SOP 06: Introduction to Rstudio Server and Github

Isaac Medina & Marc Los Huertos

September 9, 2017

1 Introduction

As students of the environment in this course, our ability to have substantive scienctific discourse rests upon our ability to draw meaningul conclusions from observations or data. In dealing with the problems of data, as in other sciences, we will utilize the language of statistics and mathematics to deal with the complexity of environmental issues. This enables us to rely on powerful statistical and computational tools that have been developed to deal with data - in particular open source software like R, Rstudio and Git.

Once you get the hang of using these programs, you will be equipped to do many kinds of interesting and powerful analyses. However, becoming facile in using these programs can feel a lot like learning how to walk. We need to approach this process in discrete steps. The following pages will explain more about what Rstudio and Github are, as well as guide you through an excercise that connects the functionality of both programs!

1.1 Purpose

This document is intended as a resource and guide to help you understand how to:

- Create projects in Rstudio and connect them with your peers so you can collaborate online using Github repositories.
- Troubleshoot when you run into problems "pushing," "pulling" and "merging" your work with your collaborators.

2 Background

2.1 What is R?

R is a powerful, open source program but combined with RStudio and Github the program becomes an archetype of a program that enables 1) collaboration, 2) transparency, and 3) accessibility.

RStudio is the user interface for R. Although R by itself is an amazing example of crowd sourcing, where a wide range of staticians and programmers have created a free programming environments with a robust range of statistical packages, the RStudio interface provides a user with the tools to track and publish their analysis process in an effecient and transparent way.

Local Install versus Server — R and RStudio can be installed on a local computer/laptop from the CRAN download mirror sites. However, we also have access to the R and RStudio Server installed on the Pomona College mainframe, where you can access it via a web browser. Wow, this is conveient!

2.2 What is Git and Github?

GitHub is a web-based Git repository hosting service. Git is a software program that allows one to track versions of code and text.

Version Control is a method to track changes in software, and often in the context of collaborative projects. The final component of R and RStudio is its capacity to create projects (RStudio's terminology) and repositories (Github's terminology) that can be shared among collaborators. In particular, the collaboration allows for contributions to be tracked via version control tools. There are a number of ways that we can access these tools, but we'll try to limit the methods to keep the process relatively "simple".

3 Connecting Rstudio and Github: The "Beginner's Luck" Excercise

Now that you know what Rstudio and Github are this excercise will guide you through the processes of connecting the two together. This process will require several steps in which you will have to go back and forth between Rstudio and Github.

Step 1: Sign into your Rstudio Server Account

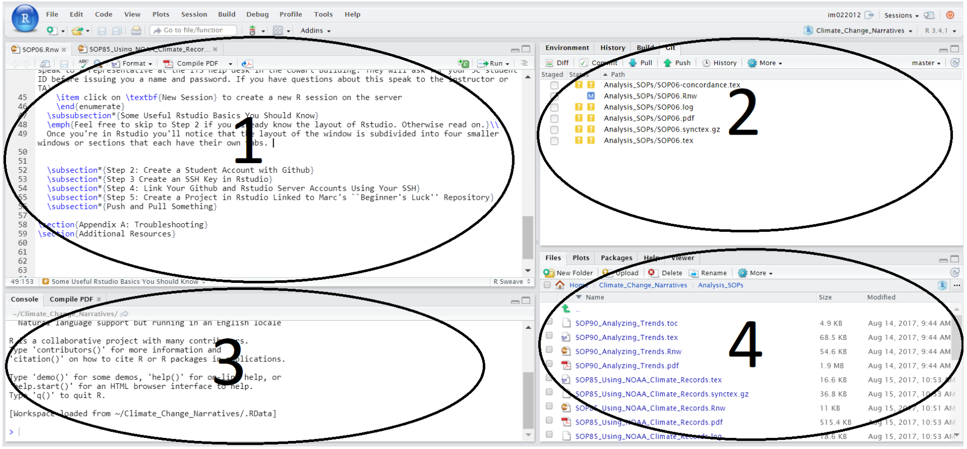
Since we will be using the server version of Rstudio all you need to do is login to your Rstudio Account:

1. Using your computer's web browser go to `https://rstudio2.campus.pomona.edu` to access your Rstudio account
2. Login using your Pomona College user ID and password [1]
3. click on **New Session** to create a new R session on the server

[1] Non-Pomona students must first speak to a representative at the ITS help desk in the Cowart Building. They will ask for your 5C student ID before issuing you a name and password. If you have questions about this speak to the instructor or TA

