Template for contribution to Computo

Example based on the Rmarkdown system

the Computo team

2021-03-12

Introduction

About this document

This document provides a Rmarkdown[1] template for contributions to the **Computo** Journal. We show how `R` or `Python` code can be included.

It also serves as a documentation for configuring the github repository which will host the Rmarkdown source of your manuscript and prove us the reproducibility of your work. To this end, we use binder to generate the final rendering of your manuscript both in HTML and PDF.

Computo template

Subissions to Computo require both scientific content (typically equations, codes and figures) and a proof that this content is reproducible. This is achieved via the standard notebook systems available for R, Python and Julia (Jupyter notebook and Rmarkdown), coupled with the binder building system.

A Computo submission is thus a git(hub) repository typically containing

- the source of the notebook, which may be a Rmarkdown document (such as the present document) or a Jupyter notebook + Myst **link to other templates when available**);
- auxiliary files, e.g.:
  - a BibTeX file (e.g. `./template-computo-Rmarkdown.bib`)

    - some static figures in the `figs/` subdirectory (e.g. `figs/picture.png`)
- configuration files for the binder environment to setup the machine that will build the final notebook file (HTML and/or PDF)

The present document explains how to:

1. format a notebook with Rmarkdown
2. (optionally) configure the binder environment

The compiled notebook (pdf file) will be generated directly in the github repository (via a github action), ready to be submitted to the Computo submission platform (https://computo.scholasticahq.com/).

Advice in writting your manuscript

First make sure that you are able to build your manuscript as a regular notebook on your system. Then you can start configure the binder environment (which we configure to use an Ubuntu machine with latest LTS release).

Formating the notebook

This section is about writing a notebook with the Rmarkdown system, typically for R users.

Rmarkdown basics

We first quickly cover the most basic features of Rmarkdown, that is, formatting text with markdown, math with $\LaTeX$ via MathJax and bibliographical references via $BibT_EX$.

Rmarkdown (http://rmarkdown.rstudio.com) is a simple formatting system for authoring HTML and PDF documents, that relies on the markdown markup language.

To render the document as HTML within Rstudio, click the **Knit** button. A document will be generated that includes both content as well as the output of any embedded R code chunks within the document. Alternatively, the shortcut Ctrl + Maj + K will produces the same result.

Mathematical formulae

$\LaTeX$ code is natively supported, which makes it possible to use mathematical formulae:

$$f(x_1, \dots, x_n; \mu, \sigma^2) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{1}{2\sigma^2}\sum_{i=1}^{n}(x_i - \mu)^2\right)$$

References

References are displayed as footnotes using bibtex, e.g. `[@computo]` will display as (Computo Team 2020), where `computo` is the bibtex key for this entry. The bibliographic information is automatically retrieved from the `.bib` file specified in the header of this document (here: `template-computo-Rmarkdown.bib`).

R code

`R` code (R Core Team 2020) chunks may be embedded as follows:

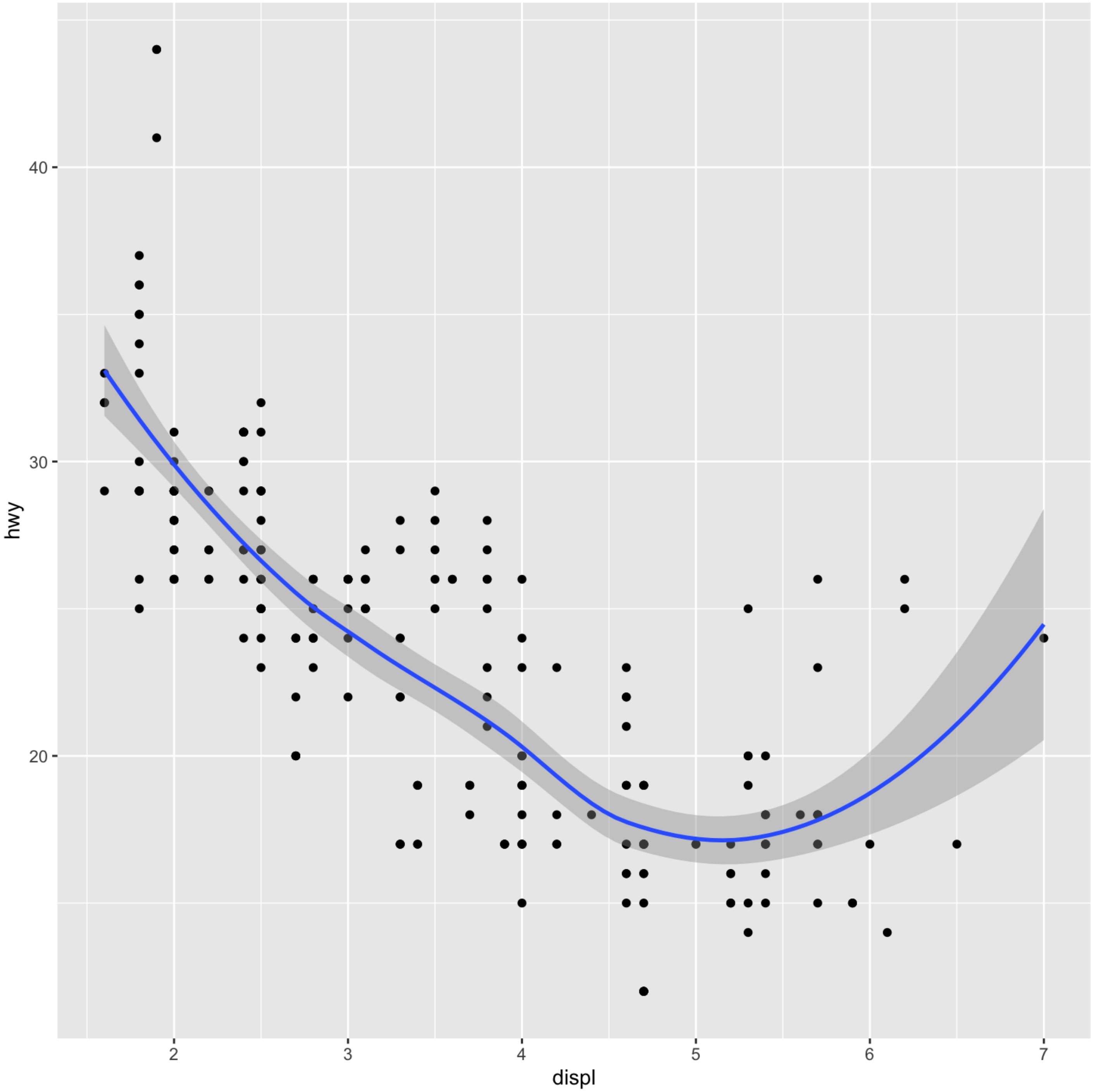
```
knitr::kable(summary(cars))
```

speed	dist
Min. : 4.0	Min. : 2.00
1st Qu.:12.0	1st Qu.: 26.00
Median :15.0	Median : 36.00
Mean :15.4	Mean : 42.98
3rd Qu.:19.0	3rd Qu.: 56.00
Max. :25.0	Max. :120.00

Including Plots

Plots can be generated as:

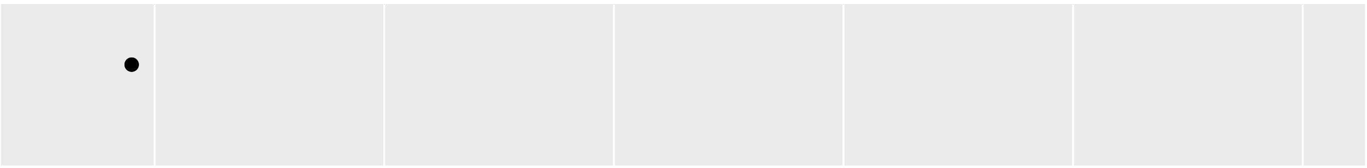
```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```

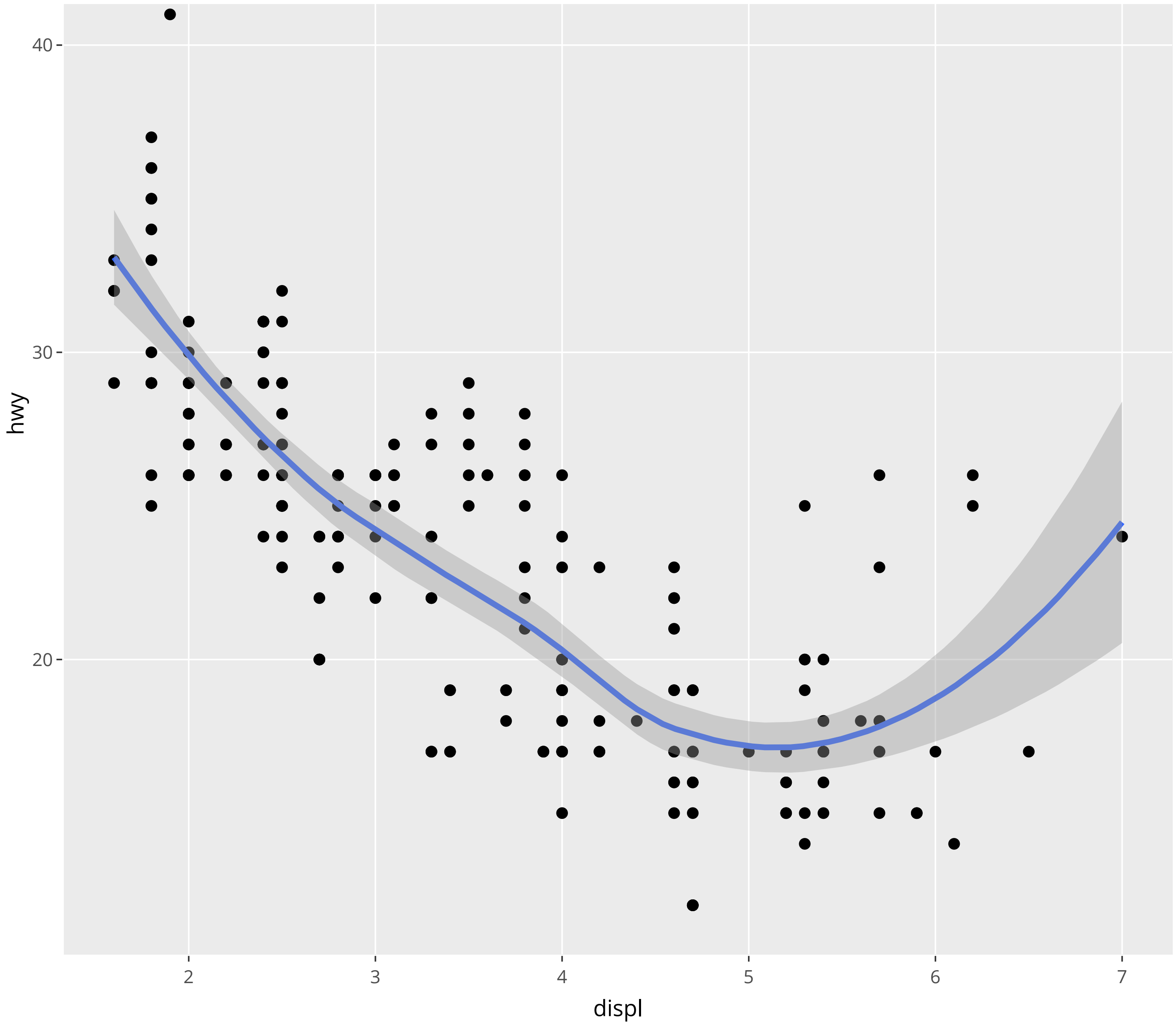


Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.

Interative plots may also be produced in the HTML output of the document:

```
library("plotly")
ggplotly(p)
```





Python Code

The R package reticulate (https://cran.r-project.org/package=reticulate) (Ushey, Allaire, and Tang 2020) includes a Python engine for R Markdown that enables easy interoperability between Python and R chunks. Below we demonstrate a small subset of the available functionalities. We refer to the vignette R Markdown Python Engine (https://cran.r-project.org/web/packages/reticulate/vignettes/r_markdown.html) for a more detailed description.

Setup

```
library("reticulate")
use_virtualenv("computo-template")
```

First make sure (here, in `R`) that the required python modules are available

