

1 Outline

2 Course Overview

3 What is R?

4 RStudio: An Integrated Development Environment (IDE) for R

4.1 USDA Computer Setup

4.2 Basic Tips for using R

4.3 Working Directories

4.3.1 Setting the Working Directory

4.3.2 RStudio Projects (.Rproj files)

4.4 Data Management in RStudio

4.5 Getting Help

4.6 Packages

4.7 Writing Scripts

4.8 Saving R Files

5 Rcmdr (R Commander): A Graphical User Interface for R

6 Review

7 Additional Readings

8 References

1 Outline

2 Course Overview

3 What is R?

4 RStudio: An Integrated Development Environment (IDE) for R

5 Rcmdr (R Commander): A Graphical User Interface for R

6 Review

7 Additional Readings

8 References

Chapter 1 Introduction to R and RStudio

Stephen Roecker, Skye Wills, Katey Yoast and Tom D'Avello

2020-02-03

Statistics for Soil Survey



Natural Resources Conservation Service and the National Cooperative Soil Survey

1 Outline

1. Course Overview
 1. Review Course Objectives
 2. Why is this training needed?
 3. Why is course organized this way?
2. What is R?
 1. Why should I use R?
 2. What can R do?
3. How do I get started?
 1. RStudio interface
 2. How to import, export, and view files
 3. How to navigate the Help tab
 4. What are packages?
 5. How to save files

2 Course Overview

2.1 Course Objectives

- Develop solutions to investigate soil survey correlation problems and update activities.
- Evaluate investigations for interpretive results and determine how to proceed.
- Create a continuous surface from point data.
- Summarize data for populations in NASIS.
- Help to pursue the question “why”

2.2 Why is this training needed?

- Long standing goal of the Soil Science Division to have a course in statistics (Mausbach, 2003)
(https://www.nrcs.usda.gov/Internet/FSE_DOCUMENTS/nrcs142p2_051833.pdf)
- Opportunities to learn these techniques are limited, especially at the undergraduate level (Hennemann and Rossiter, 2004)
(<http://www.css.cornell.edu/faculty/dgr2/Docs/ChaAm/ChaAmKeynoteHennemann.pdf>)
- Consistent methodology (data analysis, data population, sampling design, etc.)
- There is continually a greater need to use these techniques:
 - Mapping of lands at high production rates (MacMillan et al., 2007
(<https://www.sciencedirect.com/science/article/pii/S0016706107001152>); Kempen et al., 2012
(<https://access.onlinelibrary.wiley.com/doi/10.2136/sssaj2011.0424>); Brevik et al., 2016
(<https://www.sciencedirect.com/science/article/pii/S034181621630220X>))
 - Ecological Sites (Maynard et al., 2019)
(dl.sciencesocieties.org/publications/sssaj/abstracts/83/3/666)
 - Soil survey refinement (disaggregation) (Chaney et al., 2016
(<https://www.sciencedirect.com/science/article/pii/S0016706116301434>); Ramcharan et al., 2017
(<https://access.onlinelibrary.wiley.com/doi/10.2136/sssaj2017.04.0122>))

2.3 Why is course organized this way?

- Our best judgement for assembling into **24** hours what could be **6** University level courses
- Mixture of slides and script enabled web pages is new for NRCS
- The web content is a long-term investment and should serve as a permanent reference
- Feel free to provide guidance for improving the class for future offerings

3 What is R?

R is a free, open-source software and programming language developed in 1995 at the University of Auckland as an environment for statistical computing and graphics (Ihaka and Gentleman, 1996) (<https://www.stat.auckland.ac.nz/~ihaka/downloads/R-paper.pdf>). Since then R has become one of the dominant software environments for data analysis and is used by a variety of scientific disciplines, including soil science, ecology, and geoinformatics (Envirometrics CRAN Task View (<https://cran.r-project.org/web/views/Environmetrics.html>); Spatial CRAN Task View (<https://cran.r-project.org/web/views/Spatial.html>)). R is particularly

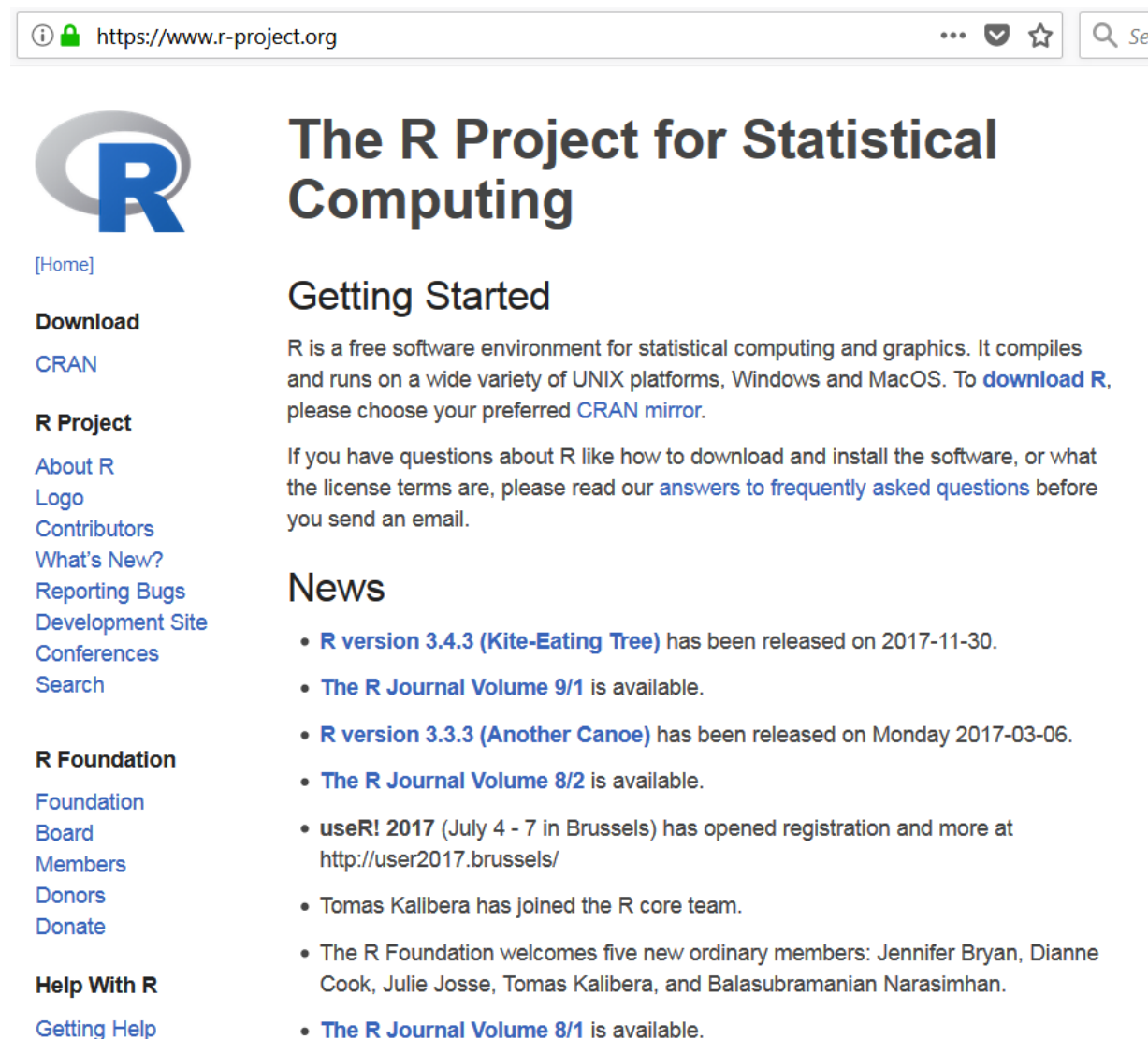
popular for its graphical capabilities, but it is also prized for its GIS capabilities which make it relatively easy to generate raster-based models. More recently, R has also gained several packages which are designed specifically for analyzing soil data.

1. a software environment:

- statistics
- graphics
- programming
- calculator
- GIS
- etc...

2. a language to explore, summarize, and model data

- functions = verbs
- objects = nouns



The screenshot shows the homepage of the R Project for Statistical Computing. The browser address bar displays 'https://www.r-project.org'. The page features the R logo on the left, followed by the title 'The R Project for Statistical Computing'. Below the title is a 'Getting Started' section with a brief description of R and links to download R and CRAN mirrors. A 'News' section lists recent releases and events. A sidebar on the left contains links for 'Download', 'R Project', 'R Foundation', and 'Help With R'.

[\[Home\]](#)

Download

[CRAN](#)

R Project

[About R](#)
[Logo](#)
[Contributors](#)
[What's New?](#)
[Reporting Bugs](#)
[Development Site](#)
[Conferences](#)
[Search](#)

R Foundation

[Foundation](#)
[Board](#)
[Members](#)
[Donors](#)
[Donate](#)

Help With R

[Getting Help](#)

The R Project for Statistical Computing

Getting Started

R is a free software environment for statistical computing and graphics. It compiles and runs on a wide variety of UNIX platforms, Windows and MacOS. To [download R](#), please choose your preferred [CRAN mirror](#).

If you have questions about R like how to download and install the software, or what the license terms are, please read our [answers to frequently asked questions](#) before you send an email.

News

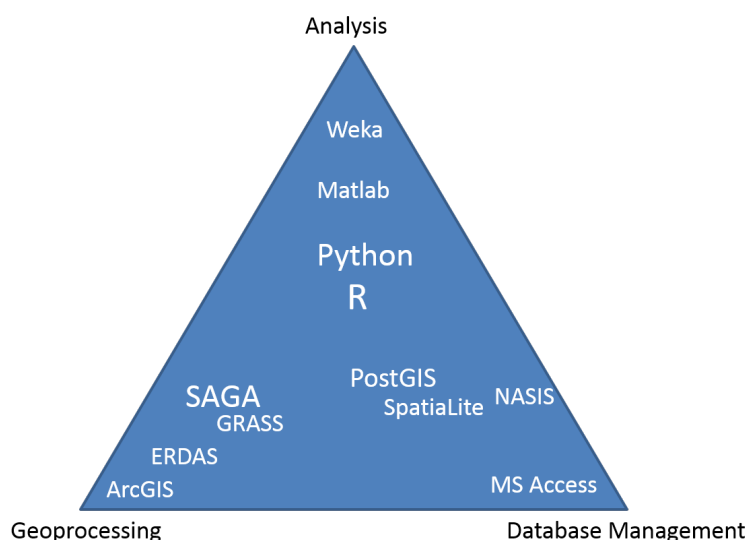
- [R version 3.4.3 \(Kite-Eating Tree\)](#) has been released on 2017-11-30.
- [The R Journal Volume 9/1](#) is available.
- [R version 3.3.3 \(Another Canoe\)](#) has been released on Monday 2017-03-06.
- [The R Journal Volume 8/2](#) is available.
- [useR! 2017](#) (July 4 - 7 in Brussels) has opened registration and more at <http://user2017.brussels/>
- Tomas Kalibera has joined the R core team.
- The R Foundation welcomes five new ordinary members: Jennifer Bryan, Dianne Cook, Julie Josse, Tomas Kalibera, and Balasubramanian Narasimhan.
- [The R Journal Volume 8/1](#) is available.

