Getting Started with Thesis Writing in R

by

Christelinda Laureijs

A short guide for curious and motivated researchers

June 17, 2024

COLOPHON

This document was typeset in R Markdown and the text was set in EB Garamond. The data and code used to produce this document are available at *https://github.com/christelinda-laureijs/reproducible-thesis*.

Contents

List of Figures

Introduction

Writing a paper in R can be challenging at first, but it is a rewarding process that will save you lots of time, effort and stress over the long term. This document provides examples and sample code for you to get started. It is also a sample RMarkdown document to see if you can knit PDFs from R and RStudio. Ideally, you should be able to click on Knit -> Knit to PDF and generate a PDF without issues.

A note for Mac Users: When you knit the document the first time, you might see error messages with something about Cairo and your graphics device. Every device is different, but you might be able to fix it by installing XQuartz and then Cairo.

What are the advantages of writing in R?

Reproducibility

Your analyses and text are all in one place, so you will always know what you did to create a specific plot or how you got a certain p-value. Others will be able to follow your process by reading your code, and if you make your code and data availble, they can replicate your analyses (and ideally end up with the same numbers!).

Efficiency

Rather than typing the same things over and over again, you can set up functions to complete tasks like completing a statistical test or creating a plot. By using in-text R code, you won't need to manually type p-values and statistics and change them each time you re-do your analysis (hurray!!). Once your code is set up, it is easy to run your paper again with new data, and generate new plots, p-values, and statistical output tables.

Since you can define your colours, fonts, and ggplot theme at the beginning of your document, you don't need to fiddle with formatting each plot. All plots will have the same formatting, and it is easy to change colours/styling for all plots at once.

No manual formatting

You don't need to worry about the layout of your document at all. You do not need to manually number your figures or subheadings, format your Table of Contents, or readjust paragraphs each time you insert or remove a picture. You can easily cross-reference figures, equations and chapters. To insert a list of figures, just type \listoffigures - see, it's very doable!

Since RMarkdown is a plain-text document, your computer will be able to handle large documents much more easily than Word. It will not get buggy or slow when you have lots of high-resolution pictures.

Beautiful typography

When you knit a document using a PDF, R uses LaTeX. LaTeXis a typesetting engine that uses mathematical algorithms to arrange text into the most ideal way according to typesetting rules. The resulting documents look very elegant in a way that is hard to quantify. LaTeX-produced documents look very distinctive and after a while, you'll be able to easily tell the differences between a document produced using LaTeXvs. Word.

Even if you stick to just the defaults, your documents will look great. If you need to use equations, LaTeXis one of the best ways to create clean, well-aligned equations.

Free

R, Rmarkdown, LaTeXand the packages you'll use are all free and open-source.

Longevity

Your documents are plain text files, which means that they don't take up much space on your computer, and you can open them up years later in any plain text editor (even Notepad!). It also means that your files will always be available, and you won't get locked into proprietary software.

Version Control

It is easy to set up version control with tools like Git and GitHub (which is also free). I would highly recommend learning how to set up a project on GitHub. Each time you make a change to the document on your computer, Git will also save the changes to the online version of your file, hosted on GitHub. It means that you'll always have a backup copy, and if something wrong happens, you can revert back to an older version and see what changes you made.

What are the disadvantages of writing in R?

Learning curve

If you're not familiar with R, RMarkdown, and LaTeX, it will take much longer to set up your paper than in Word. It will also take a while to format your preamble to make your document look exactly the way you want it. You'll likely spend a lot of time googling things and reading through answers on StackOverflow and GitHub.

Troubleshooting

Things will break down, and sometimes you'll spend way more time than you had anticipated dealing with error messages. During these times, it does seem much easier to just open a Word document and type there.

To minimize errors, I would suggest these things:

- Run each code chunk from top to bottom to catch any errors. Ensure that everything runs before knitting.
- Knit your document frequently as you go to help spot errors.
- R will knit documents using a blank R session, so everything that needs to be in the document must be defined in your script.
- Turn off the "Save workspace image" option in your R Global Options to prevent old loaded packages and hidden variables from 'hovering' in the background and creating strange rrors.

