Producing Reproducible Research Using RMarkdown and Github

Adam Dennett

12/10/2018

Table of Contents

**Health Warning - I've have only just started learning this stuff myself, so this document is bound to contain some key omissions and errors. As I continue this journey, I will update and refine and hopefully, in a few years, it might be something like a definitive guide!!**

In this practical session, you will learn how to produce work that is open, reproducible and portable using RStudio, RMarkdown, Git and Github.

* [RStudio](https://www.rstudio.com/) is a graphical user interface (that you should already be familiar with) - it contains a number of features which make it excellent for authoring reproducible and open geographic data science work.
* RMarkdown is a version of the [Markdown](https://en.wikipedia.org/wiki/Markdown) markup language which enables plain text to be formatted to contain links to data, code to run, text to explain what you a producing and metadata to tell your software what kinds of outputs to generate from your markdown code. For more information on RMarkdown, visit [here](https://rmarkdown.rstudio.com/lesson-1.html).
* [Git](https://git-scm.com/) is a software version control system which allows you to keep track of the code you produce and the changes that you or others make to it.
* [GitHub](https://github.com/) is an online repository that allows anyone to view the code you have produced (in whatever language you choose to program in) and use/scrutinise/contribute to/comment on it.

# Setting up GitHub to store your code

1. If you are working on your own computer, you will first need to install git - <https://git-scm.com/> - if you are working on the UCL Remote Desktop, you won't need to do this as it is already installed for you.
2. Go to <http://github.com> and install github (if working on your own computer). Create an account and create a new repository (call it anything you like - 'gis\_code' or something similar), making sure it is public and you check the box that says 'initialise new repository with a README' - click 'create repository' at the bottom

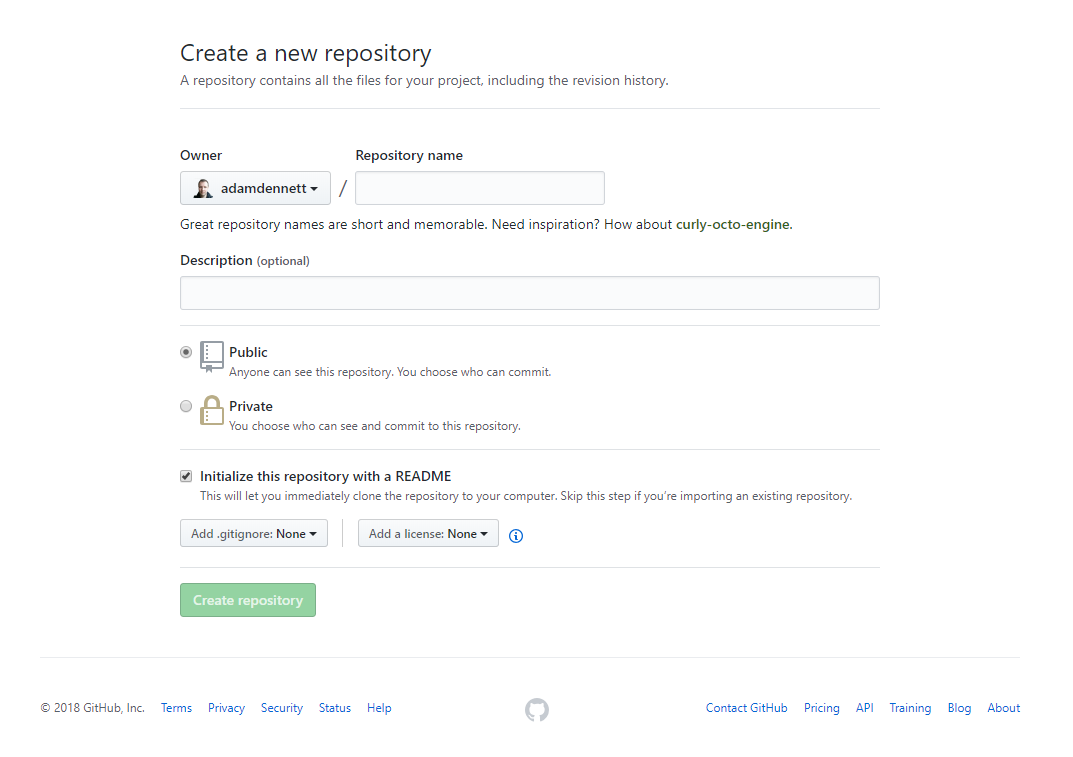


Figure 1 - Setting Up Your GitHub Repo

1. Your new repository ('repo') will be created and this is where you will be able to store your code online. You will notice that a README.md markdown file has also been created. This can be edited to tell people what they are likely to find in this repository.
2. Now you have created your repo online, you need to 'clone' it so that there is an identical copy of it in a local folder on your computer.

