Atlanta R Users

25 March 2021

REPRODUCIBILITY WITH THE RMARKDOWN/KNITR ECOSYSTEM

Melinda Higgins, PhD; Emory University, Professor

Dated 25 March 2021

Think about your own work…

- What do you want to automate?
- What could you re-use?
    - code, files, formatting, graphics, logos, header, footer, boilerplate
- What should you share with your team?
- What do you find yourself doing over and over?
    - correcting or reformatting
- If you won the lottery today (and left your job), what do you need to tell your replacement so they can pick up where you left off and complete your current tasks?

THE BIG PICTURE

text

data

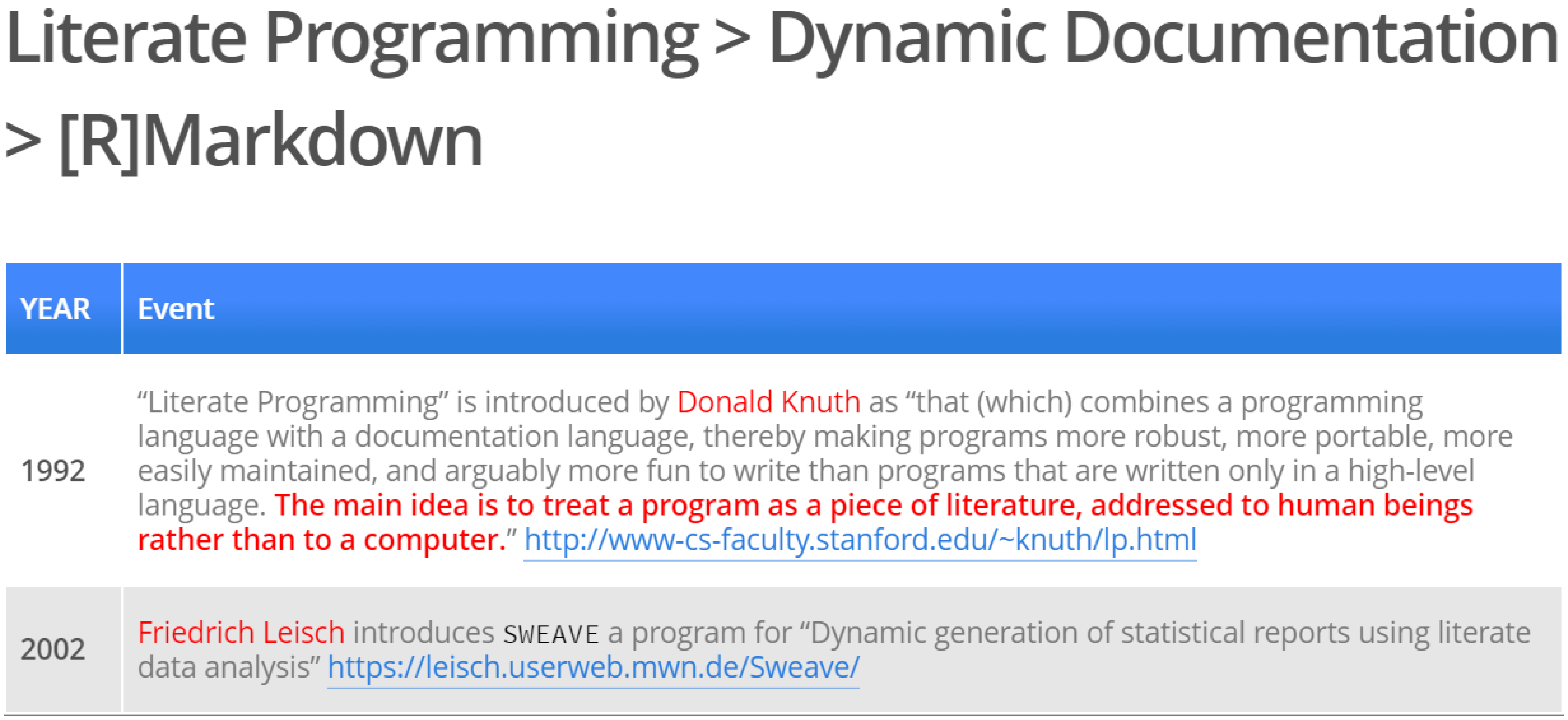
code

figures

tables

- Manuscript
- Report
- Slides
- Website
- Dashboard
- Book

A (BRIEF) HISTORY OF LITERATE PROGRAMMING



Literate Programming > Dynamic Documentation > [R]Markdown

YEAR	Event
1992	"Literate Programming" is introduced by Donald Knuth as "that (which) combines a programming language with a documentation language, thereby making programs more robust, more portable, more easily maintained, and arguably more fun to write than programs that are written only in a high-level language. **The main idea is to treat a program as a piece of literature, addressed to human beings rather than to a computer.**" http://www-cs-faculty.stanford.edu/~knuth/lp.html
2002	Friedrich Leisch introduces `SWEAVE` a program for "Dynamic generation of statistical reports using literate data analysis" https://leisch.userweb.mwn.de/Sweave/

Literate Programming > Dynamic Documentation > [R]Markdown

YEAR	Event
2004	John Gruber created the `Markdown` language in 2004 in collaboration with Aaron Swartz - their goal was to "write using an easy-to-read, easy-to-write plain text format, and optionally convert it to structurally valid XHTML (or HTML)" https://daringfireball.net/projects/markdown/
2012	Yihui Xie releases `knitr` R package released - `knitr` was inspired by `SWEAVE`
2014	`rmarkdown` R package released - extends `Markdown` to work with R/RStudio environment

SWEAVE by Friedrich Leisch

Sweave: Dynamic Generation of Statistical Reports Using Literate Data Analysis

What is Sweave?

