polar bar plots. Variations of temporal-proportion wind rose plots include contour wind roses (Munn, 1969) and standard vector-deviation wind roses (Crutcher, 1957). A temporal-proportion wind rose can be extended to pollution rose, replacing the mapping of wind speed to pollutants.

### 2.4.2 Autocorrelation plots

Trend analysis of ambient air pollutant concentrations is a critical component of the air quality management strategy (Auckland Regional Council, 2020a), which is commonly carried out preliminarily with linear models. Nonetheless, mutual independence of the



Australian beer production



**Figure 2.6:** A *ggplot2*-based autocorrelation plot (left) and partial autocorrelation plot (right) of quarterly lags with the *feasts* package for Australian beer production (Hyndman, 2021b). The sample (partial) autocorrelation function of each lag is compared to a significance threshold of  $\pm \frac{\phi^{-1}(1-\frac{\alpha}{2})}{\sqrt{n}}$ .

residuals is the core assumption of all linear models; as such, the temporal dependence of air quality data between successive observations needs to be captured (Chambers and Hastie, 1992). The autocorrelation plots visually provide preliminary insights into the type of correlation structure present in the series prior to analysis (Venables and Ripley, 2002). The autocorrelation plots are also a tool for model diagnostics and evaluation.



## Chapter 3

# Auckland air quality data

### 3.1 Introduction

Air quality index (AQI) is a critical indicator of overall air quality by measuring key air pollutant concentrations at a given time. The constitution of AQI consists of ambient air pollutants listed in the National Environmental Standards for Ambient Air Quality which defines the threshold target for calculating AQI (Auckland Regional Council, [2020a](#)). The national standard defines AQI as the maximum ambient air pollutant measurement ratio to the national target as a percentage (Auckland Regional Council, [2020b](#)). Over ten stations in Auckland monitor a subset of the listed pollutants in an hourly interval.

It is of interest to visually explore the variation and relationships of AQI and its constituent pollutants over time. The data, provided by Auckland Regional Council ([2021](#)), includes 11 air pollution parameters from 10 monitoring stations in Auckland from as North as Takapuna to as South as Patumahoe. The available parameters consist of air quality index (AQI) and ten pollutants as per Table [3.1](#), with various starting dates (since as early as 2003 in Takapuna) until April 2021.

Only six of the standard-listed air pollutants are monitored and available in the data, and each station independently monitors a subset of the six pollutants. As such, the calculation of AQI, based on available data, may be simplified to

$$AQI = 100 \times \max\left\{\frac{PM_{2.5}}{25}, \frac{PM_{10}}{50}, \frac{NO_2}{200}, \frac{SO_2}{350}, \frac{CO}{10}, \frac{O_3}{150}\right\} \quad (3.1)$$

Parameter	Unit	Note
AQI		Air quality index
BC(370)	ngm <sup>-3</sup>	Black carbon at 370nm wavelength
BC(880)	ngm <sup>-3</sup>	Black carbon at 880nm wavelength
CO*	mgm <sup>-3</sup>	Carbon monoxide concentration
NO	µgm <sup>-3</sup>	Nitrogen monoxide concentration
NO <sub>2</sub> *	µgm <sup>-3</sup>	Nitrogen dioxide concentration
NO <sub>x</sub>	µgm <sup>-3</sup>	Nitrogen oxides concentration
O <sub>3</sub> *	µgm <sup>-3</sup>	Ozone concentration
PM <sub>2.5</sub> *	µgm <sup>-3</sup>	Particulate matter with diameter <2.5µm
PM <sub>10</sub> *	µgm <sup>-3</sup>	Particulate matter with diameter <10µm
SO <sub>2</sub> *	µgm <sup>-3</sup>	Sulphur dioxide concentration

**Table 3.1:** *Parameters available in the raw data.*

\* AQI-related ambient air pollutants

It is noteworthy that the availability of air quality parameters in each monitoring station varies from year to year. Besides, the extreme values addressed in Section 3.3.1 are more frequent in earlier years. The final data set is thus taken since the year 2016.

## 3.2 Data enrichment

### 3.2.1 Supplementary meteorological data







The air quality index is presumably affected by various environmental and meteorological factors, such as wind as per Section 2.4.1, which are included as supplementary variables.

