PharmaSUG 2022 - Paper AP-189

Quarto for Creating Scientific & Technical Documents

Phil Bowsher, RStudio PBC;
Thomas Mock, RStudio PBC

ABSTRACT

Quarto is a new open-source scientific and technical publishing system built on Pandoc for creating dynamic content and reports with Python, R, Julia, and Observable. The goal of this paper is to bring awareness of Quarto to the statistical programming community and to provide an introduction to this new system as well as advice for getting started. In ancient times, a Quarto was a book that was folded to produce eight book pages. To learn more about 'Executable manuscripts' please see this article:

https://www.nature.com/articles/d41586-022-00563-z

INTRODUCTION

R Markdown was created in 2014 as an R package for creating workflows and reports with R. It has been an important addition to clinical workflows as it operates as the report engine for creating rtfs, pdfs, html documents, Word files and more. R Markdown docs use R and knitr to render/generate reports. Using other languages such as Stan is often called from R via a code chunk. Packages provide extensions in R Markdown like blogdown and bookdown, each with its own operating methods.

So why go to Quarto?

First off, R Markdown will continue to be supported. If you prefer it, keep using it. R Markdown will not be deprecated and the RMarkdown/Quarto/RStudio team will continue to provide support and bug-fixes for RMarkdown. Over time, Quarto may gain new features that will not be backported into Quarto. Importantly, for R users the knitr package is the computational engine for both RMarkdown and Quarto. Additional notes on RMarkdown vs Quarto can be found in Yihui Xie's blog (maintainer of knitr/RMarkdown packages) as well as in the Quarto FAQ.

R Markdown was primarily designed with R users in mind and is R centric. In R Markdown there is a hard-dependency on R/Knitr for computation. Note that in many situations, Quarto will be able to render existing Rmd files without modification, so when you decide to switch the transition is smooth for core formats.

Quarto is designed to be forward thinking and intended for users of current and future languages that want to create reproducible documents with Pandoc/markdown. Quarto is the next generation of R Markdown, so over time it will likely surpass R Markdown in features/capability. Quarto does not require R, is cross language and is a general interface for creating a consistent system for many languages. As of today, Quarto is focused on scientific and computing languages such as Python, R, Julia, and Javascript, but the team is committed to adding new languages/engines over time. Quarto runs computations via separate pluggable language "engines" (eg Python/Julia via Jupyter, R via knitr, Observable Javascript) which is great for Multiple Language teams. Quarto is sponsored by RStudio, PBC and the same core developer team works on R Markdown, knitr, and Quarto along with additional community contributors**.** Quarto was designed for people and teams that want to create and share documents using a plethora of languages. Users can export to over 40 different output formats. For example, interested in trying Julia? Quarto supports Julia like this:

~~~
---
title: "Plots Demo"
author: "Norah Jones"
date: "5/22/2021"
format:
  html:
    code-fold: true
jupyter: julia-1.7
---

## Parametric Plots

Plot function pair (x(u), y(u)).
See @fig-parametric for an example.

