	Environmental Analysis Teaching and Research Laboratory	Date: 1/16/2018	Number: 10 v.04
	Standard Operating Procedure	Title: Rstudio Projects and Github	
	Approved By: Marc Los Huertos	Revision Date: October 8, 2019	

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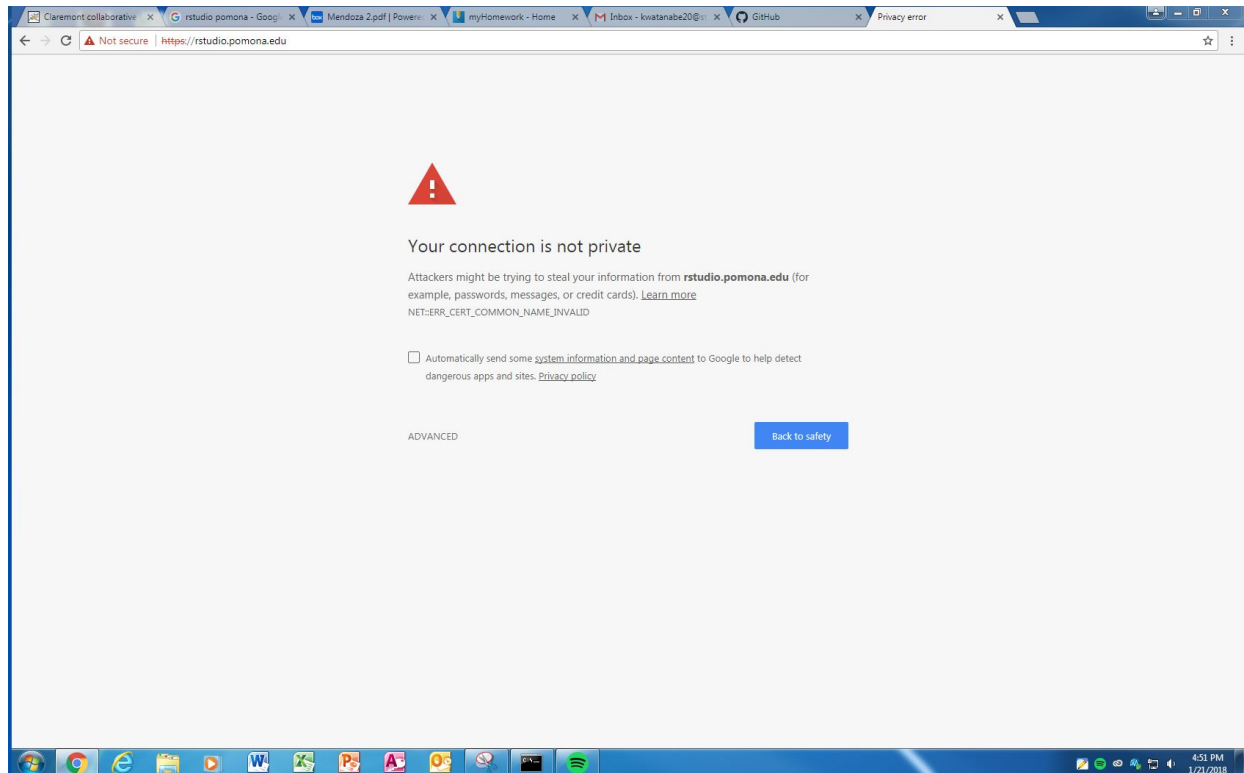
Figure 1: Privacy error can be ignored by going to the Advanced settings.

