**Abstract**

Discussing the computational tool for producing reproducible research paper we so far learned in STAT-159 class. I commented on the use of Makefile, Git, Github, Pandoc and Rmarkdown about their roles and usefulness in reproducible workflow.

The goal of this paper is to describe the experience the author using the computational tool above to produce a reproducible research paper. Through online searches and collaboration with teammates, I learned a lot about each term and would like to discuss some of my findings in this paper

**Makefile**

What is Makefile? Perhaps, many Windows system programmer does not know what it is because the Integrated Development Environment, IDE, of Windows will do the job for you. However, as being a discipline and professional programmer, it is better to master Makefile.

In software development, Make is a utility software that automatically builds program from the source code by reading files called Makefile. It is a tool converting form of files, of which the converted file called target; meanwhile, it also checks the dependencies between files. The role of Makefile is so important to the whole project’s compile rules. There are innumerable source files that respectively placed in different directory by type, function and model. Makefile defines a set of rule to specify the priority of compilation, such as, which file should be compiled first, which file should be recompiled or process more complicated execution. It can execute commands from operating system since Makefile is like a Shell Script.

The advantage of Makefile is an automating the build process; once the Makefile has been done, only have to command Make, and the whole process will be automatically built. It becomes really helpful and productive once you have immeasurable files. Make is a command tool that interprets the instruction and performs the required action from Makefile. In the most of modern software development, Make has been supplanted by the Integrated Development Environment; however, under the Unix environment, there is still so many programmers employ Make to assist software development.

**Git**

Git is a version control system that allows software developer track changes to files over time. It is a tool that manages source code history. It is used for software development that is effective and fast. More than that, Git is also a magical platform collaboration, which you can save or recall precious version of the project at any time. Keeping track of your changes and development process on your project is one of the most important features of Git.

Finally, it is important to not confuse Git with Github. Git is a command-line tool while Github is a web-based graphical interface that help you interact with Git repositories in a nice way.

**Github**

Github is a web-based for storing Git versioned files remotely, which is a repository hosting sever. It allows you visualize differences and manage remotely images, data files text files, and track changes in text. Github provides remote repositories for backup and collaboration. Such as sharing your repositories with others, accessing other user’s repositories and storing remote copies of your repositories as backup of your local copies. More than that, Github allows project member to review your code and interact with it. In that case, Github provides a better way to collaborate with teammate on a large project.

Not only Github is a Git repository hosting sever, but it also is a consequence of the existence of Git.

**Pandoc**

Pandoc is an open-source document converter, which is widely used as a writing tool. It is like an all-in-one toolkit. The function of Pandoc, Knitr, is designed to convert Markdown documents to other formats such as html， Word and PDF. Pandoc is like the extension of Markdown; as a result, the Markdown files will finally convert into the formats of files you need. Especially for research writing, suppose that you write a paper by LaTeX for submission to a plushier. Unfortunately, they reject the format of the paper but DOCX. In this case, Pandoc can make the conversion simply by typing pandoc –output document.docx document.tex on the command line. The advantage of Pandoc is that you can make conversion easily between two formats.

input formats: docbook, haddock, html, json, latex, markdown, markdown\_github,

markdown\_mmd, markdown\_phpextra, markdown\_strict, mediawiki,

native, opml, org, rst, textile

Output formats: asciidoc, beamer, context, docbook, docx, dzslides, epub, epub3,

fb2, html, html5, icml, json, latex, man, markdown,

markdown\_github, markdown\_mmd, markdown\_phpextra,

markdown\_strict, mediawiki, native, odt, opendocument, opml,

org, pdf\*, plain, revealjs, rst, rtf, s5, slideous, slidy,

texinfo, textile

[\*for pdf output, use latex or beamer and -o FILENAME.pdf]

**Rmarkdown**

General speaking, the research workflow mainly consists of three peart: prepare data by EXCEL, analyze data by R and compose the result by WORD. If the workflow was break off by any reason, such as generated wrong images, gathered wrong data or any typo on the summary, can you reproduce and update the result in WORD easily? In that way, a small change would result in a mountain of work. Fortunately, Rmarkdown is able to thoroughly reproduce and keep track of all the changes.

**The Easy Part:**

The easy part of this project is converting formats from [Markdown](https://en.wikipedia.org/wiki/Markdown) file written in [R](https://en.wikipedia.org/wiki/R_(programming_language)) and processed with Knitr. This simple conversion is done by click the Knitr button on Rstudio.

**The Challenging Part:**

The most challenging part of the project is collaboration. To do a reproducible workflow, a tiny change or bug would cause a mountain of work. For example, some project teammate use WORD to produce manuscripts while others use LaTex. It’s very hard to interact or work with other who use different format editing. Sometimes, there is miscommunication between teammate. As a result, all of above would slow down the project process.

**Resources Used**

I utilized most of the online resources, including Google searches, Wikipedia pages and YouTube videos to search definition and tutorial of Makefile, Git, Github, Pandoc and Rmarkdown. I learned a lot of the technical knowledge that I did not know.

**The most time consuming part:**

I spent 7 hours on this paper.

The most time consuming part is writing this discussion and converting the legit definition of Git, Github, etc. into my own word. This part is especially challenging

**Conclusion:**

From writing this paper, I have developed deeper understanding of how to utilize the computational tool such as Makefile, Git, Github, Pandoc and Rmarkdown to do a reproducible research. It allows me to be familiar with those computational tool and take more advantage of it, which make me to do the daily reproducible workflow more efficiently and smoothly.