"Sweave is a tool that allows to embed the R code for complete data analyses in latex documents. The purpose is to create dynamic reports, which can be updated automatically if data or analysis change. Instead of inserting a prefabricated graph or table into the report, the master document contains the R code necessary to obtain it. When run through R, all data analysis output (tables, graphs, etc.) is ***created on the fly and inserted*** *into a final latex document. The report can be* ***automatically updated if data or analysis change, which allows for truly reproducible research****."*[13]

13. Friedrich Leisch. Sweave: Dynamic generation of statistical reports using literate data analysis. In Wolfgang Härdle and Bernd Rönz, editors, Compstat 2002 - Proceedings in Computational Statistics, pages 575-580. Physica Verlag, Heidelberg, 2002. ISBN 3-7908-1517-9.

The next evolution `<- knitr`

In 2012 Yihui Xie, created and released the `knitr` package for R to extend the capabilities of `SWEAVE` beyond LaTeX.

*"The **knitr** package was designed to be a transparent engine for dynamic report generation with R, solve some long-standing problems in Sweave, and combine features in other add-on packages into one package."*[14]

14: https://yihui.org/knitr/

The next evolution `<- ... + rmarkdown`

- In 2014, RStudio released `rmarkdown` to extend the `markdown` language originally intended to write documents for the “web” *(i.e. HTML)*.[15]

- `rmarkdown` leverages Pandoc *(“universal document converter”)* [16] to convert between formats: from HTML (readable by web browsers) to DOC (such as from Microsoft Word or Google Docs) to ODT (Libre Office) to PDF (portable document format) to others like EPUB (e-books), HTML5 slide shows (slidy, ioslides), and TeX based documents and slides (Beamer).

15. https://daringfireball.net/projects/markdown/syntax
16. http://pandoc.org/index.html

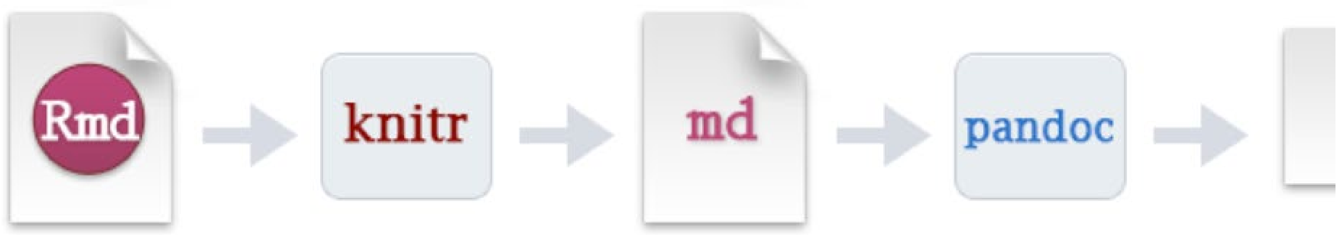
Pandoc https://pandoc.org/

…often called the *Swiss-Army knife* for converting files from one format to another. Pandoc can convert documents in markdown, reStructuredText, textile, HTML, DocBook, LaTeX, MediaWiki markup, TWiki markup, OPML, Emacs Org-Mode, Txt2Tags, Microsoft Word docx, LibreOffice ODT, EPUB, or Haddock markup to

- HTML formats: XHTML, HTML5,Slidy, reveal.js, Slideous, S5, DZSlides.
- Word processor formats: Microsoft Word docx, OpenOffice/LibreOffice ODT, OpenDocument XML
- Ebooks: EPUB version 2 or 3, FictionBook2
- Documentation formats: DocBook, TEI Simple, GNU TexInfo, Groff man pages, Haddock markup
- Page layout formats: InDesign ICML
- Outline formats: OPML
- TeX formats: LaTeX, ConTeXt, LaTeX Beamer slides
- PDF via LaTeX
- Lightweight markup formats: Markdown (including CommonMark), reStructuredText, AsciiDoc, MediaWiki markup, DokuWiki markup, Emacs Org-Mode, Textile
- Custom formats: written in lua.

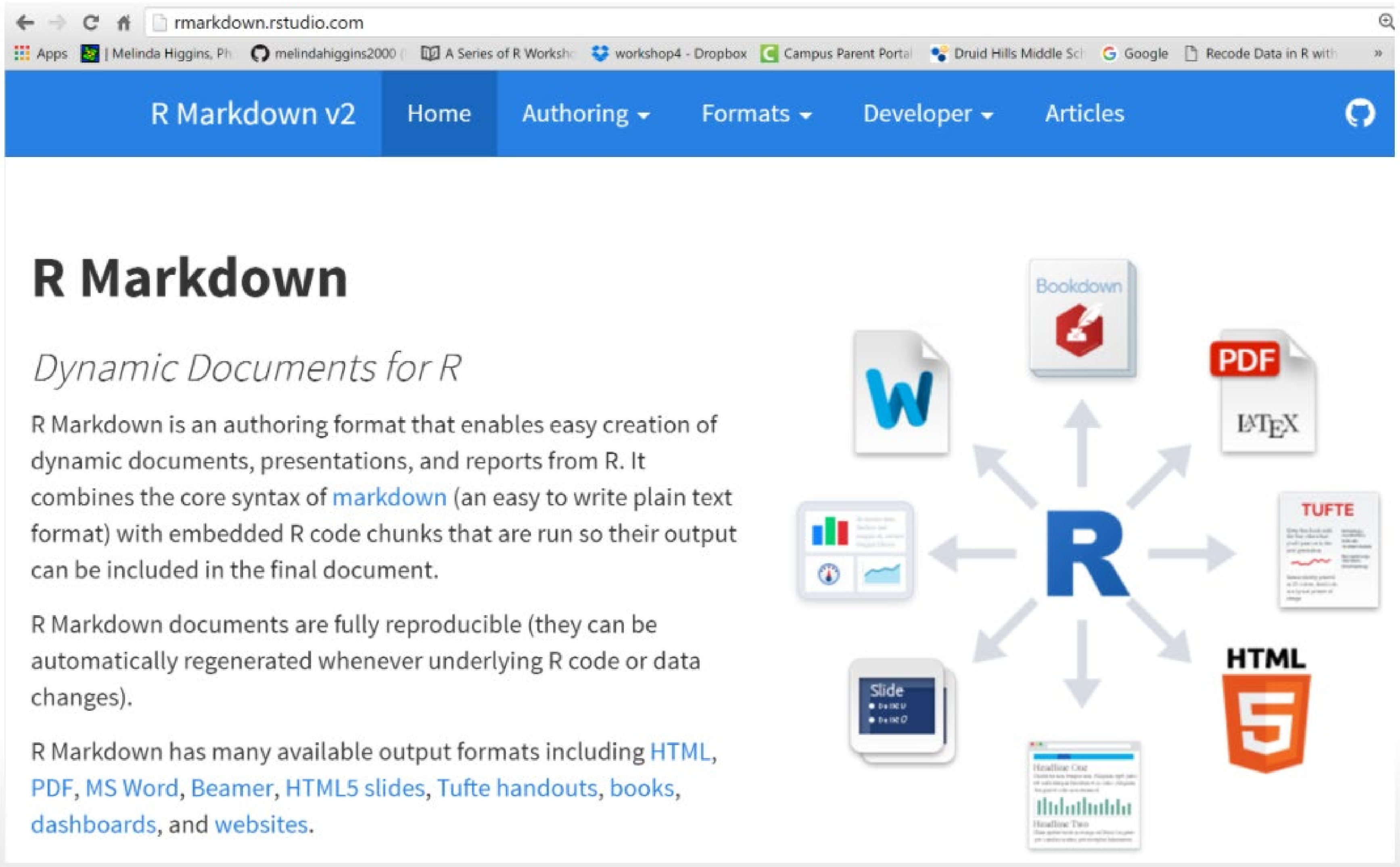
RMARKDOWN (+ PANDOC)

