**Gentzkow and Shapiro**

Read Gentzkow and Shapiro “Code and Data for the Social Sciences” in the “Helpful stuff” Github directory and answer the following

1. Briefly summarize the point of chapters 2- 8 in less than one page.

**Introduction**: Database design and it´s advantages for working properly.

**Automation (2):** Automate everything that can be automated and write a single script that executes all code from the beginning to the end.

* Interactive mode has two big issues:

1. Replicability: there is no record of the precise steps that we did.
2. Efficiency: With many files then we do not know what to run first. Reverse-engineering process is tough for complex projects.

Conclusion: Use “rundirectory.bat”. It executes from start to finish

**Version Control (3):** Store code and data under version control and run the whole directory before checking it back in.

* Demarcate versions of files is painful and fuzzy. To much work to keep track of multiple versions.
* In control version the software remembers every version that was ever checked in, and who did the changes.

**Directories (4):** Separate directories by function, separate files intro inputs and outputs and make directories portable.

* Subdirectories by functions: inputs, outputs, code, temporary or intermediate files.
* If you have the directories in the network, you can run it from every computer.

**Keys (5):** Store cleaned data in tables with unique, non-missing keys and keep data normalized as far into your code pipeline as you can.

* Physical structure of a database should communicate its logical structure.
* Relational database is useful.
* Key: is a variable or set of variables that uniquely identifies the elements of a table.
* Common error: combine variables defined at several different levels of aggregation.

**Abstraction (6):** Abstract to eliminate redundancy and to improve clarity. Otherwise, do not abstract.

* Abstraction: turning the specific instances of something into a general-purpose tool.

**Documentation (7):** Don´t write documentation you will not maintain. Code should be self-documenting.

* Common error: make improvements to the code without improving the comments.
* To be consistent, use the naming of variables and the structure of your coding to guide the reader.

**Management (8):** Manage tasks with a task management system. Remember that emails are not a task management system.

* The more people work in the project, the more difficult becomes the communication.
* It´s clear who´s the responsible for each task. Here enter platforms like Slack.

1. Why do Genztkow and Shapiro think these elements of modern empirical work are so important? What problems does each element solve?

The seven principle elements that Genztkow and Shapiro teach us are crucial es we want to reduce our wear, time and also money because sometimes in a project we have to hire more RA due to the fact that we don´t know how to optimize time cleaning data and doing code.

**Problems that each element solve:**

1. Automation: with the use of logical functions we can automatize process that sometimes we decide by hand
2. Version control: This is one of the relevant principals. Here if we are doing share projects the name of the data sets, the name of the variables is no more a problem because the project and code chunks are easy to replicate in every computer.
3. Directories: it is the solution to mess. If we apply this advice, we would have really organized projects. Generally, we waste time trying to find where do we save some file because usually we save all the files without distinguishing their function, we mix databases with code, plots, pdfs etc.
4. Keys: When we introduce variables that identify the elements of the table, we construct clear mental maps because all the possible desegregations have a path

County 🡪 State 🡪 Region 🡪 Country.

This is the main feature of a relational database.

1. Abstraction: Helps eliminating redundancy and making your project a clear one.
2. Documentation: It´s important to comment your scripts so you your co-authors and RA´s know what you did. However, knowing how much to comment is almost and art, you should not comment too much but not too less. The authors give us examples of what a comment should look like.
3. Management: Using a platform that helps you communicating with other people involved in the project helps a lot the management and effectiveness of a work. This element helps you knowing who´s responsible for each task. Websites like Slack could help us with this.

1. Give an example of the sort of problem that could arise in the course of an empirical project if someone were to fail to adopt these principles.

Santiago, Juan Esteban and Juan David are working in project that wants to do a VAR model between Covid and WTI prices. The three of them separately downloaded the databases from Yahoo finance.

