Computational Tools for Political Science Research

Summer 2024 – Vanderbilt University

Nguyen T. Ha

In the computational sessions of Math Camp, we will spend most of our time together getting comfortable with several tools that will accompany your research journey in the years to come. In our five sessions, we will cover the basics of the (1) the programming language R its common IDE RStudio, and (2) the typesetting languages LaTeX and Markdown. For the most part, we focus on these tools for practical reasons. To start, you will complete most of your statistics problem sets using R. On the other hand, despite many of its annoying quirks, LaTeX is helpful for typing up problem sets you will encounter in both the statistics and game theory sequences.

The broader goal of these sessions is to get you started on thinking about how to best organize your workflow. What we want as a collective of political scientists is to produce research that is transparent and reproducible (Lupia and Elman 2014; Elman, Kapiszewski, and Lupia 2018). Yet, research is an inherently messy process that involves many separate tasks. When starting a new project, how should you organize your brainstorming notes? How do you take systematic notes to make your literature review a breeze? Collecting, exploring, and analyzing your data will have you going back and forth between many different documents and programs. You will sometimes make mistakes, because there are simply too many things to manage. It gets tedious, too.

Fortunately for us, there are many readily available tools—many of which are free and open-source—that can assist us in our research goals. The programming and typesetting languages that we will cover in this one week are not the only tools available to us, and in fact you will probably end up adopting new skills to support your research agenda. As Prof. Brenton Kenkel once advised in his "Preparing for the Job Market" professionalization seminar, you should think of computational skills as your *force multipliers*—tools that amplifier your research output. The sooner you gain commands of these tools, the more freedom you will have to explore and express your ideas. To this end, aside from R and LaTeX, we will also briefly touch on files organization and note-taking approaches. In your problem sets, you will also practice using ChatGPT in ways that can help you learn

a new programming language.

1 Prerequisites and expectations

None! The content and exercises of the lectures are designed under the assumption that everyone has no prior experience. Bring your best self and a working computer.

To save time, though, we will want to get a few things installed before our first session on Friday, August 9.

1.1 R Installation

We will be using R inside RStudio throughout, with the exception of the first R problem set. What is the difference between R and RStudio, you ask? RStudio is an integrated software environment or an IDE. In general, you can run R code in RStudio or other IDEs such as Visual Studio Code, Python Notebook, or a Google Colab notebook. You can even run R out of a console terminal. On the other hand, you cannot run RStudio without R. RStudio is a popular IDE among R users for several reasons. First, you can view which objects are available in your working environment, whereas if you're running R with a Python Notebook or in a terminal console, you will have to list the objects by yourself. Second, many people find the the plot output pane in RStudio quite convenient. Another reason why RStudio is popular is perhaps path dependency—many of us started out learning R using RStudio

You can download the latest version of R here: https://www.r-project.org/

You will need to download RStudio next: https://posit.co/download/rstudio-desktop/

1.2 LaTeX

Similar to R, LaTeX refers to the underlying engine that you will use to compile your documents.[1] On the surface, you can use any text editor you prefer. These days, Overleaf seems to be the most popular option for personal and collaborative use. I recommend getting your first LaTeX projects started in Overleaf to get used to the syntaxes. Its main advantages are that it is a browser-based TeX editor with quick preview of your documents, and is especially convenient when it comes to troubleshooting the little things. For collaboration, Overleaf's ability to leave comments and track changes makes it similar to MS Word.

1. You may have heard about TeX instead. LaTeX is basically built on TeX, with added macros so that you don't have to spend time configuring basic formatting (paper size, margins, etc.) from scratch.

Because Overleaf is online, there is no installation needed. Recently, however, Overleaf has implemented a cap over the compilation time for its free accounts, so it can be worthwhile to find an alternative. My weapon of choice is rather old-school: the very minimal TeXworks editor. Other popular options are TeXstudio and TeXmaker. Recently, I have been getting used to writing my LaTeX documents in Visual Studio code.