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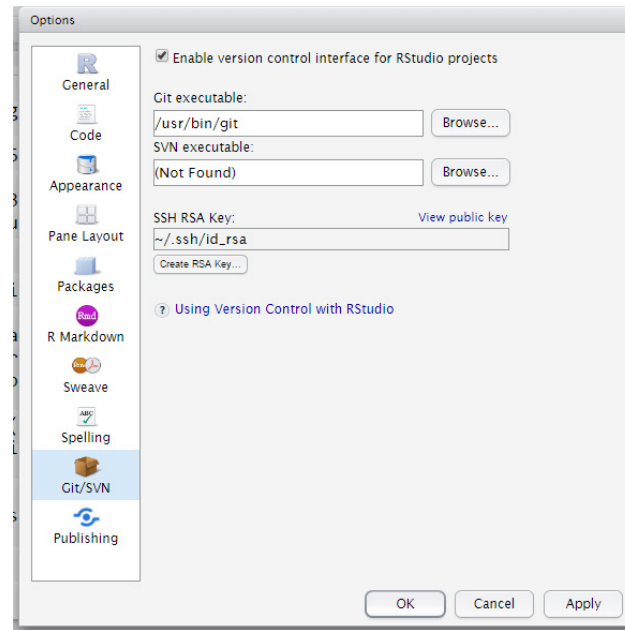
Figure 2: After natigating through the Rstudio Git/SNV menu, select 'Create an RSA Key'.

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.

2. Select the menu item "Settings."

3. Using the left menu, select the "SSH and GPG keys" button (Figure 3).

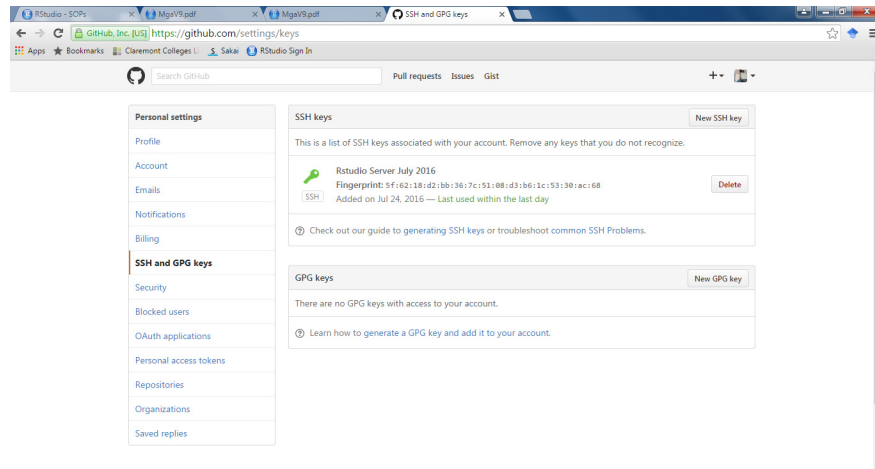4. Click on new SSH key, type in name of key (e.g. 'myRstudio SSH key')

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.

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.

Linking an RStudio Project to a Github Repository

12.15 To link a Github repository in Rstudio, you will need to "clone" the site, while we create a new project in R studio.
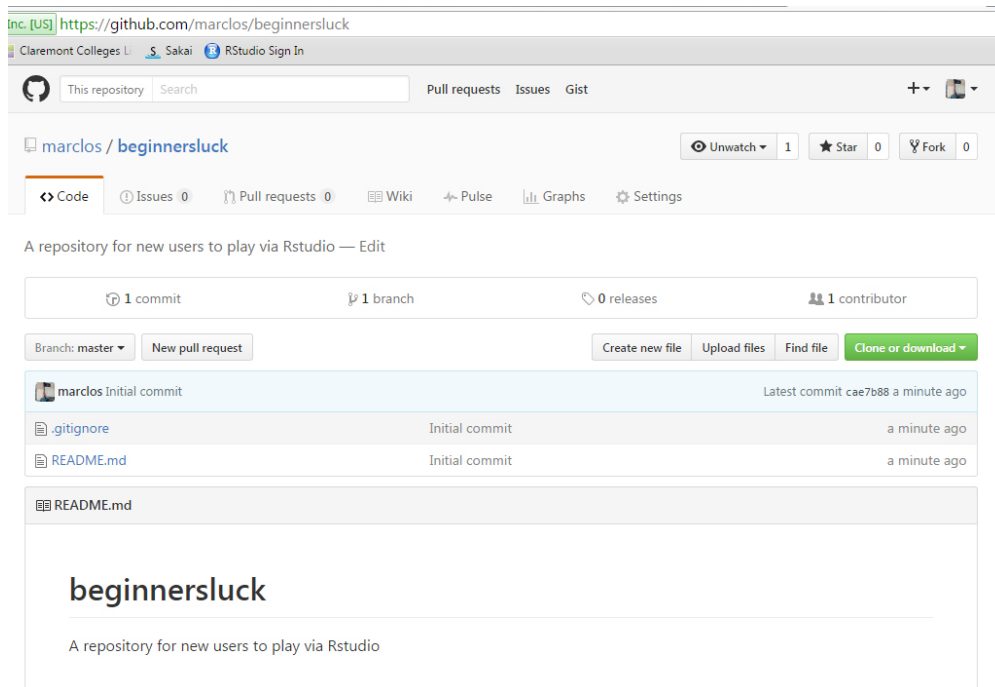
Figure 4: Step 1: Select the green button, "Clone or Download"

