GENEVA GRADUATE INSTITUTE

INSTITUT DE HAUTES ÉTUDES INTERNATIONALES ET DU DÉVELOPPEMENT

GRADUATE INSTITUTE OF INTERNATIONAL AND DEVELOPMENT STUDIES

On the effect of global environmental governance on innovative outcomes; Revisiting the Porter Hypothesis

THESIS
submitted at the Geneva Graduate Institute
in fulfillment of the requirements of the
Master of International Economics

by

Bernhard Bieri

Thesis No. 12345

Supervisor: Prof. Joëlle Noailly
Second Reader: Prof. James Hollway

Geneva
2022

INSTITUT DE HAUTES ETUDES INTERNATIONALES ET DU DEVELOPPEMENT
GENEVA GRADUATE INSTITUTE

RESUME / ABSTRACT

This *RMarkdown* template is for writing a Graduate Institute thesis.
This template's sample content include illustrations of how to write a thesis in *RMarkdown*, and largely follows the structure from this R Markdown workshop.
Congratulations for taking a step further into the lands of open, reproducible science by writing your thesis using a tool that allows you to transparently include tables and dynamically generated plots directly from the underlying data. Hip hooray!

Acknowledgements

I would like to express my heartfelt thanks to Prof. Joëlle Noailly for her supervision without which the present work would not exist. Her guidance on how to adress methodological issues as well as her insight in the field of environmental economics proved to be invaluable in the writing of this thesis. Additionally, I would like to thank Prof. James Hollway for the co-supervision of this thesis and in particular for the collaboration within the PANARCHIC project which allowed me to continuously refine my programming skills and discover the importance of social network analysis to further refine our understanding of social issues. Furthermore, I would like to thank Henrique Sposito, Jael Tan and Esther Peev, all members of the PANARCHIC team, for their help.

Bernhard Bieri
Geneva Graduate Institute
30 June 2022

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Chapter 2

Litterature Review

The present paper draws from three distinct but complementary strands of the literature on environmental policy. First and foremost, we ought to characterize what is known in the literature as the *Porter Hypothesis* and distinguish the different versions that have been devised over the years. We will strive to understand the impact of increased environmental regulations on both innovative activity of firms and on overall firm performance before completing the analysis by highlighting works that looked at the impact of the nature of environmental policy on firm-level innovation. The next strand of the literature relates to the concept of *policy certainty* which will allow us to Finally, and most central to our contribution to the litterature, lies the literature *Global Environmental Governance*.

2.1 The Porter Hypothesis

2.2 The Role of Policy Certainty on the Innovative Behavior of Firms

2.3 The Evolution of Global Environmental Governance

Chapter 3

Methods

Chapter 4

Results

4.1 An intro to R chunks

One of the most useful parts of *RMarkdown* is the ability to weave or 'knit' text (as we've seen in the previous three chapters) together with the code that produces tables and graphics for your final document. This has various advantages such as:

- *coherence*: for you, having your analysis and text together means keeping a coherent story together about why you did the analysis you did, how you got to your results, and what your interpretation of these results is
- *amendability*: this gives you an opportunity to update the code, for example by filtering the data in a different way, and rerun the code to see how any statistics, tables or graphs related to the data may have changed
- *reproducibility*: this also, importantly, supports a reproducible workflow such that you, or others, can rerun and test the code at a later point

To 'knit' such text and code together, code for particular graphs or tables should first be inserted into the text in 'chunks'. These chunks are opened with three backticks and closed in a new line with another three backticks. The code that produces the desired output is included between these lines, which can include comments etc. A key part of these chunks are the chunk parameters, which are included in braces immediately following the first set of backticks. The first parameter indicates the programming language. The second parameter, which is optional and should be separated from the first parameter only by a space, is a chunk label. It is important that this chunk label includes no spaces and is unique to this chunk. Then, separated by commas,