Variable	Unit
Relative Humidity	%
Temperature	°C
Wind Speed	ms <sup>-1</sup>
Wind Direction	°

**Table 3.2:** *Supplementary meteorological variables.*

### 3.2.2 Categorisation of air quality index

The value of the air quality index derived from relevant pollutants is numerically non-intuitive and does not provide perceptual implications to the state of air quality per se, which leads to the proposal of standard classification by evaluating potentially imposed health risk (Gore and Deshpande, 2017). Notwithstanding the absence of an officially promulgated standard for the classification of air quality index in New Zealand (New Zealand Parliamentary Counsel Office, 2004), variations of the United States standards of reporting air quality index (U.S. Environmental Protection Agency, 2020) are generally accepted worldwide. The health risk warnings-driven categorisation of the air quality index delivers an immediate message about the current status of air quality and asserts behavioural influence to the public with colours (Braun, Mine, and Clayton Silver, 1995).

AQI Level of Concern	Value of Index	Colour (Hexadecimal Code)
Good	0 to 50	 Green (#00E400)
Moderate	51 to 100	 Yellow (#FFFF00)
Unhealthy for Sensitive Groups	101 to 150	 Orange (#FF7E00)
Unhealthy	151 to 200	 Red (#FF0000)
Very Unhealthy	201 to 300	 Purple (#8F3F97)
Hazardous	301 and higher	 Maroon (#7E0023)

**Table 3.3:** *The United States definition of AQI categories and colours (U.S. Environmental Protection Agency, 2020). Note that AQI is always rounded to the nearest integer.*

### 3.2.3 Categorisation of wind direction

The implementation of the wind rose method of visualising meteorological wind data outlined in Section 2.4.1 is made interactable by **echarts4r** (Coene, 2020) without any direct call to the **openair** package. Thus, the wind direction variable is converted into a 12-level factor of equal 30-degree intervals, the default option for the `windRose()` function in **openair** (Carslaw and Ropkins, 2012), as part of the data preparation process. The process of correcting directional bias by default in **openair** is excluded as the variable is unrounded (Droppo and Napier, 2008).

### 3.3 Data quality and cleaning

The raw data consists of two separate data sets, each with a different data structure. Cleaning and manipulation are needed to ensure that the two data sets are consistent in structure and free from error. The raw data sets are individually inspected and cleaned before combination. This section outlines the issues found and methods to address them.

#### 3.3.1 Abnormal and missing values

Abnormal or missing values arise from instrumental or input errors. Upon inspection, 104,332 records were found to have a negative value. Nevertheless, all pollutants are reported in units in the form of mass per unit volume, and other parameters, except for temperature, are only sensible if positive as of Table 3.1. Therefore, 104,257 records of insensible negative values are removed. Besides, conspicuously anomalous records of AQI are found in data, including consecutive hours of  $>1,000$  AQI in Takapuna and numerous AQI values being inconsistent with Formula 3.1 based on available pollutants in the same data set. The anomalous records are nonetheless kept as-is for further verification.

In addition, preliminary inspection finds that 0.81% of records are explicitly missing. Yet after filling the implicit time gaps in the data, 53.71% of records are implied to be missing.

#### 3.3.2 Date and time

A consistent format in date and time is crucial to the accuracy of temporal data. Observations with inconsistent time format are present in the data, where some are recorded in hh:mm:ss whilst others in hh:mm. The inconsistency in the time format is correctable due to the hourly nature of the data. 0.06% of records with missing time are removed.

The time zone of New Zealand changes by +1 during daylight saving. To avoid duplicated index upon boundaries of daylight saving upon data visualisation, all time-stamps are presented in NZST (UTC+12). On the other hand, the date and time in the cleaned data file are stored as a single variable, with its format in compliance with ISO 8601 (International Organization for Standardization, 2019; Wickham, Hester, and Francois, 2018).

### **3.3.3 Duplicate records**

Temporal data should not present duplicate records. Of the 7,292,038 valid records, 239,374 (3.28%) are duplicate with 120,207 redundant records. Further checking reveals that 230,822 of the duplicates have inconsistent values. However, as the scale of the inconsistency of most duplicate records is reasonably small, the first-appearing records of each duplicate are kept.

### **3.3.4 Structural difference in raw data sets**

The primary data set, which records all parameters except for wind direction, is in long format, with each observation consisting of a single record of one parameter for one station at a given hour. Nevertheless, each observation of the wind direction data set consists of wind direction records of all stations at a given hour. Each data set is pivoted to the structure such that each observation is uniquely identified by the date-time and station with records of all parameters before combination to ensure structural consistency.

