Instructions

AUG
2018

This page will guide you on how to use Git and GitHub to clone the project repository to your computer. The repository you will clone has a skeleton of how the project should be structured. This structure is also discussed below.

Install Git

First, you need to download and install Git. Follow the instructions below depending on your operating system.

- Mac/Linux distributions:
  - Download the Git software and install it.
  - You can access it via your terminal application.
- Windows:
  - The easiest way to get it installed is via Git for Windows.
  - Download it and install it. The default configurations should be fine (you may want to change the editor to nano).
  - You should now have a git bash application that is very similar to a Mac terminal.

If you have never used Git before do not worry. The Lecture 1 - Git has everything you need to know for our usage of Git and Github.

Create a Folder for Projects and Data

Now that you have download and installed Git, open your terminal application (or git bash) and change directory (cd) to your home folder.

```
cd ~/
```

The `~` at the beginning is interpreted by the terminal as the path to your home folder. In my case, this is the same as typing `/Users/guilhermesalome`. (If you type `echo $HOME` you should get back the path to your home folder.)

Now, make a new directory (mkdir) to hold the solution to the projects:

```
mkdir Projects
```

Move inside this folder and create a new folder that will hold all the data you will use in the projects.

```
cd ~/Projects/
mkdir Data
```

You should save the data you were assigned to inside the `Data/` folder.

Getting the Repository for Project 1

Follow the next steps to get the repository for Project 1. Read all of the steps before beginning.

1. Go to GitHub and create a new account if you do not already have one.
2. Follow this link to get the repository for Project 1.
    1. When you click on the link, you will be asked to login with your GitHub account.
    2. You will then see a page with a list of names (Last Name, First Name). Click only on yours. This will link your account to your name.
    3. Then you will be asked to accept the assignment. When you do, GitHub will start cloning the project repository to your account.
    4. After the cloning is finished, you will get a link to your assignment. Follow that link.
    5. Lastly, click on the green button `Clone or download` and copy the URL to the repository.
3. Use the URL to clone the repository to your computer:

```
cd ~/Projects/
git clone URL
```

This tells git to go to the Project1 page, and clone (copy) all its contents to your folder. It will create a new folder with the project name, in this case Project1, and download all the files to that folder.

Now, if you list files (ls) in your directory:

```
ls
```

You should see a new folder for Project 1. Change the directory to that folder:

```
cd Project1Folder/
```

The Project1 folder contains the project `.pdf` file with the questions to be answered by you, and it also contains a skeleton of how you should organize the solution of your project. Let's discuss the folder structure of Project1.

Folder Structure

The folder you just cloned has the following structure:

```
Project1/
|
|-- main.m
|-- report.tex
|-- preamble.tex
|-- README.md
|-- functions/
|   |-- simDiffusionConstantCoeff.m
|   |-- simDiffusionStochasticVariance.m
|   |-- simJumpDiffusionConstantCoeff.m
|   |-- simJumpDiffusionStochasticVariance.m
|   |-- simJumpProcess.m
|   |-- simStochasticVariance.m
|-- scripts/
|   |-- plotDefaults.m
|   |-- plot1A.m
|   |-- plot1B.m
|   |-- plot1C.m
|   |-- plot1D.m
|   |-- plot2A.m
|   |-- plot2B.m
|   |-- plot2C.m
|-- figures/
    |-- 1A.png
    |-- 1B.png
    |-- 1C.png
    |-- 1D.png
    |-- 2A.png
    |-- 2B.png
    |-- 2C.png
```

Let's go over each item:

- `main.m` : this file is your main Matlab script. Running it should solve all of the problems for this project and create all necessary plots.
- `report.tex` : this file is where you will write your report in the LaTeX format. When you finish your project, compile this file to generate a pdf with the report.
- `preamble.tex` : this file contains the LaTeX settings that are included in your `report.tex` file.
- `README.md` : this file contains is displayed at the GitHub page for the project.
- `functions/` : this folder should contain your Matlab `.m` files that define new functions. These functions are used to break the exercises into smaller pieces, simplifying your `main.m` file and making it more readable/easier to debug. I created many files inside this folder. Each file holds the structure of one of the functions that solves one of the items in the project. You should complete these files to obtain the soutions.
- `scripts/` : this folder contains scripts that, like functions, should help with keeping `main.m` readable and easy to debug. During most projects this folder will usually contain scripts that generate plots/figures. The code to generate plots in Matlab can easily become big and repetitive. The idea is to move that code into scripts, say `plot1A.m` , so that we can simply call this script in the right place of `main.m` and have the plot be created. This also helps separating the computational part of the project from the plotting part. Inside this folder there is also a file called `plotDefaults.m` , it modifies some of the behavior of default plots in Matlab.
- `figures/` : this folder should contain the figures you were asked to create in this project. These figures will be used to construct your report, and this folder provides a clean way to keep them separate from everything else. I created an empty file named `1A.png` as an example. In this project you will fill that file with an actual plot.

You should now be able to understand how to organize your project.

On Making Mistakes and Debugging

The lectures are responsible for discussing the theory. The projects are meant for you to implement that theory and see it in practice. Implementing the theory means creating software to compute the estimates we are interested in, and then analyzing the results.

The process of writing the software can be divided in two stages:

- Knowing what we want the computer to do
- Telling the computer what to do

The first stage is knowing what to compute, which is given by the theory we have been studying. The second stage is about transforming that theory into computer instructions. It is in this step that mistakes occur, and we always make mistakes when programming.

> If debugging is the process of removing software bugs, then programming must be the process of putting them in. – E. W. Dijkstra

The easiest mistakes to spot are syntax mistakes. They are easy to spot because you will get an error when you run your code, and so are also easy to fix.

The hardest mistakes to identify are logical mistakes. These happen when you think your code is doing something, but you instructed it to do something else. This will results in computing wrong values, and, because there is no easy way to identify these mistakes, you will spend most of your time fixing this kind of error.

Your job is to find your own logical mistakes and fix them. Do not worry if you do not succeed at first, it is normal to make mistakes when coding. Look at your code and understand what is actually doing, and then fix it and try again.