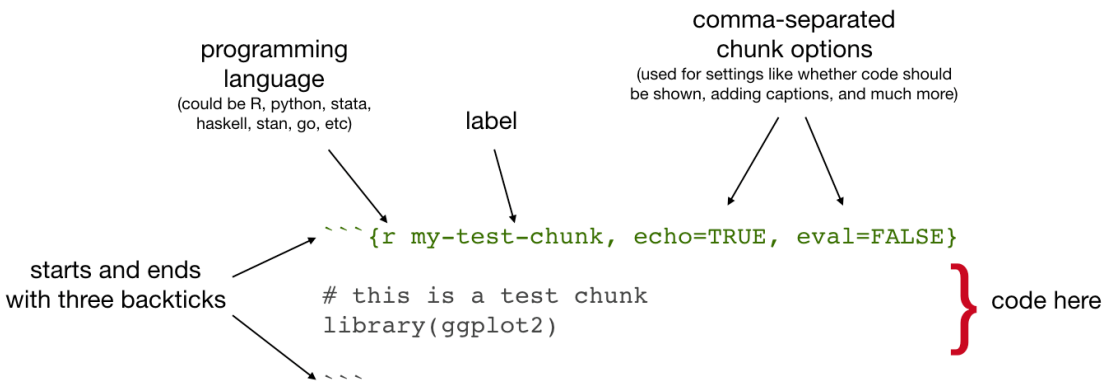


Figure 4.1: Chunk parts

are any additional parameters included. In the figure below, two examples are included: whether or not the code itself should be shown (`echo`) and whether or not the code should be evaluated or run (`eval`). Other parameters that we will see later include figure width, captions, etc.[1]

In the code chunk shown in Figure 4.1, we can see that it is loading a package library. Functions and objects loaded or created in earlier chunks are cached by default and thus held on to for use in later chunks. It is therefore quite common to have a chunk early on in your chapter (or even your whole dissertation) that loads a number of the packages you need later on. As an example, below is a short code chunk that loads a set of **R** packages that we are going to rely on for the rest of this chapter.

```
# Load packages

library(dplyr)

library(ggplot2)

library(knitr)
```

When you click the **Knit** button above in RStudio a document will be generated that includes both content as well as the output of any embedded **R** code chunks within the document. If you need a graphic or tabular material to be part of the text, you can just put it inline. If you need it to appear in the list of figures or tables, it should be placed in a code chunk.

In the remainder of this chapter, we will see how to use code chunks to include figures, load and summarise data, and present results from your analysis.

[1] More information is available at `https://yihui.org/knitr/options/`.

Figure 4.2: IHEID logo

4.2 Including external figures

First, we will treat how to include figures in *RMarkdown* in more detail. If your thesis has a lot of figures, *RMarkdown* might behave better for you than your usual word processor, which can have a tendency to . One perk is that it will automatically number the figures accordingly in each chapter. You'll also be able to create a label for each figure, add a caption, and then reference the figure in a way similar to what we saw with tables earlier. If you label your figures, you can move the figures around and *RMarkdown* will automatically adjust the numbering for you. No need for you to remember! So that you don't have to get too far into LaTeX to do this, a couple **R** functions have been created for you to assist. You'll see their use below.

4.2.1 Numbering and labelling figures

You don't need to number figures in *RMarkdown*; it'll do it automatically. And captioning a figure is awfully easy too. In the **R** chunk below, we will load in a picture that is stored as `iheid.png` in our `figures` directory. The `include_graphics()` function is from the `{knitr}` package, that does most of the heavy lifting here. We'll label the chunk 'iheidlogo' so that we can refer back to it later, and give the figure the caption "IHEID logo". You will see that here you do not need to include the parameter `echo=FALSE`, as the code including the figure is hidden by default.

```
include_graphics(path = "figures/iheid.png")
```