```{julia}
#| label: fig-parametric
#| fig-cap: "Parametric Plots"
~~~

```
using Plots

plot(sin,
     x->sin(2x),
     0,
     2π,
     leg=false,
     fill=(0,:lavender))
```
https://quarto.org/docs/computations/julia.html

Getting Started with Quarto

Like R Markdown, Quarto can work in a local laptop/desktop or on a Server. Quarto is under active development so you may want to download/update regularly (once a week) to make sure you are getting the latest features. The first production-stable (1.0) release of Quarto is planned to coincide with RStudio::conf 2022 (late July 2022).

Note that for RStudio v2022.02.0 and beyond, Quarto will be bundled with the installation, and installation of Quarto manually is only needed when deciding to upgrade the version of Quarto.

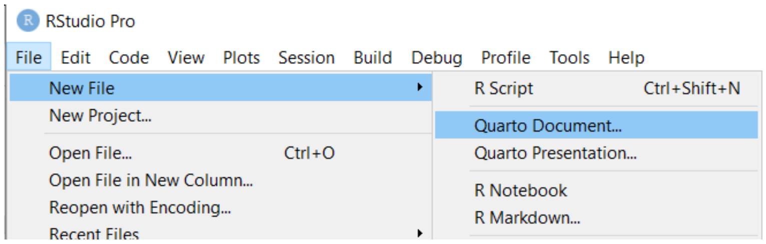
For desktop users, please see this link:

https://quarto.org/docs/get-started/

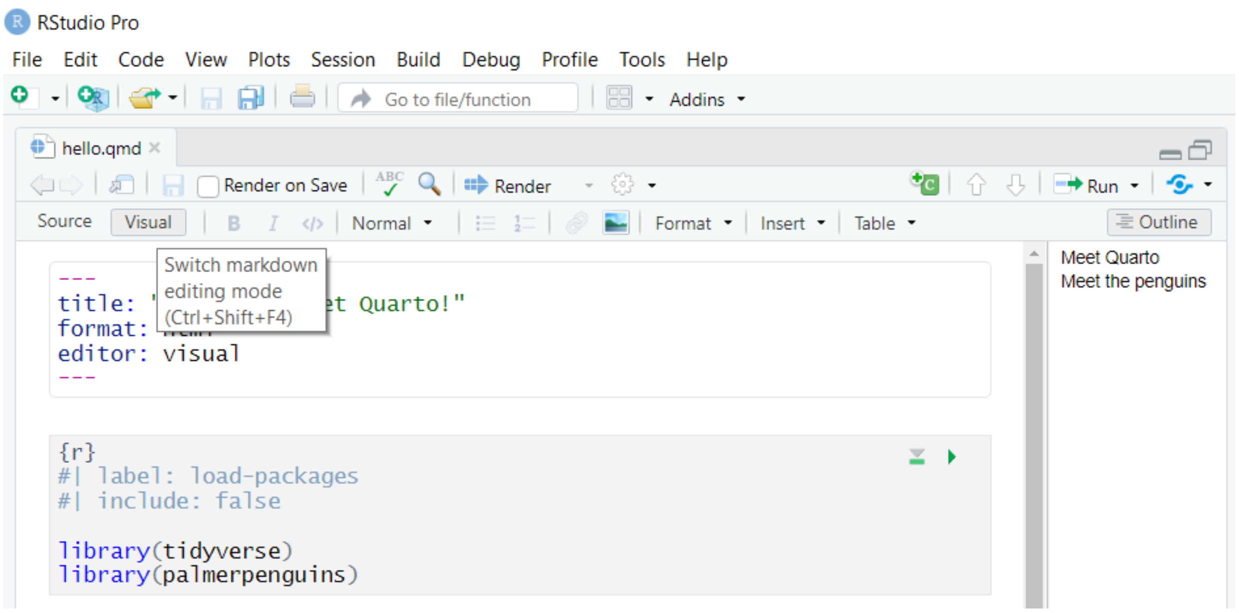
Step 1: Install Quarto