How it works



https://rmarkdown.rstudio.com/

RMARKDOWN “HUB”



R Markdown

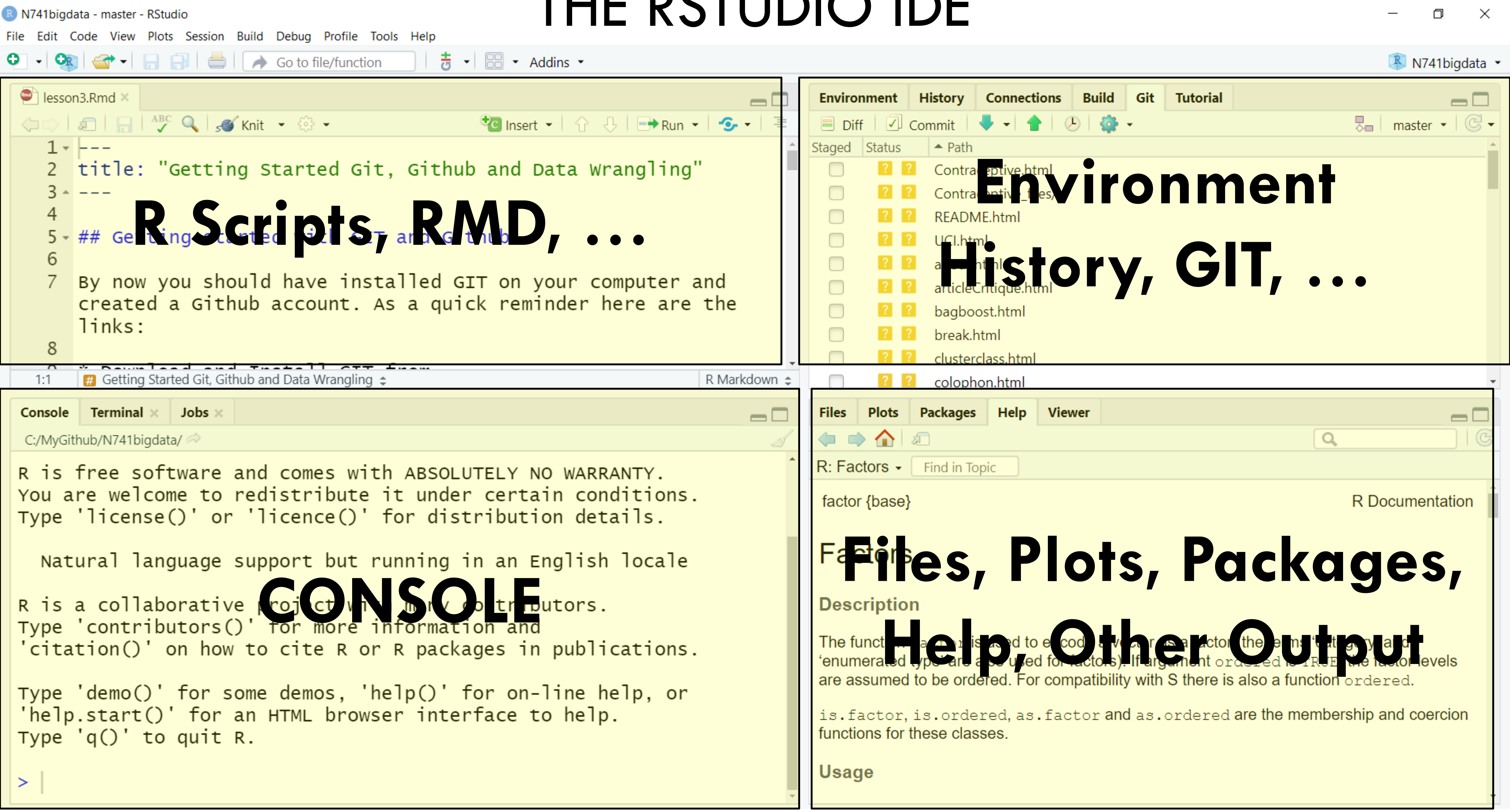
Dynamic Documents for R

R Markdown is an authoring format that enables easy creation of dynamic documents, presentations, and reports from R. It combines the core syntax of markdown (an easy to write plain text format) with embedded R code chunks that are run so their output can be included in the final document.