4.2.2 Referencing figures

Referencing figures in the text should be done with a little reference to the chunk label so that it will always refer to the right figure number, even if you add additional figures and plots before it. To reference the IHEID logo use a backslash, at-symbol, and then in parentheses

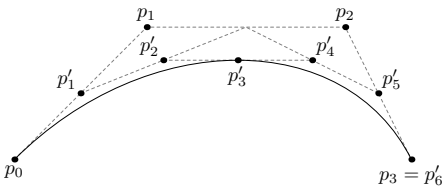


Figure 4.3: Subdiv. graph

`fig:chunk-label`, like so `\@ref(fig:iheidlogo)`. Usually some descriptor like "Figure", "Fig.", "Illustration" or other is added before this reference, so that it reads something like Figure 4.2. Note that the reference is hyperlinked in the resulting PDF, so that you can click on it to take you to the page where the original figure is printed.

4.2.3 Resizing figures

It is common to resize external image files so that they fit the format of the text. The most common way to do this was already demonstrated in the code chunk for Figure 4.1 above. There we used the parameter `out.width="\\textwidth"` to, in this case, shrink the image to the width of the text, but this same parameter specification would also expand an image to fit the width of the text where possible.

Another option is the `out.extra` chunk option. This can be used to shrink or expand an image loaded from a file more specifically by specifying `"scale= "`. Here we use the graph stored in the "subdivision.pdf" file, again found in the `figures` subdirectory. The output can be found in Fig. 4.3.

```
include_graphics("figures/subdivision.pdf")
```

We can also use the `out.extra` chunk option to enlarge figures, and even to rotate them any number of degrees around its axis. In Figure 4.4, we see an example where Figure 4.3 has been enlarged and flipped upside down.

```
include_graphics("figures/subdivision.pdf")
```

If you look closely at the chunk options, you will also see that two different captions have been provided. The `fig.scap` is a short caption that overrides the main caption, `fig.cap`, in the

Figure 4.4: A Larger Figure, Flipped Upside Down

table of contents. This can be very useful where, for example, you need to describe the figure in a caption over several lines, details that do not need to be presented in the table of contents.

4.2.4 Placing figures

One thing that may be annoying is the way *RMarkdown* handles "floats" like tables and figures (it's really LaTeX's fault). LaTeX will try to find the best place to put your object based on the text around it and, until you're really, truly done writing, it is best to just leave it where it lies. There are some optional arguments specified in the options parameter of the `label` function. If you need to shift your figure around, it might be good to look here (you can click on the word 'here') on tweaking the options argument.

Spacing out your chunks between paragraphs can help, as it can give LaTeX more options to find a suitable home for the figure or table, as LaTeX would otherwise try and keep all the text together, saving the image for later. A last trick is to write `\clearpage` directly in your *RMarkdown* document. This gives LaTeX a chance to catch up with all the 'floats' it has accumulated, and starts a new page.

4.3 Loading and exploring data

Sometimes it is not an existing image that must be imported, but instead you wish to create a table or plot of some data that you have imported. In this section, we're going to very quickly cover how to import internal and external data, how to manipulate and summarise the data, create plots from that data, and tabulate inferential results from that data.

4.3.1 Importing data

In some cases, the data you need might be available already in an **R** package. An example below is a very short code chunk that summarises a dataset that is built into **R**. Here it is particularly easy, as we do not even need to load the data (it is loaded by default).

```
summary(cars)
```

```
##      speed           dist
##  Min.   : 4.0   Min.   :  2.00
##  1st Qu.:12.0   1st Qu.: 26.00
##  Median :15.0   Median : 36.00
##  Mean   :15.4   Mean   : 42.98
##  3rd Qu.:19.0   3rd Qu.: 56.00
##  Max.   :25.0   Max.   :120.00
```

Othertimes you may need to load the data first before summarising it, with the function `data()`. There are an increasing number of data packages for **R** available, including several on a range of international and development topics.