12.16 I have created a simple repository to begin learning how to use R Studio.

12.17 Search Github for the following repository name, 'beginnersluck'. There are several, so you'll have to find the one I created, under the username, 'marclos'.

12.18 Select this repository. Note the 'README.md', click on it to view it. The suffix, md, referrs to markdown, which is a simple languague for things to display on the web.

12.19 We need to get the url and SSH key for Rstudio and to accomplish this we "Clone" the Github repository. Click on the "clone or download" button (Figure 4).

12.20 Next we will 'Clone with SSH'. **IMPORTANT:** If the header says 'Clone with HTTPS', you can toggle the selection with the small 'Use SSH' link in the upper right of the box. If you skip switching to SSH, it's really hard to fix the project afterwards.

12.21 Copy the clone information by clicking on **the little clipboard** on the right of the address (Figure 5).
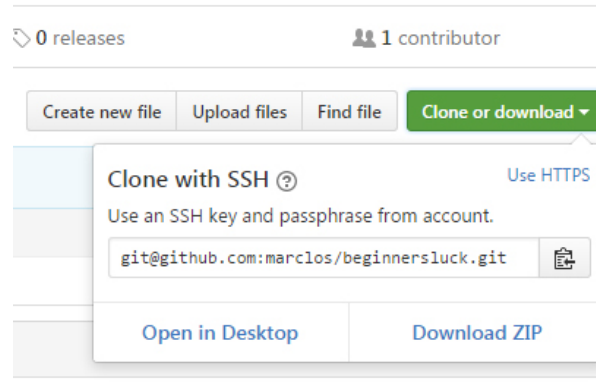
Figure 5: Step 2: Be sure you clone with SSH!

12.22 Now we return to Rstudio and create a new project. Navigate to the file toolbar and select "New Project". Where you will be greeted with three choices (Figure 6). Select 'Version Control'.
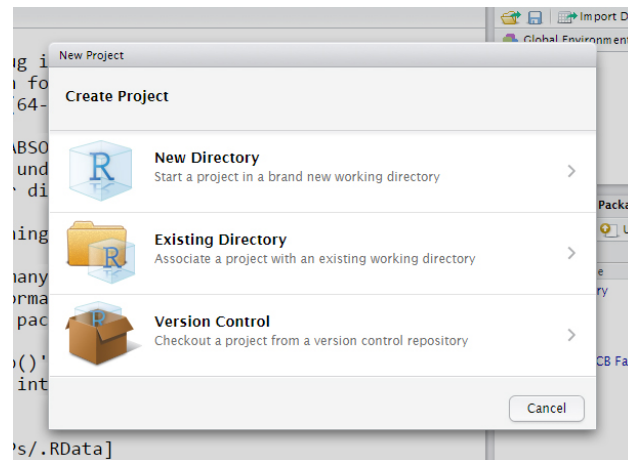

Figure 6: Step 3: When you create a new project, you have three choices. Select 'Version Control' to clone your repository.