R Markdown documents are fully reproducible (they can be automatically regenerated whenever underlying R code or data changes).

R Markdown has many available output formats including HTML, PDF, MS Word, Beamer, HTML5 slides, Tufte handouts, books, dashboards, and websites.

THE RSTUDIO IDE


N741bigdata - master - RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
N741bigdata
lesson3.Rmd
Knit
Insert
Run
Environment History Connections Build Git Tutorial
Diff Commit
master
Staged Status Path
Console Terminal Jobs
Files Plots Packages Help Viewer
R: Factors
Find in Topic
1:1
Getting Started Git, Github and Data Wrangling
R Markdown

```
---
title: "Getting Started Git, Github and Data Wrangling"
---

## Getting Started with GIT and Github

By now you should have installed GIT on your computer and
created a Github account. As a quick reminder here are the
links:

* Download and Install GIT from
```

R Scripts, RMD, …

```
Contraceptive.html
Contraceptive_files/
README.html
UCI.html
a...html
articleCritique.html
bagboost.html
break.html
clusterclass.html
colophon.html
```

Environment History, GIT, …

```
C:/MyGithub/N741bigdata/

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
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.

>
```

CONSOLE

factor {base} R Documentation

Factors

Description

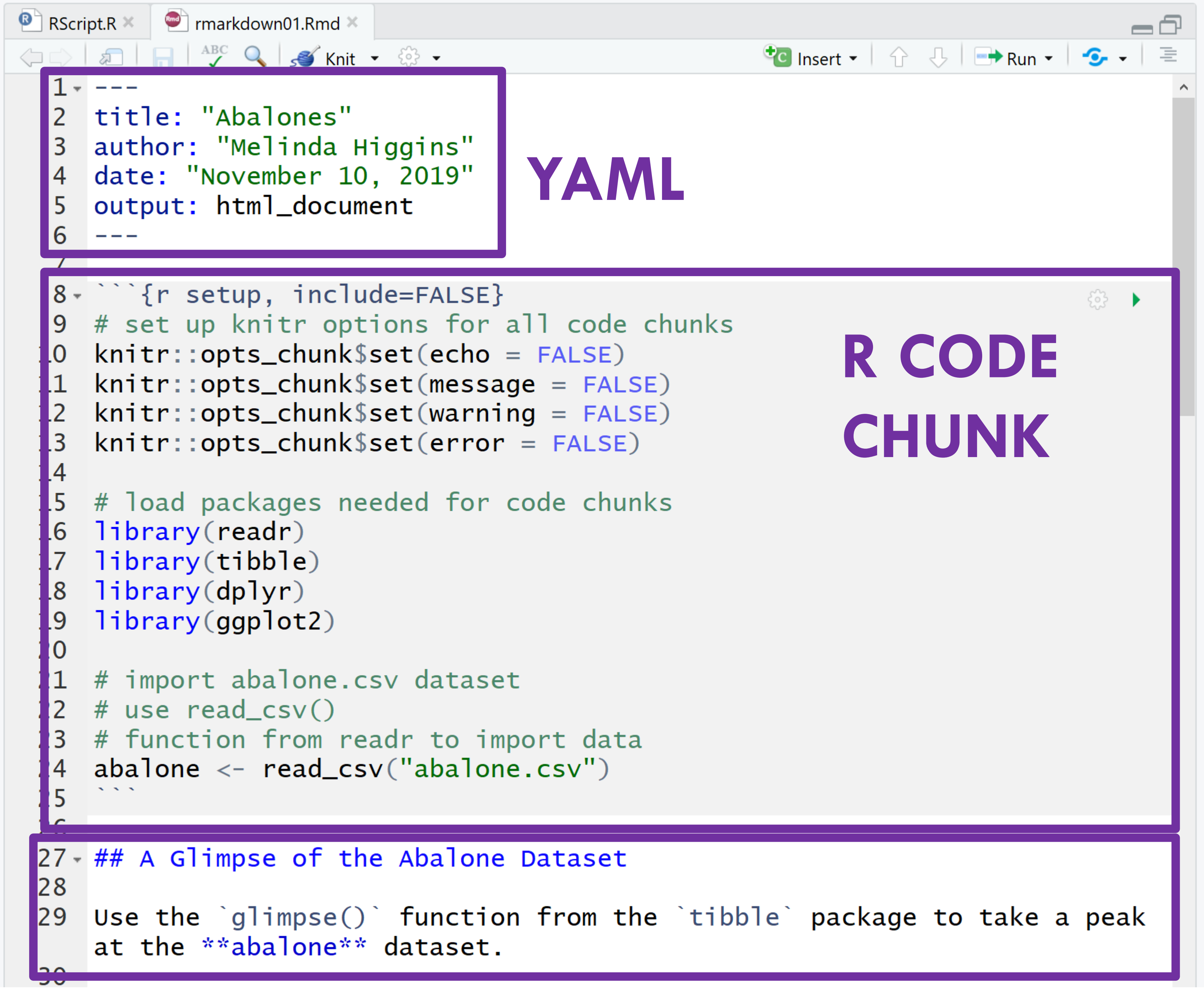
The function factor is used to encode a vector as a factor (the terms 'category' and 'enumerated type' are also used for factors). If argument ordered is TRUE, the factor levels are assumed to be ordered. For compatibility with S there is also a function `ordered`.

`is.factor`, `is.ordered`, `as.factor` and `as.ordered` are the membership and coercion functions for these classes.

