

# **CSSS/POLS 510 MLE Lab**

## **Lab 2. Intro to RMarkdown and Overleaf**

Ramses Llobet

October 6, 2023

# Agenda

1. Pre-view **Problem Set 1**
2. Intro to  $\text{T}_\text{E}^\text{X}$  and  $\text{L}^\text{A}^\text{T}_\text{E}^\text{X}$
3. R Markdown
4. Lab code practice.
5.  $\text{L}^\text{A}^\text{T}_\text{E}^\text{X}$  and Overleaf

# Pre-view Problem Set 1

► Let's quickly look over PS1.

# Intro to T<sub>E</sub>X and L<sup>A</sup>T<sub>E</sub>X

- ▶ T<sub>E</sub>X is a *typesetting engine*<sup>1</sup> designed by Donald Knuth, a computer scientist and mathematician at Stanford
  - ▶ For typesetting scientific text and mathematical formulas

---

<sup>1</sup>Modern extensions of the T<sub>E</sub>X engines include pdfT<sub>E</sub>X, XeT<sub>E</sub>X, LuaT<sub>E</sub>X, etc.

# Intro to T<sub>E</sub>X and L<sup>A</sup>T<sub>E</sub>X

- ▶ L<sup>A</sup>T<sub>E</sub>X is a *document preparation system, or a macro package*, built on top of the T<sub>E</sub>X engine, with features:
  - ▶ Typesetting journal articles, technical reports, books, and slides
  - ▶ Control over large documents containing sectioning, cross-references, tables and figures
  - ▶ Typesetting of complex mathematical formulas
  - ▶ Advanced typesetting of mathematics with AMS-LaTeX
  - ▶ Automatic generation of bibliographies and indexes
  - ▶ Multi-lingual typesetting
  - ▶ See more [here](#)

# Intro to T<sub>E</sub>X and L<sup>A</sup>T<sub>E</sub>X

- ▶ Popular *implementations, or distributions*, of T<sub>E</sub>X/L<sup>A</sup>T<sub>E</sub>X
  - ▶ MacTeX for Mac OS: <http://www.tug.org/mactex/>
  - ▶ MiKTeX for Windows: <https://miktex.org>

# Intro to T<sub>E</sub>X and L<sup>A</sup>T<sub>E</sub>X

- ▶ L<sup>A</sup>T<sub>E</sub>X vs. other word processors (e.g. Microsoft Word)
  - ▶ Microsoft Word/Power Point
    - ▶ WYSIWYG: What You See Is What You Get
    - ▶ You interact with a user interface to control the document layout while typing text
    - ▶ What is displayed on the screen resembles what will be printed
  - ▶ L<sup>A</sup>T<sub>E</sub>X
    - ▶ You provide “L<sup>A</sup>T<sub>E</sub>X commands” to specify the layout, structure, and details of the document:
    - ▶ `\command[optional parameter]{parameter}`
    - ▶ And *typeset* the document using the T<sub>E</sub>X engine and compile the output

# Intro to T<sub>E</sub>X and L<sup>A</sup>T<sub>E</sub>X

- ▶ The input for L<sup>A</sup>T<sub>E</sub>X is a plain text file (.tex)
  - ▶ You need a text editor!
- ▶ Numerous popular text editors
  - ▶ Specific: Texmaker, TeXShop, TeXstudio, TeXworks...
  - ▶ Generic: Emacs (Aquamacs), Vim, Sublime, Atom...



# RMarkdown

- ▶ All the above sound pretty complicated...
- ▶ Instead, we can use RMarkdown as text editor!
- ▶ Some useful resources:
  - ▶ [Cheat Sheet](#) for RMarkdown.
  - ▶ For an general introduction for RMarkdown, look at [Chapter 27](#) from Wickham and Grolemund (2017) - **R for Data Science**.
  - ▶ If you want a more comprehensive guide, then check Xie et al. (2023) - **R Markdown: The Definitive Guide**.
  - ▶ Another, more applied, resource is Xie et al. (2023) - **R Markdown Cookbook**.

# RMarkdown

- ▶ RMarkdown is a document format that allows you to integrate R **code** and **output** into a single document.
- ▶ Besides R code and output, it can also include **text**, **images**, and other **multimedia elements**, allowing for rich and informative documents.
- ▶ *Pandoc* is a free and open-source **document converter** that can convert documents from one markup language to another.
  - ▶ In the context of Rmarkdown, pandoc is the underlying document converter (software) that converts the R-markdown file into a final output format, such as **HTML**, **PDF**, or **Word**.

# RMarkdown

- ▶ The output format of the final document can be customized using options in the **YAML header** or external templates.

```
1 ---  
2 title: "Lab 1 - Intro to RMarkdown"  
3 author: "Your name"  
4 date: \today  
5 output:  
6   pdf_document:  
7     latex_engine: pdflatex  
8   fontsize: 12pt  
9   editor_options:  
10    chunk_output_type: console  
11 ---  
12
```

- ▶ The YAML header in RMarkdown is a block of configuration settings at the beginning of the document enclosed by three hyphens (---).
- ▶ It is used to specify document metadata and other settings such as the document title, author, output format, and more.

