Extremely Simple Guide To Collaborative Writing Between R Markdown and Google Docs Users

Prepared by: Matthew E. Vanaman

Last Updated: 03-17-2021

Table of Contents

Feel free to skip my treatise and get right to it.

Get this guide as a PDF.

We scientists write a lot. We format a lot too, because our journals have scrupulous formatting requirements. If indeed you are a scientist who writes, you have probably been as frustrated as I have with how tedious that stuff can be, especially formatting tables.

At the time of this particular writing, it's been two years since I felt the euphoria of discovering that you can write manuscripts in R using R Markdown. With the `papaja` (Preparing APA Journal Articles) package, for example, all formatting is automated, including in-text statistics, tables, and references. You simply plug in your info (author names, title, abstract, etc.) and baddo-boom badda-bing you got yourself an APA manuscript ready for submission.

To sweeten the pot even more, R Markdown just received a new upgrade adding point-and-click functionality to most of the core features writers need, including inserting comments, formatting text, creating headings, and inserting cross-references, effectively removing the need to know any coding to get access to R Markdown's flexibility, transparency, and reproducibility. It even has an as-you-write pop-up window that pulls references from Zotero - even the Wordiest Word Users That Ever Worded have to admit how bad ass of a feature that is.

Given that R provides a process for streamlining your analysis and writing from front to back, and can even be used to create online surveys, why do scientists use anything *but* R? The answer can be summarized in two words: Old Guard. Or, less prejudicially: Learning Curve. In other words, it's easy for someone like me, a graduate student, to pick up new habits, because being the squalidly grad student I am, I could be said to have established few of my own. The same cannot be said for those who have a routine that has worked just fine for them for years.

Many younger scientists, including myself, have oriented their habits around the efficiency, transparency, and reproducibility of consolidating analysis and writing into an R Markdown workflow. Some have tried it and turned back; some have been considering it, and some have just heard about it on Twitter and made a note to look into it more closely when they have time. Everyone in the small-but-rapidly-growing niche I have just described either has or will become frustrated by the fact that you lose many of the time-saving features of writing in R when collaborators will not accommodate your (would-be) workflow. Many don't believe that making the switch could save them time in the long run, and frankly why should they? All they have are your anecdotes. To be clear: they are wrong. But to be clearer, I don't hold anything against them, and neither should you, because it *is* a big leap of faith on their part to trade in Microsoft Word take our word for it.

But this doesn't really help you, R aficionado. You need to find a solution, or else you'll have to give up on writing in R Markdown. Luckily, you don't have to, because people far smarter and skilled than I have solved this problem for us. Here is an easier-than-you-thought guide to salvaging your workflow of writing in R Markdown while collaborating with an ever-more popular writing tool: Google Docs.

Note: I focus on Google Docs here because, although there are several ways to collaborate with Word users, I have never found one as simple as this workflow for collaborating with Google Docs users. And Google Docs is on the rise.

Assumptions of This Guide

I assume the following of users of this guide:

- You are the lead author, or are otherwise in charge of formatting and analysis, which is to say you are overseeing the rendering of the final manuscript in R Markdown.

- You are slightly competent in R (seriously, you only need to be slightly competent to adopt an "All R, All the Time" analysis and writing workflow, contrary to what you may have heard).

- Your collaborators prefer writing in a Google Doc, or are willing to (Google Docs has track changes, comments, and document version history, if you need to sell it. It sucks at formatting, but in this scenario, you are in charge of that with R Markdown).

Workflow

The workflow is super simple. At each stage of revision, you, Stewardperson of R, will have written the initial draft within R Markdown. Once the draft is ready for revisions from collaborators, you will use R to upload the contents of your R Markdown document to a Google Doc on your Google Drive. From there, you and your collaborators use Google Docs to revise. Once everyone is satisfied with the content, you will download the Google Doc content back to your R Markdown file, which will rewrite the contents of the file to include any changes that were made.

The pros of this workflow:

- Best of both worlds; I won't rehash the benefits of writing in R Markdown or Google Docs, the point is you get access to both.

- Collaborators are happy, having avoided the inevitable for a bit longer, though the creep toward having to learn R to escape the limitations of Google Docs and Word will march forward.

- The workflow is efficient - there is little added cost to you, R Stewardperson, of adopting this workflow. Collaborators will barely notice.

- The workflow works nicely with version control, if you use it (you absolutely should use version control, you won't go back after learning it).

The cons:

- *All* content in the R Markdown file ends up in the Google Doc. So your collaborators will have to be OK with scrolling past some code to get to the relevant sections.