There are a couple of ways of doing this, but the easy one is to use the GUI that comes packaged with your git installation.

1. The first thing you need to do is copy the Clone URL for your repo from the github website - click the green button in your repo for 'Clone or Download' and copy the link:

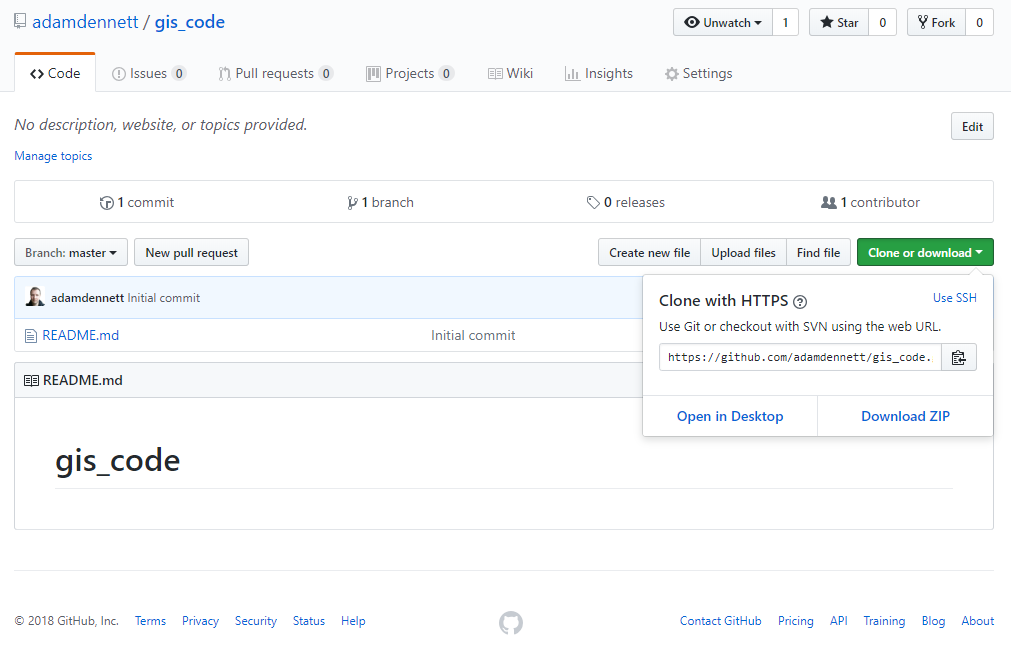


Figure 2 - Getting Your Clone Link

1. Now in the windows start menu, go to git > GUI

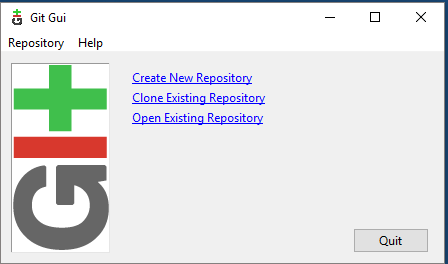


Figure 3 - Git GUI 1

1. Select 'Clone Existing Repository' and paste the link from your GitHub account into the top box and the local directory that you want to create to store your repo in the bottom box (*note, you will need to add a name for a new folder, once you have selected an existing directory*).

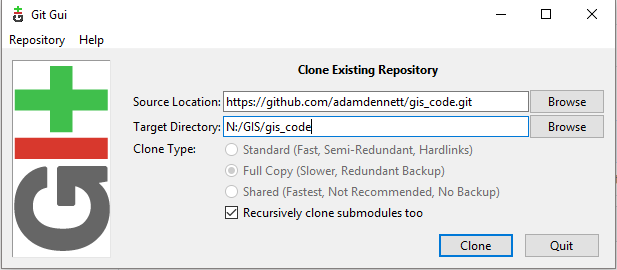


Figure 4 - Git GUI 2

1. After a few moments, you should now be able to view a copy of your GitHub repo on your local machine. This is where you will be able to store all of your code and some other files for your reproducible research.