The other option is to import data that resides in a file outside of **R**. Included in this template is a file called `flights.csv`. This file includes a subset of the larger dataset of information about all flights that departed from Seattle and Portland in 2014.[2] This subset includes only Portland flights and only rows that were complete with no missing values. Merges were also done with the `airports` and `airlines` data sets in the `pnwflights14` package to get more descriptive airport and airline names. We can load in this data set using the following commands:

[2]More information about this dataset and its **R** package is available at `https://github.com/ismayc/pnwflights14`.

```
# flights.csv is in the data directory
flights <- read.csv("data/flights.csv")
```

The data is now stored in the data frame called `flights` in the cached environment for this *RMarkdown* document in **R**. To get a better feel for the variables included in this dataset we can use a variety of functions. Here we can see the dimensions (rows by columns) and also the names of the columns.

```
dim(flights)
```

```
## [1] 398  16
```

```
names(flights)
```

```
##  [1] "month"        "day"          "dep_time"     "dep_delay"    "arr_time"
##  [6] "arr_delay"    "carrier"      "tailnum"      "flight"       "dest"
## [11] "air_time"     "distance"     "hour"         "minute"       "carrier_name"
## [16] "dest_name"
```

Another good idea is to take a look at the dataset in table form. With this dataset having more than 20,000 rows, we won't explicitly show the results of the command here, so we'll use the `eval=FALSE` chunk option to make sure the following is not run when you 'knit' the document. Still, you can press the green play button at the right of the chunk to run the chunk on demand, bringing up a new tab showing the data.

```
View(flights)
```

4.3.2 Manipulating and summarising data

While not required, it is highly recommended you use the `dplyr` package to manipulate and summarize your data set as needed. It uses a syntax that is easy to understand using chaining or 'pipe' operations (`%>%`). Below I've created a few examples of using `dplyr` to get information about the Portland flights in 2014. The example we show here does the following:

- Selects only the `carrier_name` and `arr_delay` from the `flights` dataset and then assigns this subset to a new variable called `flights2`.
- Using `flights2`, we determine the largest arrival delay for each of the carriers.

```
flights2 <- flights %>%
  select(carrier_name, arr_delay)
max_delays <- flights2 %>%
  group_by(carrier_name) %>%
  summarize(max_arr_delay = max(arr_delay, na.rm = TRUE))
```

A useful function in the `knitr` package for making nice tables in *RMarkdown* is called `kable`. It is much easier to use than manually entering values into a table by copying and pasting values into Excel or LaTeX. This again goes to show how nice reproducible documents can be! The chunk option `results="asis"` makes sure the table is produced, not the code to create the table. Tables created with the `kable()` function (in `{knitr}`) can be extended in many useful (and pretty) ways with the recommended `{kableExtra}` package.

```
kable(max_delays,
  col.names = c("Airline", "Max Arrival Delay"),
  caption = "Maximum Delays by Airline",
  caption.short = "Max Delays by Airline",
  longtable = TRUE,
  booktabs = TRUE
)
```

Table 4.1: Maximum Delays by Airline

Airline	Max Arrival Delay
Alaska Airlines Inc.	181
American Airlines Inc.	72
Delta Air Lines Inc.	84
Frontier Airlines Inc.	68
Hawaiian Airlines Inc.	6
JetBlue Airways	172
SkyWest Airlines Inc.	103
Southwest Airlines Co.	315
United Air Lines Inc.	135
US Airways Inc.	347
Virgin America	22

Note that instead of adding the caption details in the chunk options, we will be adding this in the `kable()` function, which then passes this on up. The `caption.short` argument is used to include a shorter title to appear in the List of Tables. The last two options make Table 4.1 a little easier-to-read.[3]

We can further look into the properties of the largest value here for Alaska Airlines Inc. To do so, we can isolate the row corresponding to the arrival delay of 70 minutes for Alaskan in our original `flights` dataset. We see that the flight occurred on January 1st and departed a little after 1:30 am on its way to Anchorage.

```
flights %>%
  dplyr::filter(
    arr_delay == 70,
    carrier_name == "Alaska Airlines Inc."
  ) %>%
  select(-c(
    month, day, carrier, dest_name, hour,
    minute, carrier_name, arr_delay
  ))
```

[3]Note that we can create references/links to tables using a very similar syntax here to that with figures above. We can preface the reference with "Table" or "Tab." or so, and then create the reference with the format `\@ref(tab:chunk-label)`.

