# lab01

March 27, 2021

## 1 Lab – Week One

In this lab we will:

(1) Load the lab GitHub repository locally;

(2) Set up our work environment by installing Julia and Jupyter;

(3) Cover Jupyter basics;

(4) Go over some Julia basics;

(5) Practice simple optimization application;

(6) Practice simple linear algebra applications.

The goal is to set up a stable work environment for the rest of the quarter while getting a taste of the Julia language.

Today's material is heavily based on select parts of Quantitative Economics with Julia by QuantEcon.

---

### 1.1 GitHub

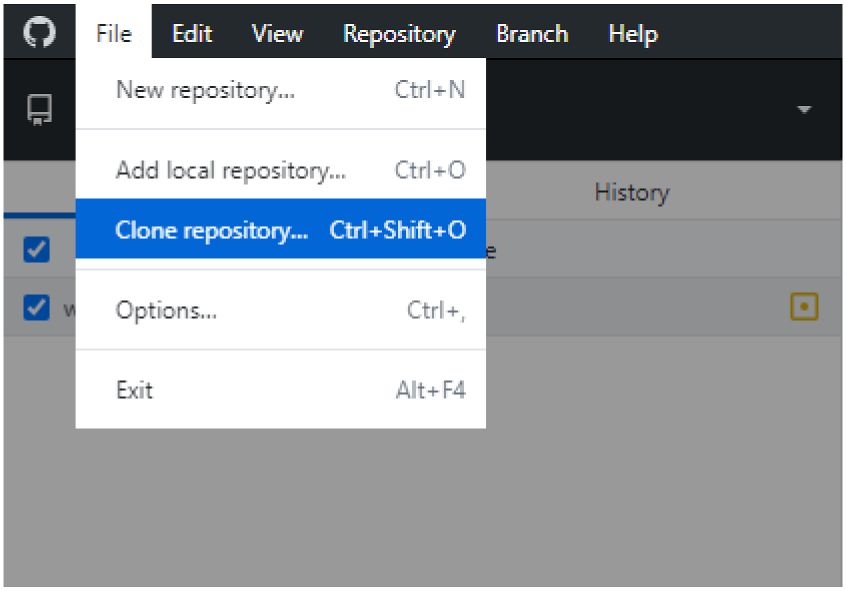
As you might have already noticed, all lab materials for this quarter live on GitHub. I've done so to allow for easy version control and replicability.

Here I will guide you through the bare minimum of the necessary GitHub functionality required to maintain up-to-date access to lab materials.

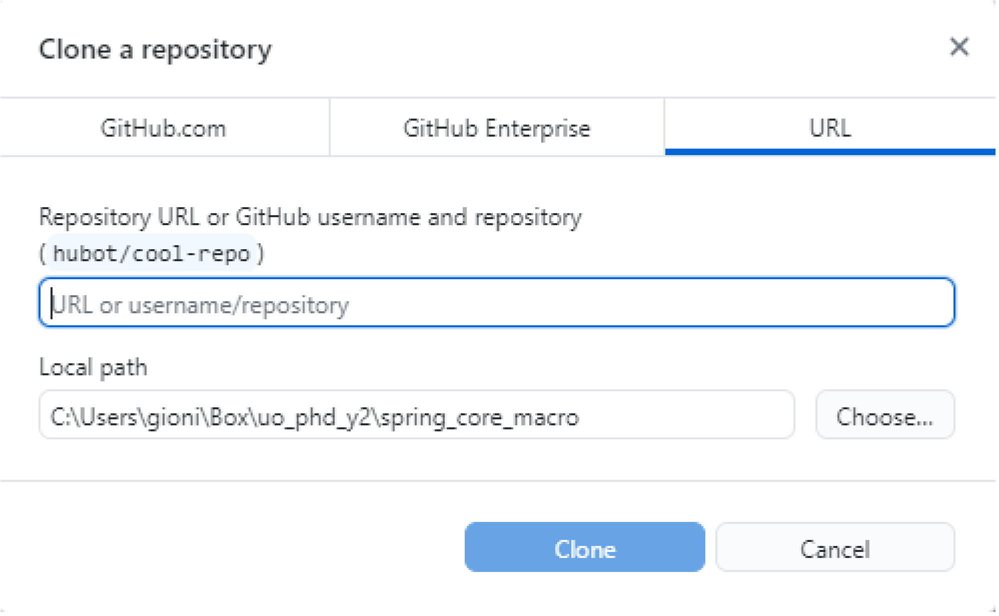
Go to the GitHub website to create an account and register for a student/educator discount (this last part isn't necessary, but can't hurt).