3.1 Why Should I Learn R?

While the vast majority of people use Microsoft Excel for data analysis, R offers numerous advantages, such as:

1. Cost. R is free! (“Free as in free speech, not free beer.”)
(<https://www.gnu.org/philosophy/free-sw.html>)
2. Reproducible Research (<http://christophergandrud.github.io/RepResR-RStudio/>)
(*self-documenting, repeatable*)
 - repeatable:
 - code + output in a single document (*‘I want the right answer, not a quick answer’ - Paul Finnell*)
 - easier the next time (humorous example
(https://www.youtube.com/watch?time_continue=1&v=s3JldKoA0zw))
 - numerous Excel horror stories of scientific studies gone wrong exist
(TED Talk (<https://www.youtube.com/watch?v=dXKbkpilQME>))
 - scalable: applicable to small or large problems
3. R in a Community
 - Numerous Discipline Specific R Groups (<https://cran.r-project.org/web/views/>)
 - Numerous Local R User Groups (including R-Ladies Groups)
(<https://jumpingrivers.github.io/meetingsR/r-user-groups.html#north-america>)
 - Stack Overflow (<https://stackoverflow.com/>)
4. Learning Resources (*quantity and quality*)
 - R books (<https://www.r-project.org/doc/bib/R-books.html>)
 - (Free Online) R Books (<https://bookdown.org/>)
5. R is ‘becoming’ the new norm (paradigm shift?) “If we don’t accept these challenges, other who are less qualified will; and soil scientists will be displaced by apathy.”
(Arnold and Wilding, 1992)
(<https://dl.sciencesocieties.org/publications/books/abstracts/sssaspecialpubl/spatialvariabil/1>)

While some people find the use of a commandline environment daunting, it is becoming a necessary skill for scientists as the volume and variety of data has grown. Thus scripting or programming has become a third language for many scientists, in addition to their native language and discipline specific terminology. Other popular programming languages include: SQL (i.e. NASIS), Python (i.e. ArcGIS), and JavaScript.



ODBC and GDAL link R to nearly all possible formats/interfaces

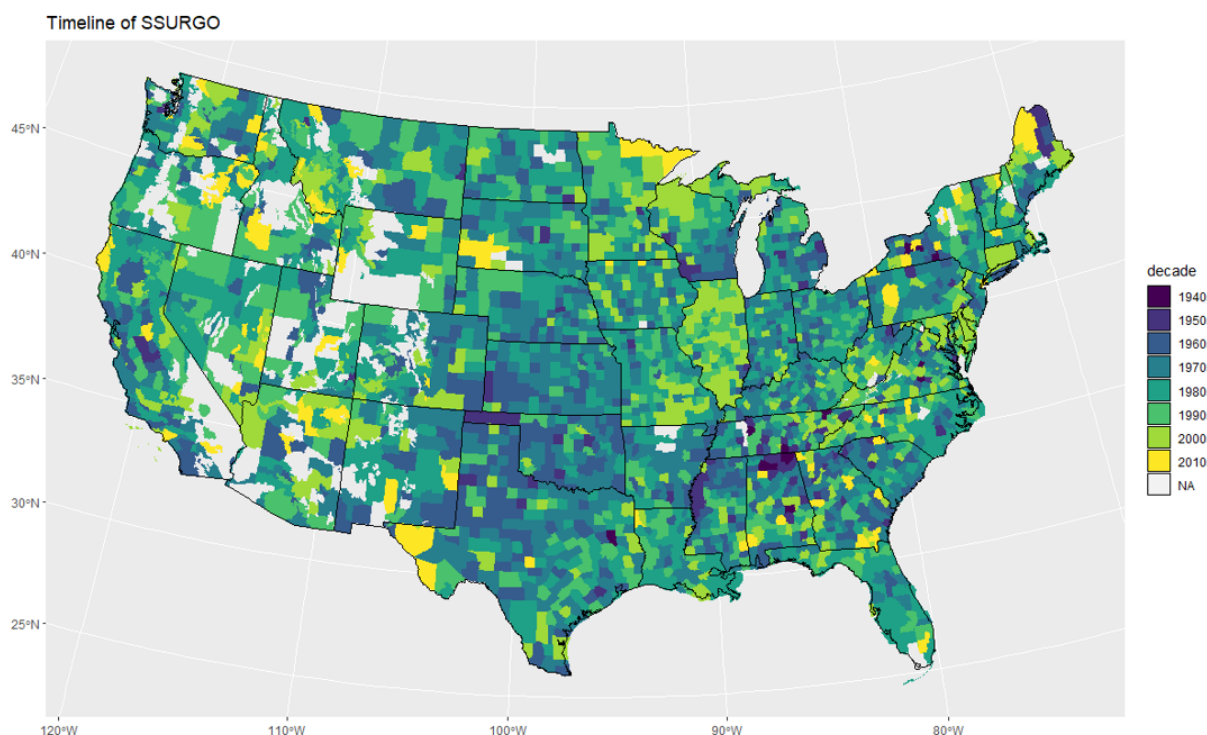
3.2 What can R do?

3.2.1 Packages

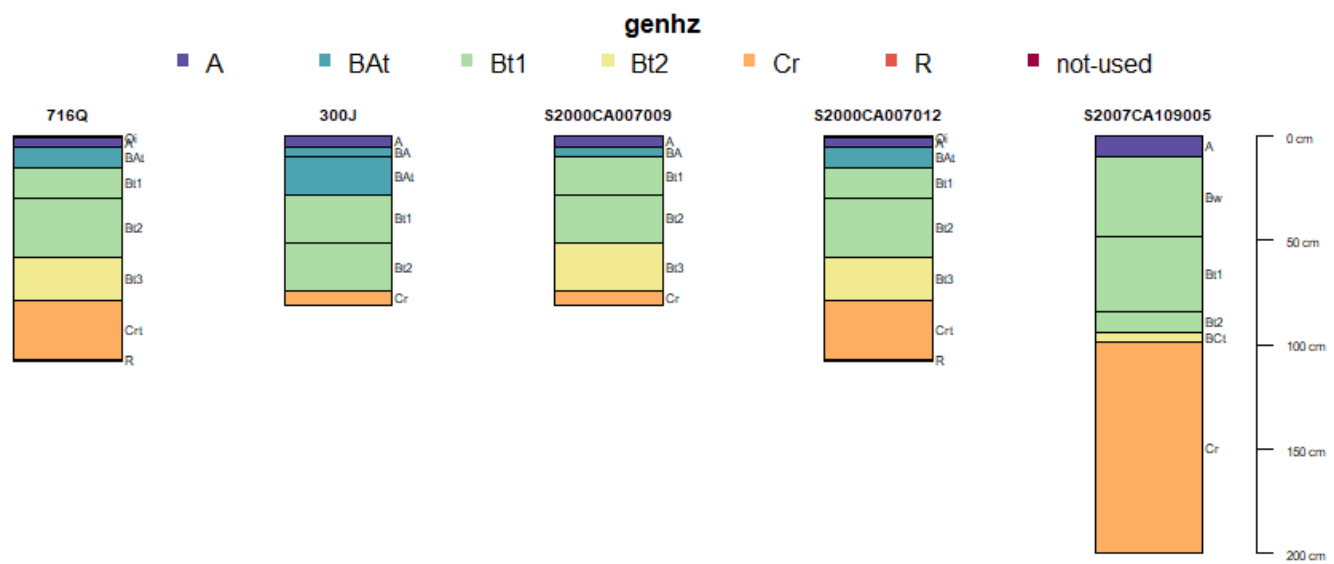
- Base R (*functionality is extended through packages*)
 - basic summaries of quantitative or qualitative data
 - data exploration via graphics
 - GIS (<https://cran.r-project.org/web/views/Spatial.html>) data processing and analysis
- Soil Science R Packages
 - aqp (<https://github.com/ncss-tech/aqp>) - visualization, aggregation, classification
 - soilDB (<https://github.com/ncss-tech/soilDB>) - access to commonly used soil databases
 - soilReports (<https://github.com/ncss-tech/soilReports>) - handful of report templates
 - soiltexture (<http://soiltexture.r-forge.r-project.org/>) - textural triangles
- Ecology (<https://cran.r-project.org/web/views/Environmetrics.html>) R packages
 - vegan (<http://vegan.r-forge.r-project.org/>) - ordination, diversity analysis, etc...
 - dismo (<http://rspatial.org/sdm/>) - species distribution modeling