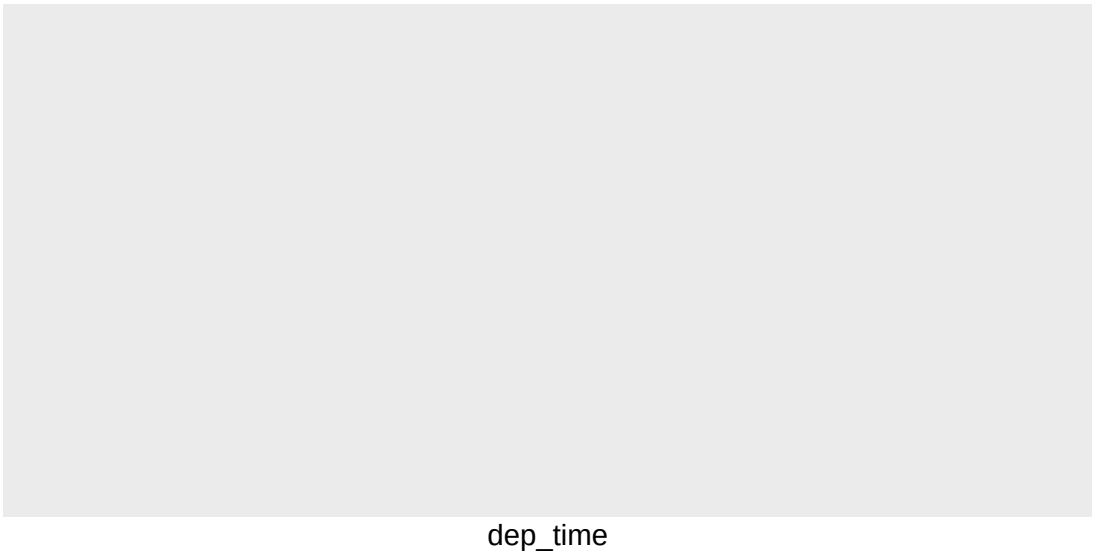
```
##   dep_time dep_delay arr_time tailnum flight dest air_time distance
## 1        1        96      235  N508AS    145  ANC      194     1542
```

4.3.3 Creating plots

Once data has been loaded or imported and manipulated or filtered as required, a common task is to visualise some key dimensions of the data to inform the reader. Here the package `{ggplot2}` plays nicely with `{dplyr}` and other `{tidyverse}` packages. `{ggplot2}` produces beautiful, high-quality academic visuals, and has been extended with a huge range of add-ons for a very broad variety of visualisation styles no matter what kind of data you have.

We're going to continue playing with the `flights` dataset from Chapter 4.3.1. First, let us show how we can visualize the arrival delay of all departing flights from Portland on March 3rd against time of departure. Note that once you open the plotting function with `ggplot()`, additional elements are chained not with the pipe operator `%>%` but `+`. Considerably more help than I can offer here on how to use `ggplot()` can be found online.

```
flights %>%
  dplyr::filter(month == 3, day == 3) %>%
  ggplot(aes(x = dep_time, y = arr_delay)) +
  geom_point()
```