Usage

Files, Plots, Packages, Help, Other Output

RMARKDOWN

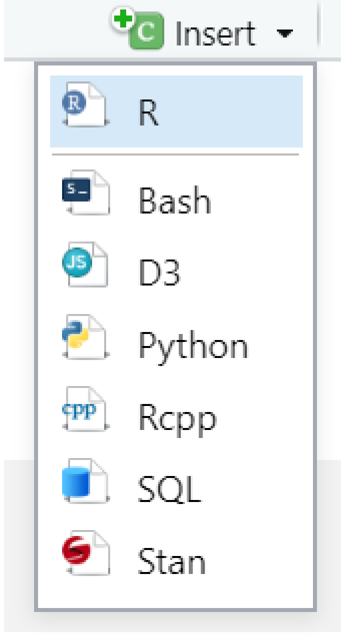


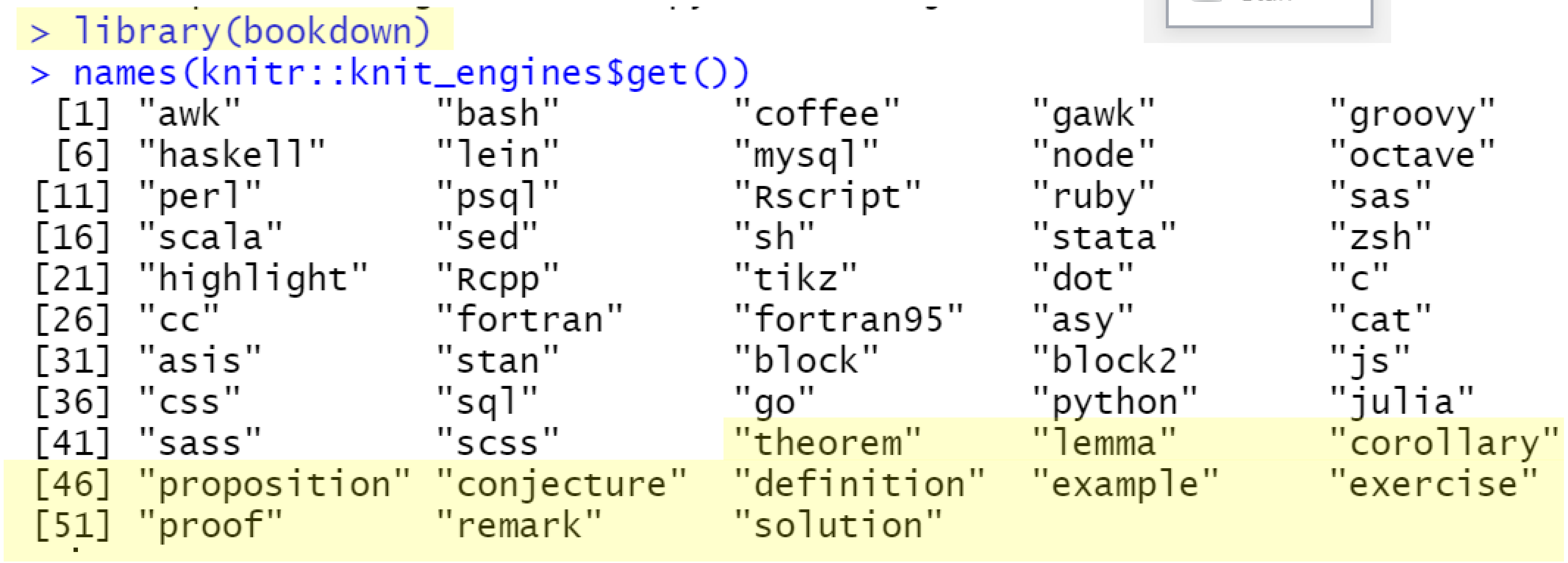
~~~
---
title: "Abalones"
author: "Melinda Higgins"
date: "November 10, 2019"
output: html_document
---

```{r setup, include=FALSE}
# set up knitr options for all code chunks
knitr::opts_chunk$set(echo = FALSE)
knitr::opts_chunk$set(message = FALSE)
knitr::opts_chunk$set(warning = FALSE)
knitr::opts_chunk$set(error = FALSE)

# load packages needed for code chunks
library(readr)
library(tibble)
library(dplyr)
library(ggplot2)

# import abalone.csv dataset
# use read_csv()
# function from readr to import data
abalone <- read_csv("abalone.csv")
```

## A Glimpse of the Abalone Dataset

Use the `glimpse()` function from the `tibble` package to take a peak
at the **abalone** dataset.
~~~