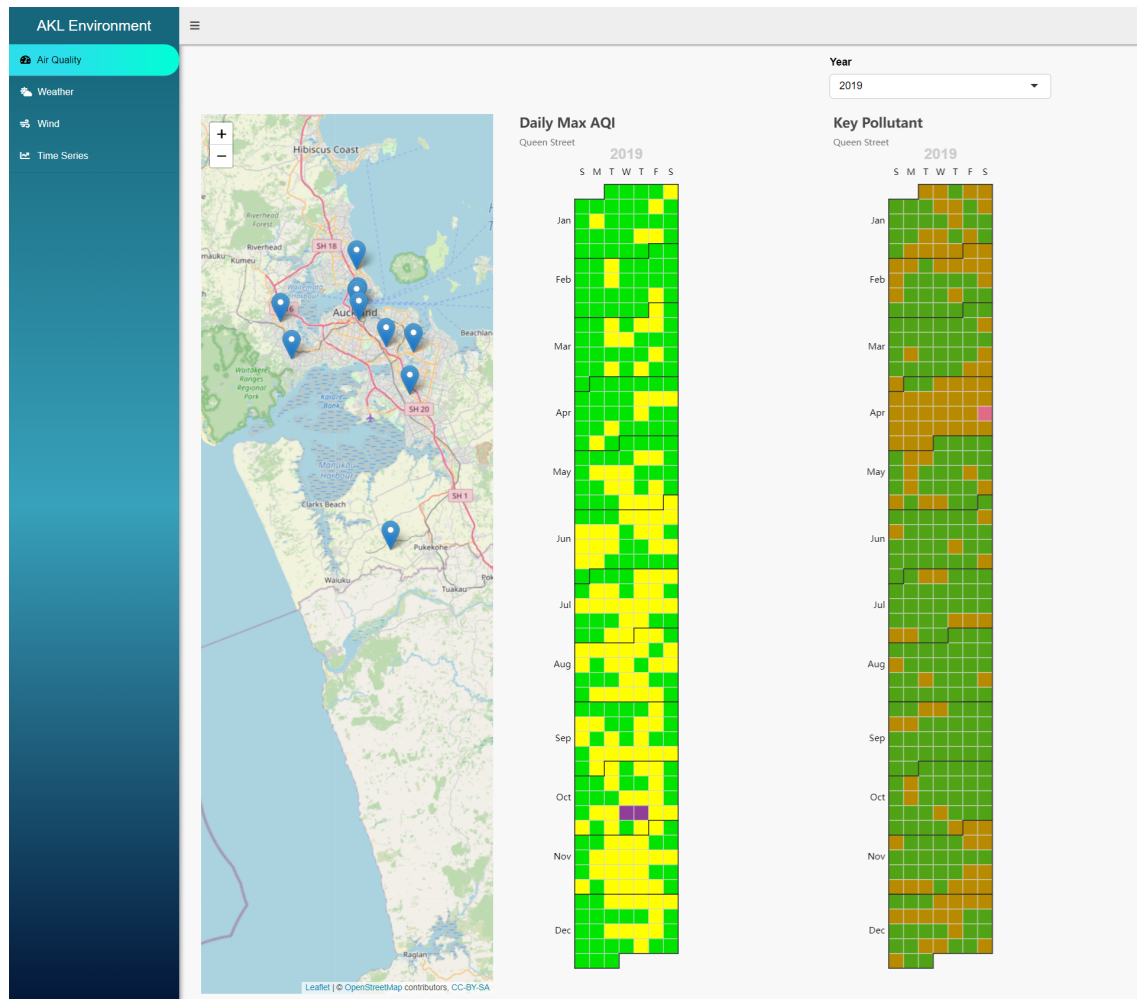




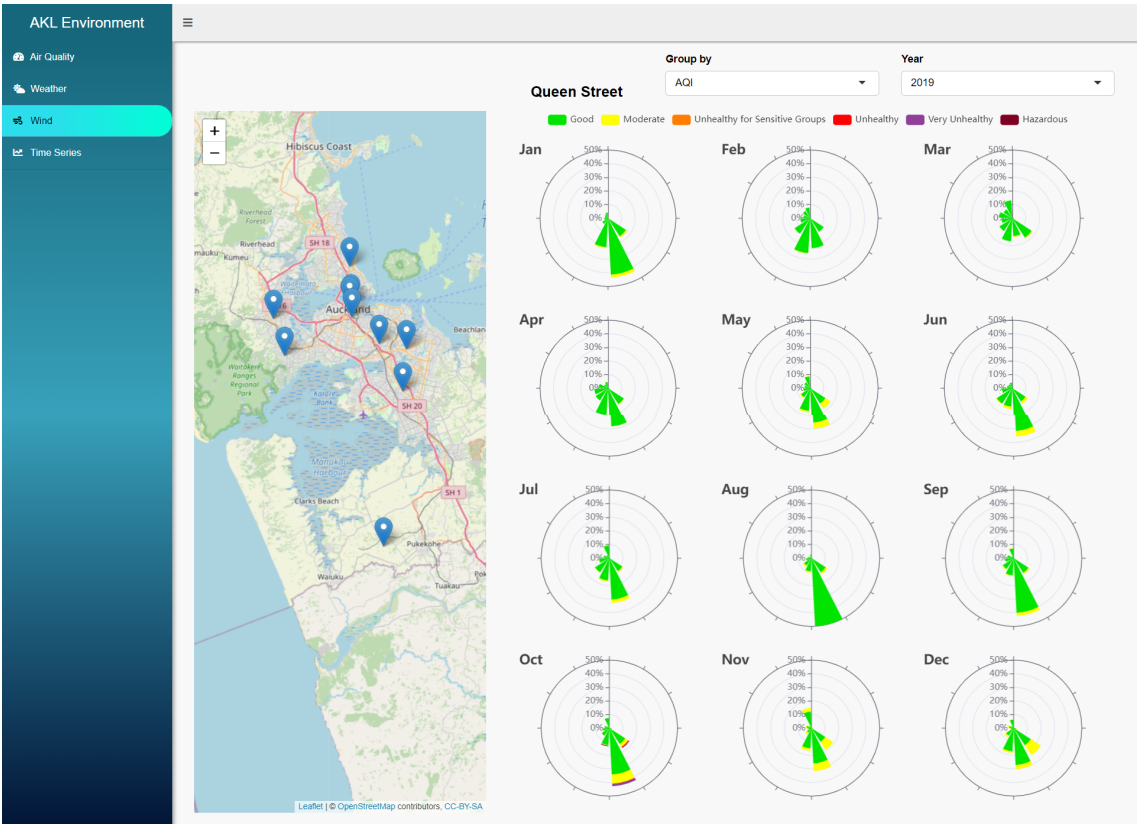
## Chapter 4

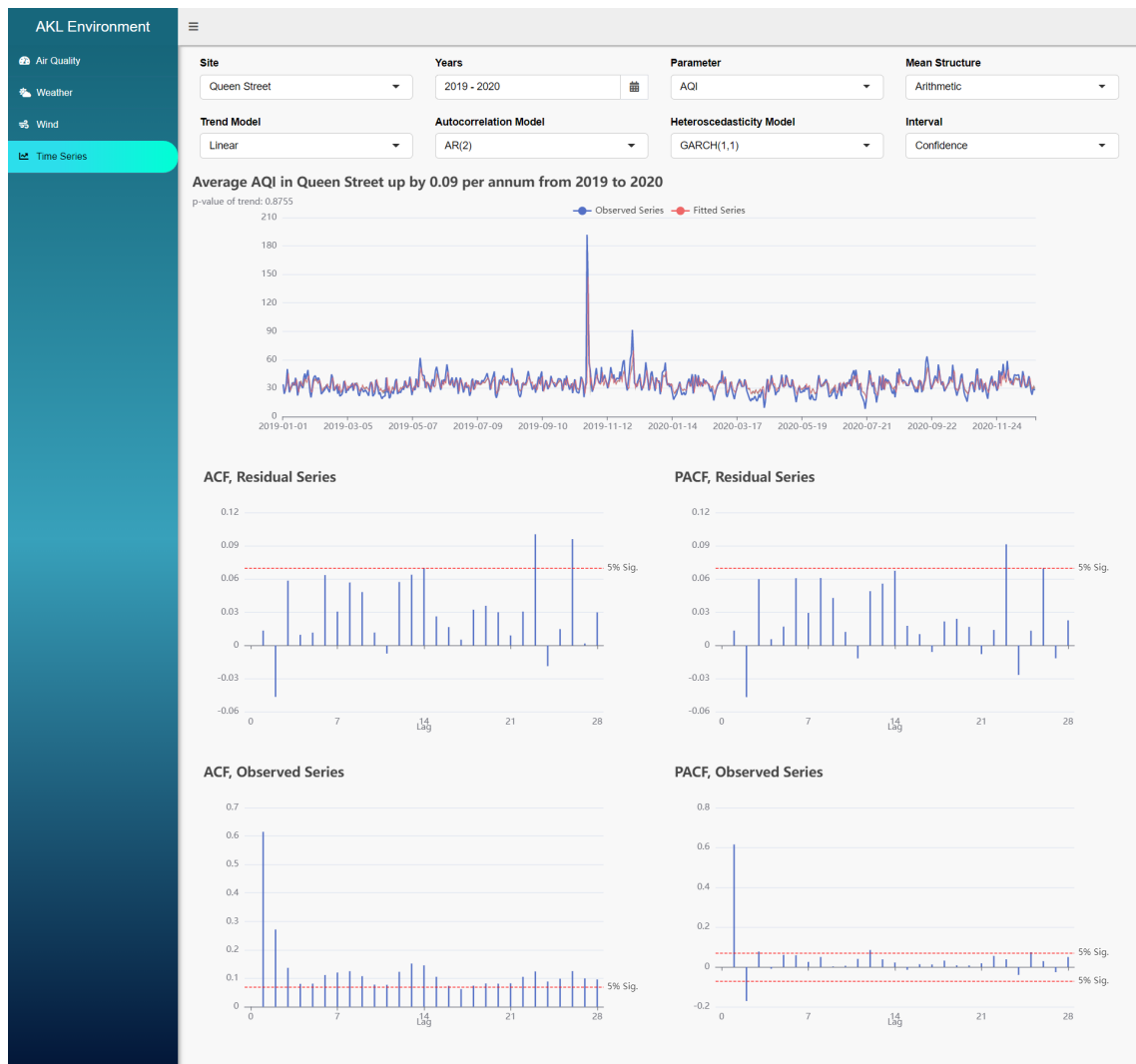
# Design layout and philosophy

- Overview of AQI
  - Spatial
  - Temporal (AQI and its constituent)
    - \* Calendar
    - \* Drill-down line plot
- Data enrichment
  - Explore relationship AQI with wind speed and direction
  - Meteorological data
- Trend analysis











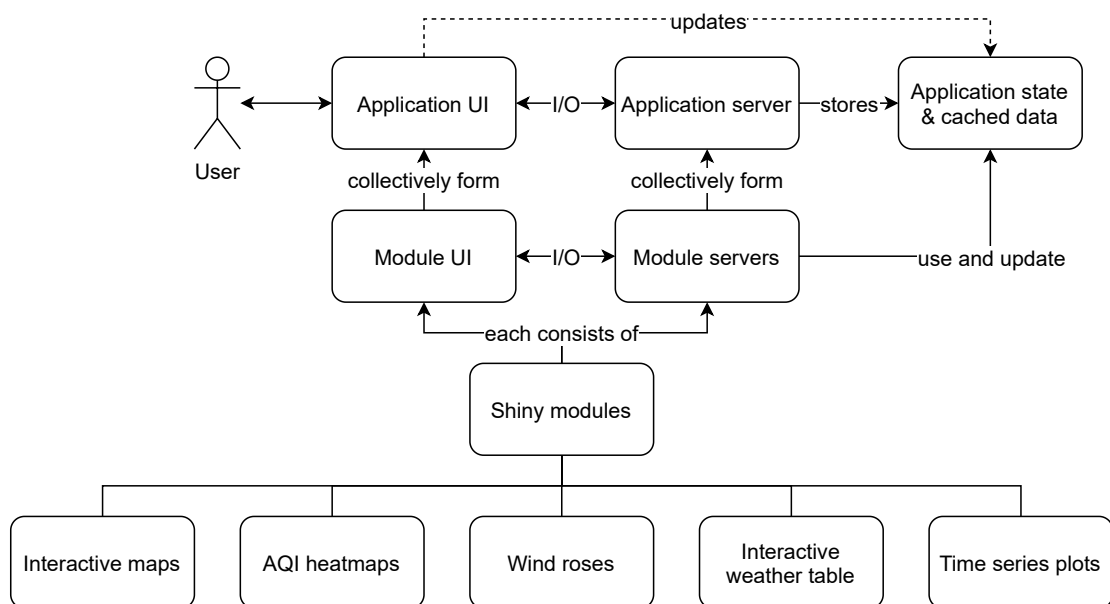
## Chapter 5

# Shiny linking and reactivity

### 5.1 Introduction

(Introduction).

This section, as a typical example, outlines the application of linking and reactivity on the user selection of monitoring stations with an interactive map in the modularised shiny application.



**Figure 5.1:** The structure of the modularised shiny application.

## 5.2 Modularisation of shiny application

The modular division of the shiny application allows the development and deployment of components to be independent of one another, enabling flexible addition and modification of functions. Each shiny module represents a dedicated functional component of the application. The modules are black boxes that are isolated and only communicate with other modules via a defined global reactive. The isolation of modules avoids unwanted interference among modules, simplifying the workflow control.