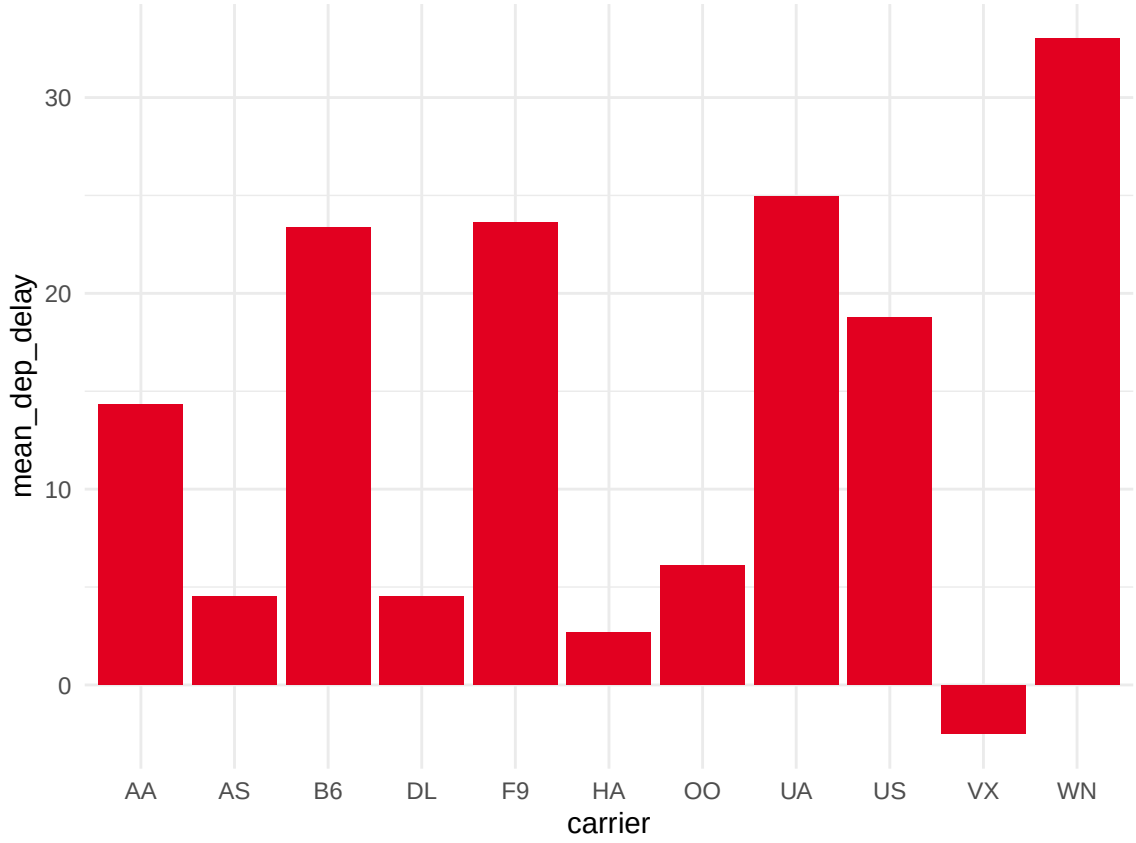
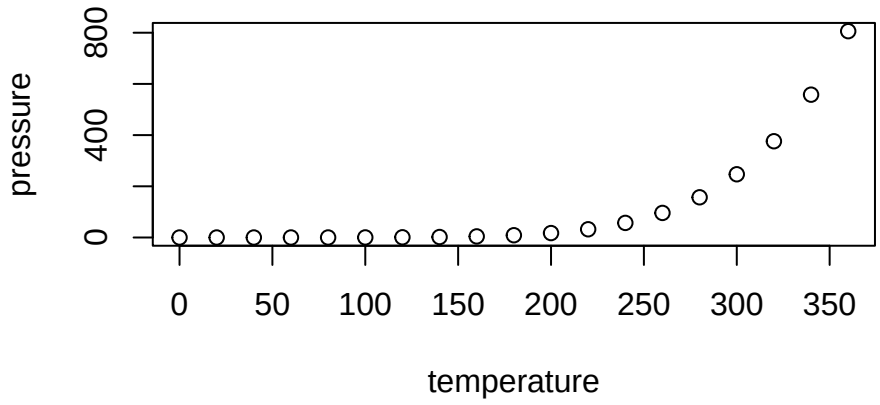