To get started with non-Overleaf options, download your TeX engine here. Then, pick one of the IDEs and follow its installation instructions.

2 Schedule and content overview

Materials relevant to the course are stored and updated each day in this Box[2] folder:
https://vanderbilt.box.com/s/g2vbaxipw3tdz5upr7wu6zhkfpntgkj5

Friday August 9th: Getting started with R and LaTeX

In this session, we will

1. make sure that everyone has what they need to get R and a TeX editor going.
2. learn the basic syntax of R.

Homework

1. Read Dr. Nick Bednar's excellent introduction to LaTeX, and complete the exercises in `TeX-homework.pdf`.
2. The second set of exercises introduces you to R outside of the RStudio environment: https://nguyentha.shinyapps.io/Intro-to-R/

Monday August 12th: Data transformation and visualization with R

Following our introduction to R in the previous week, in this session, we will learn about simple steps to clean, wrangle, and visualize data.

Homework: Read this short chapter on R style guide (yes, there's a "style" for coding!) http://adv-r.had.co.nz/Style.html.

Tuesday August 13th: Exploratory data analysis

One thing I wish I had done more in my first few years of graduate school is descriptive analysis. Often in coursework, we put a lot of emphasis on finding puzzles and building

2. Vanderbilt offers us free access to Box, which is a service that is similar to Dropbox for storing and sharing files. I recommend getting your Box account activated and using it to back-up your PC.

theories. However, without description, it is difficult to know if the "puzzles" you believe you have uncovered are actually grounded in empirical variations. In this module, we will apply the skills we have previously learn to explore an existing dataset. After that, we will learn how to report our work in R using RMarkdown.

Your problem set for today will involve all the skills we have covered so far. Follow the instructions in today's folder.

Thursday August 15th: Programming basics

In exploratory data analysis, we tend to end up with a lot of repetition. We may want to create a similar type of graph for different variables. Or we may want to relabel a bunch of variables with similar structures. Our natural inclination in these cases is often to copy-and-paste code segments. If we end up changing one detail in the code, however, we will end up having to manually make the same changes everywhere. This can be dangerous because it creates room for mistakes. Instead, we want to adopt good programming practices and avoid repetition by writing functions and use iterations. We will learn to write *loops* and *functions* to automate this process and simplify our code. And, because automation can get computationally costly, we will discuss *vectorization* as the main guiding principle for writing efficient codes. There will be no problem set for this session since we will be spending most of our time working on in-class exercises already.

Friday August 16th: Designing your own workflow/Getting started with political science research

In our final computational methods session, we will discuss the little things related to organizing your research process. For example, how should we approach a scientific paper? How can you effectively organize your reading notes? What are the best practices for managing your citations? **Finally, we will wrap up our last day of Math Camp with a short Q&A session, where Martín and Nguyên will answer any question you have about graduate school.**

3 Recommended materials and useful resources

3.1 R

While you will definitely gain more skills as you get more practice, I recommend investing some time into learning R more systematically. This will save you time down the line,

because although Stack Overflow is tremendously helpful, adapting the provided solutions to your own use cases still require understanding how R as a programming language works.

One of my favorite things about R is that it enjoys the support of a friendly and helpful community of users and library developers. We therefore have a wealth of *free* resources for learning R. Below are some that I have found useful in the past years.

- Exercises: R Tutorials by Simon Ejdemyr. This is a set of accessible tutorials that can help you get even more comfortable with R.
- Book: R for Data Science by Hadley Wickham and Garrett Grolemund offers extensive coverage of most topics relevant to political science research. I highly recommend spending a week or so reading through the book and doing all the exercises.
- Book: Advanced R by Hadley Wickham. If coding your own package or Shiny application is your goal, this book is for you. Even if that's not your goal, the first half of this book will enhance the efficiency of your code.