  - Though you could mitigate this by dropping in your analysis code into R Markdown after revisions are over.

- Collaborators with no experience in R or R Markdown will not be able to preview the formatted document, which if you are doing things the efficient way, will include the bibliography, in-text statistics, tables, and document formatting such as double spacing, APA headers, etc.

Getting Set Up

Once your draft is ready to share with others, you can use the following steps to get R setup to communicate with Google Drive, where your Google Doc will live.

1. First, install the relevant packages. You'll need two packages to do this:

   1. `googledrive`. This package is on CRAN, so you can install it the usual way using `install.packages("googledrive")`.

      1. There is a nice walkthrough of the `googledrive` package if you are interested in learning more, though you will not need to for this guide.

   2. `trackdown`. This package provides wrapper functions to the more-complicated `googledrive`. `trackdown` is not on CRAN, so you'll need to install with `devtools::install_github("ekothe/trackdown")`.

      1. There is a very brief walkthrough of the `trackdown` package if you want to know, though you will not need it for this guide.

2. Authenticate Google Drive so R can access it. There is a long and extremely tedious process for doing this manually that you can avoid by using the following steps:

   1. Restart R with a fresh environment.
   2. Run the `googledrive::drive_auth()` command (do not add arguments). In the R console, you should see a prompt asking you to select 1 or 2. Option 1 didn't work for me, so I entered option 2.
   3. Your web browser should open with a Google sign-in page. Sign in to Google, and follow the steps to allow access.
   4. In the console, you will see `TRUE` if the authentication was successful.

Note that if you are following this to the T, you will need to do Step 2 every time you want to send something to Google Drive.

Send A Draft To Google Drive

To send your document to Google Drive, use the following code:

```
# to send to google drive
trackdown::upload_rmd(
    # leave out file extension (i.e., no ".Rmd" at the end)
    file = "name-of-your-R-Markdown-file",
    gfile = "name-your-file-will-have-on-Google-Drive"
    )
```

This can take a minute. Once it's done, check out your Google Drive and your new Google Doc should be in the root (upper-most) folder of your drive. From this point on, you and your colleagues will conduct revisions in the Google Doc.

Download Revisions To Your R Markdown File

Once you and your collaborators are done revising, use the following code to send the revisions back to your local R Markdown file:

```
# to retrieve revisions from google drive
trackdown::download_rmd(
    # leave out file extension (i.e., no ".Rmd" at the end)
    file = "name-of-your-R-Markdown-file",
    gfile = "name-your-file-will-have-on-Google-Drive"
    )
```

This will overwrite your local file, but only the parts that have changed. This is handy for those who use version control: only the altered portions will be recognized as changes. As long as you committed your draft before you uploaded it to Google Drive, you will be able to see the post-Google changes in version control next to the old content.

If for some reason you want to avoid overwriting the R Markdown file, you can let a new file be created instead. The easiest way is to give a new name in the `file =` argument. This will create a new file with the new name you gave, preserving the old file but containing the contents of the Google Doc.

What's Next?

Nothing. That's it. Seriously.
Enjoy!

Appendix

Updating Google Drive File

Say you messed up the first draft, or have written some revisions yourself before your collaborators got a chance to start revising on Google Drive. Or whatever. The point is, you want to throw some more recent changes up to an extant Google Doc, which will add the changes while preserving anything that did not change. Easy, just use the code below. Beware: Google Doc's track changes will not pick this up.

```
# to update an existing google drive file
trackdown::update_rmd(
    # leave out file extension (i.e., no ".Rmd" at the end)
    file = "name-of-your-R-Markdown-file",
    gfile = "name-your-file-will-have-on-Google-Drive"
    )
```

Sending the File To A Particular Location in Google Drive

Simple: all you need to do is add the `path =` argument in `upload_rmd()`:

```

```
to send to google drive
trackdown::upload_rmd(
    # leave out file extension (i.e., no ".Rmd" at the end)
    file = "name-of-your-R-Markdown-file",
    gfile = "name-your-file-will-have-on-Google-Drive",
    path = "blah/blah-blah/name-your-file-will-have-on-Google-Drive"
)
```

## View Contents of Google Drive in R Console

Easy. It even displays in a tidy tibble. Friggin sweet, right? Just use:

```
# view contents of google drive in a tibble
# "n = n" limits the output to the first n files it finds
# "pattern = 'pattern'" returns files with the name specified as the pattern
googledrive::drive_find()
```