# Using RStudio with git

Now, as I've mentioned before, RStudio is totally bad-ass. Not only does it make R fun to use, but the lovely people who created it also built in support for things like git!

For a full and excellent tutorial on using Git with R Studio, watch this webinar: <https://www.rstudio.com/resources/webinars/rstudio-essentials-webinar-series-managing-part-2/>

If you don't want to watch the vid, I'll do a quick summary below. So, to use git, first you need to enable it in RStudio:

## Issues with the RStudio Installation on the UCL Remote Desktop

At the time of writing, because of the way that RStudio has been installed on the UCL Remote Desktop, git integration does not work. As such, if you are working on the remote desktop, DO NOT follow instructions 9 to 19 below, jump straight to instruction 20. If you are working on your own computer then fine, carry on.

# Using RStudio with git, continued...

1. Open RStudio. In RStudio Tools > Global Options, under 'Git/SVN' check the box to allow version control and locate the folder on your computer where the git.exe file is located. Allow Version Control for new Projects and navigate to where the git.exe file is on your computer. Click OK.

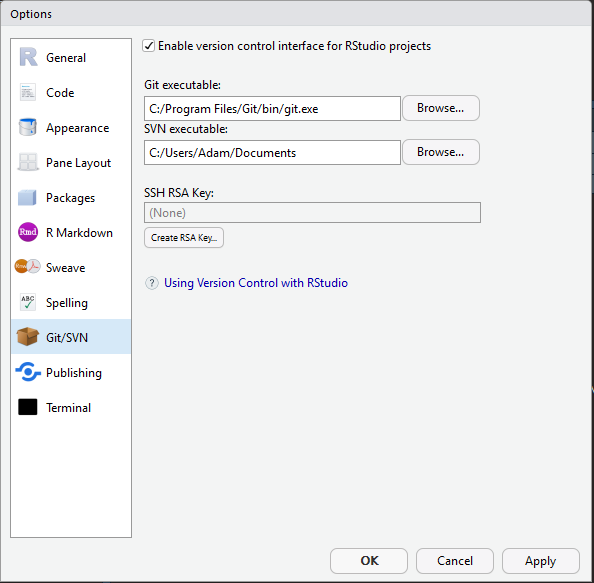


Figure 5 - Allow Version Control

1. Now in RStudio, you should create a new project in an existing directory - File > New Project > Existing directory - choose your new git repository as your new project folder. You may not need to restart RStudio.

## Saving your work to your local cloned repo

1. Open a new R Notebook in RStudio: File > New File R Notebook
2. Type some stuff (anything so that's it's not a blank empty file) at the top of the file and save it.
3. As well as saving, which saves a copy to our local directory, we will also 'commit' or create a save point for our work on git. To do this, you should click the 'git' icon and up will pop a menu like the one below:

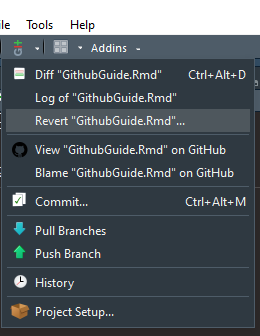


Figure 6a - Git Integration with RStudio 1

You can also click the Git tab that will have appeared in the top-right window of RStudio

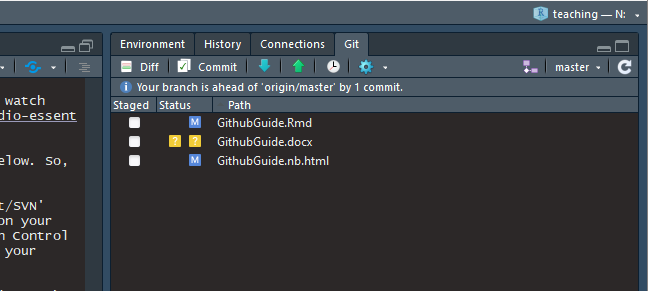


Figure 6b - Git Integration with RStudio 2

1. To Commit (save) the changes in this file to your local cloned git repository, first click 'Commit'
2. Up will then pop another window that looks a little like the one below:

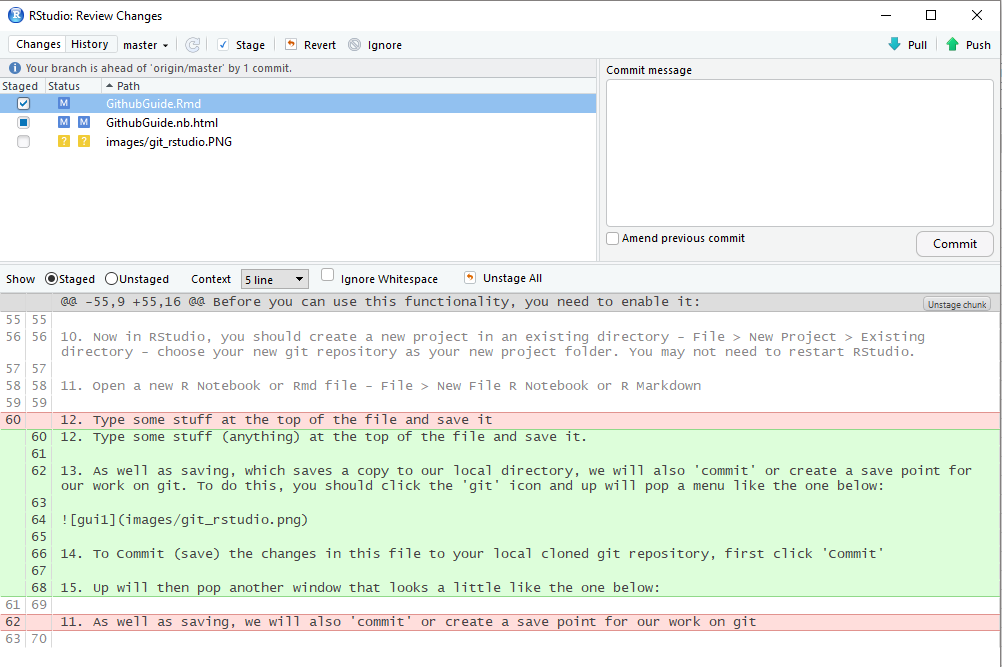


Figure 7 - Reviewing Changes

1. In this window, you will be able to review the differences between any previous saves you have made to your document and the current changes. If you are happy with the changes, you can then select the file and click 'commit' to save them to your current local repo. If you are not happy, then you can always revert back to a previous version that you know did work!
2. If I were you, I would save R Scripts or RMarkdown Documents using the usual save button quite regularly, and then every time you, then you can commit your changes then.

## Pushing your changes to GitHub

1. Once you are happy with your progress, you should also then 'Push' your changes to your online GitHub repo. This is important, both for backing up your work, but also for keeping a reproducable record of your research.
2. To do this, first make sure you have committed any changes to your local cloned repo and then click the 'Push' button to whizz your code up to your master GitHub repo - you will probably be prompted to enter your github username and password to enable this...

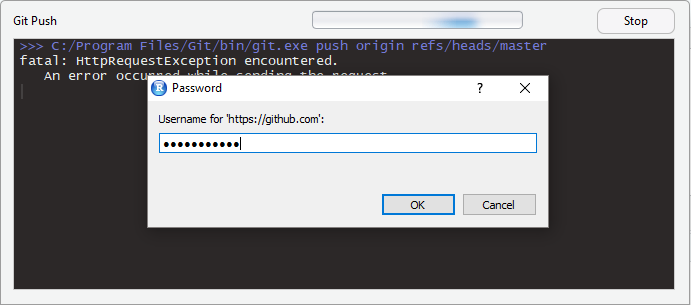


Figure 8 - Pushing to GitHub

# Git without RStudio Integration

Now, if you would like to use git but your're working on the UCL Remote Desktop or you are experiening other problems with getting git working in RStudio, fear not, you can just use your raw git installation.