"Marked" up text

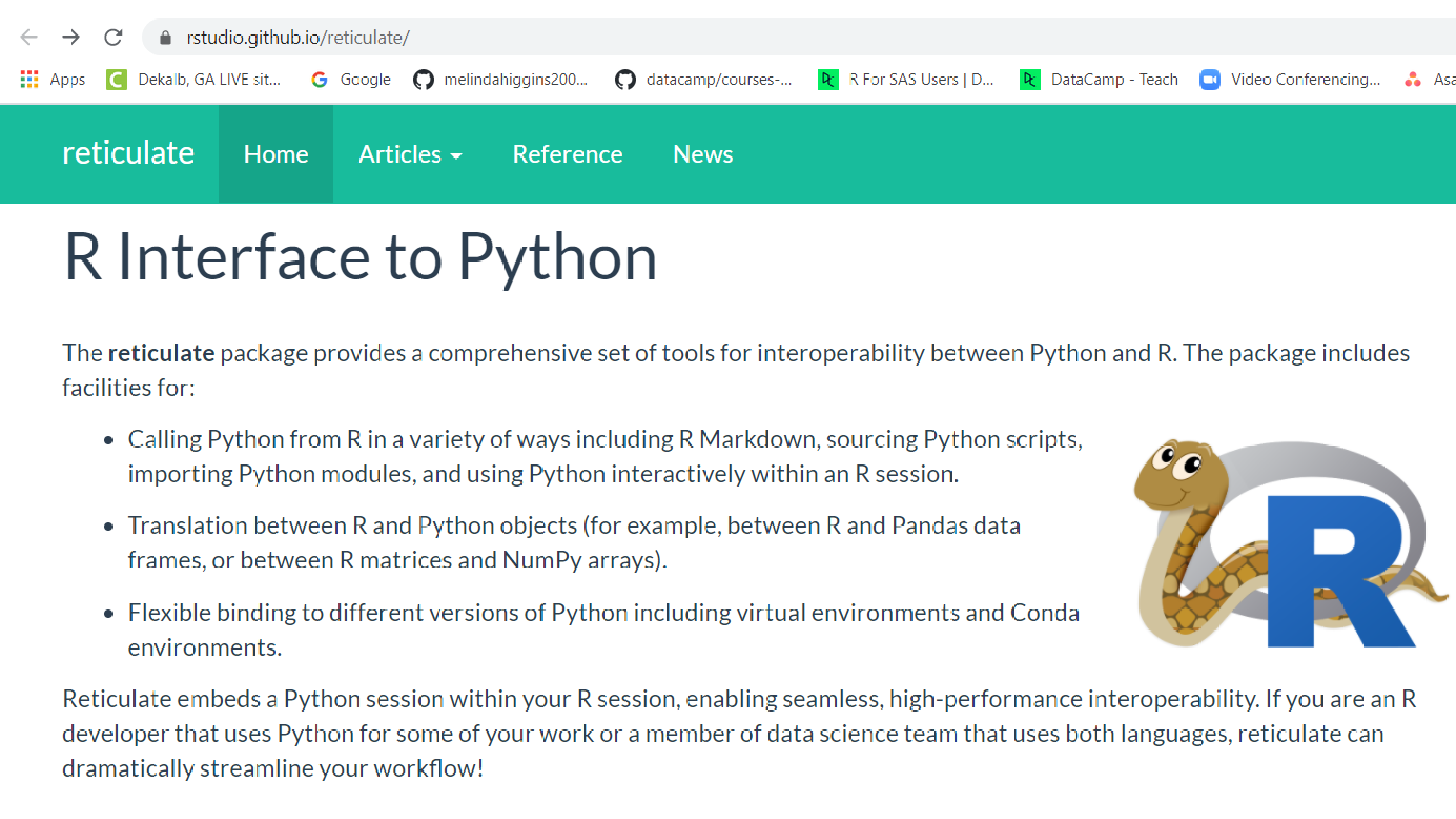
DEMOS

NOT JUST FOR R ANYMORE…





```
> library(bookdown)
> names(knitr::knit_engines$get())
 [1] "awk"         "bash"        "coffee"      "gawk"        "groovy"
 [6] "haskell"     "lein"        "mysql"       "node"        "octave"
[11] "perl"        "psql"        "Rscript"     "ruby"        "sas"
[16] "scala"       "sed"         "sh"          "stata"       "zsh"
[21] "highlight"   "Rcpp"        "tikz"        "dot"         "c"
[26] "cc"          "fortran"     "fortran95"   "asy"         "cat"
[31] "asis"        "stan"        "block"       "block2"      "js"
[36] "css"         "sql"         "go"          "python"      "julia"
[41] "sass"        "scss"        "theorem"     "lemma"       "corollary"
[46] "proposition" "conjecture"  "definition"  "example"     "exercise"
[51] "proof"       "remark"      "solution"
```



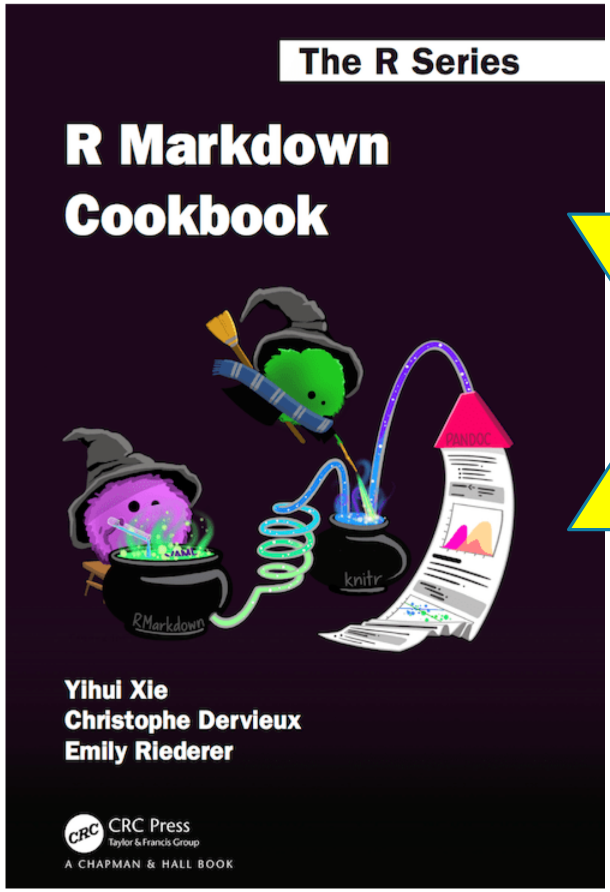
R Interface to Python

The **reticulate** package provides a comprehensive set of tools for interoperability between Python and R. The package includes facilities for:

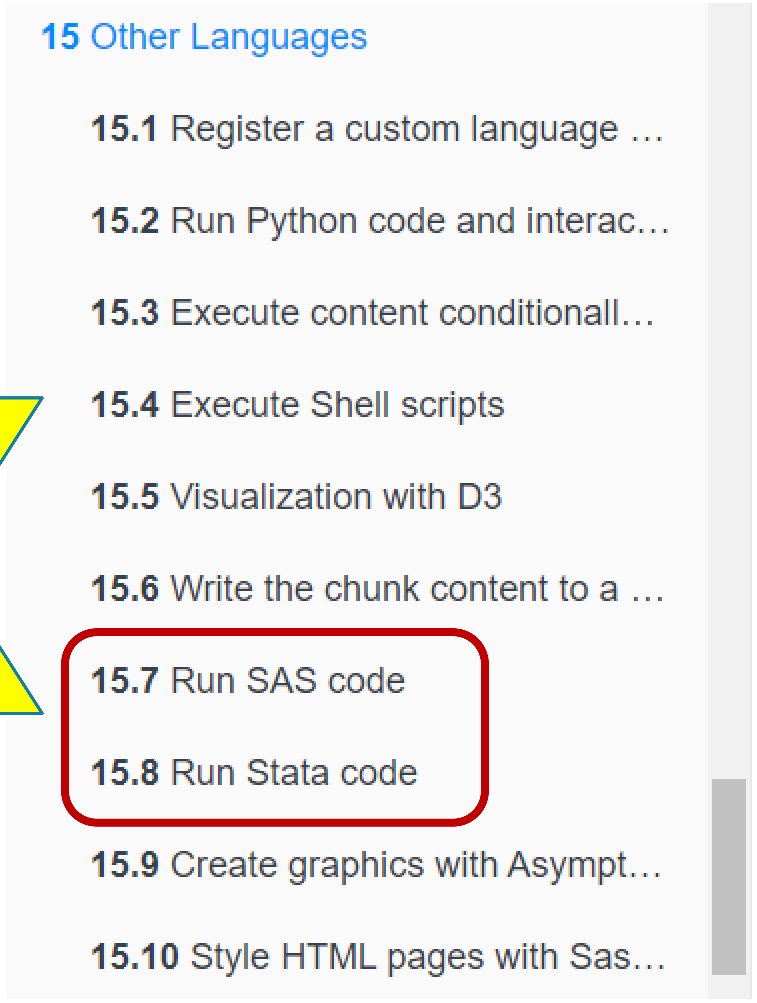
- Calling Python from R in a variety of ways including R Markdown, sourcing Python scripts, importing Python modules, and using Python interactively within an R session.
- Translation between R and Python objects (for example, between R and Pandas data frames, or between R matrices and NumPy arrays).
- Flexible binding to different versions of Python including virtual environments and Conda environments.

Reticulate embeds a Python session within your R session, enabling seamless, high-performance interoperability. If you are an R developer that uses Python for some of your work or a member of data science team that uses both languages, reticulate can dramatically streamline your workflow!

MORE THAN R AND PYTHON…



NOVEMBER 2020



15 Other Languages

15.1 Register a custom language …

15.2 Run Python code and interac…

15.3 Execute content conditionall…

15.4 Execute Shell scripts

15.5 Visualization with D3

15.6 Write the chunk content to a …

15.7 Run SAS code

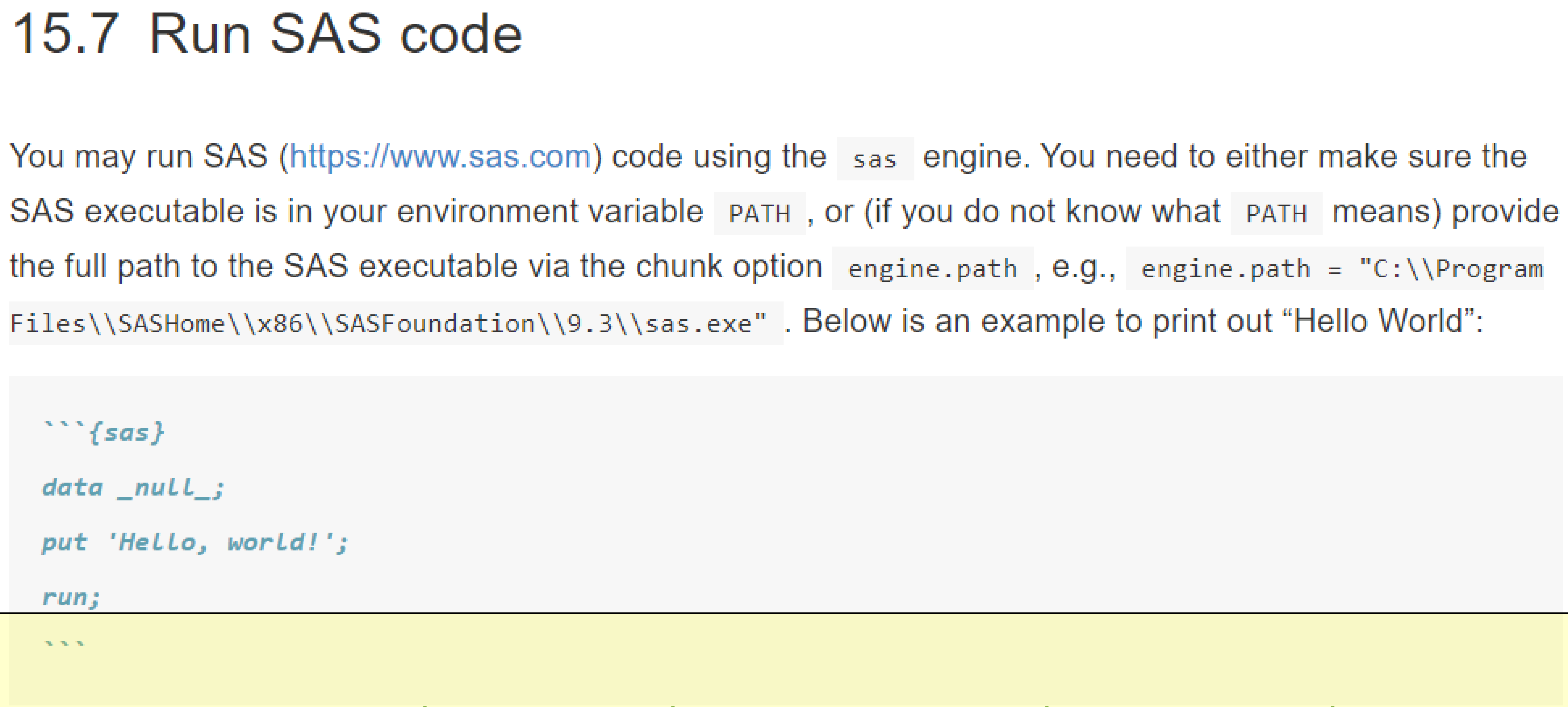
15.8 Run Stata code

15.9 Create graphics with Asympt…

15.10 Style HTML pages with Sas…

https://bookdown.org/yihui/rmarkdown-cookbook/other-languages.html

MORE THAN R AND PYTHON...