3.2.2 Soil Science Applications

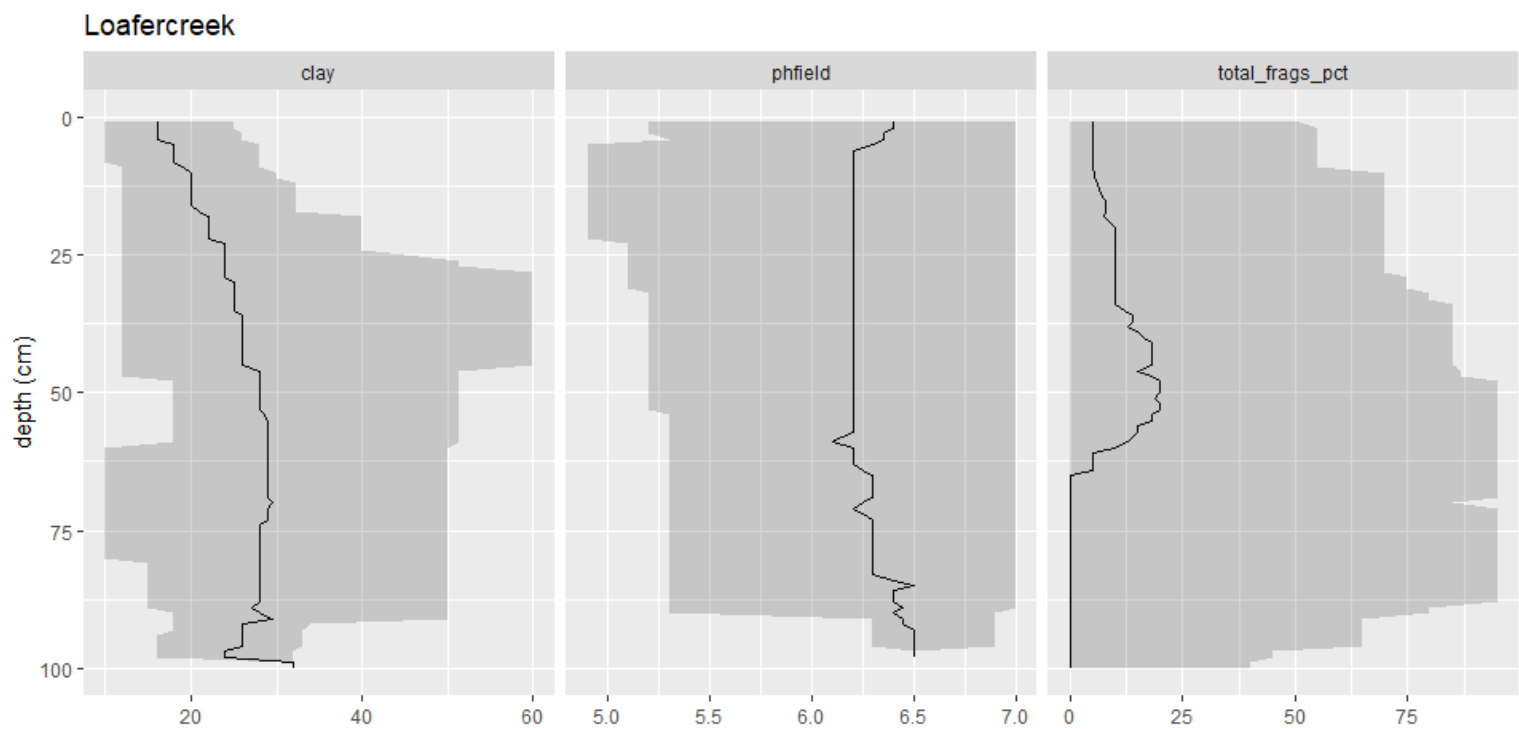
3.2.2.1 Create Maps



3.2.2.2 Draw Soil Profiles



3.2.2.3 Draw Depth Plots



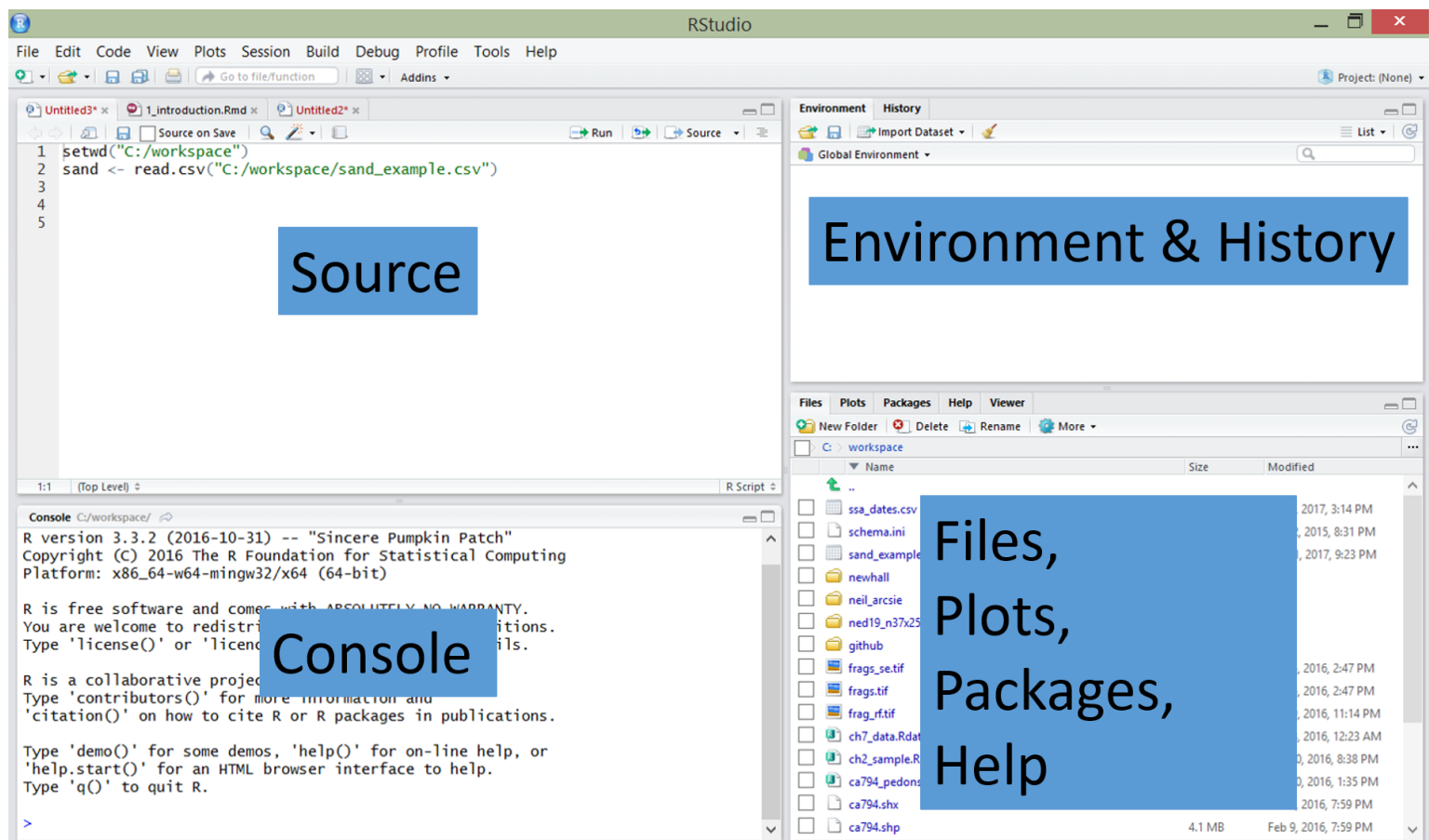
3.2.2.4 Estimate the Range in Characteristics (RIC)

variable	genhz	pct10	median	pct90
clay	A	12	16	22
clay	BA _t	16	20	25
clay	Bt ₁	18	24	32
clay	Bt ₂	23	30	44
clay	Cr	15	15	15
phfield	A	6	6	7

phfield	BAt	5	6	6
phfield	Bt1	5	6	7

4 RStudio: An Integrated Development Environment (IDE) for R

RStudio is an integrated development environment (IDE) that allows you to interact with R more readily. RStudio is similar to the standard RGui, but is considerably more user friendly. It has more drop-down menus, windows with multiple tabs, and many customization options. The first time you open RStudio, you will see three windows. A forth window is hidden by default, but can be opened by clicking the **File** drop-down menu, then **New File**, and then **R Script**.. Detailed information on using RStudio can be found at at RStudio's Website (<https://support.rstudio.com/hc/en-us/sections/200107586-Using-RStudio>).



RStudio Windows /

Tabs	Location	Description
Console Window	lower-left	location where commands are entered and the output is printed
Source Tabs	upper-left	built-in text editor

Environment Tab	upper-right	interactive list of loaded R objects
History Tab	upper-right	list of key strokes entered into the Console
Files Tab	lower-right	file explorer to navigate C drive folders
Plots Tab	lower-right	output location for plots
Packages Tab	lower-right	list of installed packages
Help Tab	lower-right	output location for help commands and help search window
Viewer Tab	lower-right	advanced tab for local web content

4.1 USDA Computer Setup

R and RStudio have been installed on all USDA computers which have NASIS installed. R and RStudio are typically updated and CCE-approved once a year. The versions on USDA machines may be one to three releases behind the latest version available for public download. Having an outdated version of R rarely creates a problem, although warnings may appear.

4.2 Basic Tips for using R

- R is command-line driven. It requires you to type or copy-and-paste commands after a command prompt (`>`) that appears when you open R. After typing a command in the R console and pressing **Enter** on your keyboard, the command will run. If your command is not complete, R issues a continuation prompt (signified by a plus sign: `+`). Alternatively you can write a script in the script window, and select a command, and click the **Run** button.
- R is case sensitive. Make sure your spelling and capitalization are correct.
- Commands in R are also called functions. The basic format of a function in R is:
`object <- function.name(argument_1 = data, argument_2 = TRUE) .`
- The up arrow (`^`) on your keyboard can be used to bring up previous commands that you've typed in the R console.
- The `$` symbol is used to select a particular column within the table (e.g., `table$column`).
- Any text that you do not want R to act on (such as comments, notes, or instructions) needs to be preceded by the `#` symbol (a.k.a. hash-tag, comment, pound, or number symbol). R ignores the remainder of the script line following `#`.

For example:

```
plot(x, y) # This text will not affect the plot function because of the comment
```

4.2.1 Brief Example

```
# Addition  
1 + 1
```

[1] 2

```
# Multiplication  
10 * 10
```

[1] 100

```
# Compute Logarithm  
log10(100)
```

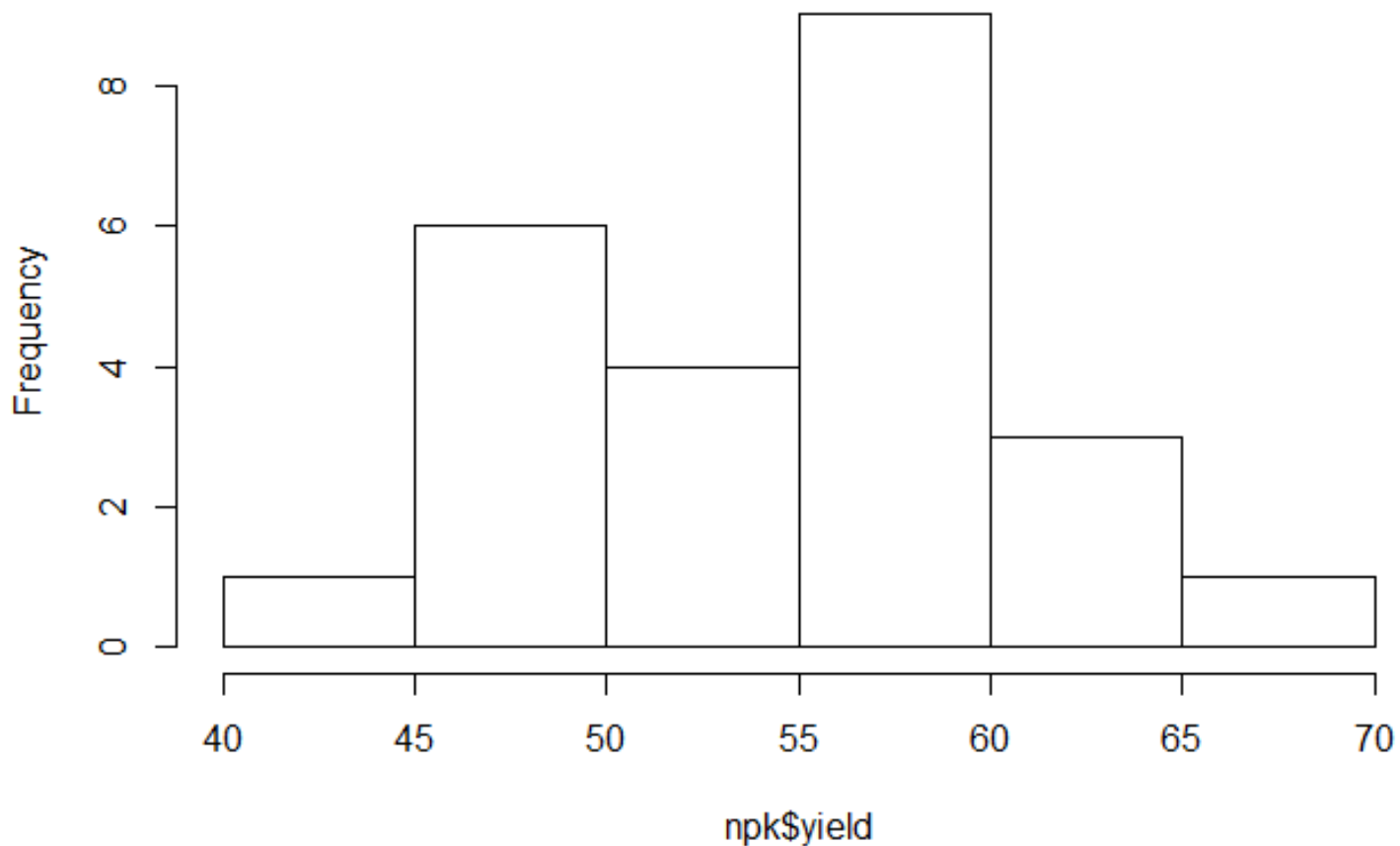
[1] 2

```
# Print Text  
"Hello World"
```

[1] "Hello World"

```
# Plot Histogram  
hist(npk$yield)
```

Histogram of npk\$yield



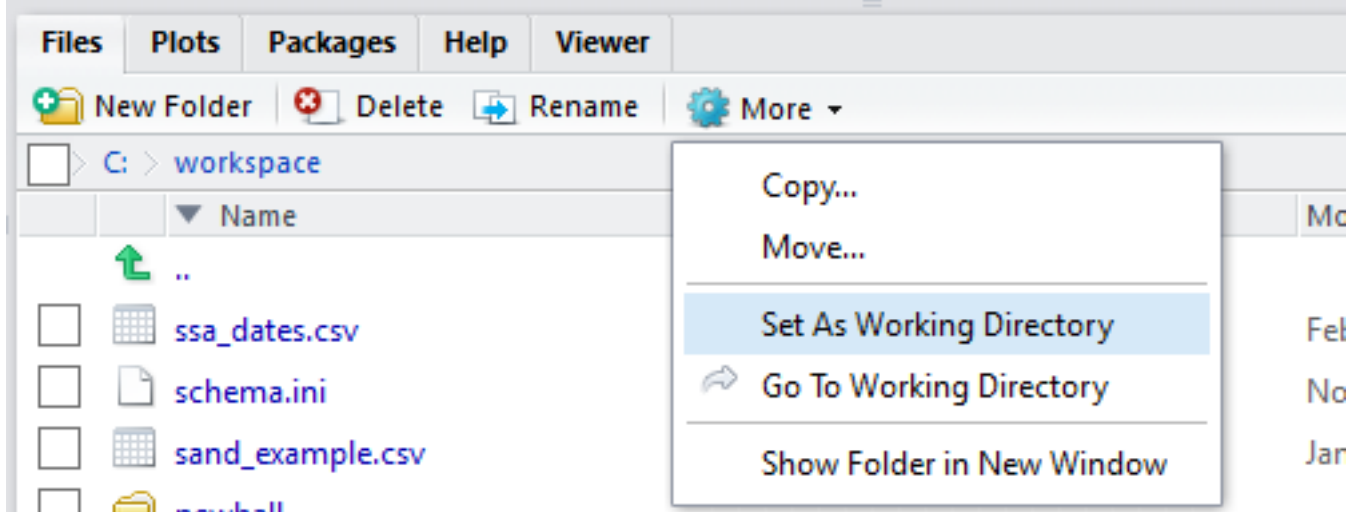
```
# Assignment  
test <- 1  
  
# or  
  
test = 1
```

4.3 Working Directories

4.3.1 Setting the Working Directory

Before you begin working in R, you should set your working directory (a folder to hold all of your project files); for example, "C:\workspace2\...".

