1. Scope and Application

1.1 R, RStudio, and Github combine as a resource for data analysis and display.

1.2 We explain how these resources can be set up to create collaborative projects using Github repositories.

1.3 We do not expect you to be an expert on how to use R or Rstudio. However, some experience with a computer language will be helpful.

2. Summary of Method

2.1 This SOP provides instructions to create Rstudio projects and obtain Github repositories into Rstudio.

2.2 The SOP also provides some guidance on how to troubleshoot should push/pull problems arise.

2.3 The use of Github is not always intuitive and we can not cover all of the topics in this SOP. So, you should plan to rely on a wide range on-line resources to help.

Contents

3. Acknowledgements

As usual we acknowledge the students who have tried to follow and made suggestions on how to improve this guide. If you have suggestions or edits to improve the document, please let the author(s) know so they can make changes and you can be acknowledged.

The following have made substantial contributions:

- Aparna Chintapalli

4. Definitions

4.1 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.

4.2 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 convenient!

4.3 GitHub is a web-based Git repository hosting service.

4.4 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 terminalogy) 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".

5. Background

5.1 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, 3 reproducibility, and 3) accessibility:

Collaboration You can easily collaborate with others and track who contributed to various parts of the repository. When you are working in a team and updating various text and code, it's important to have version control to ensure that you can trace how the text and code was change.

Transparency The program allows others to see how you analyzed the data and thus, allows clear access to what you have done.

Reproducibility that when you conduct an analysis you can reproduce the results that can be used to validate your results.

Accessibility Accessible from any computer via webserver. Becuase we can rely on consistant behavoir and even web-based servers, the operating systems and computer hardware rarely limit what you can do in R.

5.2 However, becoming experienced in using these programs is like learning how to walk. We need to approach this process in discrete steps. Some people will experience weeks of mistakes before they are confident in their abilities to use these programs, others will quickly learn to run with the programs. How quickly you can feel comfortable with these programs will depend on many factors, but will be greatly improved by the time you invest!

5.3 I recommend reading the following sites to better understand Git and Github

- Rstudio Youtube
- Github Youtube
- Github for Beginners
- Github Fw
- Understanding the GitHub Flow
- http://blog.osteele.com/posts/2008/05/my-git-workflow/

6. Interferences

6.1 R has an updated version about every six months. When performing advanced analyses, there are times that new versions will no longer run a code. Thus, older versions of R must be maintained. This is a unheard of issue for new users – thankfully. It almost never occurs because new users rarely are running advanced codes!

6.2 RStudio Server needs a functional network connection. If the network is down, then Rstudio Server is inaccessible. This can be a source of frustration.

7. Health and Safety

7.1 Some risks include carpal tunnel syndrome.

7.2 Frustation! After running into errors that seem to baffle you, go for a run, get dinner, or take a nap. Often you'll find a work around after you take a break.

Safety and Personnnel Protective Equipment

7.3 Good posture and well designed work station

8. Personnel & Training Responsibilities

8.1 Researchers using this SOP should be well versed in webbrowser and file storage practices.

9. Required Materials and Apparati

9.1 Laptop or destop computer

9.2 Access to Pomona College's SSO (single sign on) and server

9.3 Github account

9.4 Patience

10. Estimated Time

10.1 This set up procedure requires 45 minutes.

11. Introducing Rstudio

RStudio Server Login

11.1 The Rstudio Server is hosted by Pomona College, thus you'll need a Pomona College login.

11.2 For non-Pomona College students, you'll need a new login each semester, which can be obtained from the help desk at Cowart Information Technology Services building.

11.3 Depending on your location, you can login to the server with `rstudio.campus.pomona.edu` if you are on the Pomona's network (or can use the VPN while off campus). However, if you are off campus and can't use the Pomona network, use this address: `rstudio.pomona.edu` from off campus including non-Pomona networks.

11.4 From non-Pomona college networks you might get an error (Figure 1). If you get this error, go to Advanced and accept the privacy settings to continue.