Some Useful Rstudio Basics You Should Know

Feel free to skip to Step 2 if you already know the layout of Rstudio. Otherwise read on.
Once you're in Rstudio you'll notice that the layout of the window is subdivided into four smaller windows or sections that each have their own tabs.



```
speak to a representative at the ITS help desk in the Cowart Building. They will ask for your JC student
ID before issuing you a name and password. If you have questions about this speak to the instructor or
TA}
    \item click on \textbf{New Session} to create a new R session on the server
    \end{enumerate}
  \subsubsection*{Some Useful Rstudio Basics You Should Know}
  \emph{Feel free to skip to Step 2 if you already know the layout of Rstudio. Otherwise read on.}\\
  Once you're in Rstudio you'll notice that the layout of the window is subdivided into four smaller
windows or sections that each have their own tabs.


  \subsection*{Step 2: Create a Student Account with Github}
  \subsection*{Step 3 Create an SSH Key in Rstudio}
  \subsection*{Step 4: Link Your Github and Rstudio Server Accounts Using Your SSH}
  \subsection*{Step 5: Create a Project in Rstudio Linked to Marc's ``Beginner's Luck'' Repository}
  \subsection*{Push and Pull Something}

\section{Appendix A: Troubleshooting}
\section{Additional Resources}
```

Staged	Status	Path
□	? ?	Analysis_SOPs/SOP06-concordance.tex
□	M	Analysis_SOPs/SOP06.Rnw
□	? ?	Analysis_SOPs/SOP06.log
□	? ?	Analysis_SOPs/SOP06.pdf
□	? ?	Analysis_SOPs/SOP06.synctex.gz
□	? ?	Analysis_SOPs/SOP06.tex

~/Climate_Change_Narratives/

```
  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[Workspace loaded from ~/Climate_Change_Narratives/.RData]

>
```

Home > Climate_Change_Narratives > Analysis_SOPs

Name	Size	Modified
..		
SOP90_Analyzing_Trends.toc	4.9 KB	Aug 14, 2017, 9:44 AM
SOP90_Analyzing_Trends.tex	68.5 KB	Aug 14, 2017, 9:44 AM
SOP90_Analyzing_Trends.Rnw	54.6 KB	Aug 14, 2017, 9:44 AM
SOP90_Analyzing_Trends.pdf	1.9 MB	Aug 14, 2017, 9:44 AM
SOP85_Using_NOAA_Climate_Records.tex	16.6 KB	Aug 15, 2017, 10:53 AM
SOP85_Using_NOAA_Climate_Records.synctex.gz	36.8 KB	Aug 15, 2017, 10:53 AM
SOP85_Using_NOAA_Climate_Records.Rnw	11 KB	Aug 15, 2017, 10:51 AM
SOP85_Using_NOAA_Climate_Records.pdf	515.4 KB	Aug 15, 2017, 10:53 AM
SOP85_Using_NOAA_Climate_Records.log	18.6 KB	Aug 15, 2017, 10:53 AM

Note: If your Rstudio window doesn't show four windows as pictured here, don't worry! It's just that it's your first time in Rstudio and you haven't created a new file to work on. Using the toolbar at the top click on File then New File and then R Script to get the four windows as pictured here.

1. The first window is your **Document** window. Everytime you open or start a new file it will open up here. Rstudio has the capacitity to write many types of files including:
   - R markdown files (.Rmd) which are useful to quickly and easily embed R code into different formats. Once you've written up your R code you can use the KnitR package to easily convert them to a good looking, Microsoft Word, PDF or HTML file.
   - R Sweave (.Rnw) and TeX (.TeX) files to use the typesetting functionalities of the LaTeXlanguage to produce documents with embedded R code and output.
   - Shiny App files to make interactive web applications and data visualization tools
2. The Second Window is where you'll find your **Environment** and **Git** tabs. This window is used to track changes to your R "environment" and your local repository. You will learn to rely on this window once you actually begin loading and working with data in R. For now think of it

as a the place which shows you all the data, variables and functions your working with in R.

3. The third window is the **Console** window. This is like the command line in Rstudio for executing R code. Although you can write your code in your document (Window 1), anytime you actually wish to run a line of code or script it must first go into the console.

   Additionally this window displays the results of most computations and commands given to R. For example if you type 1 + 2 in the console and hit enter it should display a 3 as output.

4. Window 4 has several tabs including **Files, Plots, Help** to name a few. The Files tab is a graphical user interface (GUI) to help you open, edit and manage the files in your directory. While working in Rstudio you will oftentimes find yourself using data, images and information from different files so this tool is very handy for keeping track of where things are saved.

   Anytime you make a plot, image or graphic of your data it will be displayed in the Plots tab. The Help tab is used to get more information about specific R commands that go into the console.

Step 2: Create a Student Account with Github

Now that you know how to sign into Rstudio, Open a new browser tab or window and follow these steps to get started using Github:

1. Go to `https://github.com/join`

2. Fill in your information in Step 1 to create a new account. Be sure to use your student email address (with a .edu ending) in case you ever want to upgrade your account to a discounted student account.

3. in step 2 choose the basic/free plan. Should you ever want to create your own private repositories in the future you can upgrade to a discounted student account using the request form here: `https://education.github.com/` For now, the basic plan will suffice for the needs of our class.

4. in step 3 fill out the remaining questions, you can choose research and project management as your interests for using github

5. **Verify your email address** by clicking on the link that was automatically sent to your email address

Step 3: Create an SSH Key in Rstudio

Now that you have an active Github account, return to Rstudio. We will now attempt link your Rstudio Account to your Github account using an SSH key.