12.23 Once you have selected the Version Control, you have one more choice to make, luckily this is pretty easy, select "Git" (Figure 7).
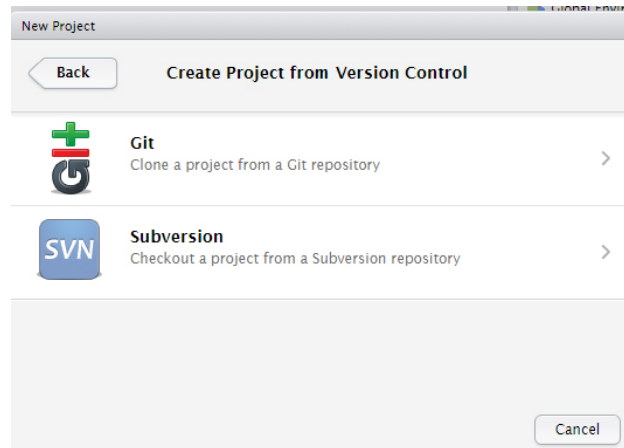
Figure 7: Step 4: Select "Git" to clone the repository.

When you arrive in the "Clone Git Repostory" window, you can simply paste your clipboard into the Repository URL, use `Ctrl-V` to paste in the text. This usually fills in the project directory name too. But if not, you can type in the name, in this case, "beginnersluck" (Figure 8). Then click on the "Create Project." button.
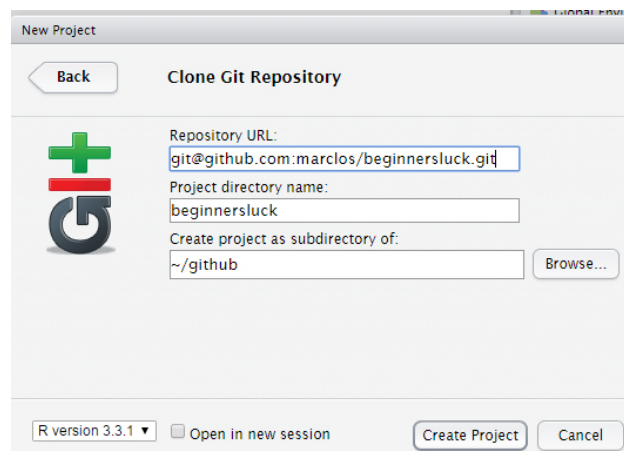

Figure 8: Step 5: You should have something useful in each of the boxes.

12.24 For many new users, Rstudio may claim some problem establishing authenticity (Figure 9). Type 'yes'. Then hit okay. I am not exactly sure why this step exists.

```
Clone Repository                                              [          ]  [ Stop ]

Initialized empty Git repository in /home/CAMPUS/smjx2015/beginnersluck/.git/

    ┌─────────────────────────────────────────┐
    │  Password                                │
    │                                          │
    │  The authenticity of host 'github.com    │
    │  (192.30.253.112)' can't be established. RSA key │
    │  fingerprint is                          │
    │  16:27:ac:a5:76:28:2d:36:63:1b:56:4d:eb:df:a6:48. │
    │  Are you sure you want to continue connecting │
    │  (yes/no)?                               │
    │  [_____] ⚷✓  │
    │                                          │
    │         [ OK ]      [ Cancel ]           │
    └─────────────────────────────────────────┘
```
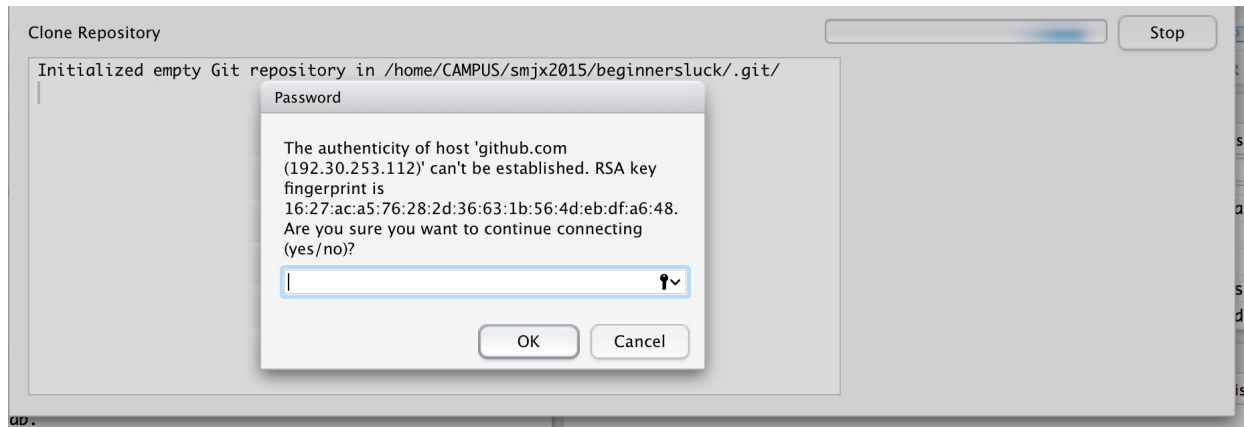
Figure 9: Step 6: You should have something useful in each of the boxes.