- Frequently use Run -> Restart R and clear output to prevent objects from cluttering your workspace and causing dependencies. For example, if you define a variable in the console, but not your document you won't realize the problem until you have a fresh R session.
- Use knitr::knit_exit() to stop knitting early. It can be a useful way to identify the specific line of code that is causing knitting issues.
- ! Make sure that your PDF is closed when you knit. If you're knitting the file and the PDF is still open, it will break the code.

Separate content and layout

You don't get to see what your document looks like until you knit it. This can be disconcerting for some people but great for others because you aren't distracted by formatting. Although you should knit your document often, don't fuss around with formatting until your paper is almost done. Things like paragraph spacing will change, and it is better to wait until the end.

At the beginning, you may worry about floats (things like pictures, plots, and tables). LaTeXwill 'float' these over the text and then plop them down in a way that minimizes the number of paragraph breaks and blank spots. This means that your floats will often be further from where you want them. It is very, very difficult to 'force' floats to go into a specific spot, and the place that LaTeXchooses is often the best layout-wise. Always use references like Figure 1, rather than "the figure below".

Collaboration issues

Unlike Word, you can't use track changes, comments, or shared files. The best way to simulate this is to set up a GitHub repository and add your supervisor as a collaborator. They could then use pull requests to suggest changes. This may cause issues if you have a supervisor who doesn't know how to use GitHub. You also may not be able to make this work if your supervisor doesn't want to or know how to comment a PDF.

How-Tos

Inserting citations

Here is an example of an in-text citation: Insulin is a hormone that regulates blood glucose levels, digestive processes, and body weight (De Meyts, 2000). When you write [@demeyts2000], the citation will be rendered as an in-text citation. This document uses APA formatting because of the 'apa.csl' file. If you want a different format, you can download the appropriate CSL file online.

Inserting Plots

You can plot figures in R. This is one of the best parts - R will generate the plots each time the document knits so you don't have to repeatedly copy and paste pictures into your paper!

Inserting Images

You can insert images through knitr. fig.cap can be quite long, and you can also add a label in the form of '\\label{text}' When you write a reference to Figure '\\label{insulin-pathway}', LaTeXwill print out Figure 2.

Important: To use `fig.cap` and `fig.scap`, you must define an `out.width` in the chunk options. If not, R will not recognize these as LaTeXcommands.

