Template for contribution to Computo

Example dedicated to `Julia` users

Jane Doe [1]   Statistics, Name of Affiliation one
John Doe   Computer Science, Name of Afficiliation two

Date published: 2/22/23   Last modified: 2/22/23

Abstract

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Curabitur posuere vestibulum facilisis. Aenean pretium orci augue, quis lobortis libero accumsan eu. Nam mollis lorem sit amet pellentesque ullamcorper. Curabitur lobortis libero eget malesuada vestibulum. Nam nec nibh massa. Pellentesque porttitor cursus tellus. Mauris urna erat, rhoncus sed faucibus sit amet, venenatis eu ipsum.

Keywords: key1, key2, key3

Contents

---

[1]Corresponding author: janedoe@nowhere.moon

1 Introduction

1.1 About this document

This document, accompanied with the [hopefully finely tuned git repos](#), provides a template for writing contributions to **Computo** (Computo Team 2020). We show how `Julia` code can be included and how the repository can be set up for triggering github actions for rendering the document, with dependencies handled by the built-in `Pkg` manager.

1.2 Setup a github repository for preparing your submission

You can start by clicking the "use this template" button, on the top of the page of the [github repository associated to this document](#). Of course, you can set your repository private during the preparation of your manuscript.

1.3 Quarto

[Quarto](#) is a versatile formatting system for authoring documents integrating markdown, LaTeX and code blocks interpreted either via Jupyter or Knitr (thus supporting Python, R and Julia). It relies on the [Pandoc](#) document converter.

1.4 Requirements

You need [quarto](#) installed on your system and the [Computo extension](#) to prepare your document. For the latter, once quarto is installed, run the following to install the extension in the current directory (it creates a `_extension` directory which is ignored by git thanks to `.gitignore` by default):

```
quarto add computorg/computo-quarto-extension
```

`Julia` and `Jupyter` must be installed on your computer.

1.5 Link with your usual tools

Quarto is expecting a `.qmd` markdown file, but will also works with a standard [Jupyter notebook](#) file if you are used to it (it will just require to add the proper YAML metadata[2]).

Note: *More advanced Jupyter-related functionality like Myst/Jupyter book are not supported in this Quarto setup. The markdown syntax inside the Jupyter notebook should follow the Quarto syntax (c.f. [below](#)). If you are more comfortable with using Myst/Jupyter book, we provide a [specific template](#) but it will requires more formatting work for Computo editorial team, thus highly encourage authors to use the Quarto templates.*

2 Formatting

This section covers basic formatting guidelines for quarto documents.

---

[2]the same metadata as in the `template-computo-julia.qmd` file in the first cell, type "Raw", of the notebook

To render a document, run `quarto render`. By default, both PDF and HTML documents are generated:

```
quarto render template-computo-julia.qmd # will render both to html and PDF
```

> 💡 **Note**
>
> To check the syntax of the formatting below, you can use the `</> source` button at the top left of this document.

2.1 Basic markdown formatting

Bold text or *italic*

- This is a list
- With more elements
- It isn't numbered.

But we can also do a numbered list

1. This is my first item
2. This is my second item
3. This is my third item

2.2 Mathematics

2.2.1 Mathematical formulae

[LaTeX](#) code is natively supported[3], which makes it possible to use mathematical formulae:

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

It is also posible to cross-reference an equation, see Equation [1](#):

$$
\begin{aligned}
D_{x_N} &= \frac{1}{2} \begin{bmatrix} x_L^\top & x_N^\top \end{bmatrix} \begin{bmatrix} L_L & B \\ B^\top & L_N \end{bmatrix} \begin{bmatrix} x_L \\ x_N \end{bmatrix} \\
&= \frac{1}{2}(x_L^\top L_L x_L + 2 x_N^\top B^\top x_L + x_N^\top L_N x_N),
\end{aligned}
\tag{1}
$$

2.2.2 Theorems and other amsthem-like environments

Quarto includes a nice support for theorems, with predefined prefix labels for theorems, lemmas, proposition, etc. see [this page](#). Here is a simple example:

Theorem 2.1 (Strong law of large numbers)**.** *The sample average converges almost surely to the expected value:*

$$\overline{X}_n \xrightarrow{a.s.} \mu \qquad when\ n \to \infty.$$

---

[3]We use [lualatex](#) for this purpose.