Figure 4.5: Mean Delays by Airline

Next Figure 4.5 presents a bar graph with the mean flight departure delays by airline from Portland for 2014. A table linking these carrier codes to airline names is available at `https://github.com/ismayc/pnwflights14/blob/master/data/airlines.csv`.

```
mean_delay_by_carrier <- flights %>%
  group_by(carrier) %>%
  summarize(mean_dep_delay = mean(dep_delay))
ggplot(mean_delay_by_carrier, aes(x = carrier, y = mean_dep_delay)) +
  geom_bar(position = "identity", stat = "identity",
           fill = iheiddown::iheid_palette("IHEID", 1)) +
  theme_minimal()
```

You don't have to use `{ggplot2}` though. For example, here is a way to use the base **R** graphics package to produce a plot using the built-in `pressure` dataset:



4.3.4 Tabulating inferential results

Another common task researchers have is the presentation of results obtained from applying various statistical methods to their data. Since **R** makes available a very broad range of statistical methods, and all too often these output objects in different structures and formats, there is unfortunately no single package that reliably and prettily prints results. Still, I can offer a few suggestions here that cover as much as possible, in addition to `{kable}` and `{kableExtra}` mentioned above.

- `{stargazer}` for well-formatted regression tables, with multiple models side-by-side, as well as for summary statistics tables, data frames, vectors and matrices.
- `{xtable}` offers a straightforward but extensible framework.
- `{memisc}` provides tools for the management of survey data, as well as the creation of tables of summary statistics and model estimates.
- `{Hmisc}` for data description and predictive modeling.
- `{finalfit}` brings together the day-to-day functions we use to generate final results tables and plots when modelling.
- `{tableone}` i.e., description of baseline patient characteristics.
- `{flextable}` provides a framework for easily create tables for reporting and publications (PDF documents only with package `{pagedown}`).
- `{huxtable}` creates LaTeX and HTML tables with a friendly, modern interface.

- `{texreg}`
- `{apsrtable}`
- `{arsenal}`

I (James Hollway) would welcome any feedback on any other packages you find useful (more are being released all the time), and which of these worked for your purposes.

4.3.5 Tables from external data

In addition to the tables that can be automatically generated from data in **R**, you can also create tables directly using *pandoc*. This might be useful if you don't have values specifically stored in **R**, but you'd like to display them in table form.

Below is an example.

Pay careful attention to the alignment in the table and hyphens to create the rows and columns. More information is available at `https://pandoc.org/README.html#tables`.

Table 4.2: Correlation of Inheritance Factors for Parents and Child

Factors	Correlation between Parents & Child	Inherited
Education	-0.49	Yes
Socio-Economic Status	0.28	Slight
Income	0.08	No
Family Size	0.18	Slight
Occupational Prestige	0.21	Slight

The addition of the `(\#tab:inher)` option to the end of the table caption allows us to then make a reference to Table `\@ref(tab:label)`. Note that this reference could appear anywhere throughout the document after the table has appeared.

Chapter 5

Conclusion

Appendix: The Echoes of the Code

The goal of this appendix is to echo the code you used in your thesis for a greater sense of transparency and replicability of your research. Note that `ref.labels` can be set to any label. Hence, you can filter the code you want replicated in the appendix by setting labels to the desired code chunks in the various chapters. See this excellent resource for more information.