This is probably the biggest change for R Markdown users as R Markdown is installed via the R console. Instead of being an R package, Quarto is a Command Line Interface (CLI), and is a separate software installed independently of a specific programming language.

Step 2: Choose your tool and get started
Keep in mind that Quarto works in many different tools and environments, such as RStudio and VSCode etc. Quarto works within the RStudio IDE just like R Markdown.

A. For RStudio, download and install the latest release of RStudio (v2022.02)
B. Within RStudio, open a Quarto document:



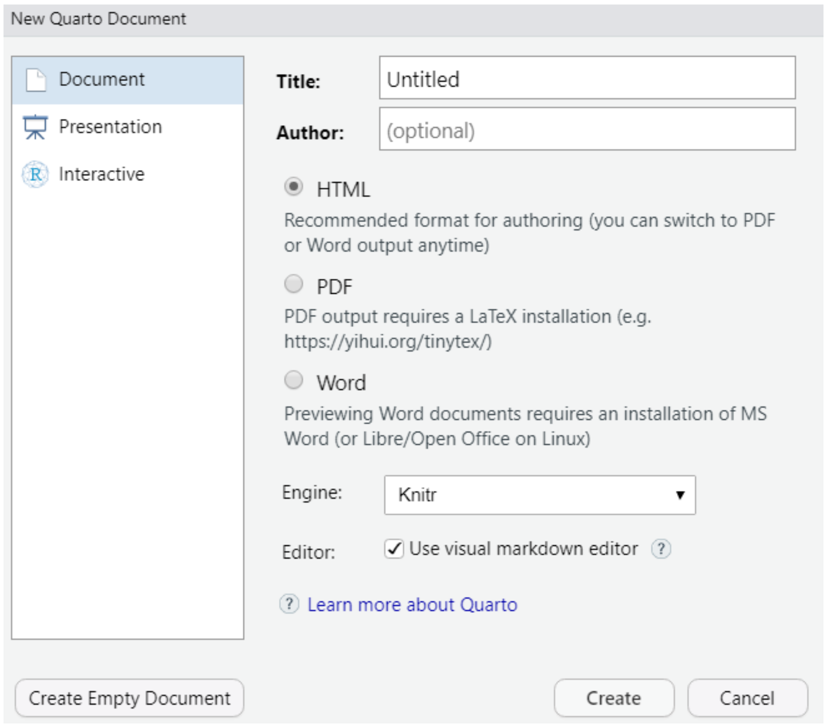
Please note that when authoring inside RStudio, Quarto also has a visual editor interface for all of Pandoc markdown, including tables, citations, cross-references, footnotes, divs/spans, definition lists, attributes, raw HTML/TeX, and more. It is nice for easily editing/formatting of Markdown tables, changing citations and even copy and pasting from Google docs. You can read more about writing Technical documents with the visual editor here: https://quarto.org/docs/visual-editor/technical.html