See Theorem 2.1.

2.3 Julia Code

Quarto uses either Jupyter or knitr to render code chunks. This can be triggered in the yaml header. In this tutorial, we use Jupyter, (Julia and Jupyter must be installed on your computer)

```
---
title: "My Document"
author "Jane Doe"
jupyter: julia-1.8
---
```

julia code chunks may be embedded as follows:

```julia
using Random
rng = MersenneTwister(1234)
x = randn(rng, ComplexF32, (10,1))
x
```

```
10×1 Matrix{ComplexF32}:
   0.61330706f0 - 0.63762915f0im
  -0.34964928f0 - 0.6384568f0im
   0.61122406f0 + 1.5640336f0im
    0.3767559f0 - 0.19214594f0im
    0.3552041f0 - 0.36556262f0im
   -0.3963343f0 - 0.013641349f0im
   0.09055523f0 + 1.3101153f0im
  -0.58531713f0 + 0.077849716f0im
  -0.17760806f0 + 0.26142728f0im
   0.05099396f0 - 1.0630851f0im
```

2.4 Figures

Plots can be generated as follows:

```julia
using Plots

plot(sin,
     x->sin(2x),
     0,
     2 ,
     leg=false,
     fill=(0,:lavender))
```
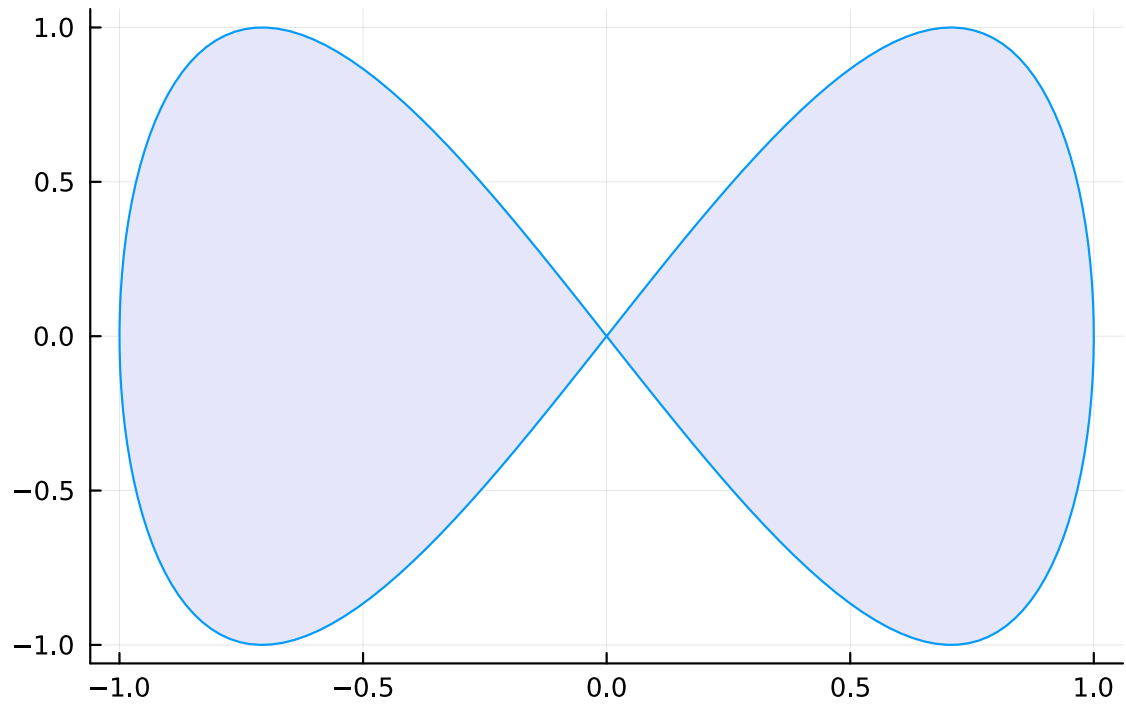
Figure 1: Parametric Plots