To change the working directory in RStudio, select main menu **Session >> Set Working Directory >> ...**. Or, from the "Files" tab click **More >> Set As Working Directory** to use the *current location of the "Files" tab* as your working directory.



Setting the working directory via the menus is the same as doing it in the Console with the `setwd()` command:

```
setwd("C:/workspace2")
```

This directory is where all your input data should be stored and also is the default location for plot files and other output.

Essentially, you want to have the inputs for your code to be found in the working directory so that you can refer to them using relative file paths (<https://excelquick.com/r-programming/importing-data-absolute-and-relative-file-paths-in-r/>). Relative file paths make it easier if you move the folder containing your script(s) around. Or, if you share it with someone else, they will have little issue getting your code to work on their own file system.

NOTE: Beware when specifying any file paths that **R** uses forward slashes `/` instead of back slashes `\`. Back slashes are reserved for use as an escape character (<https://campus.datacamp.com/courses/string-manipulation-in-r-with-stringr/string-basics?ex=4>).

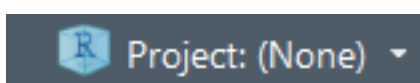
To check the file path of the current working directory (which should now be `"C:\workspace2"`), type:

```
getwd()
```

4.3.2 RStudio Projects (.Rproj files)

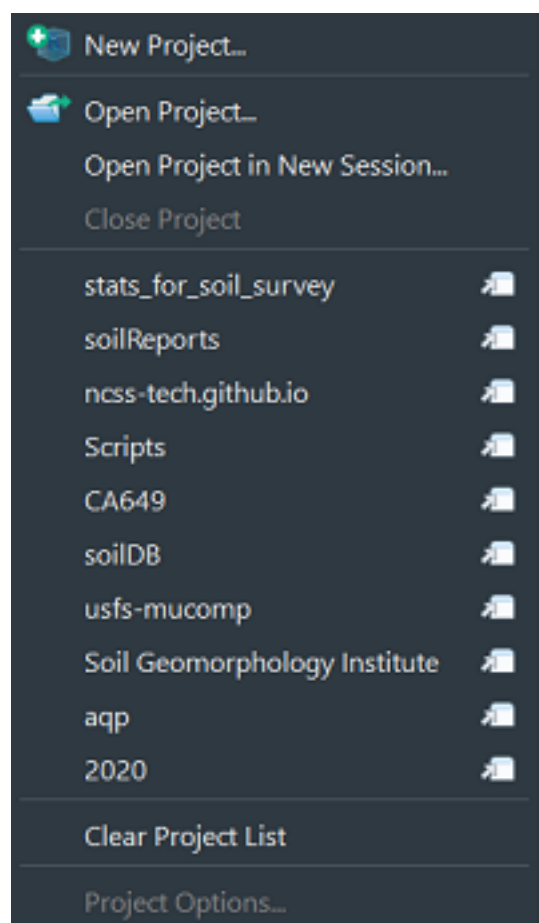
You can also manage your working directory using RStudio Projects. An RStudio Project file (*.Rproj*) is analogous to, for example, a *.mxd* file for ArcMap. It contains information about the specific settings you may have set for a "project".

You open or create projects using the drop down menu in the top right-hand corner of the RStudio window (*shown below*)



RStudio Project Menu

Here is what a typical Project drop-down menu looks like:



RStudio Project Menu (expanded)

- You can create new projects from existing or new directories with “New Project...”.
- When you click “Open Project...”, your working directory is *automatically set to the .Rproj file’s location* – this is *extremely* handy
- Any projects you have created/used recently will show up in the “Project List”

Keeping working directories simple and specific to a single “project” is a good practice that helps keeps your code and input data organized, and helps you come back to a project after some time away from it.

4.4 Data Management in RStudio

4.4.1 Importing Data

After your working directory is set, you can import data from .csv, .txt, etc. One basic command for importing data into R is `read.csv()`. The command is followed by the file name and then some optional instructions for how to read the file.

First, create an example file by **copying** the contents below. **Paste** the content into Notepad and **save** the file as sand_example.csv in your C:\workspace2 folder.

```
location,landuse,horizon,depth,sand
city,crop,A,14,19
city,crop,B,25,21
```

city,pasture,A,10,23
city,pasture,B,27,34
city,range,A,15,22
city,range,B,23,23
farm,crop,A,12,31
farm,crop,B,31,35
farm,pasture,A,17,30
farm,pasture,B,26,36
farm,range,A,15,25
farm,range,B,24,29
west,crop,A,13,27
west,crop,B,29,25
west,pasture,A,11,21
west,pasture,B,31,26
west,range,A,14,23
west,range,B,24,24

This dataset can either be imported into R using the **Import Dataset** button from the Environment tab, or by typing the following command into the R console:

```
sand <- read.csv("C:/workspace2/sand_example.csv")  
  
# if your workspace was already set you could simply use the filename,  
like so  
  
sand <- read.csv("sand_example.csv")
```

4.4.1.1 Note on Microsoft Excel files

R can import Excel files, but generally speaking it is a bad idea to use Excel. Excel has a dangerous default which automatically converts data with common notations to their standard format without warning or notice. For example, the character “11-JUN” entered into a cell automatically becomes the date 6/11/2017, even though the data is still displayed as 11-JUN. The only way to avoid this default behavior is to manually import your data into Excel via the **Data Tab>Get External Data Ribbon**, and manually set the data type of all your columns to text. Failure to do so has resulted in numerous retracted research articles (Washington Post Article (https://www.washingtonpost.com/news/work/wp/2016/08/26/an-alarming-number-of-scientific-papers-contain-excel-errors/?utm_term=.9352a35dca6f)).

4.4.2 Exporting Data

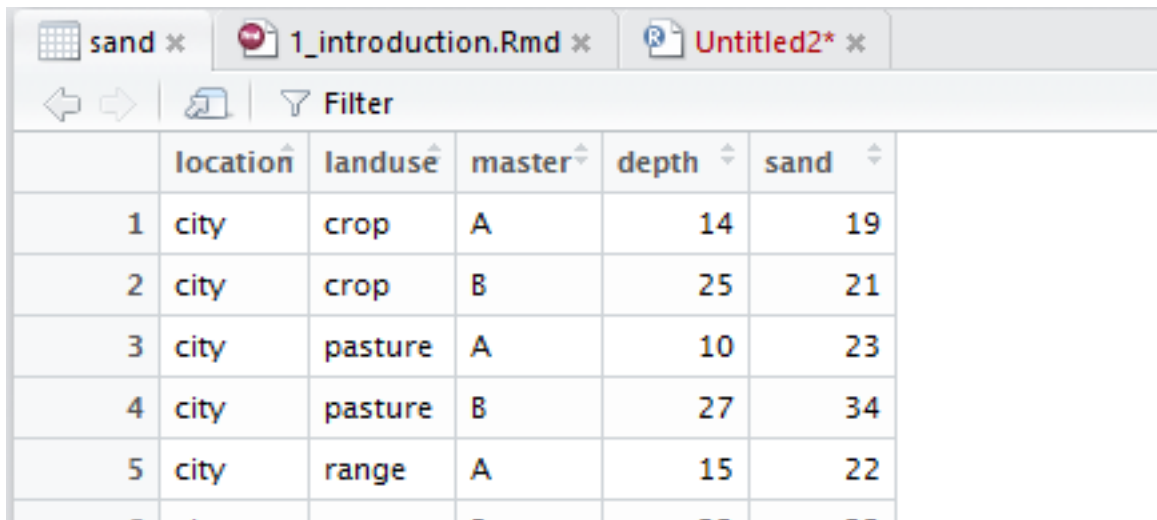
To export data from R, use the command `write.csv()` function. Since we have already set our working directory, R automatically saves our file into the working directory.

```
write.csv(sand, file = "sand_example2.csv")
```

or use the write.table() function to export other text file types

4.4.3 Viewing and Removing Data

Once the file is imported, it is imperative that you check to ensure that R correctly imported your data. Make sure numerical data are correctly imported as numerical, that your column headings are preserved, etc. To view data simply **click** on the sand dataset listed in the Environment tab. This will open up a separate window that displays a spreadsheet like view.



	location	landuse	master	depth	sand
1	city	crop	A	14	19
2	city	crop	B	25	21
3	city	pasture	A	10	23
4	city	pasture	B	27	34
5	city	range	A	15	22

Additionally you can use the following functions to view your data in R.

Function	Description
<code>print()</code>	prints the entire object (avoid with large tables)
<code>head()</code>	prints the first 6 lines of your data
<code>str()</code>	shows the data structure of an R object
<code>names()</code>	lists the column names (i.e., headers) of your data
<code>ls()</code>	lists all the R objects in your workspace directory

Try entering the following commands to view the sand dataset in R:

```
str(sand)
```

```
names(sand)
```

