# Reproducible Manuscripts and collaborative work in R

Christoph Scheffel

## Preliminary Notes

This tutorial is intended to facilitate collaborative work on R analyses and RMarkdown manuscripts. Manuscripts can be written using RMarkdown. Results from analyses in R can be automatically integrated into the text. Furthermore, you can include tables or figures created in R directly to your RMarkdown manuscript. This is extremely helpful and provides a great benefit. One drawback, however, is that collaborative work is made more difficult because annotations and comments are difficult to insert. This can be at least partially solved with the help of Git Hub. Since there is a multitude of implementation possibilities, this tutorial does not claim to be exhaustive. All information listed here have been drawn and prepared from various documentations and tutorials, for example:

- https://happygitwithr.com/index.html
- https://support.rstudio.com/hc/en-us/articles/200532077-Version-Control-with-Git-and-SVN
- https://resources.github.com/whitepapers/github-and-rstudio/
- https://static1.squarespace.com/static/5757268f7da24f26ca7b21d2/t/5c7587114192021796d7cc84/1551206162093/R_Overleaf_Integration.pdf
- A Reproducible Data Analysis Workflow with R Markdown, Git, Make, and Docker, https://psyarxiv.com/8xzqy/

First, you will make all the necessary preparations in RStudio and GitHub. Then you will learn how to work collaboratively on scripts and texts.

# Set up Git and RStudio

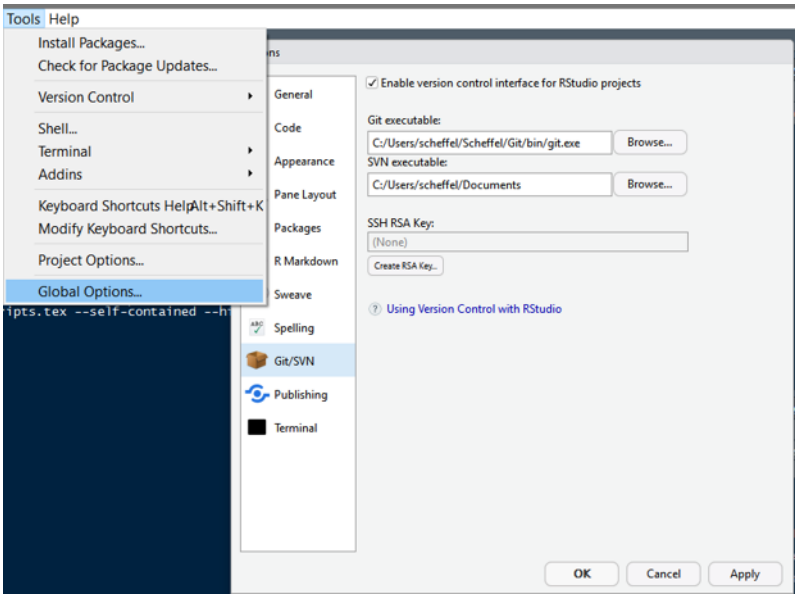
## GitHub and Git

First, you have to sign up at GitHub (https://github.com/) and download Git (if you have trouble installing Git, I would recommend this tutorial).

*Attention! If you use Windows, Git will automatically be installed into you C:/Program Files/ directory. Often, the space in the directory name causes problems. Therefore, I would recommend to choose a differenct directory for installation.*

## RStudio

Now that Git is installed, you need to link it to RStudio. Open RStudio and go to Tools -> Global Options. Navigate to Git/SVN, check Enable version control interface for RStudio projects and paste the path to your Git executable in the corresponding field and klick OK.

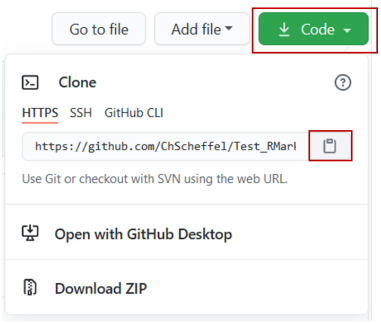



Now everything is set up and you can start a new project!