### 5.2.1 Module environment

Each of the shiny modules consists of its independent UI and server, much similar to a miniature “shiny environment”, called within the application UI and server.

```
map_aqi_ui <- function(id) {  
  ns <- NS(id)  
  tagList(leafletOutput(ns("map_aqi"), height = "1190px"))  
}  
  
map_aqi_mod <- function(id, state) {  
  module <- function(input, output, session) {  
    output[["map_aqi"]] <- renderLeaflet(...)  
    ## Reactivity and event handling  
  }  
  moduleServer(id, module)  
}
```

The shiny modules are separated via namespacing, which is achieved by the definition of the functions of module servers. The namespace of each module is uniquely identified and called by a caller-defined `id` for both the module UI and server, which allows for the reuse of modules under different `id`. The namespace is called by a module-specific function, defined by the `NS(id)`, in the module UI and stored in the shiny session, which is the basis of referral of graphics outputs in the module UI rendered in the module server.



In the context of the interactive-map module, the interactive map with the output id of "map\_aqi" is called dependently via the module-specific namespace function `ns()`.

Once defined, the modules are called in the application UI and server.

```
app_server <- function(input, output, session) {  
  ## Application Initialisation  
  ## Other shiny module servers  
  map_aqi_mod("map_aqi", app_state)  
  ## Reactivity and event handling  
}  
  
app_ui <- dashboardPage(  
  header,  
  sidebar,  
  body = dashboardBody(  
    tabItem(tabName,  
      fluidRow(column(map_aqi_ui("map_aqi"))),  
      ## Other modular UI  
    ),  
    ## Other tabs  
  )  
)
```

## 5.2.2 Event observation and handling

The observation of events initiates evaluations in reactive programming, enabling the implementation of user-input monitoring in shiny applications, achieved via the **shiny** `observeEvent()` function.

```
map_aqi_mod <- function(id, state) {  
  module <- function(input, output, session) {  
    ## Shiny output
```

```
observeEvent(input[["map_aqi_marker_click"]], {  
  state[["map_onclick"]] <- input[["map_aqi_marker_click"]][["id"]]  
})  
}  
moduleServer(id, module)  
}
```

Contextually, the interactive-map module monitors user clicks on the map items defined by the *event expression*. The JavaScript-based interactive web map rendered by the **leaflet** package (Cheng, Karambelkar, and Xie, 2021) automatically generates callback inputs to the shiny session whose change is treated as an event upon user map clicks, which triggers the evaluation of the *handler expression*. Other than callback events generated from interactive graphics, observable events include direct user inputs from the main UI.

### 5.2.3 Module communication

The communication between the namespace-isolated modules is facilitated with a **shiny** reactive object which collectively stores the current states of the application. The reactive object is shared and updated simultaneously by all modules and holds the *one version of truth* at any given point in time. The reactive object is also subsettable such that the update of values is achievable with the assignment operator in the same way that objects are updated in a `list` object (Section 5.2.2).

```
aqi_heatmap_mod <- function(id, state = app_state) {  
  module <- function(input, output, session) {  
    ## Use app_state by calling state[[app_state_name]]  
    ## Update app_state by calling state[[app_state_name]] <-  
    ## Shiny output  
  }  
  moduleServer(id, module)  
}
```

The reactive object `app_state` is stored in the application server and called upon the evaluation of all module server functions as an argument, and the update of which in any module is global for all others. Thus, the information of the user map clicks can be shared instantaneously for the graphics modules.

## 5.3 Reactive caching

### 5.3.1 Data caching

Exhaustive loading of the data with over 1 million observations incurs an undesirably long initialisation time upon application launch. As the application interface displays interactive graphics with data from only one location at a time, the initialisation upon application launch loads only the data of the default initial location (i.e., Queen Street, which is changeable). Upon detecting the user selection of a location with unloaded data, the data is updated by appending the corresponding data set to the cached data.

```
initial_app_state <- list(  
  data = append_data(NULL, "queen_street"),  
  map_onclick = "Queen Street"  
)  
  
app_server <- function(input, output, session) {  
  app_state <- eval_tidy(new_quosure(expr(  
    reactiveValues(!!!initial_app_state)  
  )))  
  ## Shiny module servers  
  ## Reactivity and event handling  
}  
  
append_data <- function(data, loc) {  
  paste0("data/", gsub("_", "-", loc), ".csv") %>%  
    read_csv(...) %>%
```

```
## Wrangling
bind_rows(data)
}

app_server <- function(input, output, session) {
  ## Application Initialisation
  ## Shiny module servers
  observeEvent(app_state[["map_onclick"]], {
    if (!loc %in% app_state[["data"]][["location"]]) {
      app_state[["data"]] <- append_data(app_state[["data"]], loc)
    }
    ## Data processing
    ## Update shiny inputs
  })
}
```