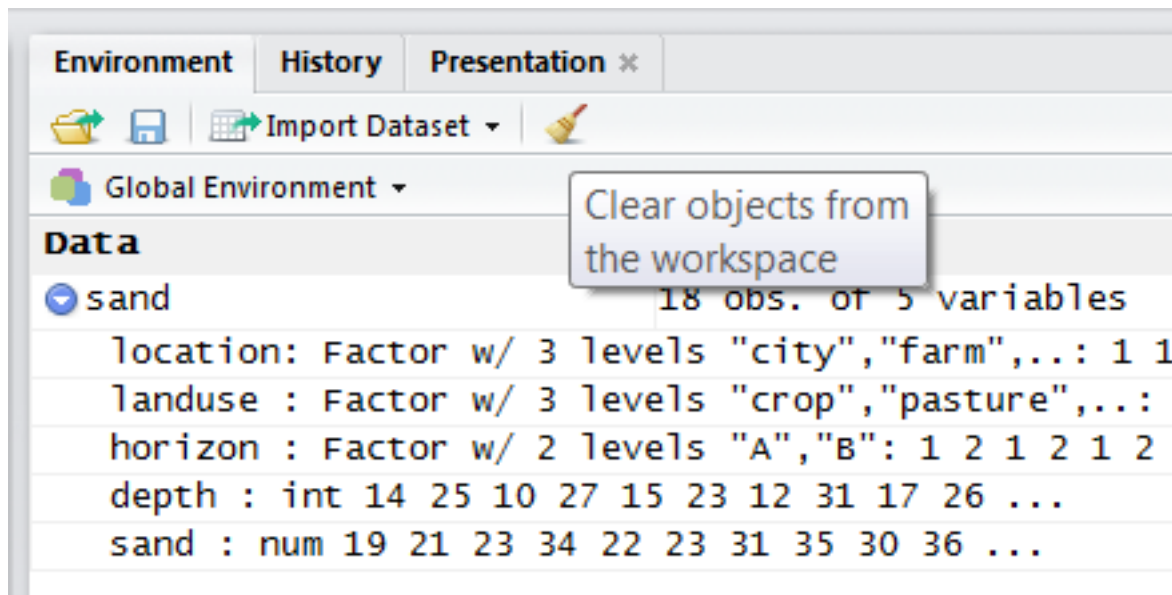
```
head(sand)
```

```
ls()
```

A data object is anything you've created or imported and assigned a name to in R. The Environment tab allows you to see what data objects are in your R session and expand their structure. Right now sand should be the only data object listed. If you wanted to delete all data objects from your R session, you could **click the broom icon** from the Environments tab. Otherwise you could type:

```
# Remove all R objects
rm(list = ls(all = TRUE))

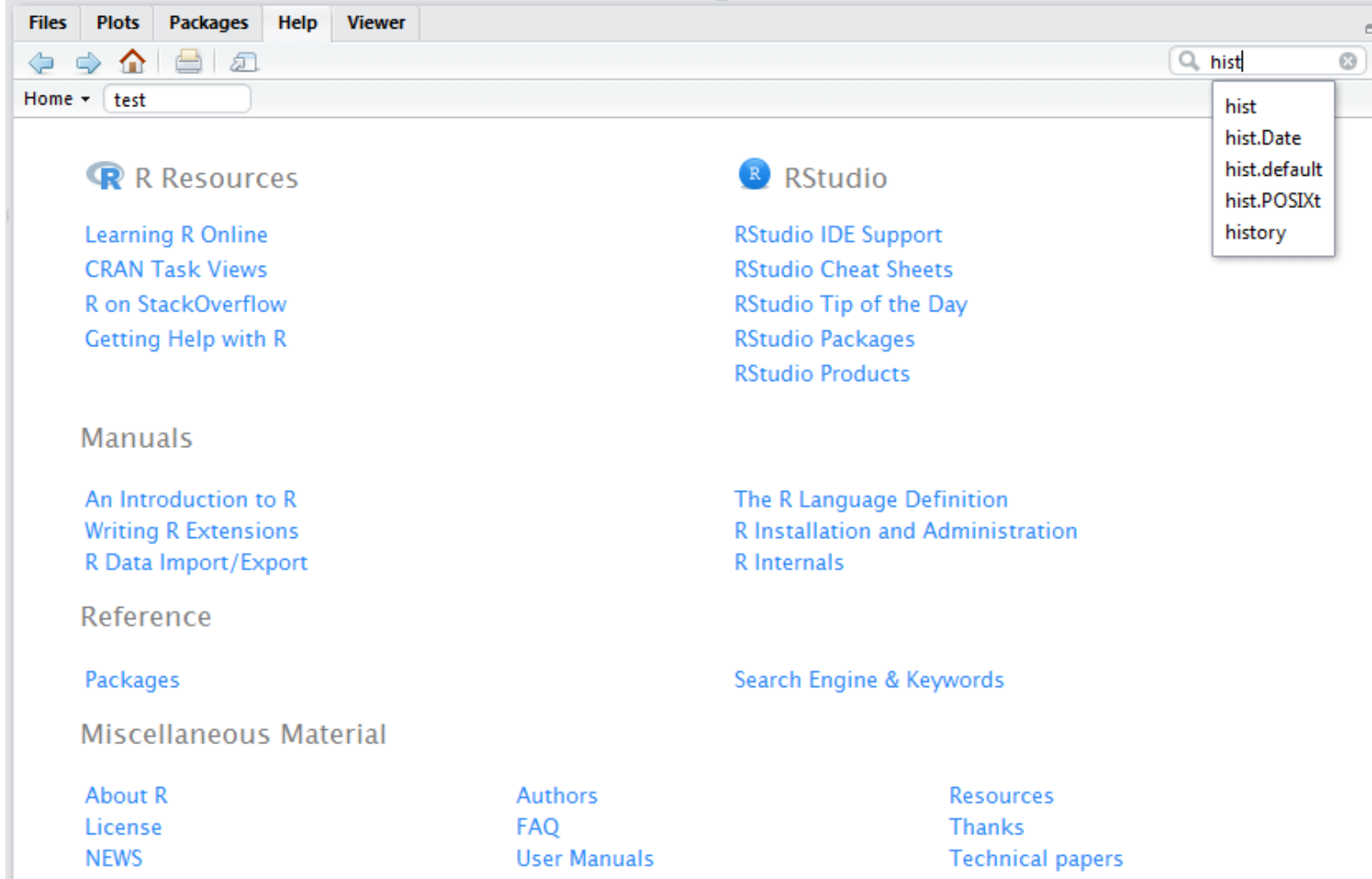
# Remove individual objects
rm(sand)
```



4.5 Getting Help

R has extensive documentation (<https://cran.r-project.org/manuals.html>), numerous mailing lists (<https://www.r-project.org/mail.html>), and countless books (<https://www.r-project.org/doc/bib/R-books.html>) (many of which are free and listed at end of each chapter for this course). The built in help files are sometimes cryptic, and the online replies can be terse, but if you seek help you shall find it. To learn more about the function you are using and the options and arguments available, learn to help yourself by taking advantage of some of the following help functions in RStudio:

1. Use the Help tab in the lower-right Window to search commands (such as hist) or topics (such as histogram).



2. Type `help(read.csv)` or `?read.csv` in the Console window to bring up a help page. Results will appear in the Help tab in the lower right-hand window. Certain functions may require quotations, such as `help("+")`.

```
# Help file for a function
help(read.csv) # or ?read.csv

# Help files for a package
help(package = "soiltexture")
```

4.5.1 Exercise: Examine the `read.csv()` function

Read the help file for the `read.csv()` function. How would you disable the first row in the sand dataset from being interpreted as a header? Report your answer to your mentor.

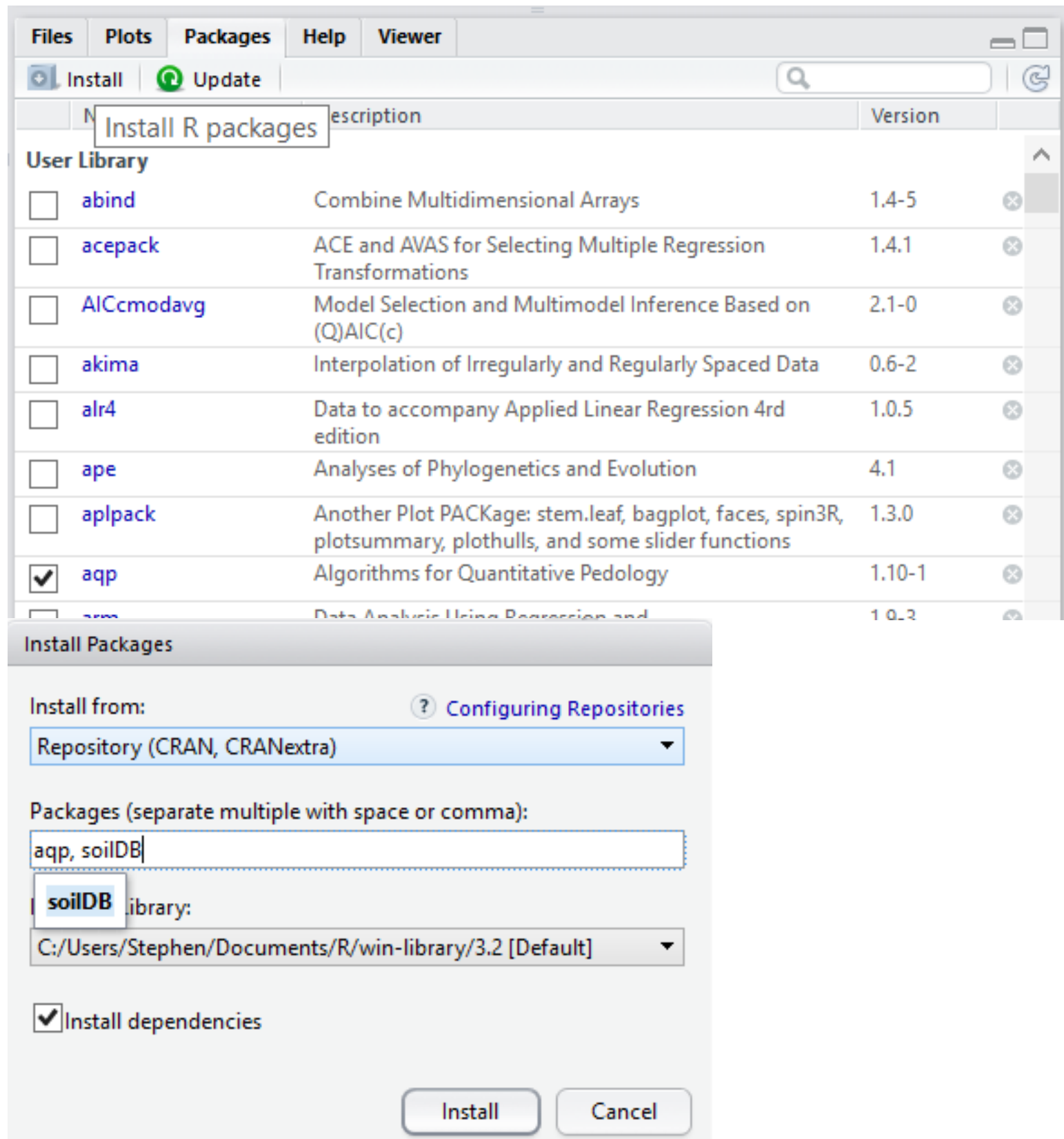
4.6 Packages

Packages are collections of additional functions that can be loaded on demand. They commonly include example data that can be used to demonstrate those functions. Although R comes with many common statistical functions and models, most of our work requires additional packages.

4.6.1 Installing Packages

To use a package, you must first install it and then load it. These steps can be done at the command line or using the Packages Tab. Examples of both approaches are provided below. R packages only need to be installed once (until R is upgraded or re-installed). Every time you start a new R session, however, you need to load every package that you intend to use in that session.

Within the **Packages** tab you will see a list of all the packages currently installed on your computer, and 2 buttons labeled either “Install” or “Update”. To install a new package simply select the **Install** button. You can enter install one or more than one packages at a time by simply separating them with a comma.