# Projects in Git and R

### Create new projects

First you need to set up a new repository on GitHub. Sign in on GitHub, go to the + in the top right corner and klick on New repository. Choose an appropriate name that you can easily remember. As there will be created folders later on, I would not recommend to use blank spaces in your names. Further, I recommend adding a *README* file where you can write extensive descriptions of your projects. Klick on Create Repository and you are good to go! Make note of the GitHub address for you repository. Klick on Code and now on the Notepad symbole next to the HTTPS address. The address to our repository is now stored in the clipboard.

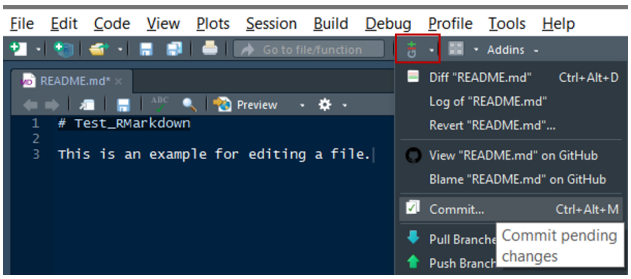



### Integrate projects in RStudio

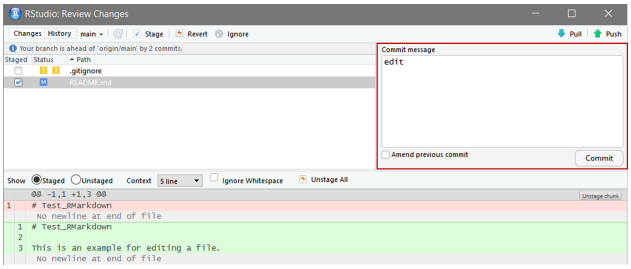
Open RStudio an click on File -> New Project. Select the option Version Control -> Git. Now paste the address of your Git repository in the first line. A new folder with the exact name of your Git repository will be created as subdirectory. *Do not change the folder name!* The location can be specified in the third line. All files from the Git repository are now copied to the local folder and you should be able to see all files from this repository in the FILES tap inside of RStudio. Now your R project is linked to the Git repository and you can start working.

## Committing

You can create and edit files in RStudio, as you are used to. For example you could open and edit the README file, you created in the Git repository. Now you have to commit changes to the Git repository. Click on the Version Control button in the control panel and select Commit....



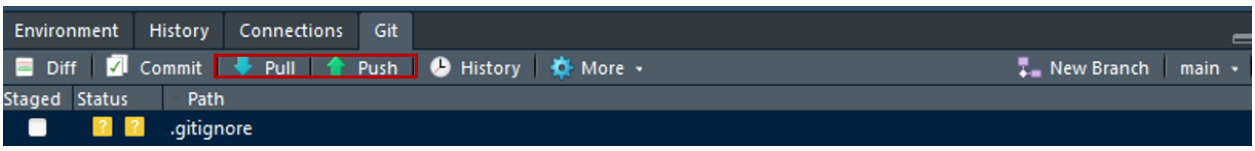
A new window opens were you can review all changes made. Add a comment explaining briefly all changes you made since the last commit and click on COMMIT.




All committed files should now disappear in the *Git* tap.

## Pushing and Pulling

If you start working with your project, you should always *pull* the latest version from your Git repository from GitHub. Further, once you committed all your changes, you can *pull* the latest version of the Git repository from GitHub. After you finished *pulling* you can *push* your changes to the Git repository (i.e. upload your changes). *Pulling* and *pushing* ist the essential feature for collaborative work! So make shure you and your collaborators always work with up to date files.
NOTE: The sequence of buttons on the RStudio git panel (that you can see in the picture) matches the sequence of actions necessary to submit changes to Git repository.



## Collaborative writing with RMarkdown scripts

In this section you will learn how to collaborative writing works with your reproducible manuscripts.

### Tips and tricks

Lastly here are some tips and tricks that that can facilitate collaborative writing:

1. In your RMarkdown file, use a new line for each sentence. Otherwise, GitHub will mark the whole paragraph although a co-worker only changed one sentence, because all sentences were in the same line.
2. Work with small commits! For every change or comment you make, commit the change. If you do all your changes and pull the whole file at the end, it gets very chaotic.
3.