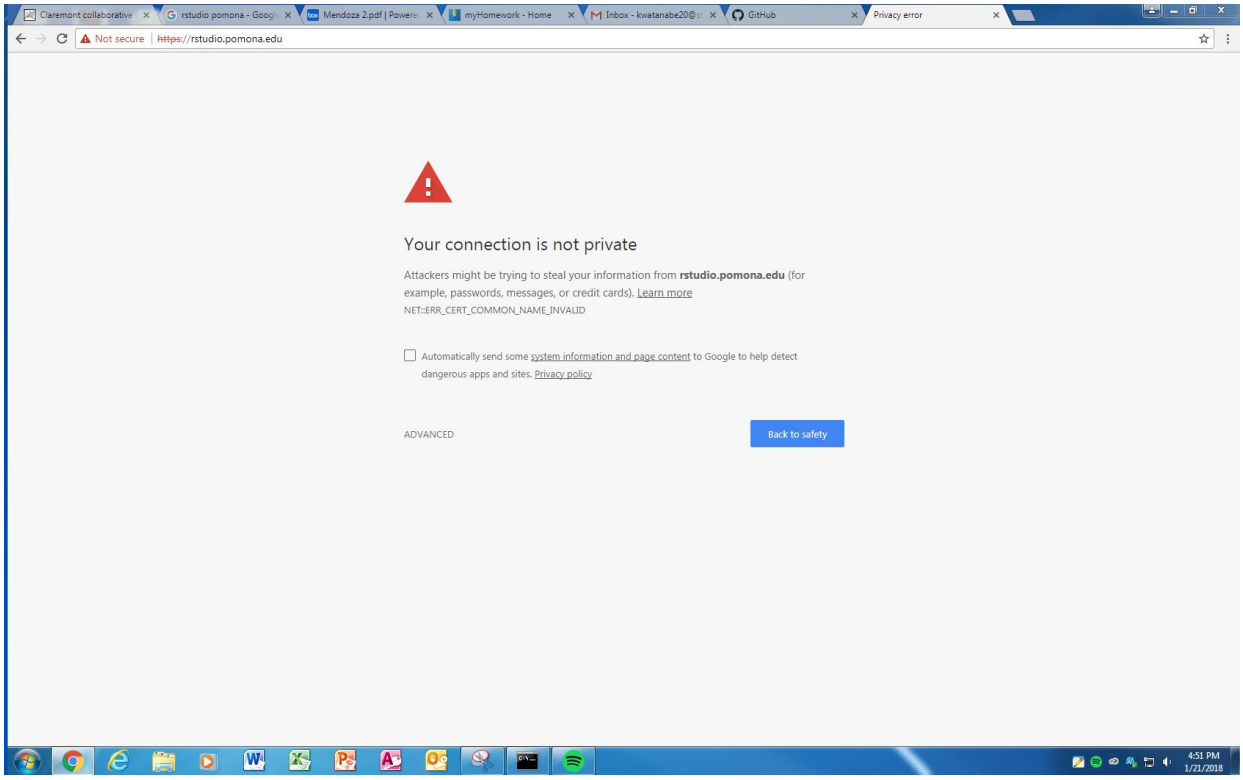

Figure 1: Privacy error can be ignored by going to the Advanced settings.

4 Rstudio Panels

Working with Rstudio is a bit confusing at first – there is lots going on! First, let's unpack the four default pannels that come up when you start Rstudio.

11.5 Each of the panels have tabs, but they can vary depending on what is active in Rstudio.

11.6 To see all of the panels, create the fourth window, we'll need to add create a new text file. Go to File/New File/R Sweave which opens a LaTeX file template. Save the file as "Test.Rnw" and now you have four windows.

Upper Left — Text and Code – Rnw, TeX, Txt, and Rmd files This window usually has the text files you are using to write and edit code and text. If there is not text open, this window will not be present and the console will take up the entire left side. A few things to note: You can have Rnw, Tex, dataframe previews, and text files open in this panel. I suggest you close ones you are not using to avoid working on stuff that you mean to.

Lower Left – Console and Compile PDF windows This is the R console, where you can enter (type directly or submit from the panal above) commands in to R from above. Without a Rmd/Rnw file open this panel might take the entire left side of the program's interface. This is where you can type or submit R commands and were

results are displayed. Also, any errors are reported in red here. If you have compiled a LaTeX file, the log file of the process will also be displayed.

Lower Right – Files/Plots/Packages/Help/Viewer This panel is handy to see what files are in your workspace, but also shows that packages that have been installed. When you make a plot, it will show up in this panel, but there are times that the plot is too big for the window. You might get a warning if that is the case, where you can usually resize the window to make the plat show up. Finally, it's a handy place to get the help manual to show up.

Upper Right – Environment/History/Git Panel You can use the "Environment" tab to see the files that R as access to. The "History" tab displays a history of the commands you have submitted and finally the "Git" tab can be used to manage files between the Github server and the local pomona server.

12. Setting Up Accounts and Communications Keys

12.1 Create Github Account: Go to Github.com and create an account. I suggest you use your Claremont e-mail adddress because this can come in handy later, should you want to create private repositories, which are free for college students! But you need to make a special request for this (`https://education.github.com/discount_requests/new`. For our purposes, you will not need a private repository and you can always request on later. Your email address can be used to prove your student status.

12.2 R, Rstudio, and Github can easily be installed on a desktop computer, however, please start by using the server versions via Rstudio.

12.3 Rstudio Server located at **rstudio.campus.pomona.edu**. Remember address only works if you are on Pomona's network.

12.4 Login using your Pomona College username and password. This is the address for when you are on campus and using a pomona login name.