To find out what packages are installed on your computer, use the following commands:

```
library()
```

```
# or
```

```
installed.packages()
```

One extremely useful package for soil scientists is the `soiltexture` package, which as you would expect allows you to plot soil textural triangles. The following command shows how to install this package if you do not have currently have it downloaded:

```
# CRAN (static version)
```

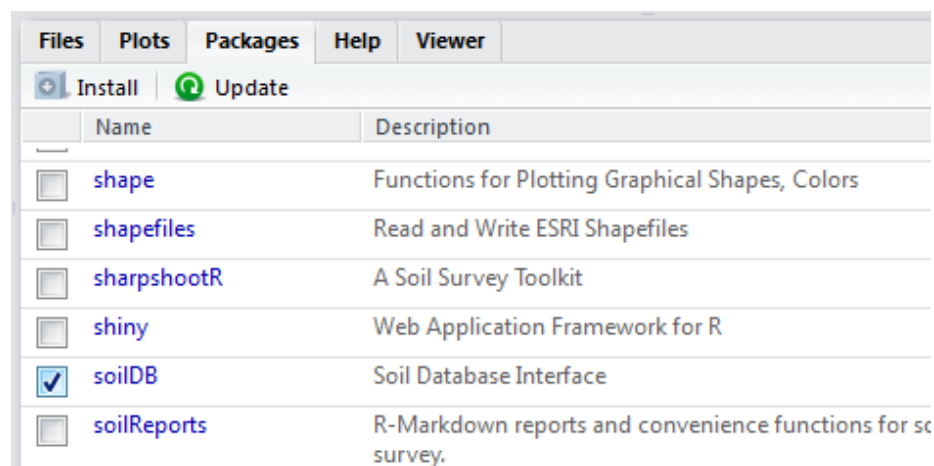
```
installed.packages(c("aqp", "soilDB", "soilReports", "soiltexture"))
```

```
# GitHub (development version)
```

```
devtools::install_github("ncss-tech/soilDB", dependencies = FALSE, upgrade_dependencies = FALSE, build = FALSE)
```

4.6.2 Loading Packages

Once a package is installed, it must be loaded into the R session to be used.



```
library(soiltexture)
```

You can also load packages using the **Packages Tab**, by **checking** the box next to the package name. Documentation about the `soiltexture` package is available from the help functions in R.

```
help(package = "soiltexture")
```

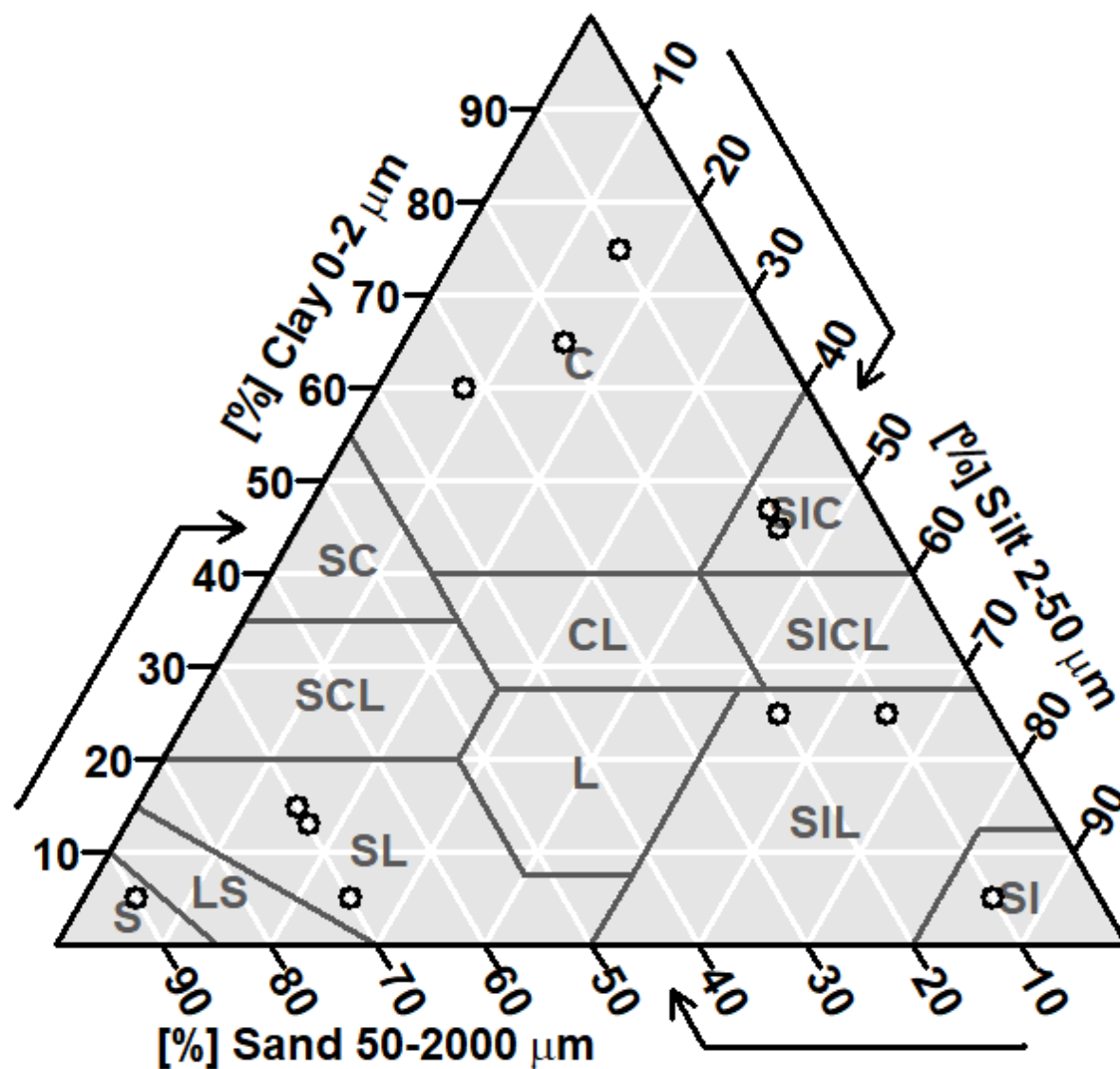
This help command sends you to a webpage. Scroll down and select the link “TT.plot”. This link brings up a webpage that has instructions on how to use the `TT.plot()` function in R.

The basic usage of the `TT.plot()` function is: `TT.plot(class.sys, tri.data)`. The “class.sym” argument specifies a character string naming the textural classification system.

```
# Copied from soiltexture vignette
# Create a dummy data frame of soil textures:
example <- data.frame(
  CLAY = c(05,60,15,05,25,05,25,45,65,75,13,47),
  SILT = c(05,08,15,25,55,85,65,45,15,15,17,43),
  SAND = c(90,32,70,70,20,10,10,10,20,10,70,10),
  OC = c(20,14,15,05,12,15,07,21,25,30,05,28)
)

TT.plot(class.sys = "USDA-NCSS.TT", tri.data = example)
```

Texture triangle: USDA (NCSS)



For more examples see the the soiltexture vignette (http://cran.r-project.org/web/packages/soiltexture/vignettes/soiltexture_vignette.pdf). Vignettes are a short tutorials that provide detailed examples and descriptions of packages. Unfortunately not all packages come with a vignette.

4.7 Writing Scripts

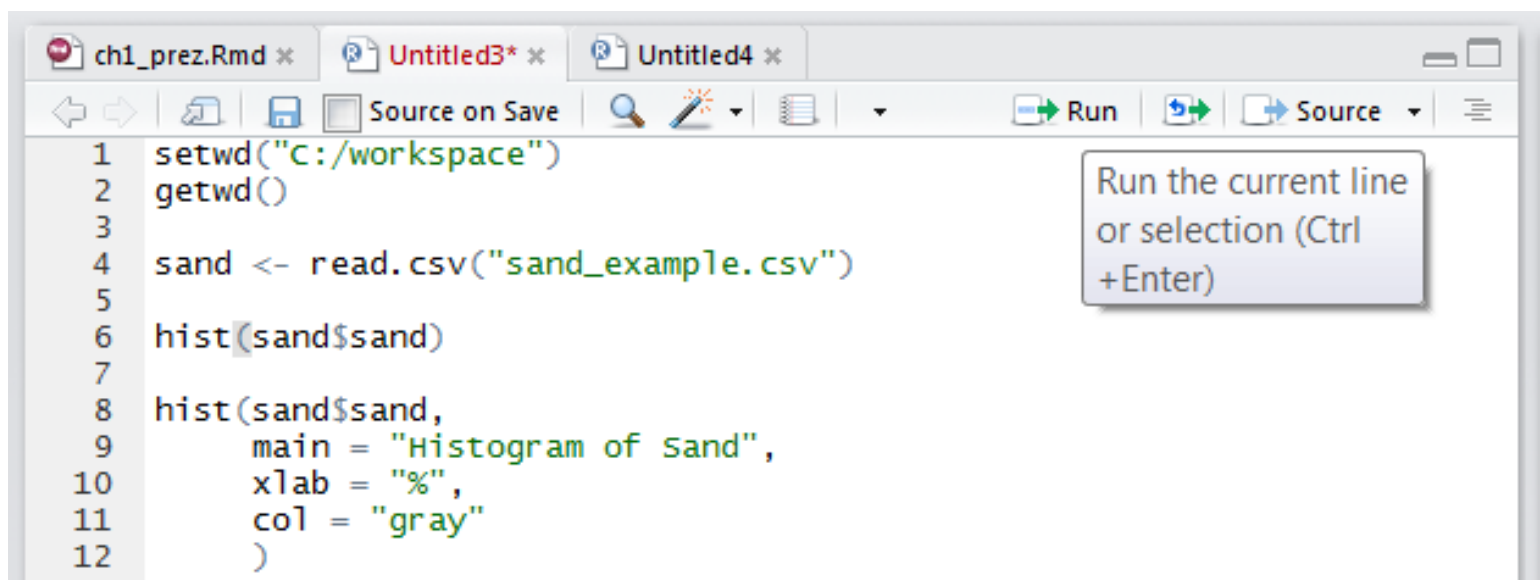
RStudio's Source Tabs serve as a built-in text editor. Prior to executing R functions at the Console, commands are typically written down (or scripted). Scripting is essentially showing your work. The sequence of functions necessary to complete a task are scripted in order to document or automate a task. While scripting may seem cumbersome at first, it ultimately saves time in the long run, particularly for repetitive tasks (humorous YouTube Video on Scripting (https://www.youtube.com/watch?time_continue=1&v=s3JldKoA0zw)). Benefits include:

- allows others to reproduce your work, which is the foundation of science
- serves as instruction/reminder on how to perform a task
- allows rapid iteration, which saves time and allows the evaluation of incremental changes
- reduces the chance of human error

4.7.1 Basic Tips of Scripting

To write a script, simply open a new R script file by clicking **File>New File>R Script**. Within the text editor **type** out a sequence of functions.

- Place each function (e.g. `read.csv()`) on a separate line.
- If a function has a long list of arguments, place each argument on a separate line.
- A command can be executed from the text editor by placing the cursor on a line and typing **Ctrl + Enter**, or by **clicking** the Run button.
- An entire R script file can be executed by **clicking** the Source button.



4.8 Saving R Files

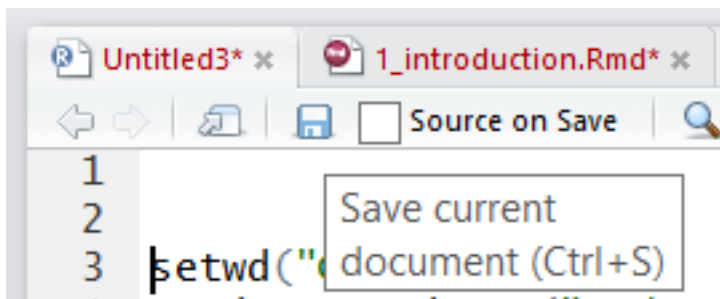
In R, you can save several types of files to keep track of the work you do. The file types include: workspace, script, history, and graphics. It is important to save often because R, like any other software, may crash periodically. Such problems are especially likely when

working with large files. You can save your workspace in R via the command line or the File menu.

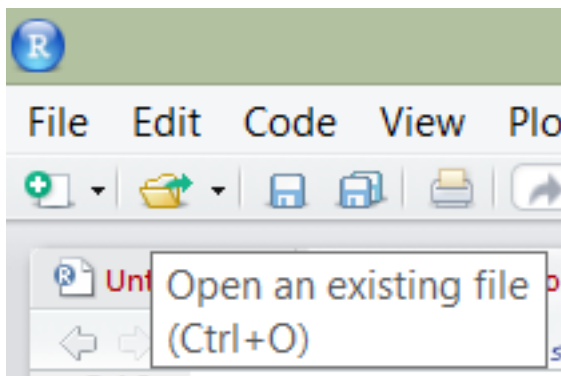
4.8.1 R script (.R)

An R script is simply a text file of R commands that you've typed.

