Into the Tidyverse

An Introduction to Packages for R

Welcome!

If you're reading this handout, that means you've taken your first steps into the world of R - and the world of data science writ large. Congratulations! It's a fun and exciting field to learn about, and very useful even if you don't intend on launching a new career, startup, or bizarre cryptocurrency. It can seem a bit daunting at first, but with an experiment-oriented mindset and a willingness to try, fail, and try again, you'll get the hang of things in no time.

This handout is intended to accompany the first module of Day 1 of the American University Data Science Professional Development Course. As such, you should already have the following:

- A computer (Mac, Windows, or the Linux distribution of your choice)
- R installed on said computer (Found here: https://cran.r-project.org/)
- RStudio installed on said computer (Found here: https://www.rstudio.com/)
- A willingness to learn! (Found within yourself)

If you're missing any of these components, click the links above (if you're reading this on a computer) or do a quick Google search to pull them up and start the installation. Your PD instructor or student assistant can help you if you have any questions or issues!

Now that we've double-checked our instruments, we're ready to take flight. Let's get started.

Oy, What's All This Then?

During the module you might have heard the instructor mention that R (the programming language) uses 'Libraries', or 'Packages', to accomplish certain tasks. This handout will be covering the installation and usage of one such Package, but it can be generalized to almost any other package that is built for R. First, it might be helpful to go over what a Package is and why it's necessary.

R has a number of powerful, built-in functions that the language can execute natively - that is, without any add-ons or modifications. These functions are shared across any installation of R (of the same version); to continue the language metaphor, they are common verbs fundamental to R. Like any language, though, its users eventually will want to make new words, new verbs, new ways of describing and interacting with the world. Collections of these new verbs - these functions - are referred to as Packages, which are stored in Libraries (though the terms are sometimes used interchangeably). R's built-in functions are powerful, but limited in scope. By installing new packages, you can add to the vocabulary at your disposal, performing old tasks more quickly, or more precisely, or performing entirely new tasks altogether.

There's two steps to using a Package in R. First, you have to actually install the package - finding it online, downloading it, and installing it to your developer environment (in this case, RStudio). The second step is performed whenever you intend to use the package, and that's to load it in - essentially telling your R session that you want to add a new set of words to today's vocabulary. The first step only needs to be done once, but the second needs to be done whenever you start a new R session - R doesn't 'remember' packages that have been loaded in previous sessions.