12.5 To set up, change Rstudio's weave option to 'knitr'. Navigate to Tools/Global Options/Sweave, then change the 'weave Rnw files' box at the top from Sweave to knitr. This will have minor, but real changes in how files are compiled. Knitr has relatively new and more robust option. Apply and close window.

12.6 If using the downloaded Rstudio, start R Studio from the program files and this will automatically start an R console window.

Creating a Connection between Github and RStudio

12.7 Open Rstudio – which can be either as a local installation or using an RStudio Server via your webbrowser. However for this class, I continue to recommend that

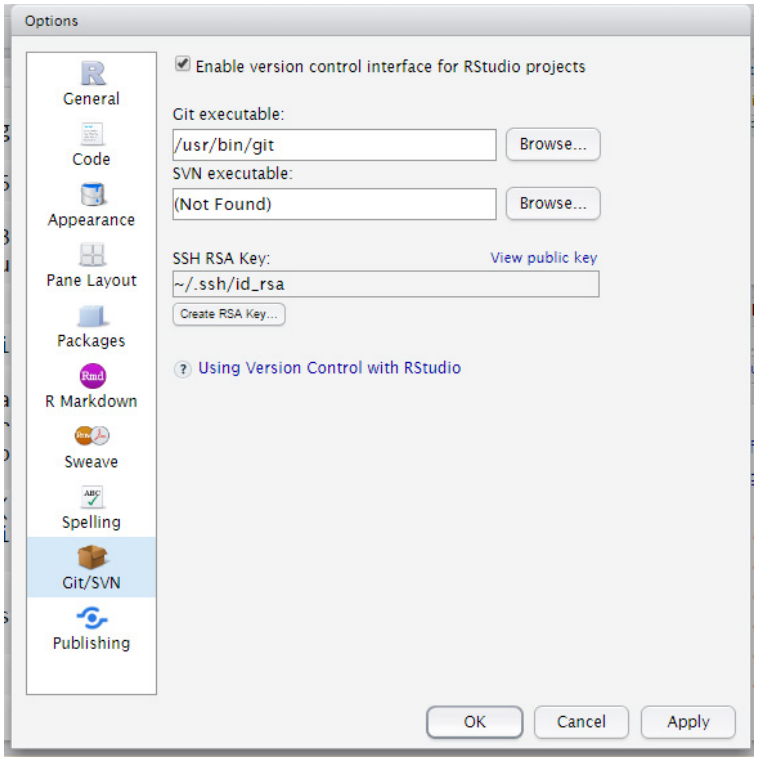

Figure 2: After natigating through the Rstudio Git/SNV menu, select 'Create an RSA Key'.

you use the RStudio Server.

12.8 To use Rstudio's version control, you must create a SSH Key that is used to open a secure connections between RStudio and Github.

12.9 To create a Key, follow the Rstudio menus Tools/Global Option and navigate to the 'Git/SVN' in the left menu.

12.10 "Create RSA key".

12.11 You have the option to create a passphrase. I suggest you ignore this for now, but should you have projects that need extra security, this is a good idea.

12.12 After you hit the "Create" button an image will be generated in a pop-up window – As far as I can tell, this image is some sort of encryption thing – but I don't see any reason why we should care. So, close this pop-up window.

12.13 Next, select the 'View the public key' and copy the contents of the pop-up window using ctrl-c. Now we need to paste that key into Github. NOTE: If you closed all the windows, you have to nagivate back to the Git/SVN menu via Global Options.

12.14 Add RSA/SSH Key to Github

1. Go to Profile, in the upper right top of the Github page. Unless you have uploaded a picture of yourself, profile image is a tetris looking icon.

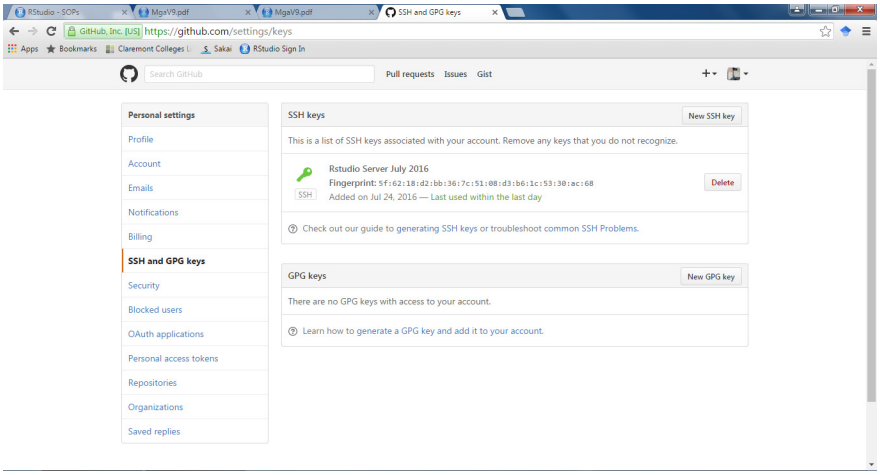

Figure 3: Creating the connection between Rstudio and Gitub requires a 'key'. We generate the key in Rstudio and then we tell Github what the key is.