To prevent the graphics from rendering before the required data is loaded, an availability check of the data from the specified location is performed before the evaluation continues.

```
aqi_heatmap_mod <- function(id, state = app_state) {
  module <- function(input, output, session) {
    ## Data processing
    output[["aqi_heatmap"]] <- renderEcharts4r({
      loc <- make_clean_names(state[["map_onclick"]])
      req(loc %in% state[["data"]][["location"]])
      ## Render shiny output
    })
  }
  moduleServer(id, module)
}
```

### 5.3.2 Graphics caching

Interactive graphics, notwithstanding already optimised for web applications, may take a considerable amount of time to load. As such, the caching of rendered graphics in the current session improves the performance of reloading the same graphics by calls to the **shiny** `bindCache()` function, with the keys being the arguments. The keys are variables from which a collectively unique graphic is rendered.

```
aqi_heatmap_mod <- function(id, state = app_state) {  
  module <- function(input, output, session) {  
    ## Reactivity and event handling  
    ## Data processing  
    output[["aqi_heatmap"]] <- renderEcharts4r(expr) %>%  
      bindCache(...)  
  }  
  moduleServer(id, module)  
}
```

### 5.3.3 Client-side interactive linking



## **Chapter 6**

# **Modelling**





## **Chapter 7**

# **Conclusion and future works**

### **7.1 Future work**

#### **7.1.1 Statistical modelling**

#### **7.1.2 Aspect ratio adaptation**

#### **7.1.3 Connection to dynamic database**

#### **7.1.4 Graphics performance**

### **7.2 Final words**



## **Appendix A**

### **Additional stuff**

You might put some computer output here, or maybe additional tables.

Note that line 5 must appear before your first appendix. But other appendices can just start like any other chapter.



# Bibliography

- Auckland Regional Council (2020a). *Auckland Ambient Air Quality Guidelines*. Accessed: 2021-09-22. Auckland, New Zealand. <https://drive.google.com/drive/folders/1D-WsQ3ISmYWjJlSI29aA7rlgYPrz0v6F>.
- Auckland Regional Council (2020b). *Auckland Ambient Air Quality Targets*. Accessed: 2021-09-22. Auckland, New Zealand. <https://unitaryplan.aucklandcouncil.govt.nz/Images/Auckland%20Unitary%20Plan%20Operative/Chapter%20E%20Auckland-wide/1.%20Natural%20Resources/E14%20Air%20quality.pdf>.
- Auckland Regional Council (2021). *The Auckland Environmental Data*. Accessed: 2021-05-25. Auckland, New Zealand. <https://environmentauckland.org.nz/Data/DataSet>.
- Becker, RA, WS Cleveland, and MJ Shyu (1996). The Visual Design and Control of Trellis Display. *Journal of Computational and Graphical Statistics* 5(2), 123–155.
- Braun, CC, PB Mine, and N Clayton Silver (1995). The influence of color on warning label perceptions. *International Journal of Industrial Ergonomics* 15(3), 179–187.
- Carslaw, DC and K Ropkins (2012). openair - An R package for air quality data analysis. *Environmental Modelling & Software* 27-28(0), 52–61.
- Chambers, JM and TJ Hastie (1992). *Statistical Models in S*. Wadsworth & Brooks/Cole.
- Chang, W, J Cheng, J Allaire, C Sievert, B Schloerke, Y Xie, J Allen, J McPherson, A Dipert, and B Borges (2021). *shiny: Web Application Framework for R*. R package version 1.6.0. <https://CRAN.R-project.org/package=shiny>.
- Cheng, J, B Karambelkar, and Y Xie (2021). *leaflet: Create Interactive Web Maps with the JavaScript 'Leaflet' Library*. R package version 2.0.4.1. <https://CRAN.R-project.org/package=leaflet>.