1. In the Start Menu, open the git GUI. Start > Git > Git GUI. You should open the existing repository that you have just created.
2. Whenever you have made some changes to your files in your cloned repo, you can use git to review the changes and 'Commit' (save) them and then 'Push' them up to your master repository on GitHub.
3. To review and commit your changes, in the commit menu, simply:
4. scan for changes
5. stage them ready for committing
6. commit the changes
7. push the changes to your GitHub repo

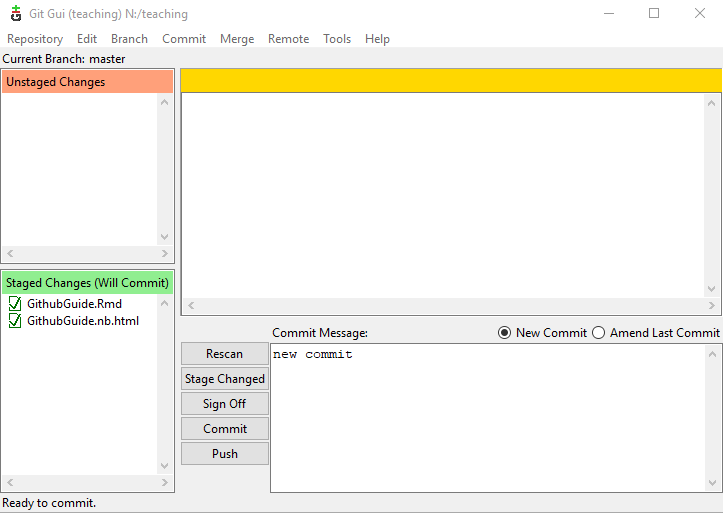


Figure 9 - Git Gui Commit

1. Try it. Add some new text to your .Rmd file. Save it. Rescan in the Git GUI to check for changes. Stage those changes one-by-one. Commit them (remembering to input a commit message). Then once committed, try using Remote > Push to send the changes to github (\*note, you will be asked for your username and password to complete this.)

## Possible git problems - Merging

1. There are many many possible git problems. I've just started with this and have come across loads (mainly to do with merging) already. As you continute to commit and push work, especially when using different clones, you may well run into things like merge conflicts and a whole load of other problems. I've just been trying to fix a merge conflict for the last hour and eventually got there. It's not easy. It's not pretty. But as ever, there is usually a guide somewhere on the internet to help you.

For merge conflicts, try here: <https://help.github.com/articles/resolving-a-merge-conflict-using-the-command-line/>

For general github help, try here: <https://help.github.com/>

1. An initial tip from me - looks like whenever you are working with a local github repo, if you come back to your work after shutting down and restarting, then the first thing you should do is 'Pull' from your GitHub repo before going any further. If you don't, you may well have all kinds of fun to solve with merging files together afterwards.

Good luck!!

# Reproducible Research Using R Markdown

OK, so now you have set everything up so that you can become a reproducable research ninja! All that remains is to do some reproducable research!

For the definitive guide on R Markdown, please read [R Markdown: The Definitive Guide](https://bookdown.org/yihui/rmarkdown/) - obviously! It will tell you everything you need to know, far beyond what I am telling you here.

There is also an excellent guide on the R Studio website -<https://rmarkdown.rstudio.com/lesson-1.html>

And a quick cheatsheet here: <https://github.com/rstudio/cheatsheets/raw/master/rmarkdown-2.0.pdf>

And an older one here: <http://www.rstudio.com/wp-content/uploads/2016/03/rmarkdown-cheatsheet-2.0.pdf>

One of the awesome things about R Markdown is it can be converted into a range of different formats - html for webpages, word documents, PDFs, blogs, books - virtually everything!

Now, earlier on in this exercise, I got you to open a new R Notebook or R Markdown file. They are both R Markdown Documents, but the Notebook allows chunks of code to be run independently.