You may want to save your scripts (whether they were written in R Editor or another program such as Notepad) so that you can reference them in the future, edit them as needed, and keep track of what you've done. To save R scripts in RStudio, simply **click the save button** from your R script tab. Save scripts with the .R extension. R assumes that script files are saved with only that extension. If you are using another text editor, you won't need to worry about saving your scripts in R. You can open text files in the RStudio text editor, but beware copying and pasting from Word files as discussed below.



To open an R script, **click the file icon**.

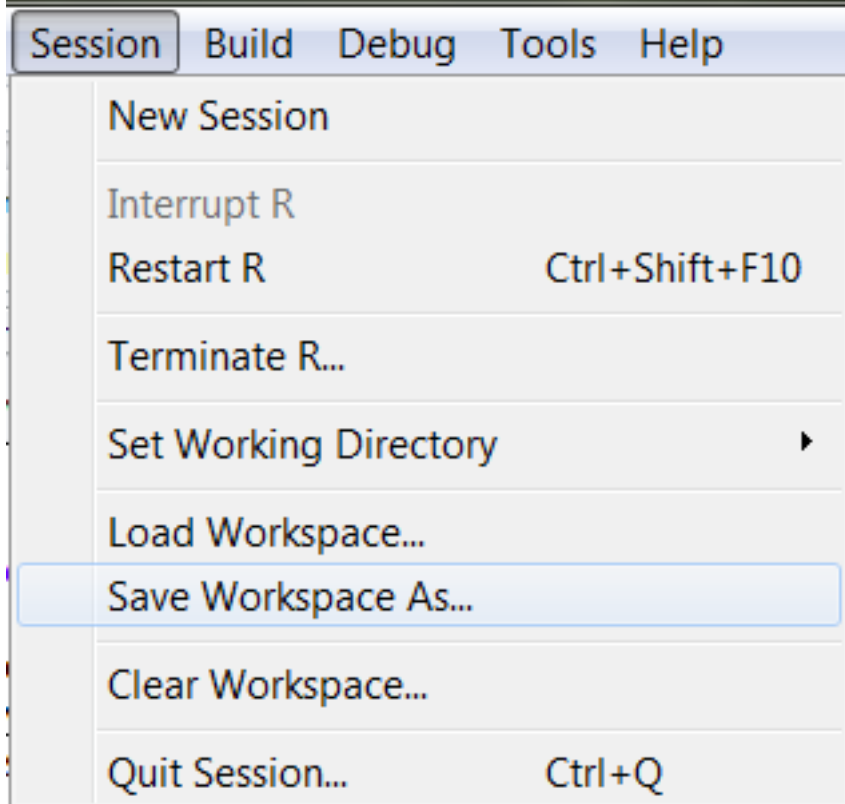


4.8.1.1 Note on Microsoft Word Files

Using Microsoft Word to write or save R scripts is generally a bad idea. Certain keyboard characters, such as quotations “”, are not stored the same in Word (e.g. “” vs. “”). The difference is largely indistinguishable to the human eye, but will not run in R.

4.8.2 Workspace (.Rdata)

The R workspace consists of all the data objects you've created or loaded during your R session. When you quit R by either typing `q()` or exiting out of the application window, R will prompt you to save your workspace. If you choose yes, R saves a file named .RData to your working directory. The next time you open R and reload your .Rdata workspace, all of your data objects will be available in R and all of the commands that you've typed will be accessible by using the up-arrow and down-arrow keys on your keyboard. You can also save or load your workspace at any time during your R session from the menu by clicking **Session>Save Workspace As..**, or the save button on the **Environment Tab**.

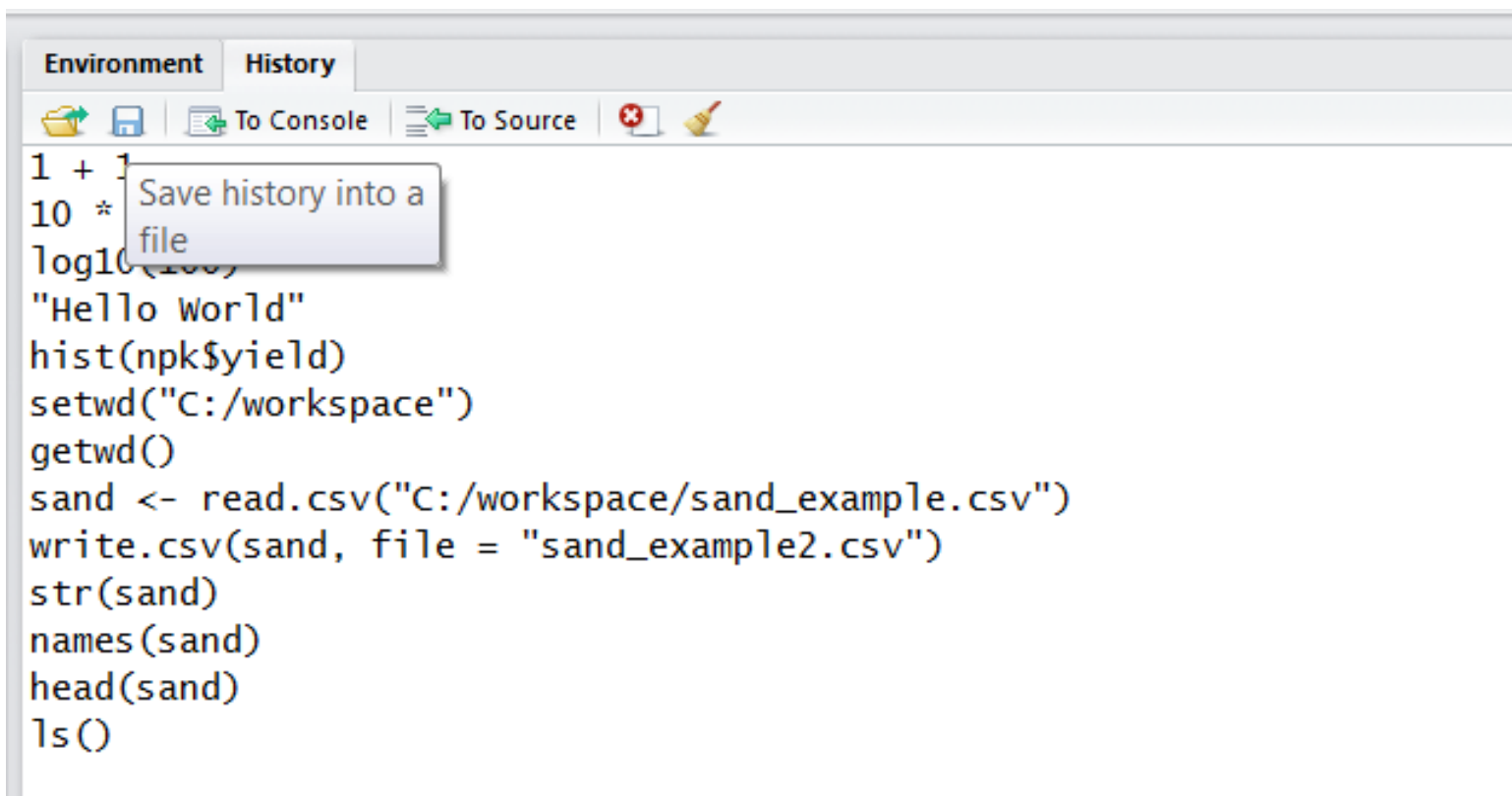


The R command for saving your workspace is:

```
save.image(file="workspace.RData")
```

4.8.3 R history (.Rhistory)

An R history file is a copy of all your key strokes. You can think of it as brute force way of saving your work. It can be useful if you didn't document all your steps in an R script file. Like an R file, an Rhistory file is simply a text file that lists all of the commands that you've executed. It does not keep a record of the results. To load or save your R history from the History Tab click the **Open File** or **Save** button. If you load an Rhistory file, your previous commands will again become available with the up-arrow and down-arrow keys.



You can also use the command line to load or save your history.

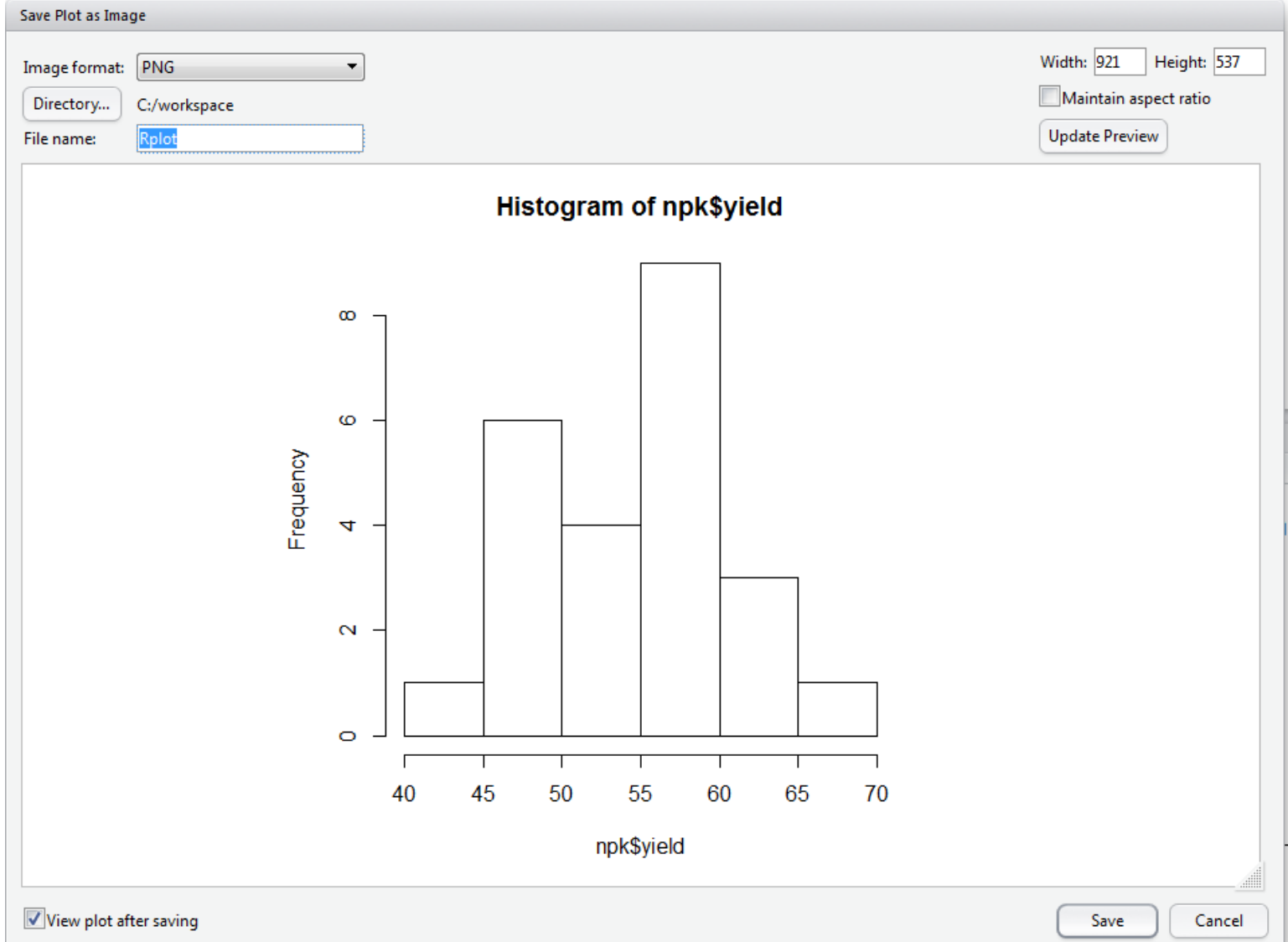
```
savehistory(file = "sand.Rhistory")
loadhistory(file = "sand.Rhistory")
history(max.show=Inf) #displays all previous commands
```

4.8.4 R Graphics

Graphic outputs can be saved in various formats.