12.25 Now you should see a few files from the repository on your Rstudio window. Open them up and see what they are to get familiar with them.

12.26 If nothing comes up, your pop-up blocker is activated. Turn off the pop-up blocker for this site and try again.

12.27 The easiest place to begin, is to 'Pull' a repository once you have cloned it.

12.28 Any user can pull public repsitories.

12.29 Everytime you want to use something from a respository you want to pull to get the most up-to-date versions.

12.30 Now that you have created connectivity you can create your own repository. But first you need to create a project in Rstudio. I suggest you the following steps as you begin:

Pulling a Repository

12.31 Pulling a Repository is downloading all the changed files into your local (or server) directory. Using the downward facing blue arrow that labeled "Pull" in the Git tab in Rstudio.

13. Becoming a Collaborator – Wait for Instructions Before Beginning This.

13.1 To contribute to a project, you must be able "Push" and "Pull" to the respository.

13.2 When you are working on someone else's repository they will need to add you as a collaborator. And to accomplish this, it is easiest to send your user name to the owner, so she can add you to the collaborator list.

13.3 Once this has been done you can "Push" and "Pull" to the respository, but let's learn more about how to use R to create this project.

14. Working on Projects in Rstudio

Best Practices

Pull When you open RStudio, the first thing you should do is "Pull" from the repository to ensure your files are up-to-date. When you "Pull", you will get one of three results:

Already up-to-date. This means that your files have not been updated on the Github site since you last pulled the files. If you suspect someone has worked on the files, but are not getting those changes, it means that your collaborator has failed to "Push" these changes onto the repsository. If this is the case, you might need to go to the troubleshooting question to address this problem.

Successful Updates If your files are successfully updated from the repository, you will see:

summary of updates

master 6038765 Started to explain set up procedures

5 files changed, 434 insertions(+), 8 deletions(-)

create mode 100644 06_Rstudio_Github/Rstudio-and-Github-concordance.tex

create mode 100644 06_Rstudio_Github/Rstudio-and-Github.log

create mode 100644 06_Rstudio_Github/Rstudio-and-Github.pdf

create mode 100644 06_Rstudio_Github/Rstudio-and-Github.tex

Unsuccesful "Pull"

Commit

Push As you might guess, when you "Push" you can also have several outcomes:

Everything up-to-date is certainly simple.

Successful "Push" "To git@github.com:marclos/SOPs.git

e2efe6f..0b18250 master -¿ master"

Unsuccessful "Push"

Basically git commit "records changes to the repository" while git push "updates remote refs along with associated objects". So the first one is used in connection with your local repository, while the latter one is used to interact with a remote repository.

Here is a nice picture from Oliver Steele, that explains the git model and the commands:

Git Data Transport Commands
http://osteele.com
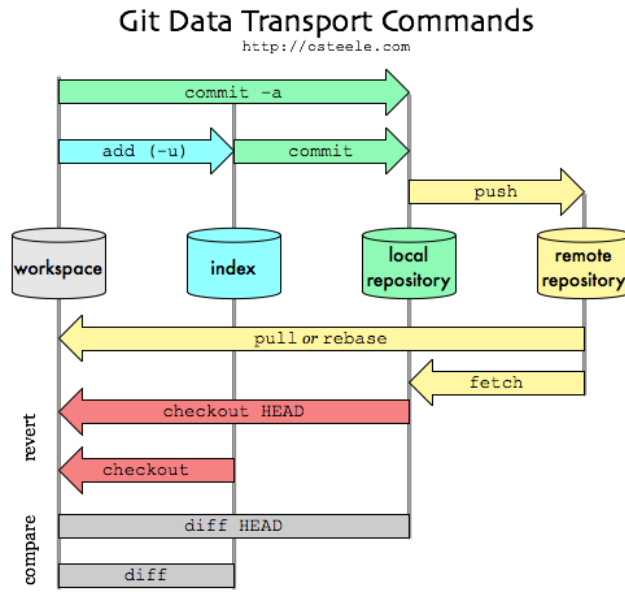
15. Creating a shared R Project