1. In an active Rstudio session click on **Tools** located in the toolbar at the top

2. In the dropdown menu select **Global Options**
3. In the new popup window navigate to **Git/SVN**
4. Click on **Create RSA Key...** button
5. The system will now ask you if you want to create a passphrase to go along with the new key. We recommend not using any passphrases for your first time. **Do not type a passphrase**
6. Click on **Create**
7. A window should pop up showing you the *private key* that Rstudio has generated close this window
8. You should be back in the popup window for Git/SVN global options. Click on **View Public Key**
9. **Copy** the text in the popup window and close it

Step 4: Link Your Github and Rstudio Server Accounts Using Your SSH

Now that you've copied the *Public Key* associated with your SSH key, we need to notify the Github server to be on the lookout for this key. Return to your Github account and follow these steps to link your Github account to Rstudio.

1. Sign into your Github account `https://github.com/login`
2. At the top right corner next to your profile picture click on the carrot to bring out the dropdown menu
3. Go to **Settings**
4. in the settings page click on **SSH and GPG Keys**
5. In the top right corner click on the button **New SSH key**
6. New sections should now appear where you can add a **Title** and **Key**. In the Title section type in "Rstudio Server". In the Key section **paste in the public key** you copied from Rstudio.
7. Click on **Add SSH Key**

Step 5: Create a Project in Rstudio Linked to Marc's "Beginner's Luck" Repository

If all was successful in the above steps your Rstudio and Github accounts should be able to link! Let's test this out by linking your Rstudio account to one of Marc's existing project repositories.

1. In an an active Rstudio session click on **File**
2. select **New Project** from the dropdown menu
3. in the popup window select **Version Control** and then select **Git**
4. You should now be prompted to type in a *Repository URL* ...
5. To get the URL open a new window or tab and go to `https://github.com/marclos/beginnersluck` ...This will take you to the page hosting Marc's github repository "Beginner's Luck"
6. Click on the green button **Clone or Download**
7. Copy repository's URL
8. Go back into Rstudio and **Paste in the URL from Github**
9. Under Project Directory Name be sure to give the project an appropriate title. For this example you can try "Beginner's_Luck"
10. Click on **Create Project**

Congratulations! You just created an Rstudio project linked to Marc's Repository. Marc and anyone else who's linked to this repository can update the files in it. You'll immediately notice that there are new files in your directory if you look at the File Navigation GUI in the lower right of the screen (Window 4). All these files are things that Marc and the other collaborators have uploaded to the repository. Uploading, editing, deleting and making changes to any of the files in the repository requires a couple more learning steps.

Push and Pull Something

You now have all the files in your directory but these files are really like copies of the ones stored in Marc's online repository. This is why it's called a "clone" of the repository. You are able to to open and directly edit the files right in Rstudio. However, if you want your other collaborators to see the changes you've made you have to **"Push"** the saved changes made to those files into the **"Master"** copy in the repository. Your collaborators, such as Marc, will be able to **"Pull"** the changes to the files and view them. The process of pulling and pushing changes to a project's repository requires several steps. Here's a excersice you can do to help you get used to it.

1. In Rstudio open the project you created for the Beginner's Luck Repository
2. Click on the **Git** tab (window 2 of Rstudio) ... This is the tab where you will go to manage all things dealing with Github and collaboration in your project.
3. Once you click on Git

4 Appendix A: Troubleshooting

Please send us a note of you have trouble, you might use the github issues button to document the problem.

4.1 https versus ssh

You may run into an error that forces you to login each time you want to uplad anything. This is because the cloning process was not done corerectly...

Changing a remote's URL MAC WINDOWS LINUX The git remote set-url command changes an existing remote repository URL.

Tip: For information on the difference between HTTPS and SSH URLs, see "Which remote URL should I use?" The git remote set-url command takes two arguments:

An existing remote name. For example, origin or upstream are two common choices. A new URL for the remote. For example:

If you're updating to use HTTPS, your URL might look like:

If you're updating to use SSH, your URL might look like:

Switching remote URLs from SSH to HTTPS

Open Git Bash.

Change the current working directory to your local project.

List your existing remotes in order to get the name of the remote you want to change.

git remote -v origin origin Change your remote's URL from SSH to HTTPS with the git remote set-url command.

Verify that the remote URL has changed.

git remote -v

The next time you git fetch, git pull, or git push to the remote repository, you'll be asked for your GitHub username and password.

If you have two-factor authentication enabled, you must create a personal access token to use instead of your GitHub password.

You can use a credential helper so Git will remember your GitHub username and password every time it talks to GitHub. Switching remote URLs from HTTPS to SSH

Open Git Bash.

Change the current working directory to your local project.

List your existing remotes in order to get the name of the remote you want to change.

git remote -v Change your remote's URL from HTTPS to SSH with the git remote set-url command.

Verify that the remote URL has changed.

git remote -v

4.2 Identifying Yourself

Tell Git who you are. Remember Git is a piece of software running on the server. This is distinct to GitHub, which is the repository website. In RStudio, click Tools:Shell…
Enter:

```
git config --global user.email "githubemailadddress@pomona.edu"
git config --global user.name "githubusername"
```

Use your GitHub username.

5 Additional Resources