This might be particularly useful when you perform model selection to output intermediary steps here instead of in the code to avoid cluttering your report.

```
knitr::write_bib(c(.packages(), "bookdown"), "bib/packages.bib")
knitr::include_graphics("figures/chunk-parts.png")
# Load packages
library(dplyr)
library(ggplot2)
library(knitr)
include_graphics(path = "figures/iheid.png")
include_graphics("figures/subdivision.pdf")
include_graphics("figures/subdivision.pdf")
summary(cars)
# flights.csv is in the data directory
flights <- read.csv("data/flights.csv")
dim(flights)
names(flights)
View(flights)
flights2 <- flights %>%
  select(carrier_name, arr_delay)
```

Appendix

```
max_delays <- flights2 %>%
  group_by(carrier_name) %>%
  summarize(max_arr_delay = max(arr_delay, na.rm = TRUE))
kable(max_delays,
  col.names = c("Airline", "Max Arrival Delay"),
  caption = "Maximum Delays by Airline",
  caption.short = "Max Delays by Airline",
  longtable = TRUE,
  booktabs = TRUE
)
flights %>%
  dplyr::filter(
    arr_delay == 70,
    carrier_name == "Alaska Airlines Inc."
  ) %>%
  select(-c(
    month, day, carrier, dest_name, hour,
    minute, carrier_name, arr_delay
  ))
flights %>%
  dplyr::filter(month == 3, day == 3) %>%
  ggplot(aes(x = dep_time, y = arr_delay)) +
  geom_point()
mean_delay_by_carrier <- flights %>%
  group_by(carrier) %>%
  summarize(mean_dep_delay = mean(dep_delay))
ggplot(mean_delay_by_carrier, aes(x = carrier, y = mean_dep_delay)) +
  geom_bar(position = "identity", stat = "identity",
           fill = iheiddown::iheid_palette("IHEID", 1)) +
```

```
  theme_minimal()

plot(pressure)
```

Appendix: The Echoes of the Code redux

Add as many appendices as you like.

References

Darwin, Charles (1859). *On the Origin of Species by Means of Natural Selection or the Preservation of Favoured Races in the Struggle for Life*. London: John Murray.

Hollway, James (2020). *iheiddown: A RMarkdown class for IHEID dissertations*. R package version 0.0.1. URL: https://github.com/jhollway/iheiddown.

Lottridge, Danielle et al. (2012). "Browser design impacts multitasking". In: *Proceedings of the Human Factors and Ergonomics Society 56th Annual Meeting*. ISBN: 9780945289418. DOI: 10.1177/1071181312561289.

R Core Team (2022). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing. Vienna, Austria. URL: https://www.R-project.org/.

Satchell, Christine and Paul Dourish (2009). "Beyond The User: Use And Non-Use in HCI". In: *Proceedings of the Annual Conference of the Australian Computer-Human Interaction Special Interest Group (OZCHI '09)* November, pp. 9–16. ISSN: 14712970. DOI: 10.1145/1738826.1738829. URL: http://douri.sh/publications/2009/nonuse-ozchi.pdf.

Shea, Nicholas et al. (2014). "Supra-personal cognitive control and metacognition". In: *Trends in Cognitive Sciences* 18.4, pp. 186–193. ISSN: 1364-6613. DOI: 10.1016/j.tics.2014.01.006. URL: http://dx.doi.org/10.1016/j.tics.2014.01.006.

Solomon, Nick (2020). *thesisdown: An updated R Markdown thesis template using the bookdown package*. R package version 0.0.2.

Von Goethe, Johann Wolfgang (1829). *Wilhelm Meisters Wanderjahre oder die Entsagenden*. de. Cotta.

Wu, Tim (2016). *The Attention Merchants: The Epic Scramble to Get Inside Our Heads*. Knopf Publishing Group.

References

Xie, Yihui (2016). *bookdown: Authoring Books and Technical Documents with R Markdown*. ISBN 978-1138700109. Boca Raton, Florida: Chapman and Hall/CRC. URL: https://github.com/rstudio/bookdown.

Xie, Yihui (2022). *bookdown: Authoring Books and Technical Documents with R Markdown*. R package version 0.26. URL: https://CRAN.R-project.org/package=bookdown.