# RMarkdown

- ▶ **Code chunks** are sections of R code that can be executed and embedded within an RMarkdown document.

```
78
79 ```{r name, error=TRUE, warning=FALSE}
80 # brau brau, derp herp
81 head(data)
82 ```
83
```

- ▶ To insert an R code chunk, use the shortcut key:
  - ▶ Windows: Ctrl + Alt + I
  - ▶ macOS: Cmd + Option + I
- ▶ Code chunks can be customized with various **chunk options**.

# RMarkdown

## ► Frequently used chunk options

Option	Description
include	If FALSE, knitr will run the chunk but <b>not</b> include the chunk in the final document
echo	If FALSE, knitr will <b>not</b> display the code in the code chunk above it's results in the final document.
error	If FALSE, knitr will <b>not</b> display any error messages generated by the code.
message	If FALSE, knitr will <b>not</b> display any messages generated by the code.
warning	If FALSE, knitr will <b>not</b> display any warning messages generated by the code.

## Recommendation for Homework

Option	HW setting
include	TRUE
echo	TRUE
error	FALSE
message	FALSE
warning	FALSE

- **Note:** set the function `knitr::opts_chunk$set()` with any general setting without repeating it in every code chunk.

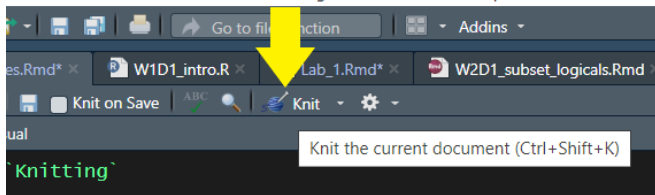
```
20
21 {r setup, include=FALSE}
22 knitr::opts_chunk$set(include = TRUE, echo = TRUE, message = FALSE, warning =
  FALSE, error = FALSE)
23
```

# Knitting

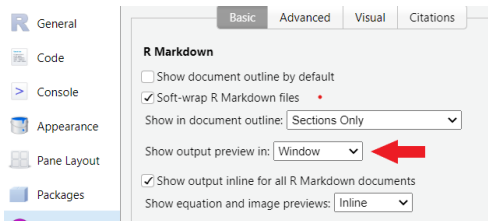
- ▶ In RMarkdown, **rendering** a document means converting the source RMarkdown file into its final output format (using pandoc).
- ▶ To render a document, we need to Knit, knitting is the process of taking the RMarkdown file and converting it into a single, cohesive document that can be rendered into different formats (HTML, PDF, etc).
  - ▶ To produce **PDF file**, you need TeX files.
- ▶ Easy way: Install the tinytex package:  
`install.packages("tinytex")`. Then run  
`tinytex::install_tinytex()`.

# Knitting

- To knit:



- Auxiliary window for output preview:





# Code practice

1. Open R-Markdown file [RMarkdownSample.Rmd](#), we will review the code together.
  2. Open R-Markdown file [Lab02CodePractice.Rmd](#).
- ▶ Please note that the Lab Code key solutions are not included in the compressed ZIP file. I will upload the key along with the lab recording within 24 hours **after** completing the lab.

# Intro to $\text{\LaTeX}$ with Overleaf

- ▶ Alternatively, we have Overleaf: <https://www.overleaf.com/>
  - ▶ An online  $\text{\LaTeX}$  editor
    - ▶ Integrated PDF preview pane
    - ▶ Quality of life features: auto-complete commands, auto-close brackets, keyboard shortcuts, etc.
    - ▶ Numerous templates: journal articles, books, CVs, slides, posters, etc.
    - ▶ Easy collaboration (But not free)
    - ▶ Integrated with Zotero and Mendeley for bibliography management
    - ▶ Integrated with Git for version control

# Intro to $\text{\LaTeX}$ with Overleaf

- ▶ Before we dive in, useful resources
  - ▶ [The Not So Short Introduction to  \$\text{\LaTeX} 2\_{\epsilon}\$](#)  (Oetiker et al., 2023)
    - ▶ Learn  $\text{\LaTeX}$  in 280 pages / minutes
  - ▶ [‘Overleaf’ documentation](#)
    - ▶ Contains intro to basic  $\text{\LaTeX}$ , Overleaf, and many practical guides
  - ▶ [TeXat StackExchange](#)
  - ▶ General: [Mathematics](#) and [Tables](#) and [TikZ](#)
  - ▶ Beamer Theme: [here](#)
  - ▶ Bibliography: [natbib](#), [doi2bib](#), [text2bib](#)
  - ▶ Other: [here](#)

# Intro to L<sup>A</sup>T<sub>E</sub>X with Overleaf

- ▶ Some useful templates:
  - ▶ Thesis: [here](#)
  - ▶ Homework: [my sample](#).
  - ▶ Working paper: [Kenya's sample](#), and [Chris's sample](#)
  - ▶ Academic journal: [here](#)
  - ▶ Presentation slides (Beamer): [here](#) and [here](#)
  - ▶ Poster: [here](#)
  - ▶ CV: [here](#) and [here](#)
  - ▶ Graphs, trees, diagrams (TikZ): [here](#) and [here](#)