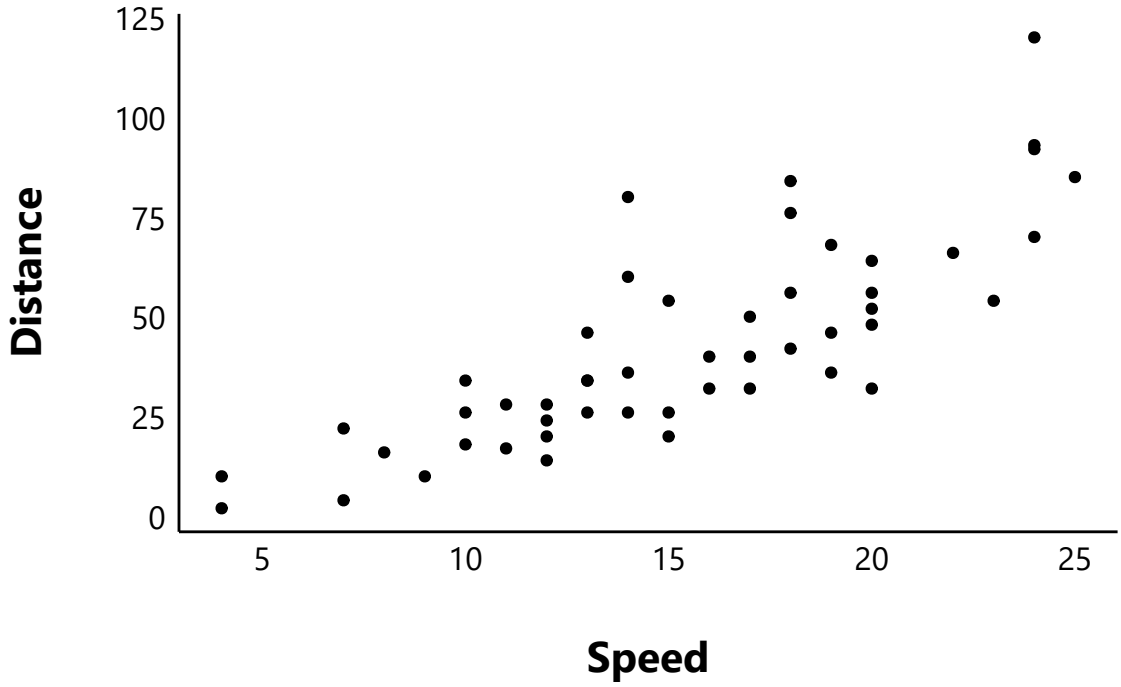

Figure 1: Here is an example of a figure caption

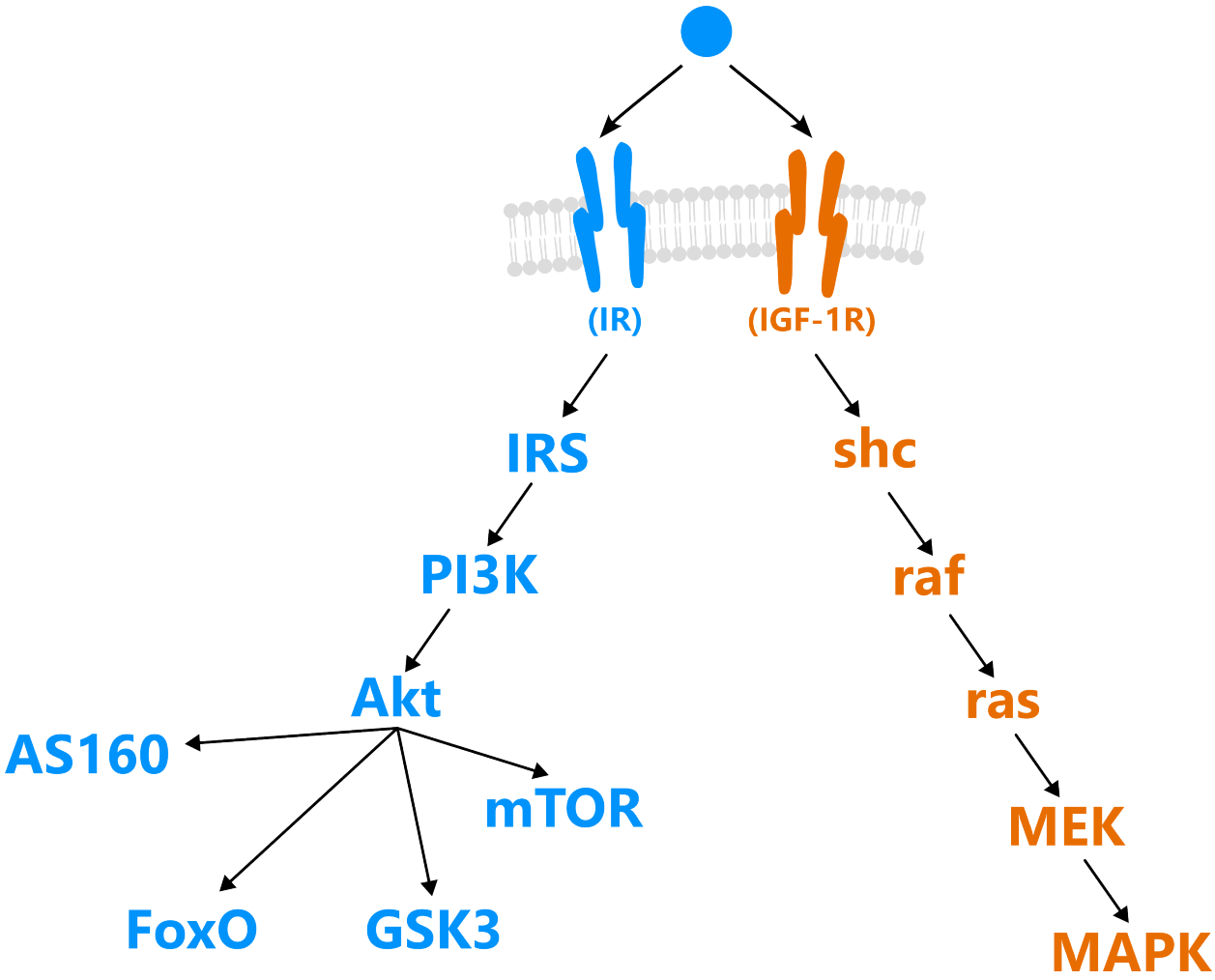

Figure 2: When insulin binds to a cell, it activates a series of molecular pathways involved in energy metabolism and gene expression...[if your figure caption is too long, you should define a short figure caption (fig.scap) which will go into the list of figures]...although it looks like there are two distinct pathways, the downstream components of the P13K and MAPK pathways frequently interact with one another (De Meyts, 2000).

In-text R code

You can embed R code within normal text to create dynamic reports. For example, if you write ‘ r nrow(cars) ‘, this will be printed out as 50.

E.g. There were 50 cars in the dataset.

You can take this even further to embed p-values, t-test statistics, and other values within the body text. This will save you lots of time in the results section. It also reduces the likelihood of mistakes.

It will take some time to find out how to extract p-values from different data types. In RStudio, always check what class your model is. Then, you can google things like “Extract p-value from an object of class anova in r”.

Other helpful functions will include str() to see the structure of the model. This can provide a list of the parts of the model and give you an idea of which variables you need to select the specific p-value that you want.

Tables

There are many packages that will allow you to create publication-quality tables. I’ve found that the packages kable and broom are particularly useful.

LaTeX Basics

You can learn a lot about LaTeXas you customize your paper. This is a very valuable skill, and it will be particularly useful if you’re planning to stay in academia.