- Coene, J (2020). *echarts4r: Create Interactive Graphs with 'Echarts JavaScript' Version 4*. R package version 0.3.3. <https://CRAN.R-project.org/package=echarts4r>.
- Cook, D and DF Swayne (2007). *Interactive and Dynamic Graphics for Data Analysis*. New York, NY: Springer Science & Business Media, Inc.
- Crutcher, HL (1957). On the Standard Vector-Deviation Wind Rose. *Journal of the Atmospheric Sciences* **14**(1), 28–33.
- Droppo, JG and BA Napier (2008). Wind Direction Bias in Generating Wind Roses and Conducting Sector-Based Air Dispersion Modeling. *Journal of the Air & Waste Management Association* **58**(7), 913–918.
- Gore, RW and DS Deshpande (2017). An approach for classification of health risks based on air quality levels. In: *2017 1st International Conference on Intelligent Systems and Information Management (ICISIM)*, pp.58–61.
- Hyndman, R (2021a). *Forecasting: Principles and Practice*. 3rd ed. Melbourne, VIC, Australia: Monash University. <https://otexts.com/fpp3>.
- Hyndman, R (2021b). *fpp3: Data for "Forecasting: Principles and Practice" (3rd Edition)*. R package version 0.4.0. <https://CRAN.R-project.org/package=fpp3>.
- International Organization for Standardization (2019). *ISO 8601: Date and Time Format*. <https://www.iso.org/iso-8601-date-and-time-format.html>.
- Liu, J, J Li, and W Li (2016). Temporal Patterns in Fine Particulate Matter Time Series in Beijing: A Calendar View. *Scientific Reports* **6**(32221).
- Microsoft Corporation (1995). *Windows NT 3.51 Product Overview*. Accessed: 2021-10-11. Seattle, WA. <https://web.archive.org/web/20071225070412/http://support.microsoft.com/kb/124814>.
- Munn, RE (1969). Pollution wind-rose analysis. *Atmosphere* **7**(3), 97–105.
- New Zealand Parliamentary Counsel Office (2004). *Resource Management (National Environmental Standards for Air Quality) Regulations 2004*. Accessed: 2021-10-13. Wellington, New Zealand. <https://www.legislation.govt.nz/regulation/public/2004/0309/latest/DLM286835.html>.
- O'Hara-Wild, M, R Hyndman, and E Wang (2021). *feasts: Feature Extraction and Statistics for Time Series*. R package version 0.1.7. <https://CRAN.R-project.org/package=feasts>.

- R Core Team (2021). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing. Vienna, Austria. <https://www.R-project.org>.
- Rahman, NHA and MH Lee (2020). Air Pollutant Index Calendar-Based Graphics for Visualizing Trends Profiling and Analysis. *Sains Malaysiana* **49**(1), 201–209.
- Sievert, C (2020). *Interactive Web-Based Data Visualization with R, plotly, and shiny*. Chapman and Hall/CRC. <https://plotly-r.com>.
- The Apache Software Foundation (2020). *Apache ECharts: An Open Source JavaScript Visualization Library*. <https://echarts.apache.org/handbook>.
- Tufte, ER (1983). *The Visual Display of Quantitative Information*. Graphics Press.
- U.S. Environmental Protection Agency (2020). *Technical Assistance Document for the Reporting of Daily Air Quality - the Air Quality Index (AQI)*. Accessed: 2021-10-13. Research Triangle Park, NC. <https://www.airnow.gov/sites/default/files/2020-05/aqi-technical-assistance-document-sept2018.pdf>.
- Venables, WN and BD Ripley (2002). *Modern Applied Statistics with S*. 4th ed. Springer-Verlag.
- Wang, E, D Cook, and RJ Hyndman (2020). A new tidy data structure to support exploration and modeling of temporal data. *Journal of Computational and Graphical Statistics* **29**(3), 466–478.
- Wickham, H (2016). *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York. <https://ggplot2.tidyverse.org>.
- Wickham, H (2021). *Mastering Shiny*. Sebastopol, CA: O'Reilly Media. <https://mastering-shiny.org>.
- Wickham, H, J Hester, and R Francois (2018). *readr: Read Rectangular Text Data*. R package version 1.3.1. <https://CRAN.R-project.org/package=readr>.
- Wilke, CO (2019). *Fundamentals of Data Visualization*. Sebastopol, CA: O'Reilly Media. <https://clauswilke.com/dataviz>.
- Zhang, B, L Jiao, G Xu, S Zhao, X Tang, Y Zhou, and C Gong (2018). Influences of wind and precipitation on different-sized particulate matter concentrations (PM<sub>2.5</sub>, PM<sub>10</sub>, PM<sub>2.5-10</sub>). *Meteorology and Atmospheric Physics* **130**(3), 383–392.