1. Go back to the notebook you created earlier in step 11. We are now going to insert some code from the practical last week and run it.
2. In RStudio, you can either select Code > Insert Chunk or you can Click the 'Insert' button and insert an R Chunk

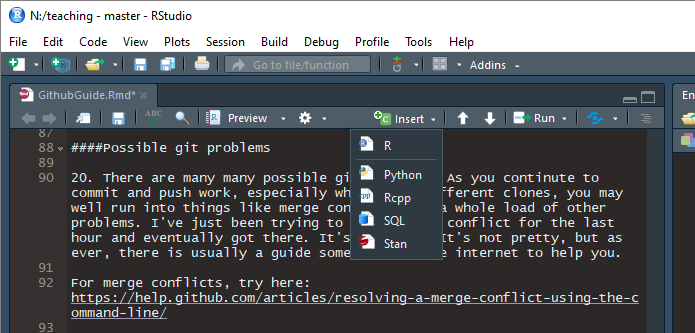
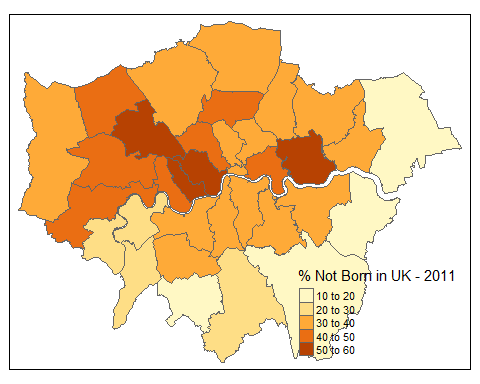


Figure 8 - Insert R Chunk

1. A box will appear and in this box, you will be able to enter and run your R code. Try pasting in:

library(tidyverse)  
library(geojsonio)  
library(sf)  
library(tmap)  
library(tmaptools)  
  
#read some data attributes  
LondonData <- read\_csv("https://files.datapress.com/london/dataset/ward-profiles-and-atlas/2015-09-24T14:21:24/ward-profiles-excel-version.csv", na = "n/a")  
#read some geometries  
EW <- geojson\_read("http://geoportal.statistics.gov.uk/datasets/8edafbe3276d4b56aec60991cbddda50\_2.geojson", what = "sp")  
#pull out London  
LondonMap <- EW[grep("^E09",EW@data$lad15cd),]  
#convert to a simple features object  
LondonMapSF <- st\_as\_sf(LondonMap)  
#append the data to the geometries  
LondonMapSF <- append\_data(LondonMapSF,LondonData, key.shp = "lad15cd", key.data = "New code", ignore.duplicates = TRUE)  
#plot a choropleth  
qtm(LondonMapSF, fill = "% Not Born in UK - 2011")



1. When including code chunks in your work, there are various options that allow you to do things like include the code, but not run it: display the output but not the code, hide warnings etc. Most of these can be input automatically by clicking the cog icon in the top-right of the chunk.

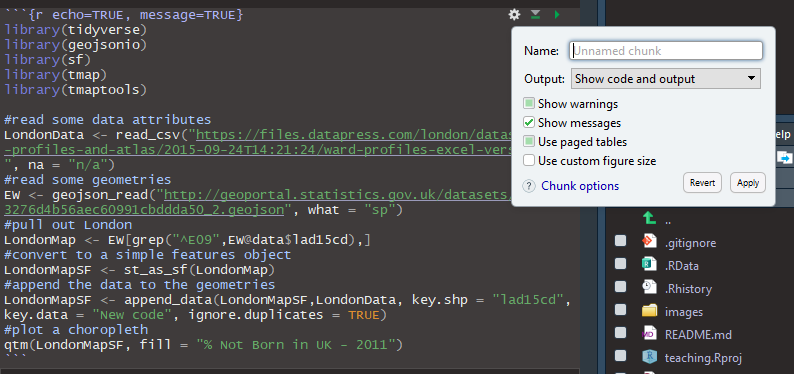


Figure 9 - Chunk Options

1. Various other options and tips can be found in the full R Markdown guide on RStudio here: <https://rmarkdown.rstudio.com/lesson-1.html> and in this the cheatsheets linked to above.

# Adding References to Your Work

Now, as you build up your documentation for the work you are doing, it is likely that you will want to include references to support your arguments.

Building a bibliography in your R Markdown document is quite simple, you will just need to install some additional packages that plug into RStudio and learn how to export your bibliography from your reference management software.

## Exporting a bibliography from Zotero

Here I will assume that you are going to be using Zotero as your reference management software. If you use other software, there are probably similar options for exporting available to you, so don't worry.

The aim is to get a list of references that are stored in a format known as BibTeX - <http://www.bibtex.org/>

I won't go into BibTeX now, but if you go on to write your dissertation using [LaTeX](https://www.latex-project.org/), then BibTeX is the format that you will use to store your references.

