Updating R

Neil J. Hatfield

Last Updated: May 03, 2023

The main thing that I want to do in this document is to provide you each with some useful tools to keep your R updated.

1 Release Cycle

R uses a fairly regular update cycle. Major releases happen each year roughly around April 1st. As of writing this document, R version 4.3.0 was released on April 21st, 2023. Patch releases (changing the last number in the version) happen as needed to fix issues once or twice a year between major updates.

A habit I use is that I check updates to R before the start of each semester and then about the half-way point of the semester. I refrain from doing updates to R in the second-half of a semester (unless I have no other option) as a safety procaution (I don't want to break anything with limited time to fix things).

2 Installed Packages

Particularly with major updates (e.g., 4.1.x to 4.2.y or 4.2.y to 4.3.0), you will lose all of the packages that you installed. However, with the following code, you can capture that list so you don't have to try to remember all of your packages.

```
# Create a data frame of all of your installed packages----
packages <- as.data.frame(installed.packages())

# Save to your desktop ----
write.csv(
  x = packages,
  file = "~/Desktop/mypackages.csv",
  row.names = FALSE
)
```

Until you're done updating R, keep track of where you put the `mypackages.csv` file. After you've installed the new version, you'll need to read that file into R.

3 Updating R

Updating R is fairly straightforward: you just need to install the newest version from CRAN website: https://cran.r-project.org/.

3.1 OS Specific Notes

3.1.1 Mac OS (Apple) Users

Make sure that you select the correct build for your computer. If you click on the Apple menu (the Apple icon in the upper left corner of your computer screen) and select About This Mac, you'll get a popup window that should have an icon and your OS version listed. Look for a line that says "Processor" or "Chip".

- If you see “Intel Core” in the Processor/Chip line, select the `R` installer package for “Intel Macs”.
- If you see “Apple M#” in the Processor/Chip line, select the `R` installer package for “Apple silicon Macs”

Once you’ve downloaded the file, continue with the installation process. You may need to (re-) install XQuartz.

3.1.2 Windows Users

For Windows users, you’ll want click the link and select “base” and follow directions from there. (You may need to install the appropriate version of Rtools if you plan to build or compile packages on your local machine; the link is on the same page as “base”.)

Once you’ve downloaded the installer, continue with the installation process. When that finishes, Window users might need to take an extra step to ensure that RStudio uses the newest version of R.

To see if you need to make this change,

1) Launch RStudio
2) Type `version` in the Console and hit the `enter/return` key.
3) In the resulting output, look to see if the major and minor lines match the version of `R` you just installed.

- If they match, you’re done.
- If they don’t’ match, you need to tell RStudio which version to use

Telling RStudio which version of `R` to use can be done in the Global Options setting:

1) Click on the Tools menu of RStudio
2) Click on Global Options...
3) In the General menu, under the Basic tab, the first field should be labeled as R version. Click the Change... button.
4) Click the radio button next to Choose a specific version of R and then select the most up-to-date version in your list. If you do not see the version you just installed. Try quitting RStudio and restarting your computer. If it still does not appear, you might need to download and install `R` again.
5) Click OK through all of the windows. You will need to restart RStudio for the change to take place.

4 Getting Your Packages Back

Once you’ve updated `R`, it’s time to re-install your packages. The following code will help you do just that.

```
# Get the list of base R packages currently installed ----
currentPackages <- as.data.frame(installed.packages())

# Read in your saved list of packages ----
## You might need to change the file path
mypackages <- read.csv(file = "~/Desktop/mypackages.csv", header = TRUE)

# Remove the packages you don't need to install from your list ----
needInstalling <- mypackages[which(!(mypackages$Package %in% currentPackages$Package)), ]

# Install packages ----
install.packages(needInstalling$Package)
```

Be sure to watch out for any error or warning messages that get displayed in your console. You might need to scroll up to see them. These can identify packages in your list that need to be installed through other means (e.g., via the `remotes` or `devtools` package or from another repository such as BioConductor).

When that process finishes, you should be good to go.