15.1 Learning to create a shared R project will help you create an understanding of the work flow, where we

1. Create a New Repository (if necessary)

2. Clone the respository

3. Pull

4. create a test file

5. Push

15.2 At this point, you can start using the Github respitories that have already been created.

16. Advanced Topics

Create and knitr new Rmd file

TBD

Downloading R, Rstudio, and Github as desktop programs

16.1 Note new users: Please use the server mode to begin using this software. Later you can use the desktop versions.

16.2 To accomplish install the programs on the local harddrive, download and install the programs in the following order:

1. R https://cran.r-project.org/

2. RStudio https://www.rstudio.com/

3. Github desktop https://desktop.github.com/

Creating a New Repository

16.3 Creating a respository in Github is pretty easy, but there are a couple of decision points worth pointing out.

16.4 Click on the green "New Repository" button.

16.5 A new page comes up and provides space for the name of the new repository. If you leave spaces, Github will automatically insert dashes between the words.

16.6 Decide if you want the site to be public or private. NOTE: Usually to get a private site, you have to pay, however, you can apply for an academic account where you can have private accounts.

16.7 Check the box if you want a README.md file created. I always create one – you can do it later, but it's nice to document what you are doing, so I suggest you create one.

16.8 Some files are not worth push and pulling all the time, but these files depend on the program you are using. In this case, we can have github ignore all files that are associated with R that don't really contribute to the project itself. So, I suggest you add these files to the .ignore file. You can add these later too, but it's a bit tricker.

16.9 Finally, Github allows you to create a licence to protect your work. I need to do some research about the differences, but I usually pick 'GNU General Public Licence' for no good reason, besides it sounds good.

16.10 The respository is now created, but Rstudio doesn't have access to it yet. You'll need to pull the respistory.

Clone New Project

TBD

17. Troubleshooting

Identity Error
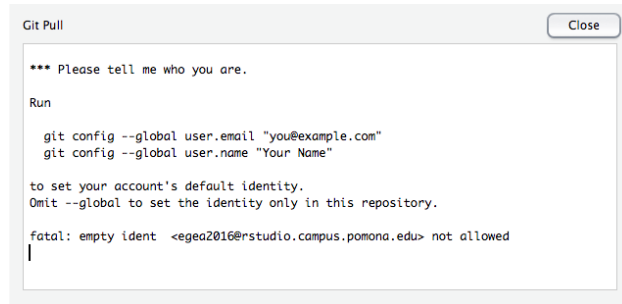
17.1 An identity error is...

Figure 10: Enter email and name

17.2 If you recieve an error as shown in the figure above, we can fix this using the Git shell. IN the "Git" tab, click on the gear icon and select "Shell..."

17.3 A new window will be shown and then type after the prompt:
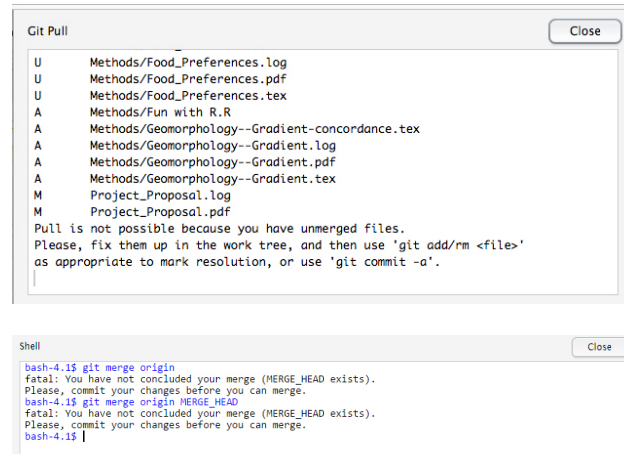'git config ... 'git config... '