1. Having built up a bibliograpy of relevant literature in zotero, you should export it in BibTeX format and save it to the same directory that your R Markdown project is located in. Go to File > Export Library and select BibTeX as your format.

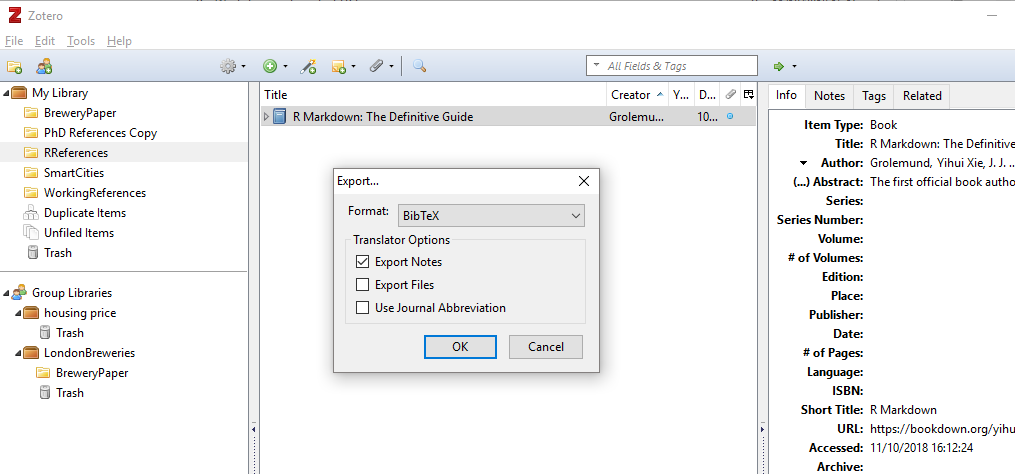


Figure 10 - Zotero Export

1. Once your BibTex file is saved into the same directory as your Markdown document, you can then tell it where to find the bibliography in the YAML text at the very top of the file:

---  
title: "Producing Reproducible Research Using RMarkdown and Github"  
output:  
 html\_notebook: default  
 word\_document: default  
   
author: Adam Dennett  
bibliography:   
 - RReferences.bib  
   
---

## Adding References Using citr

1. In order to add add citations from your bibliography, you can use the citr package, which creates an addin within RStudio

install.packages("citr")  
library(citr)

1. In the 'Addins' menu near the top of RStudio, you should (once RStudio has been restarted) have a citr option for 'Insert citations' and including them in your work.

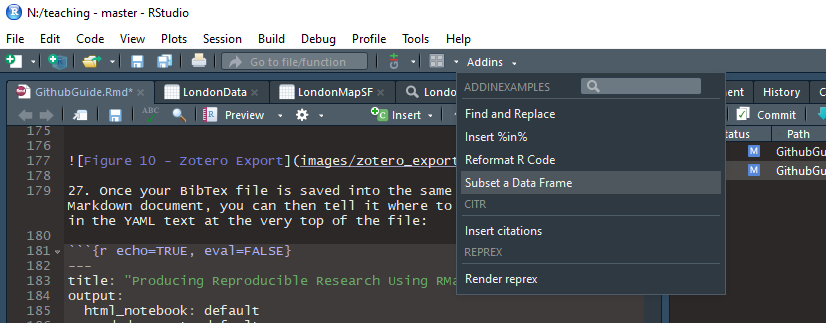


Figure 11 - Citr Addin

1. Try adding some citations from your BibTeX library... You will find that once you have added them into the text, citr will build a bibliography at the bottom of your document for you - neat, hey?!

Here is a reference for a seminal and totally game changing paper by Dennett and Page (2017). Here is another reference (Xie, Allaire, and Grolemund 2018) that heped me write this post. And look at how they all stack up alphabetically below...

## References

Dennett, Adam, and Sam Page. 2017. “The Geography of London’s Recent Beer Brewing Revolution.” *The Geographical Journal* 183 (4): 440–54. doi:[10.1111/geoj.12228](https://doi.org/10.1111/geoj.12228).

Xie, Yihui, J.J. Allaire, and Garrett Grolemund. 2018. *R Markdown: The Definitive Guide*. <https://bookdown.org/yihui/rmarkdown/>.