After finishing the above step, download and install GitHub Desktop. Open it and log into your GitHub account locally.

In the GitHub Desktop interface, go to `File > Clone Repository`:



The following box should pop up, in which we need to navigate to the `URL` tab:



Clone a repository

Repository URL or GitHub username and repository
( `hubot/cool-repo` )

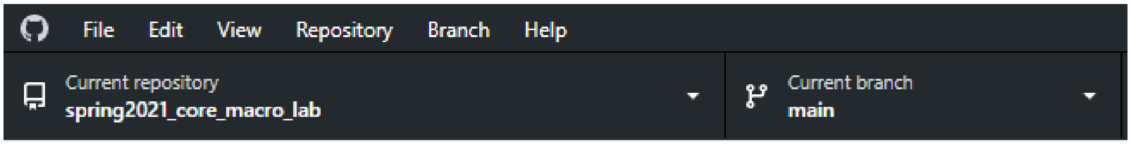
URL or username/repository

Local path

C:\Users\gioni\Box\uo_phd_y2\spring_core_macro

Paste `https://github.com/gionikola/spring2021_core_macro_lab.git` into the URL, and then choose a folder in which you would like to keep the lab materials.

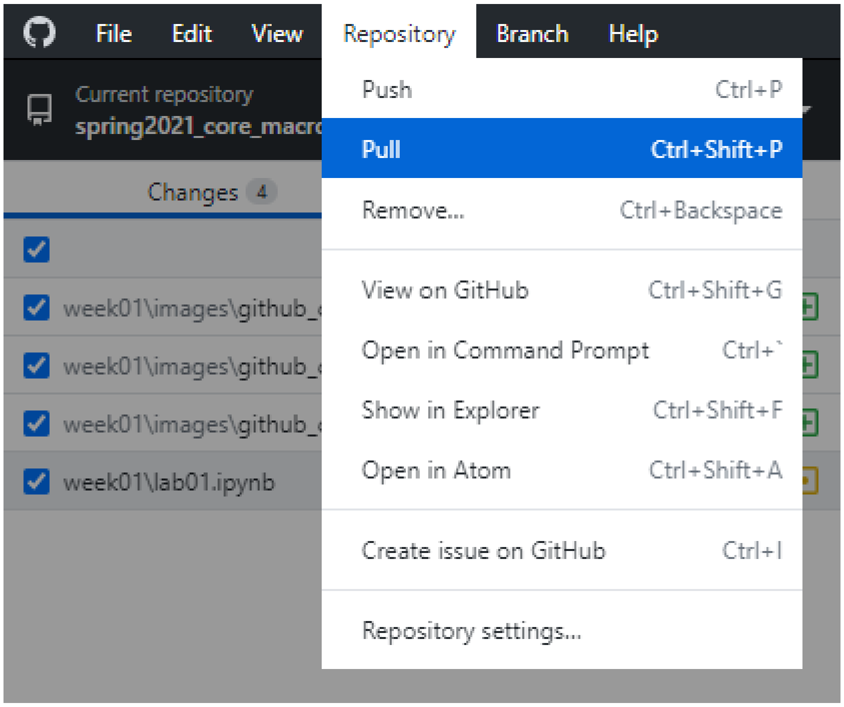
Once this is done, make sure that the `Current repository` is set to `spring2021_core_macro_lab`:



Current repository
spring2021_core_macro_lab

Current branch
main

Now we can automatically keep all lab documents up-to-date by periodically opening up GitHub Desktop and going to `Repository > Pull`:



## 1.2 Basic Setup

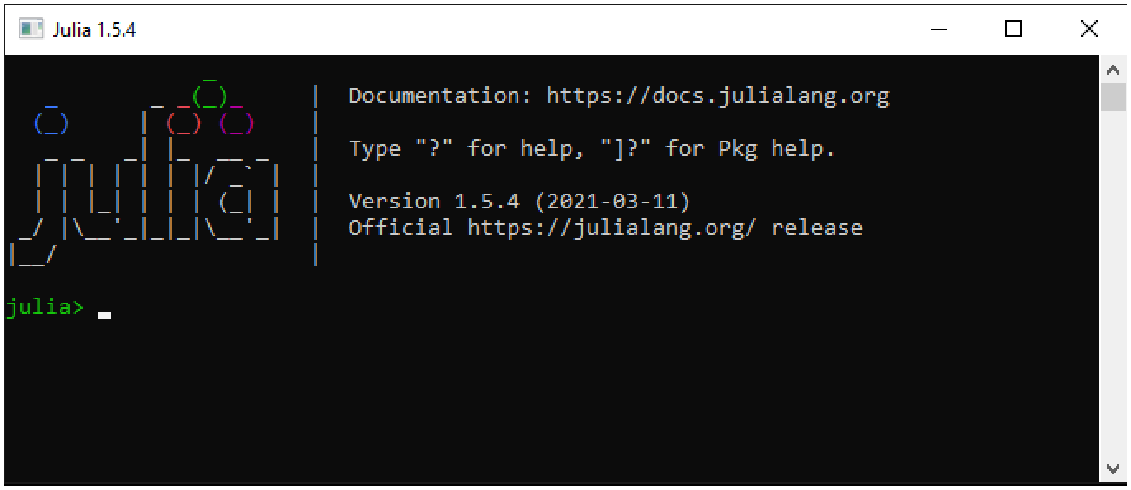
To set up the necessary work environment for these labs, we need to do the following three things:

- Download and install Julia;
- Download and install Jupyter;
- “Connect” Jupyter with Julia.

We will first need to download and install the current stable version of Julia by visiting this page and following the instructions for the appropriate operating system.

Then we will need to visit this page to download and install Anaconda, which essentially a package management tool that includes Jupyter. To learn more about Anaconda, read the Wikipedia page.

Once both Julia and Anaconda are installed, we can perform the final step of linking Jupyter with Julia. Open Julia – we should see the following window pop up:

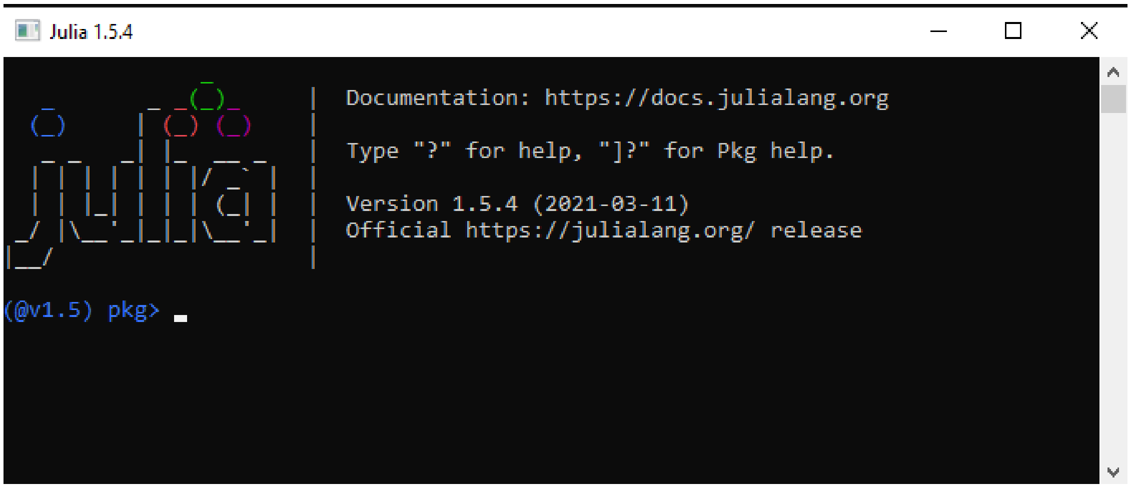


```
               _
   _       _ _(_)_     |  Documentation: https://docs.julialang.org
  (_)     | (_) (_)    |
   _ _   _| |_  __ _   |  Type "?" for help, "]?" for Pkg help.
  | | | | | | |/ _` |  |
  | | |_| | | | (_| |  |  Version 1.5.4 (2021-03-11)
 _/ |\__'_|_|_|\__'_|  |  Official https://julialang.org/ release
|__/                   |

julia> 
```