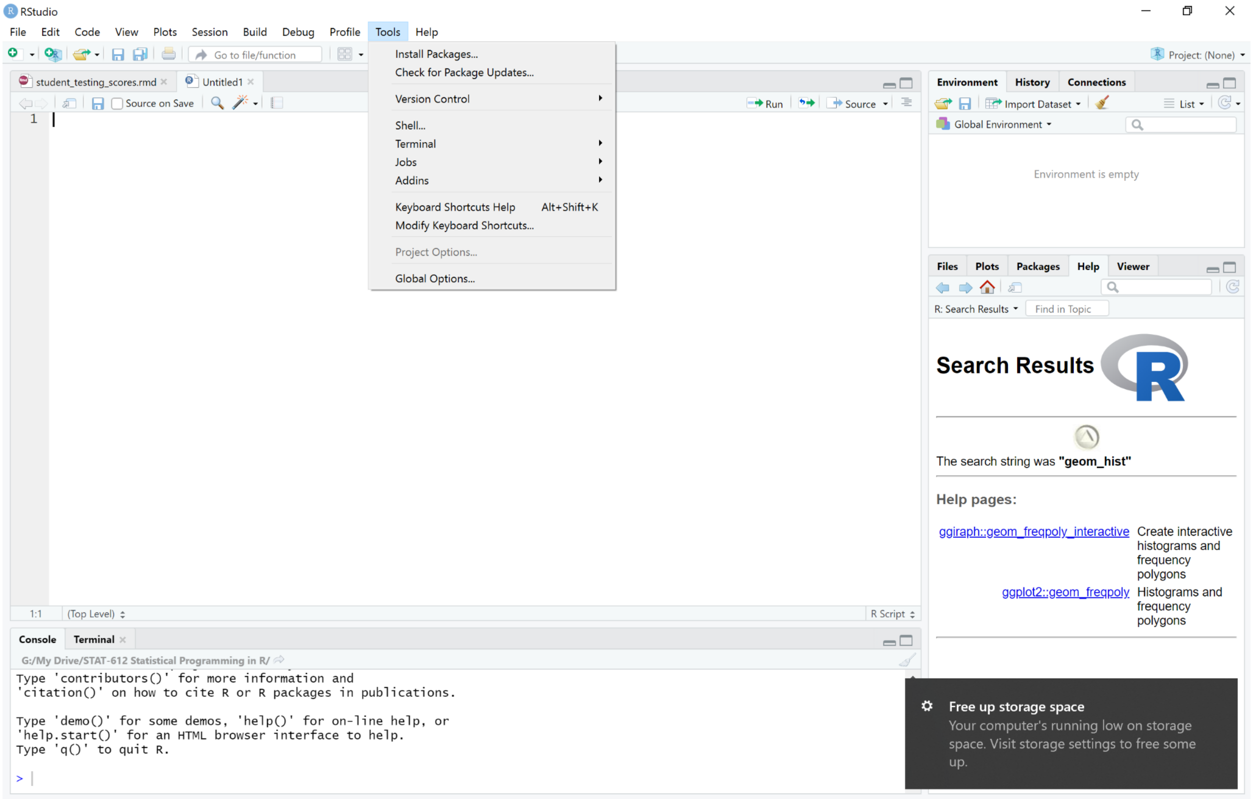
So that's our (very) brief overview of R packages. Let's take a look at the one we'll actually be using.

Tidyverse

In data science applications, we often have to deal with data that is very messy. Data is encoded incorrectly, values are misspelled, datatypes are mismatched compared to what we expect to see, and so on. There are enough difficulties just with the data we use - we shouldn't make more work for ourselves by creating messy code that's difficult to follow, let alone debug. Enter Tidyverse. Less a package and more a *collection* of packages, Tidyverse is unified by a shared design philosophy and attitude towards data. Fundamentally, Tidyverse seeks to simplify writing functional code in R, making it easier for others to understand and easier for the author to diagnose problems. Tidyverse functions are all fairly consistent in their syntax - the way you use one function in one package will be broadly similar to the way you use any other function in any other Tidyverse package - and in many cases improve on the functionality of base R. Included in the Core Tidyverse set of packages are things like *ggplot2*, a set of simple and aesthetically appealing graphing functions; *dplyr*, a framework for the ‘grammar of data manipulation’, allowing you to perform otherwise-tedious data manipulation in just a few seconds; and *stringr*, a comprehensive package for text analysis. Tidyverse and its constituent packages are legitimately some of the most useful things I’ve ever worked with in R, and familiarizing yourself with its functions is a good start to your journey into data science.

Installation

Installing packages in R through RStudio is very easy! If you click on the “Tools” menu in the toolbar, you’ll see a few options pop up…



```
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> 
```

Search Results

The search string was **"geom_hist"**

Help pages:

ggiraph::geom_freqpoly_interactive	Create interactive histograms and frequency polygons
ggplot2::geom_freqpoly	Histograms and frequency polygons

Free up storage space
Your computer's running low on storage space. Visit storage settings to free some up.

Select “Install Packages”. In the dialog box that pops up, double check that your search repository is CRAN, and that “Install Dependencies” is checked. Type in “Tidyverse” the package search bar, click the package name, and click “Install”. Once installation is complete, you’ve done the first of the two steps. Next, when you want to use Tidyverse in an R project, you’ll just have to include “library(tidyverse)” at the start of the project (where the other ‘imports’ go).

And that’s it! Tidyverse is now installed on your computer, and you can import it for any project you start in your working directory. These same steps can be followed for any other package in the CRAN repository.