Most LaTeXcommands start with a forward slash and include arguments in curly brackets. Many of them are intuitive:

\setcounter{tocdepth}{2} to set the table of content depth to header 2 \tableofcontents to insert a table of contents

If you want to make any cross references, you can define them using labels. You may have seen this in the *Inserting Images* section. For example, you can write Figure '\\label{insulin-pathway}' to automatically include the correct figure number.

It's not recommended to do too much formatting within your document. Ideally, all of your formatting styles should be defined in the preamble.

Changing the formatting

If you want to change how any part of this document looks, go to Templates/MtA-Thesis-Preamble.tex.

You can learn a lot about LaTeXjust by customizing the preamble. Most LaTeXcommands start with a forward slash and include arguments in curly brackets. Many of them are intuitive:

\listoffigures

Other Tools

To create high-quality schematics, I would highly recommend Inkscape, which is a free and open source vector editor. You're probably familiar with using the shapes tools in PowerPoint. Inkscape is like this, but you have much more control over the alignment, and many helpful tools that allow you to do more advanced techniques.

References

De Meyts, P. (2000). The Insulin Receptor and Its Signal Transduction Network. In K. R. Feingold, B. Anawalt, M. R. Blackman, A. Boyce, G. Chrousos, E. Corpas, W. W. de Herder, K. Dhatariya, K. Dungan, J. Hofland, S. Kalra, G. Kaltsas, N. Kapoor, C. Koch, P. Kopp, M. Korbonits, C. S. Kovacs, W. Kuohung, B. Laferrère, ... D. P. Wilson (Eds.), *Endotext*. MDText.com, Inc.