17.4 These commands should sync rstudio to Github based on your email and name.

Push, Pull, and Merge Errors

17.5 At first when you're a newcomer to working on projects within Rstudio connected to a GitHub repository, you may forget the "best practices" of pulling, committing and pushing described above. You might have already gone through all the steps involved in setting up your GitHub account, linking your workspace in Rstudio with the correct project, and begun work on a specific file, but if you forget to update your workspace each time you come back to modify a file (especially in the case of coming back the next day to a file in Rstudio and forgetting to pull changes other collaborators may have made), you will run into problems committing, and pushing your changes. In this most common case, it's not likely that you'll notice any problems until you try to commit your changes. Thus, we'll begin this section on trouble shooting problems with Github and Rstudio at the committ level.

17.6 Alternatively, if you and others edit the same section, Github is going to need some "human" decision making to negotiate these changes. In Github jargon, this requires one to 'merge' changes.

17.7 (un)Fortunately, students are very good at discovering ways that the pull/commit/push process can be disrupted. As each case comes to my attention, I ask for a screen shot and a description, so we can trouble shoot each problem when they present themselves.

'Commit' Problems

17.8 In the scenario described above, you will get an error code when trying to committ your new changes to a file.

'Merge' Errors

17.9 If you find merge errors it often means that you and someone else has modified a file and they need to be reconciled.

17.10 Before trying to commit changes, We need to merge changes in the work tree.

17.11 To do this go to the list in the upper right hand corner of workspace in Rstudio studio (select the Git tab). In this list you should see your file changes, as well as other changes.

17.12 Select the files that have blue squares with your initials. The important part is to find the places where there is a multiple squares (ie. Blue squares, orange squares and red squares). Multiple squares signify that those sections there were changes made to the file that over lap and there should be a U next to the checkbox on those lines. Click those checkboxes.

17.13 Then commit and push changes.

'Pull' is rejected

17.14 Merging changes... how??
Pull is not possible because you have unmerged files. Please, fix them up in the work tree, and then use 'git add/rm ¡file¿' as appropriate to mark resolution, or use 'git commit -a'.

Ideally, if one gets a merge conflict, he should resolve them manually, and commit the changes using git add file.name && git commit -m "'removed merge conflicts". Now, another

user has updated the files in question on his repository, and has pushed his changes to the common upstream repo.

It so happens, that your merge conflicts from (probably) the last commit were not not resolved, so your files are not merged all right, and hence the U(unmerged) flag for the files. So now, when you do a git pull, git is throwing up the error, because you have some version of the file, which is not correctly resolved.

To resolve this, you will have to resolve the merge conflicts in question, and add and commit the changes, before you can do a git pull.

Dealing with non-fast-forward errors

17.15 Sometimes, Git can't make your change to a remote repository without losing commits. When this happens, your push is refused. If another person has pushed to the same branch as you, Git won't be able to push your changes.

17.16 You can fix this by fetching and merging the changes made on the remote branch with the changes that you have made locally:

- $ git fetch origin

- # Fetches updates made to an online repository

- $ git merge origin YOUR_BRANCH_NAME

- # Merges updates made online with your local work

17.17 Or, you can simply use git pull to perform both commands at once.

'Push' fails

17.18 Below are several potential remedies:

merge asdfasdf

Deleting a Project in R Studio If you are willing to sacrifice the changes you made or have mailed them to a collaborator to deal with you can delete the entire project in Rstudio. To accomplish this delete all files in directory, clear workspace and console, don't save, then go to session tab: Terminate R, and hopefully that will do the trick. Then commence with starting a new project.

18. Collaboration and Version Control

Adding Collaborators

18.1 Go to Profile, repository, toolbar in repository click settings, click on collaboration. Add usernames of collaborators to give them permission to collab.

Workflow tracking

18.2 Collaborators can create ways that each one is responsible for certain activities..

18.3 Branching...

19.   References

19.1