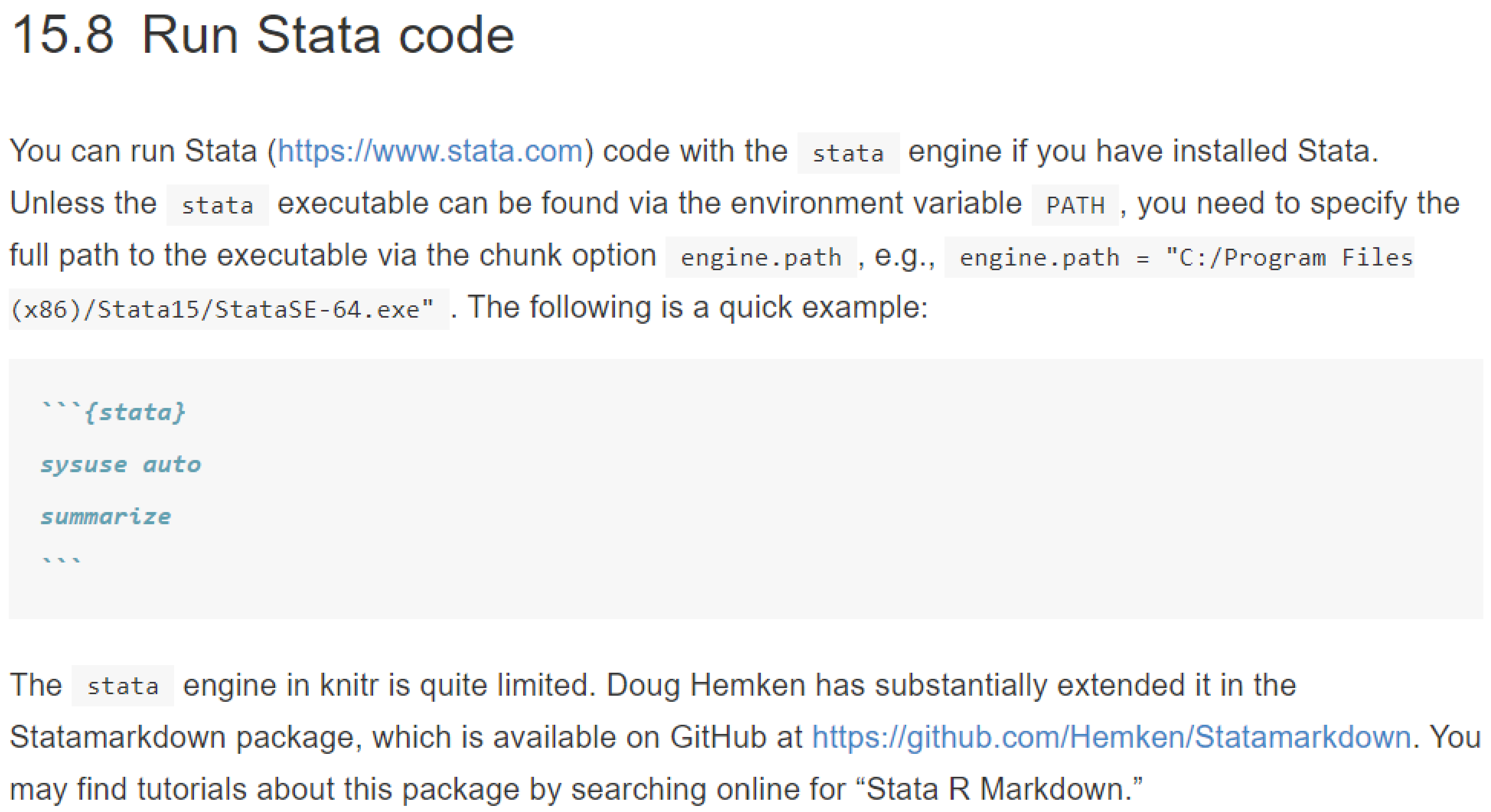
15.7 Run SAS code

You may run SAS (https://www.sas.com) code using the `sas` engine. You need to either make sure the SAS executable is in your environment variable `PATH`, or (if you do not know what `PATH` means) provide the full path to the SAS executable via the chunk option `engine.path`, e.g., `engine.path = "C:\\Program Files\\SASHome\\x86\\SASFoundation\\9.3\\sas.exe"`. Below is an example to print out “Hello World”:

~~~
```{sas}
data _null_;
put 'Hello, world!';
run;
```
~~~

Also see
https://www.ssc.wisc.edu/~hemken/SASworkshops/Markdown/SASmarkdown.html
https://cran.r-project.org/web/packages/SASmarkdown/

MORE THAN R AND PYTHON...



15.8 Run Stata code

You can run Stata (https://www.stata.com) code with the `stata` engine if you have installed Stata. Unless the `stata` executable can be found via the environment variable `PATH`, you need to specify the full path to the executable via the chunk option `engine.path`, e.g., `engine.path = "C:/Program Files (x86)/Stata15/StataSE-64.exe"`. The following is a quick example:

~~~
```{stata}
sysuse auto
summarize
```
~~~

The `stata` engine in knitr is quite limited. Doug Hemken has substantially extended it in the Statamarkdown package, which is available on GitHub at https://github.com/Hemken/Statamarkdown. You may find tutorials about this package by searching online for “Stata R Markdown.”

QUESTIONS?

My contact info:

Melinda.higgins@emory.edu

https://melindahiggins.netlify.app/