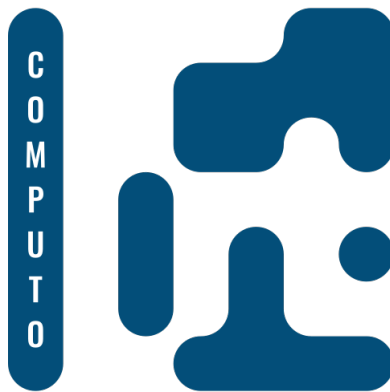It is also possible to create figures from static images:

Figure 2: Computo logo (label)

Note: *Until Quarto version 1.3+ is released, including a remote image (from a web URL) in a document (like the image above) will work in the rendered HTML document but will generate an error when building the PDF document (c.f. related bug report).*

2.5 Tables

Tables (with label: `@tbl-mylabel` renders Table 1) can be generated with markdown as follows

Table 1: my table caption

Tables	Are	Cool
col 1 is	left-aligned	$1600
col 2 is	centered	$12
col 3 is	right-aligned	$1

2.6 Handling references

2.6.1 Bibliographic references

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

2.6.2 Other cross-references

As already (partially) seen, Quarto includes a mechanism similar to the bibliographic references for sections, equations, theorems, figures, lists, etc. Have a look at this page.

2.7 Advanced formatting

Advanced formatting features are possible and documented (including interactive plots, pseudo-code, (Tikz) diagrams, Lua filters, mixing R + Python in the same document), but are beyond the scope of this simple introduction. We point several entries in this direction.

> ⚠️ More information
>
> - The Quarto web site for comprehensive documentation, including:
>   - Tutorial
>   - User guide
>   - Options reference
> - The template distributed with the Computo Quarto extension, which uses such advanced features.
> - Our mock version of the t-SNE paper, a full and advanced example using Python and the Jupyter kernel.
> - The previously published papers in Computo can be used as references.

3 Finalize your submission

3.1 Handle `Julia` dependencies with `Pkg`

To make your work reproducible, you need to fix the packages and environment used to run your analysis. In `Julia`, the built-in `Pkg` package manager is a method of choice. https://towardsdata-science.com/how-to-setup-project-environments-in-julia-ec8ae73afe9c.

In this tutorial, we simply need to add the `Plots` library and its dependencies. It is important that you also add `IJulia` to your project, since it is a required dependency for quarto to correclty render

your document via Jupyter.

Once added via the `Pkg` manager of Julia (`add Plots; add IJulia`), the files `Project.toml` and `Manifest.toml` in the current project directory contains all the information required.

3.2 Continuous integration

The repository associated with this template is pre-configure to trigger an action on push that performs the following:

1. Check out repository on the `ubuntu-latest` machine
2. Install quarto and dependencies, including the Computo extension
3. Install Julia (1.8.1) and dependencies based on the files `Project.toml` and `Manifest.toml`
4. Render your .qmd file and Publish the results on a gh-page (both HTML and PDF)

The file .github/workflows/build_n_publish.yml is largely inspired from this file.

Once this is successful, you are ready to submit your manuscript to the Computo submission platform.

> ⚠️ **Warning**
>
> The first time, you possibly need to create the branch for the action to work. This can be done by running the following command from your computer, in your git repository:
>
> ```
> quarto publish gh-pages
> ```
>
> Then, set the branch `gh-page` as the source of your github page, and trigger the action to check that everything works fine.

3.3 Data and large files

If your submission materials contain files larger than 50MB, **especially data files**, they won't fit on a git repository as is. For this reason, we encourage you to put your data or any materials you deem necessary on an external "open data" centered repository hub such a Zenodo or OSF.

References

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