```
if (!py_module_available("seaborn")) py_install("seaborn")
if (!py_module_available("pandas")) py_install("pandas")
if (!py_module_available("matplotlib")) py_install("matplotlib")
```

Using python

Example of python code and associated output:

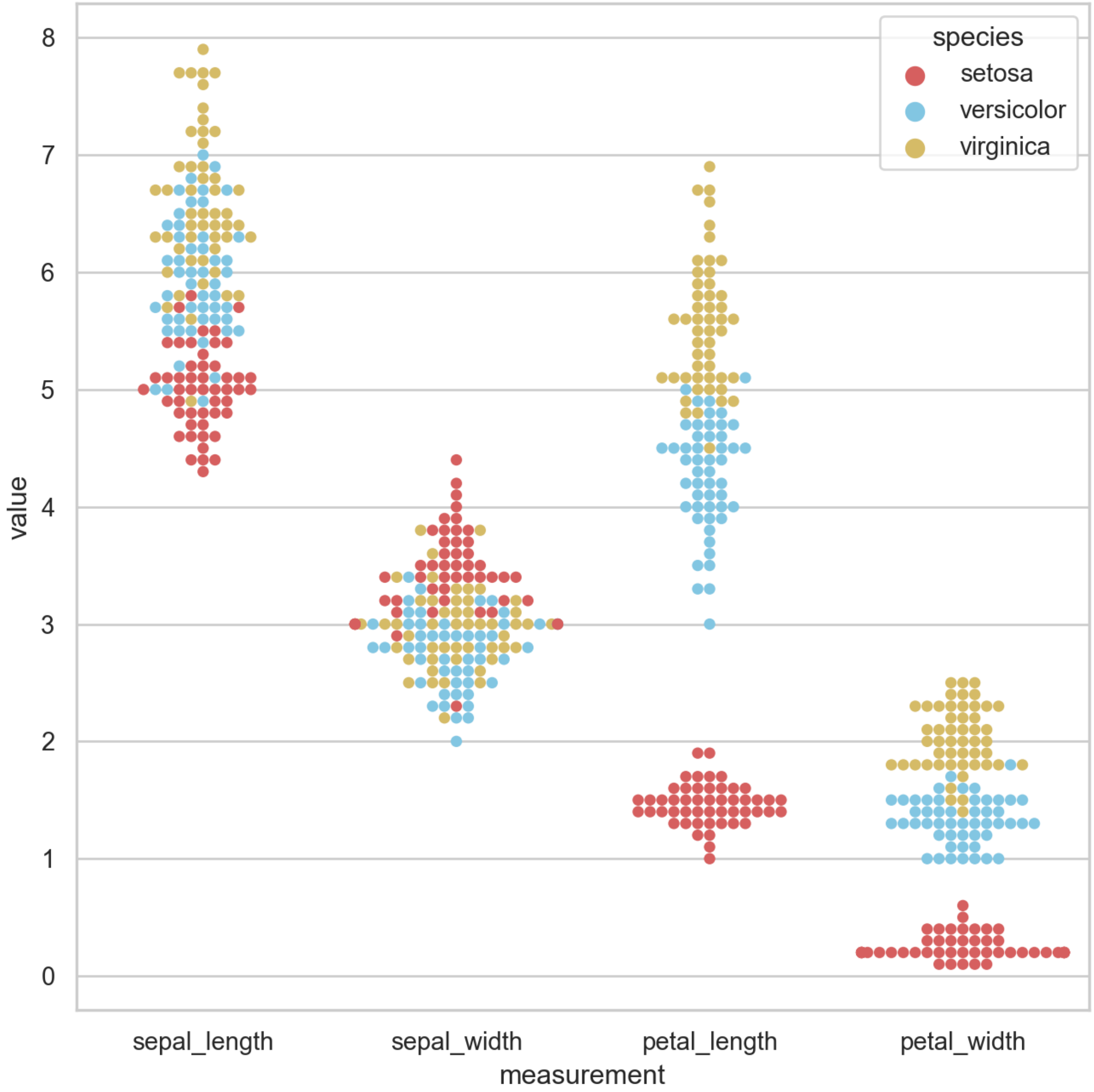
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

sns.set(style="whitegrid", palette="muted")

# Load the example iris dataset
iris = sns.load_dataset("iris")

# "Melt" the dataset to "long-form" or "tidy" representation
iris = pd.melt(iris, "species", var_name="measurement")
plt.figure()

# Draw a categorical scatterplot to show each observation
sns.swarmplot(x="measurement", y="value", hue="species", palette=["r", "c", "y"], data
        =iris)
plt.show()
```



Communication between R and python chunks

All objects created within Python chunks are available to R using the `py` object exported by the reticulate package, e.g.:

```
rmarkdown::paged_table(py$iris)
```

species <chr>	measurement <chr>	value <dbl>
setosa	sepal_length	5.1
setosa	sepal_length	4.9

species <chr>	measurement <chr>	value <dbl>
setosa	sepal_length	4.7
setosa	sepal_length	4.6
setosa	sepal_length	5.0
setosa	sepal_length	5.4
setosa	sepal_length	4.6
setosa	sepal_length	5.0
setosa	sepal_length	4.4
setosa	sepal_length	4.9

1-10 of 600 rows Previous **1** 2 3 4 5 6 ... 60 Next

Conversely, all objects created within R are available from Python using the `r` object exported by the reticulate:

First, let us create an object within R:

```
data(volcano)
```

```
rmarkdown::paged_table(as.data.frame(volcano))
```