Juan Esteban started the project and send the script by email to Santiago and Juan David; the code chunks do not run properly because the three named differently the databases. After this was solved, Santiago is confused because he does not understands the name that Juan Esteban gave to the variables. Juan Esteban put “Llwtip” but obviously it was almost impossible that Santiago knew it that JE was referring to log of wti price lagged one time. Due to the fact that they had to turn in this project in one week, Juan David also started working at the same time of Santiago and when they showed each other they advances the realized that some things were the same because it wasn’t clear who was responsible of each task.

**Conclusion of the history:** If the group would have applied all the advices, especially the ones of management, documentation and version control it would have been easier to do the work.

1. How do you plan to incorporate these solutions into your own work?

* The last element of the authors would be the first one that I would implement. I think that having WhatsApp groups where you talk about the projects, do jokes and talk about your life at the same time is really bad for having clearness of the responsibilities of each member of the project and the advances that any of them have done. Having a platform (like slack) or a channel for talking of each subdivision of the project would be great.
* I would use platforms such as Github for the control version advice, with that the groups have a virtual desktop that is homogenous for everyone.
* I would be more organize applying advices like automation, keys, and documentation. With well-defined directories.

**Git**

These next questions concern the software “git” and “github”.

1. Briefly explain what git and github are used for, how they are similar and how they are different:

**Git:** it is used for version control which allow us to work together with other people without many troubles, usually it is use by people who write code. It does not require the use of GitHub. It is the fundamental tool

**GitHub:** website that permits that you uploaded the Git repositories online. This is useful as a backup. Another advantage is that other people can navigate in your repos and you in theirs, so it makes easier collaboration. It is also cool because gives you a visual interface of Git. It works like a cloud where you pull your works and you pull others work.

Similarity: Both have the main objective of version control. Provide almost the same features but with different interfaces. They are not competitors, but they compliment each other, that why they are a good combination.

Difference: Git is open source tool that you install locally, while Github is an online service which the people that use Git can connect and share resources. In few words, GitHub is a cloud-based platform built around Git. However, GitHub have some features that Git do not has like online editing of files, lightweight task tracking, email notifications and alerting etc.

I have also included a deck of slides by Frank Pinter in “Helpful stuff”. The completion of this section will satisfy the Github requirement of the course, not counting any additional assignments that use Github. You must have it done by **Wednesday** **June 10th** to receive the 10% credit.

1. Name a benefit of using git to organize your empirical research. What types of common problems can occur if you do not use git?

A good benefit for an empirical project id the Distributed Development, which allows each person to work on a copy that points back to a central repository. Due to the common problems that I mentioned in the story that I wrote in the third question of the first part like the fact that in projects where mora than one persona works it results a problem that is difficult to work at the same time while we know who changed the work. This allows us to have a track of all the movements that any person does to the work and we have a backup of it.

1. What about using git is challenging for you for right now? What steps can you take to minimize those challenges such that you can adopt git for this class?

Being totally sincere, I had seen GitHub two times in my life and what I did was to download the information but not to create an account and clone the files. That´s why this is the first time that I introduce myself to Git. Yesterday I create the account and start digging a little bit through the functionality. Today with the guide of Genztkow and Shapiro I understood the importance of Git and Github. It has been difficult to work on it because it is a new world. However, a friend of mine have done some work in GitHub so when I have some doubts, I see tutorials in YouTube but if that does not function I call my friend for some help.

Steps:

1. Mess around and dig into the functionality. Try to do it alone
2. Look for tutorials in Youtube or for some articles that are on Internet. Even the articles that Scott share in the file of helpful stuff.
3. Call a friend or write in slack for some help.
4. Name the four main Git operations. What does each operation do and how are is each operation different from one another?

* Stage: tells Git to add changes to the repo history
* Commit: confirm Git that you are sure of doing the change, so that it should be part of the repo history
* Pull: Get the changes made on the GitHub repo. Pull refers to bring from the cloud the changes that other have done
* Push: “Upload” to the GitHub repo the local changes. Push refers to drive the local changes that you have done to the repo.