2. Select the menu item "Settings."
3. Using the left menu, select the "SSH and GPG keys" button (Figure 3).
4. Click on new SSH key, type in name of key (e.g. 'myRstudio SSH key')
5. Paste in the RSA key, from your clipboard, into the window below.
6. Hit green "add SSH key" button.
7. Github and R can now communicate.

13. Creating and Updating Your Own Repository

How To Create a Fork in Github and Connect It to RStudio

A fork in GitHub is when you copy someone elses original repository into your own repository so that when you make edits, you are only making edits in your own pages. When you want to share your changes with the original repository, in this case the marclos/Environmental-Sciences-in-East-Asia repository, you can create a pull request which will be explained in the next set of directions.

In Github

- To start, you should ne logged into GitHub and have the marclos/Environmental-Sciences-in-East-Asia repository open and you should be logged into RStudio as well.
- To create a fork, navigate to the marclos/Environmental-Sciences-in-East-Asia repository home page and look to the top right of the window
- Click the button that says fork and you should be taken to a new page that has your account name followed by /Environmental-Sciences-in-East-Asia

- If you see a new page, you have successfully created a new fork where you will make edits from now on

RStudio

Now we have to connect your fork to your RStudio account so that all your work can be saved

- Find the green button that says code and copy the SSH link
- Then go to RStudio and create a new project
- File new project → version control → Git
- Then paste the link you copied into Repository URL and give your project a title like Environmental-Sciences-In-East-Asia-LastName and save the project in your home directory
- You should see a little box pop-up and load a bunch of things onto your RStudio
- Scan the bar at the top of your screen until you find the tools button, click it and then click shell
- In the shell type the following code

  `Git clone` (your SSH link you coped from GitHub earlier)

  `Git remote -v`
- This checks to make sure you correctly cloned the project. You should see something like this:
- Origin git@github.com:erta2020/Environmental-Sciences-in-East-Asia (fetch)
- Origin git@github.com:erta2020/Environmental-Sciences-in-East-Asia (push)
- Next we need to add Marcs repository to this project so that you can push and pull later on
- Go back to GitHub and under your fork repository name on the top left of your screan, click where it says marclos/Environmental-Sciences-in-East-Asia and you should be taken back to Marcs page
- Click the green code button and copy the SSH link before going back to RStudio
- Back in the shell, type the code below

  `Git remote add upstream` (the SSH key you just copied)

  `Git remote -v`
- Now you should see four lines, two labeled origin which is your repository and two labeled upstream which is marcs repository.
- Now you have successfully connected your RStudio to your new GitHub repository

How to Create Pull Requests

You want to make pull requests after you make changes to your work that you want to be reflected in everyone elses repository. You will need to create a full request to do this. For this class you will make one pull request each week, on Fridays so that your work can be looked at by Marc over the weekend.

Committing The Changes

- Click git in the top right box and then press commit
- After a new window pops up, make sure you click the small square next to the changes you made on the left-hand side
- Type a message in the commit message box that explains quickly what changes you made, then press commit
- Close the pop-up box and press push and then once that finishes press close again and go back to the main RStudio page
- Now open GitHub and navigate to your repository
- Click the button that says pull requests which will take you to a new page
- Click the green new pull requests button and you should see the change you just committed in RStudio
- Click create pull request
- Write a similar message that you types in the commit box earlier and finish the request by pressing create pull request
- Marc then has the opportunity to review the changes you made and accept them into the main repository so that your classmates can see your new changes

How to Pull Marcs Repository Updates Into Your Repository

This is what you need to do after your classmates changes have been added to Marc's main repository. Because you have a fork, the pull requests you make dont automatically appear to other students in their forks, so you have to pull that information into your fork to make sure that you can also see it. For this class, you will update your RStudio every Monday after Marc has reviewed your Friday pull requests

You can check to see if your Fork is ahead or behind Marcs repository by going to your GitHub fork and seeing if it says that your branch is "behind/head marclos by # of commits". Once you finish this and if you do everything correctly, your GitHub page will say that your repository Is even with Marc's.

Using The Shell in RStudio

- In RStudio, open a "shell" by looking for the "tools" button along the top of your screen and then clicking shell
- Write the following code in the shell, pressing enter before each new bullet point

```
\item Git pull upstream master
```

- This pulls all the updates made in Marcs repository

```
\item Git push origin master
```

- This finishes the updates and pushes them so they are saved in your repository
- Press the green push button in the top right box under git to make sure everything gets pushed to GitHub