Format	Function
pdf	pdf("graphic.pdf")
window metafile	win.metafile("graphic.wmf")
png	png("graph.png")
jpeg	jpeg("graph.jpg")
bmp	bmp("graph.bmp")
postscript	postscript("graph.ps")

To save a graphic: (1) Click the **Plots Tab** window, (2) click the **Export** button, (3) **Choose** your desired format, (3) **Modify** the export settings as you desire, and (4) click **Save**.



The R command for saving a graphic is:

```
png(file = "npk_yield.png")
plot(npk$yield)
dev.off()
```

The first line of this command creates a blank file named sand with a JPEG extension. The second line plots the data object that you want to create a graphic of (here it is conveniently the same name as the JPEG file we are creating). The third line closes the graphics device.

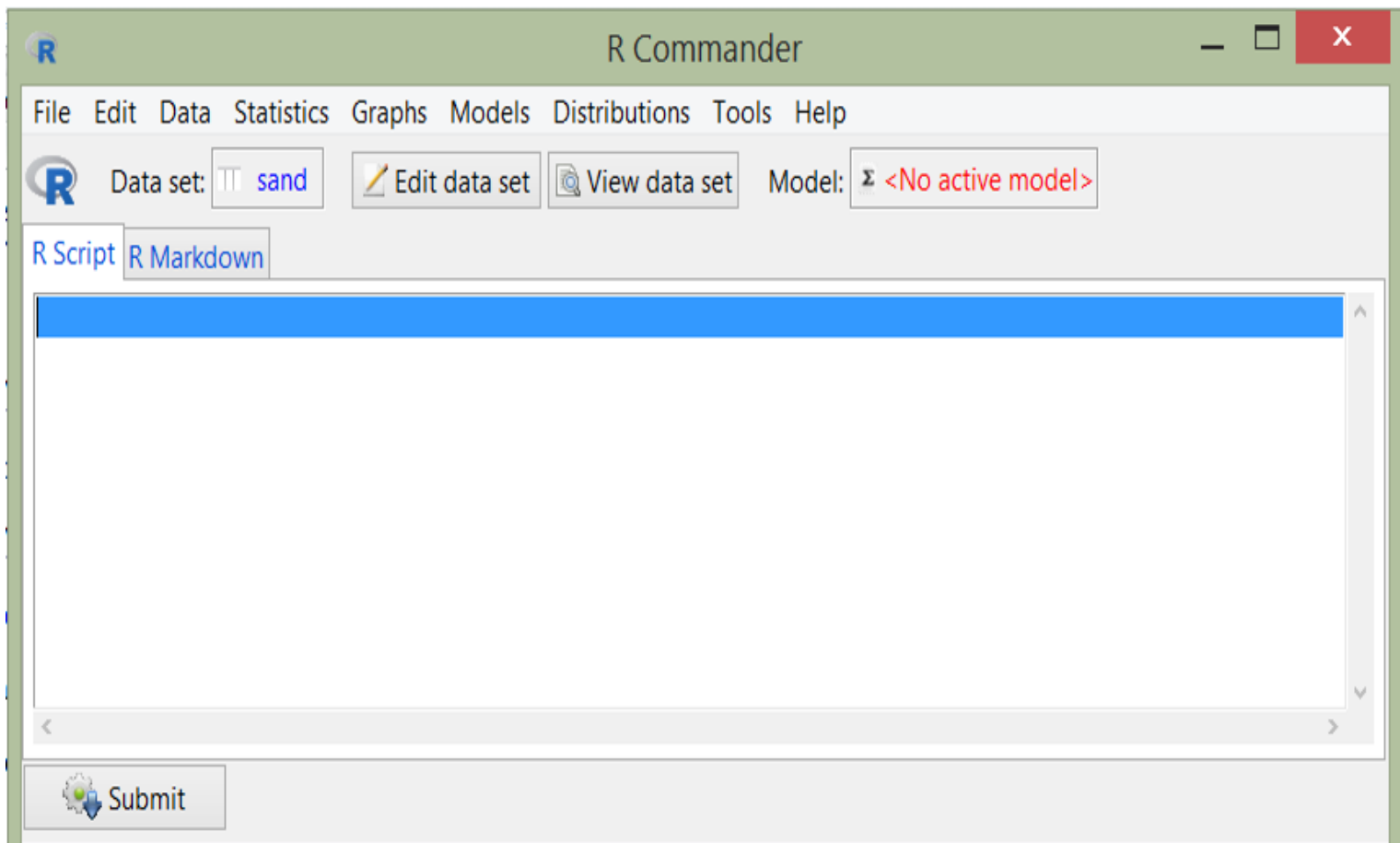
5 Rcmdr (R Commander): A Graphical User Interface for R

While we recommend the use of RStudio for some of the reasons listed above, many people new to R might benefit from a graphical user interface (GUI) that allows the user to run basic functions using a point and click interface. Luckily for beginners R has the R Commander (Rcmdr) GUI, which is similar to JMP (https://www.jmp.com/en_us/learning-library/using-jmp.html). Rcmdr was created by John Fox (<http://socserv.socsci.mcmaster.ca/jfox/Misc/Rcmdr/index.html>) for his introductory statistics students so they could see how the software worked without learning a large number of function names and arguments. Rcmdr is a great way to begin familiarizing

yourself with R and statistics within a GUI environment. Regretably we know of no GUI that allows users to perform the majority of soil survey applications demonstrated in this course, and thus won't Rcmdr won't be covered. For those who wish to pursue Rcmdr, alternative instructions can be viewed at Andy Chang & G. Jay Kerns website (http://gchang.people.ysu.edu/r/R_Instructions.htm).

To take a quick peak at Rcmdr, it can be opened by **entering** the following command into the R console.

```
install.packages(Rcmdr)
library(Rcmdr)
```



6 Review

Given what you now know about R, try to answer the following questions:

1. Can you think of a situation where an existing hypothesis or conventional wisdom was not repeatable?
2. What are packages?
3. What is GitHub?
4. Where can you get help?
5. What does the file .Rprofile do?
6. What do you hope to get out of this class?

7 Additional Readings

- Introductory R Books
 - Quick-R (<https://www.statmethods.net/>)
 - R Cookbook (<http://www.cookbook-r.com/>)
 - Reproducible Research with R & RStudio (not free) (<http://christophergandrud.github.io/RepResR-RStudio/>)
- Advanced DSM R Books
 - Predictive Soil Mapping with R (<https://envirometrix.github.io/PredictiveSoilMapping/>)
 - Using R for Digital Soil Mapping (not free) (<http://www.springer.com/us/book/9783319443256>)
- Soil Science R Applications
 - aqp and soilDB tutorials (<http://ncss-tech.github.io/AQP/>)
 - ISRIC WOrld Soil Information Example Training Courses (<https://www.isric.org/utilise/capacity-building/training-courses#examplecourses>)
 - ISRIC World Soil Information YouTube Channel (<https://www.youtube.com/channel/UCNi1XYjdXWF9eAjvG40KqWg>)
 - OpenGeoHub Courses (<https://opengeohub.org/course>)
 - OpenGeoHub YouTube Channel (<https://www.youtube.com/channel/UC6HFFFYiV4zEYJlQMIXemWA/featured>)
 - David Rossiter's Cornell Homepage (<http://www.css.cornell.edu/faculty/dgr2/>)
 - Pierre Roudier (<https://pierrerroudier.github.io/teaching/index.html>)
- Soil Sciences and Statistics Review Articles
 - Arkely, R., 1976. Statistical Methods in Soil Classification Research. *Advances in Agronomy* 28:37-70.
<https://www.sciencedirect.com/science/article/pii/S0065211308605520>
(<https://www.sciencedirect.com/science/article/pii/S0065211308605520>)
 - Mausbach, M., and L. Wilding, 1991. *Spatial Variability of Soils and Landforms*. Soil Science Society of America, Madison.
<https://dl.sciencesocieties.org/publications/books/tocs/sssaspecialpubl/spatialvariabil>
(<https://dl.sciencesocieties.org/publications/books/tocs/sssaspecialpubl/spatialvariabil>)
 - Wilding, L., Smeck, N., and G. Hall, 1983. Spatial Variability and Pedology. In : L. Wilding, N. Smeck, and G. Hall (Eds). *Pedogenesis and Soil Taxonomy I. Concepts and Interactions*. Elsevier, Amsterdam, pp. 83-116.
<https://www.sciencedirect.com/science/article/pii/S0166248108705993>
(<https://www.sciencedirect.com/science/article/pii/S0166248108705993>)

8 References

Brevik, E.C., J.A. Homburg, B.A. Miller, T.E. Fenton, J.A. Doolittle, and S.J. Indorante, 2016. Selected highlights in American soil science history from the 1980s to the mid-2010s. *Catena* 146:128-146.

Chaney, N., E. Wood, A.B. McBratney, J.W. Hempel, T.W. Nauman, C.W. Brungard, and N.P. Odgers, 2016. POLARIS: A 30-meter probabilistic soil series maps of the contiguous United States. *Geoderma* 274(15):54-67.

<https://www.sciencedirect.com/science/article/pii/S0016706116301434>

(<https://www.sciencedirect.com/science/article/pii/S0016706116301434>)

Hennemann, G.R., and Rossiter, D.G., 2004. Training needs for the next generation of soil surveyors. International Conference on Innovative Techniques in Soil Survey; 22-26 March 2004, Cha-Am, Thailand.

<http://www.css.cornell.edu/faculty/dgr2/Docs/ChaAm/ChaAmKeynoteHennemann.pdf>

(<http://www.css.cornell.edu/faculty/dgr2/Docs/ChaAm/ChaAmKeynoteHennemann.pdf>)

Kempen, B., D. Brus, J. Stoorvogel, G. Heuvelink, F. de Vries, 2012. Efficiency Comparison of Conventional and Digital Soil Mapping for Updating Soil Maps. *Geoderma* 76(6):2095-2115. <https://access.onlinelibrary.wiley.com/doi/10.2136/sssaj2011.0424>

(<https://access.onlinelibrary.wiley.com/doi/10.2136/sssaj2011.0424>)

Ihaka, R., and R. Gentleman. 1996. R: A language for data analysis and graphics. *Journal of Computational and Graphical Statistics* 5(3):399-314.

<https://www.stat.auckland.ac.nz/~ihaka/downloads/R-paper.pdf>

(<https://www.stat.auckland.ac.nz/~ihaka/downloads/R-paper.pdf>)

MacMillan, R., D. Moon, and R. Coupe, 2007. Automated predictive ecological mapping in a Forest Region in B.C., Canada, 2001-2005. *Geoderma* 140(4):353-373.

www.sciencedirect.com/science/article/pii/S0016706107001152

(www.sciencedirect.com/science/article/pii/S0016706107001152)

Mausbach, 2003. The Importance of Statistical Documentation - Keeping Soil Survey Information Relevant in the 21st Century. 2003 National Cooperative Soil Survey Conference, Plymouth, MA.

https://www.nrcs.usda.gov/Internet/FSE_DOCUMENTS/nrcs142p2_051833.pdf

(https://www.nrcs.usda.gov/Internet/FSE_DOCUMENTS/nrcs142p2_051833.pdf)

Ramcharan, A., T. Hengl, T. Nauman, C. Brungard, S. Waltman, S. Wills, and J. Thompson, 2017. Soil Property and Class Maps of the Conterminous United States at 100-Meter Spatial Resolution. *Soil Science Society of America Journal*, 82(1):186-201.

<https://access.onlinelibrary.wiley.com/doi/10.2136/sssaj2017.04.0122>

(<https://access.onlinelibrary.wiley.com/doi/10.2136/sssaj2017.04.0122>)