3.2 LaTeX

Fans of LaTeX usually argue that it allows users to compose documents without the hassle of worrying about formatting. Therefore, LaTeX should save you time and improve your concentration. But is LaTeX always more efficient? A study from 2014 by Knauff and Nejasmic (2014, 1) compares LaTeX and MS Word users and finds that, on average, "LaTeX users were slower than Word users, wrote less text in the same amount of time, and produced more typesetting, orthographical, grammatical, and formatting errors". Further, although LaTeX users reported more satisfaction using LaTeX, even proficient users encounter these issues, resulting in lower productivity.

The reality is that, in our discipline, LaTeX aesthetics have become some sort of virtual signaling. I once made a perfectly fine presentation in PowerPoint, but a professor suggested that I should start using Beamer instead because many people think Beamer presentations look more professional.[3] You will also find that most academic CVs are written in LaTeX. It has *that* look that you'll recognize the moment you see it. That said, you absolutely do not need to buy into the LaTeX kool aid. It is just a tool that you should exploit to optimize your workflow.

With LaTeX, we want to **work smart, not hard**. When you find yourself spending too much time tinkering with formatting in LaTeX, it is perhaps time to consider other "cheat" options. Below are a few common sources of LaTeX troubles you will encounter, and some

3. One of the best job talks that I've attended at our department used Prezi!

corresponding hacks to make your life a tad easier.

3.2.1 Tables

Never create a LaTeX table from scratch. When I need a quick table that has nothing to do with statistics, I use the following: https://www.tablesgenerator.com/. Another possible option is to create a table in Excel, import it to R, and export it to LaTeX. You can do this with the R package `xtable`, which allows us to print many different types of data frames, tables, and matrices as TeX code.

To export your regression model results from R to LaTeX, use `stargazer` and `modelsummary`. These are the two most common packages for exporting statistical outputs in R to LaTeX. `modelsummary` is newer and better-maintained.

See this Stack Overflow thread for more suggestions.

3.2.2 Diagrams and graphs

Tikz is the most well-known package for creating diagrams in LaTeX. However, it has a slightly steep learning curve. Luckily, great people on the Internet have created multiple tools for you to actually save time and focus on your own research (instead of spending 2 hours creating a diagram).

- Daniel Kumor's Tikz tool: https://dkumor.com/posts/technical/2018-08-15-causal-tikz/
- Package `dagitty` drawing directed acyclic graph (DAG) in R: https://daggity.net/primer

3.2.3 Converting between LaTeX and other formats

Every once in a while I see folks on Twitter asking for tips on converting LaTeX-based `.pdf` documents into `.doc`. The solution can be quite simple: open your TeX-based PDF document in Word. If you're looking for a more complicated solution, then let me introduce you to Pandoc.

My most common use case for Pandoc is still to convert LaTeX files into a Word document. When would you ever need to do that? Well, perhaps you've taken a liking to LaTeX and want to do everything in it, but your new co-author or advisor prefers to edit in Word. Perhaps you'd like to run your `.pdf` document through your choice of spellchecker. But Pandoc's strength is in its ability to convert to and from many different formats. If, for instance, you truly prefer the simplicity of Markdown, then you will need Pandoc to produce your final outputs. You will eventually be asked to use RMarkdown in your coursework—in fact, RMarkdown uses Pandoc under the hood to produce `.pdf` outputs. To be fair,

Pandoc does not do it all: it can parse TeX but cannot interpret all packages and document classes.

To use Pandoc you will need to tap into *bash*. To convert your `draft.tex` file into `draft.docx`, for example, you can type the following into your terminal:

```
pandoc -s draft.tex -o draft.docx
```

References

Elman, Colin, Diana Kapiszewski, and Arthur Lupia. 2018. Transparent social inquiry: implications for political science. *Annual Review of Political Science* 21 (1): 29–47.

Knauff, Markus, and Jelica Nejasmic. 2014. An efficiency comparison of document preparation systems used in academic research and development. *PloS one* 9 (12): e115069.

Lupia, Arthur, and Colin Elman. 2014. Openness in political science: data access and research transparency: introduction. *PS: Political Science 38; Politics* 47 (1): 19–42. https://doi.org/10.1017/S1049096513001716.