V1 <dbl>	V2 <dbl>	V3 <dbl>	V4 <dbl>	V5 <dbl>	V6 <dbl>	V7 <dbl>	V8 <dbl>	V9 <dbl>	V10 <dbl> ▸
100	100	101	101	101	101	101	100	100	100
101	101	102	102	102	102	102	101	101	101
102	102	103	103	103	103	103	102	102	102
103	103	104	104	104	104	104	103	103	103
104	104	105	105	105	105	105	104	104	103
105	105	105	106	106	106	106	105	105	104
105	106	106	107	107	107	107	106	106	105
106	107	107	108	108	108	108	107	107	106
107	108	108	109	109	109	109	108	108	107
108	109	109	110	110	110	110	109	109	108

1-10 of 87 rows | 1-10 of 61 columns Previous **1** 2 3 4 5 6 ... 9 Next

This object is accessible from Python:

```
print(r.volcano)
```

```
## [[100. 100. 101. ... 104. 104. 103.]
##  [101. 101. 102. ... 105. 104. 104.]
##  [102. 102. 103. ... 105. 105. 104.]
##  ...
##  [ 98.  98.  98. ...  94.  94.  94.]
##  [ 97.  98.  98. ...  94.  94.  94.]
##  [ 97.  97.  97. ...  94.  94.  94.]]
```

Other languages

Theoretically, you can include many others languages into Rmarkdown including Julia and C++. If you are comfortable enough to configure binder and prove us the reproducibility of your code, feel free to use any other language.

Configuring binder

Basic binder setup

System libraries

`apt.txt`

R packages

`environment.yml` for packages available in conda, or `install.R` for other packages (including git(hub) packages available via `remotes` or Bioconductor packages availabl via `BiocManager`).

Building

Make a commit which include de string “do_build” in its message. (try to lower the footprint…)

Session information

```
sessionInfo()
```

```
## R version 4.0.3 (2020-10-10)
## Platform: x86_64-apple-darwin13.4.0 (64-bit)
## Running under: macOS Catalina 10.15.7
## 
## Matrix products: default
## BLAS/LAPACK: /usr/local/miniconda/envs/computorbuild/lib/libopenblasp-r0.3.12.dylib
## 
## locale:
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
## 
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base     
## 
## other attached packages:
## [1] reticulate_1.18 plotly_4.9.3    ggplot2_3.3.3  
## 
## loaded via a namespace (and not attached):
##  [1] Rcpp_1.0.6        pillar_1.5.1      compiler_4.0.3    highr_0.8        
##  [5] tools_4.0.3       digest_0.6.27     viridisLite_0.3.0 jsonlite_1.7.2   
##  [9] lattice_0.20-41   nlme_3.1-152      evaluate_0.14     lifecycle_1.0.0  
## [13] tibble_3.1.0      gtable_0.3.0      mgcv_1.8-34       pkgconfig_2.0.3  
## [17] rlang_0.4.10      Matrix_1.3-2      DBI_1.1.1         crosstalk_1.1.1  
## [21] yaml_2.2.1        xfun_0.20         httr_1.4.2        withr_2.4.1      
## [25] stringr_1.4.0     dplyr_1.0.5       knitr_1.31        htmlwidgets_1.5.3
## [29] generics_0.1.0    vctrs_0.3.6       grid_4.0.3        tidyselect_1.1.0 
## [33] data.table_1.14.0 glue_1.4.2        R6_2.5.0          fansi_0.4.2      
## [37] rmarkdown_2.7     tidyr_1.1.3       farver_2.1.0      purrr_0.3.4      
## [41] magrittr_2.0.1    scales_1.1.1      ellipsis_0.3.1    htmltools_0.5.1.1
## [45] splines_4.0.3     assertthat_0.2.1  colorspace_2.0-0  labeling_0.4.2   
## [49] utf8_1.1.4        stringi_1.5.3     lazyeval_0.2.2    munsell_0.5.0    
## [53] crayon_1.4.1
```

References

Computo Team. 2020. “Computo: Reproducible Computational/Algorithmic Contributions in Statistics and Machine Learning.”

R Core Team. 2020. *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing. https://www.R-project.org/ (https://www.R-project.org/).

Ushey, Kevin, JJ Allaire, and Yuan Tang. 2020. *Reticulate: Interface to Python*. https://github.com/rstudio/reticulate (https://github.com/rstudio/reticulate).

---

1. https://rmarkdown.rstudio.com/ (https://rmarkdown.rstudio.com/)↩