A Brief Introduction to GitHub for Social Scientists using Stata and Dropbox[*]

Heitor S. Pellegrina

May 5, 2018

1 Introduction

Github is widely used online platform that computer scientists use to keep track of their codes and to collaborate in projects. It is useful to understand the difference between Git from GitHub. Git is a version control system that stores all the history of a code. With Git, programmers can check what they changed in a code, when, and they can go back to previous versions of their project if they do not like the results of some changes that they made. Github is a web-based repository that keeps all the history of a code on a cloud using the Git system. Since everything is in a cloud, it allows many people to colaborate in the formulation of a code.

1.1 Why Should I use GitHub+Dropbox instead of just using Dropbox?

Dropbox is a great resource for teamwork in smaller projects. Many people work on a project and everything is automatically syncronized across computers. However, as projects multiply, become larger and gain many collaborators, there are several problems. First, if you have worked with Dropbox with many collaborators you probably noticed the typical generation of "conflict" copies. This happens when Dropbox is not sure which version to use because there was more than one sincronization at the same time. As I explain below, GitHub tends to avoid this problem because files are not automatically sincronized. Second, Dropbox does not have a good system to track codes. If something changes, we do not know who changed a code and what exactly changed. GitHub has an interface that makes it easy to identify where exactly codes changed.

[*]This document should not be considered as representative of high standards of project management! I hope that this material can be useful for Social Scientists like me who struggled with the material that is available for social scientists on the internet. In particular, I could not find a specific document that addressed the integration of Dropbox with GitHub when a project uses larger datasets. Comments are very welcome!

Third, if you want to bring back previous versions of the project, GitHub can do it quickly, but with Dropbox, we can only bring back specific files one at a time. While some of these problems may be more likely to appear in large projects (imagine writing the codes for facebook for example...), as big empirical projects become more frequent in Economics, I believe that the use of Github will inevitably become more frequent as well. The main cost of using GitHub is that you have to be much more systematic about the way how you work since you have to manually sincronize your changes with the cloud.

1.2 What is the Difference between Dropbox and GitHub?

In practice, the main difference between Dropbox and GitHub is that now we have an intermediary step for the syncronization of files called *"commit"*. In Dropbox, we have

$$\text{Master Repo (on the web)} \iff \text{Local Repo (in your computer),}$$

where Master Repository is the cloud where the codes and their history is maintained and local repository is a copy of the Master Repository that you keep in your computer. In Git, we have

$$\text{Master Repo (on the web)} \iff \text{Commit} \iff \text{Local Repo (in your computer),}$$

The way how sincronization works with Github is as follows. You first download all the repository containing the codes for the project to your computer using a process called *"clone"* if it is the first time that you are downloading the project and a process called *"Pull"* if you are just updating the project to your local repo. Both are essentially downloading the files from the internet. After you make your changes to the files, you can upload them to this intermediary step called commit. You can upload these changes to the cloud using a process called "*Push*".

1.3 Can I use GitHub as a Repository for my Datasets?

GitHub is a tool for keeping track of codes, but it is not supposed to be a repository for datasets. This is where an integration of Dropbox with Github is useful. GitHub can keep track of all the codes that is written in a project, but not the datasets and output of the project (figures, tables and pdfs). For example, a researcher can keep track of do-files that clean the data, keeping all the versions of the code, but only keep the final version of the cleaned dataset and not its history. The idea is that, by keeping track of the code, we can always go back to previous versions of the project and re-clean previous datasets if the project is well organized.

1.4 Preliminaries for this Manual

Before you move forward, make sure that you do the following

- Go to Github and set up an account.
- Follow their introduction guide.
- Go to Git website and install git.

2 Commands for Terminal

There are two ways to sincronize your files with the master repository. First, you can with the Master repository using a software called GitHub Desktop. This is a software where you can "point-and-click" your way through sincronization. I found this quite user friendly. However, there were a couple of functionalities that I was not able to obtain with it. Second, you can use the Terminal in Mac[1] or the command line in windows. They may seem scary at first. But, in practice, you do not have to be anythin near a sophisticated programmer to use it. There are just a few commands that you have to memorize for your daily work (or just use the list below).

- MAC
  - *ls* (this command will **list** the files in the current directory)
  - *cd Dropbox* (this command will change the **current directory** and enter the folder called "Dropbox")
  - *cd ..* (this will go a folder back in the hierarchy of folders)
  - *mkdir projects-git* (this command will **make a directory** called projects-git)
  - *git clone https://github.com/hpellegrina/hello_world* (this command will make a clone of this repository into your machine)
  - *git add table_1_summary_stat.do* (this command will add to the branch the do.file called *table_1_summary_stat.do*)
  - *git add -A -m "Adding all files"* (this command just **adds** to the branch all the new files you have and creates a **message**)
  - *git commit -m "Adding table_1_summary_stat.do"* (this command will commit the changes that you made to the folder, the phrase inside the quotation marks is the comment that you are going to attach to your commit)

[1]If you are using a Mac computer and do not know what Terminal is, read this website here.

  - *git push* (this command will push the changes that you made to the git cloud, you will see that github asks for you authentication here)
  - *git pull* (this command will pull any changes that you made to the git cloud)

How a day of work on a task may look like.

- You start by going to the directory where the project is in your terminal. Type *git pull* to update the directory with everything that everybody else may have changed.
- You make all the changes in the directory related to your task.
- You type *git add -A* to add all the changes that you made to your branch.
- You commit the changes using *git commit - m "Issue 7: added table_1_summary_stat.do"*
- You type *git pull*.

3 Hello-World! I'm an Economist!

Following the tradition in the computer science. Below you will have your first commit using GitHub, Stata and Dropbox. I taylored this commit to include some of the tricks that I have been using to integrate Dropbox with GitHub.

1. Before you start, make sure that you have a GitHub account and that you have installed Git into your machine. To double check if it's properly installed, go to terminal in MAC and type "git". If it is installed, it will show a list of commands that you can use with git.
2. Create a repository called "projects-my-hello-world-econ-git".
3. Create a folder in Dropbox or GoogleDrive and call it "projects-hello-world-econ-data-repo". Put the data called "macro_indicators.dta" that is inside the folder "datastorage-for-practice" into the folder that you just created.[2]
4. Open the terminal and go to the Dropbox folder that you have using the commands *cd* and *ls*. In the Dropbox folder, type *git clone https://github.com/hpellegrina/projects-hello-world-econ* . This will clone my folder to your computer.
5. Type *git clone projects-my-hello-world-econ-git* (this is where you are going to commit the project based on my clone).

[2]In a real project, we will avoid keeping any data file in "git" folders. I'm keeping this .dta file here only for demonstration purposes.

6. Type *cd projects-my-hello-world-econ.*

7. Create a "fake" link to the Dropbox folder. In my case, I typed *ln -s /Users/heitorpellegrina/ Dropbox/ projects-my-hello-world-econ-data-repo /Users/heitorpellegrina/ Dropbox/projects-my-hello-world-econ/datastorage*. For PC users, you have to type *mklink /J datastore "C:\Users\ USERNAME\ Dropbox\projects-my-hello-world-econ-data-repo"*

8. Outside terminal, copy all the documents from my project into your own folder.

9. Now, run the stata do-file from your folder and create the new figure. You may have to change the location of the directory inside the .do file.

10. Now, it's time to commit your changes. Go back to terminal and type *git add -A*.

11. Type *git status*, check if everything that you changed is actually there.

12. Type *git commit -m "Added first figure for Hello-World! I'm an Economist!"*

13. It's done! You can check your repository on the web in your account. It should contain all the additional files.