This is the Julia REPL (Read-Evaluate-Print Loop).

Now type `]` to enter package mode, then type `add IJulia`.

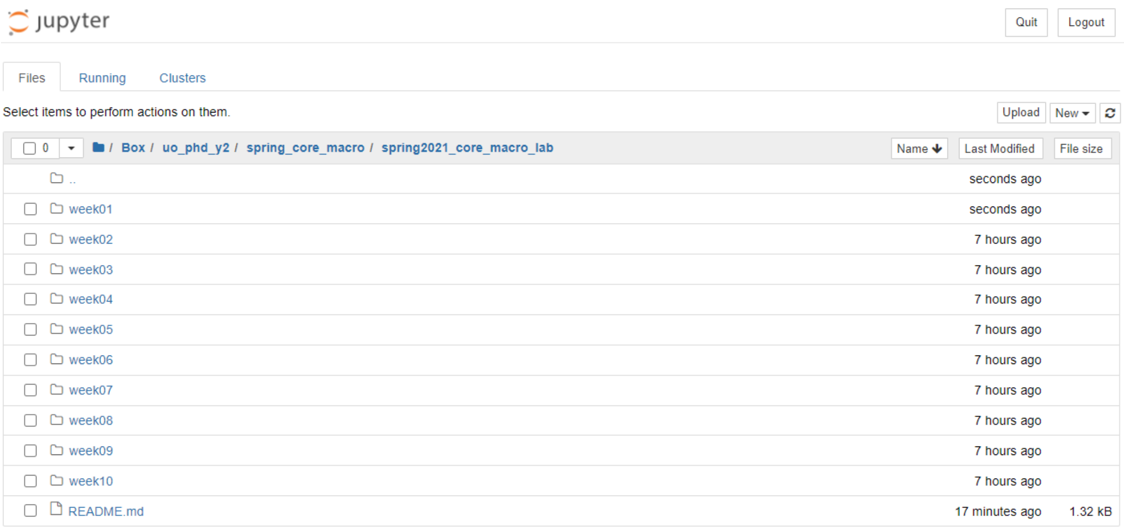


```
               _
   _       _ _(_)_     |  Documentation: https://docs.julialang.org
  (_)     | (_) (_)    |
   _ _   _| |_  __ _   |  Type "?" for help, "]?" for Pkg help.
  | | | | | | |/ _` |  |
  | | |_| | | | (_| |  |  Version 1.5.4 (2021-03-11)
 _/ |\__'_|_|_|\__'_|  |  Official https://julialang.org/ release
|__/                   |

(@v1.5) pkg> 
```

We then click `Enter` on the keyboard to run the command. This will install the IJulia kernel, which should automatically link Julia with Jupyter. After the installation is complete, we click `Backspace` on the keyboard to leave package mode.

We can now type `using IJulia; notebook()` into the REPL to launch Jupyter Notebook. If this running this command leads to an error or the prompt `install Jupyter via Conda, y/n? [y]:`, then we can alternatively launch Jupyter Notebook by finding where it lives on our machine and/or typing `jupyter notebook` into the Anaconda Prompt. In any case, we should see something similar to the following tab open in our web browser (but in a different directory):



Select items to perform actions on them.

/ Box / uo_phd_y2 / spring_core_macro / spring2021_core_macro_lab

| Name | Last Modified | File size |
|---|---|---|
| .. | seconds ago | |
| week01 | seconds ago | |
| week02 | 7 hours ago | |
| week03 | 7 hours ago | |
| week04 | 7 hours ago | |
| week05 | 7 hours ago | |
| week06 | 7 hours ago | |
| week07 | 7 hours ago | |
| week08 | 7 hours ago | |
| week09 | 7 hours ago | |
| week10 | 7 hours ago | |
| README.md | 17 minutes ago | 1.32 kB |

We can now manually navigate to our preferred directory (folder) and create new notebooks, along with other types of documents. When we open a document, it will open as a new tab in our web browser.

At this point we are all set-up and ready to code!

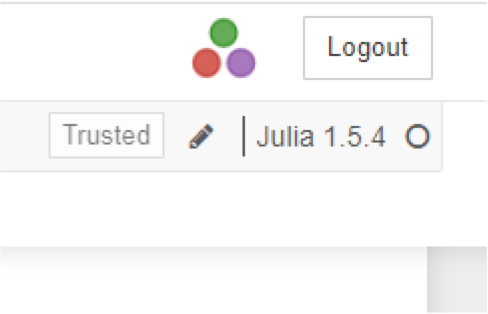
---

## 1.3 Jupyter Basics

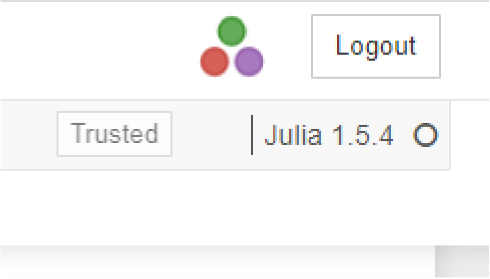
The keyboard shortcuts window says it best: "The Jupyter Notebook has two different keyboard input modes. Edit mode allows you to type code or text into a cell and is indicated by a green cell border. Command mode binds the keyboard to notebook level commands and is indicated by a grey cell border with a blue left margin."

At any given point in time in the Jupyter interface we are either in edit mode or command mode. We can tell which mode we are in using two indicators: (1) the color of the outline around a selected cell, and (2) the presence/absence of a pencil symbol next to the active kernel name at the upper-right corner of the notebook.

If we are in edit mode, then we should see the following:



Otherwise we should see the following:

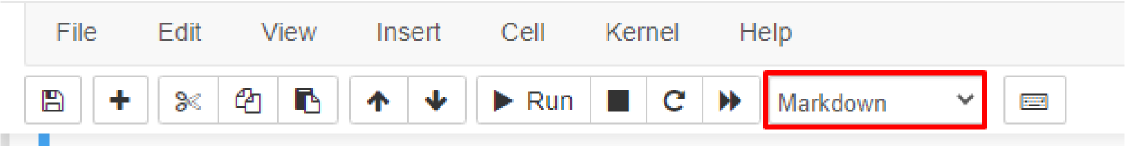


The following is a list of some very useful shortcuts:

- `Esc` triggers command mode if currently in edit mode;
- `Enter` triggers edit mode if currently in command mode;
- `Ctrl-Enter` runs selected cell;
- `Shift-Enter` runs selected cell and selects the next cell below;
- `Alt-Enter` runs selected cell and creates a new cell below;
- `D, D` deletes selected cell.

Check out more shortcuts by going into command mode and clicking the `H` key.

There are three types of cells (1) code, (2) Markdown, and (3) raw. We will mostly be interested in code and Markdown cells, so we disregard the final category. We can tell what type of cell we are dealing with by either (1) looking up at our toolbar



or by (2) looking to the left of the cell to check whether there is a `In [ ]:` present – if so, then we have ourselves a code cell, otherwise it is a Markdown cell.

A couple more useful shortcuts:

- `M` changes a selected cell type to Markdown if in command mode;
- `Y` changes a selected cell type to code if in command mode.

Jupyter is simple and intuitive enough that the above should be enough to get us going.

---

### 1.4 Julia Basics

---

### 1.5 Optimization

---

## 1.6 Linear Algebra