```
---
title: "Meet Quarto!"
format: html
editor: visual
---
```

```
{r}
#| label: load-packages
#| include: false

library(tidyverse)
library(palmerpenguins)
```

C. Name and select your output type:



Title: Untitled

Author: (optional)

● HTML
Recommended format for authoring (you can switch to PDF or Word output anytime)

○ PDF
PDF output requires a LaTeX installation (e.g. https://yihui.org/tinytex/)

○ Word
Previewing Word documents requires an installation of MS Word (or Libre/Open Office on Linux)

Engine: Knitr

Editor: ✓ Use visual markdown editor

Learn more about Quarto

For Linux Server admins, please see this link:

https://quarto.org/docs/get-started/

There you will find the download for linux:

Linux quarto-0.9.426-linux-amd64.deb

Here is an example shell script to install Quarto on a server: Note that a .tar.gz file is also available if preferred at: https://github.com/quarto-dev/quarto-cli/releases.

```
QUARTO_VERSION=0.9.426

echo "--> Installing Quarto v${QUARTO_VERSION}"
curl -L -o /quarto.deb -L \

"https://github.com/quarto-dev/quarto-cli/releases/download/v${QUARTO_VERSION}/quarto-${QUARTO_VE
RSION}-linux-amd64.deb"

apt install /quarto.deb

rm -f /quarto.deb
```

After Quarto has been installed, the admin will need to confirm it has been added to the $PATH. Once added to the $PATH, Quarto will be available from the terminal and in RStudio.

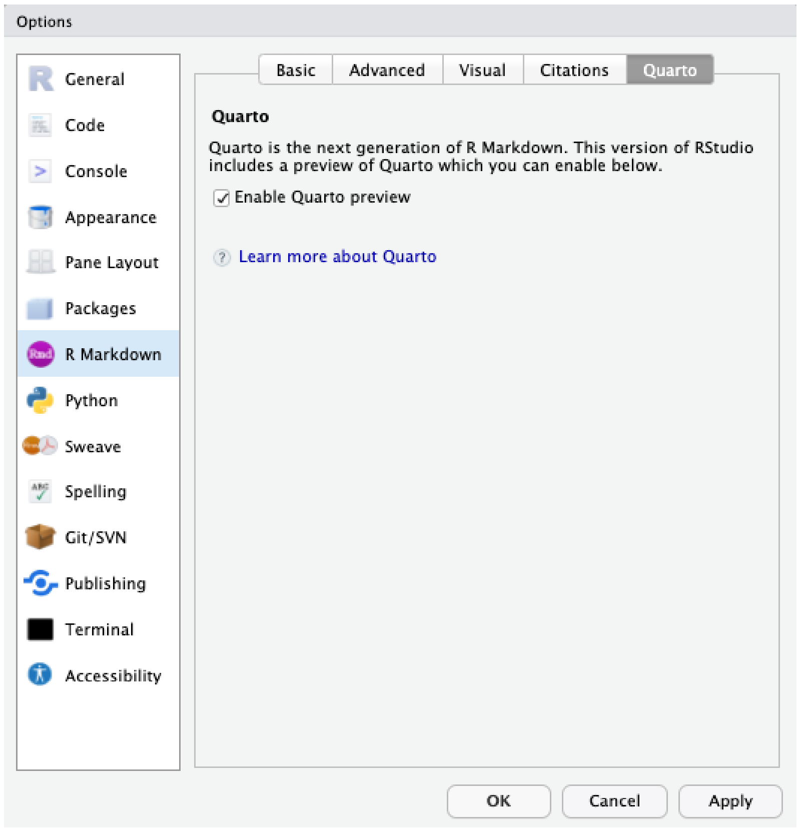
Users/admins can confirm access to the quarto executable via the terminal:

```
$ quarto --help
```

RStudio users on a server will see the option to open a Quarto document by visiting:

File -> New File -> Quarto Document

If Quarto Document is not an available option, first confirm that quarto is on the $PATH. If it is available on the $PATH, next open the Global Options pane in the RStudio IDE, navigate to the RMarkdown tab and then to the Quarto sub-tab. “Enable Quarto Preview” should be checked to activate Quarto features in the RStudio IDE.



Quarto

Quarto is the next generation of R Markdown. This version of RStudio includes a preview of Quarto which you can enable below.

☑ Enable Quarto preview

Examples and Notes on output formats

Quarto can produce a wide variety of output formats. Below is the code for an HTML example. If you are an existing R Markdown user, at first glance you may not notice a big difference. The first indication that you are using Quarto instead of RMarkdown is the `format: html` in the YAML instead of `output: html_document`.

~~~
---
title: "Quarto HTML Basics"
author: "Norah Jones"
date: "March 22nd, 2021"
toc: true
format:
  html:
    html-math-method: katex
    code-tools: true
    self-contained: true
execute:
  warning: false
---

## Introduction

This a Quarto document. To learn more about Quarto see <https://quarto.org>.

Click the **Code** button in the header to see the full source code of this document.

Here we call the R `summary()` function---the function's output is included
immediately below:

```{r}
summary(cars)
~~~

Plot Output

You can also embed plots, for example:

```{r}
#| label: fig-pressure
#| fig-cap: "Pressure"
#| code-fold: true

library(ggplot2)
dat <- data.frame(cond = rep(c("A", "B"), each=10),
                  xvar = 1:20 + rnorm(20,sd=3),
                  yvar = 1:20 + rnorm(20,sd=3))

ggplot(dat, aes(x=xvar, y=yvar)) +
    geom_point(shape=1) +
    geom_smooth()
```

Note that the `code-fold: true` parameter was added to the code chunk to hide the code
by default (click "Code" above the plot to see the code).

The use of the `label` and `fig-cap` options make this a cross-referenceable figure
(see @fig-pressure).

Interactivity

You can also add interactive plots. For example:
```
```

~~~
```{r}
#| label: fig-temperatures
#| fig-cap: "New Haven Temperatures"

library(dygraphs)
dygraph(nhtemp) %>%
  dyRangeSelector(dateWindow = c("1920-01-01", "1960-01-01"))
~~~

Tables

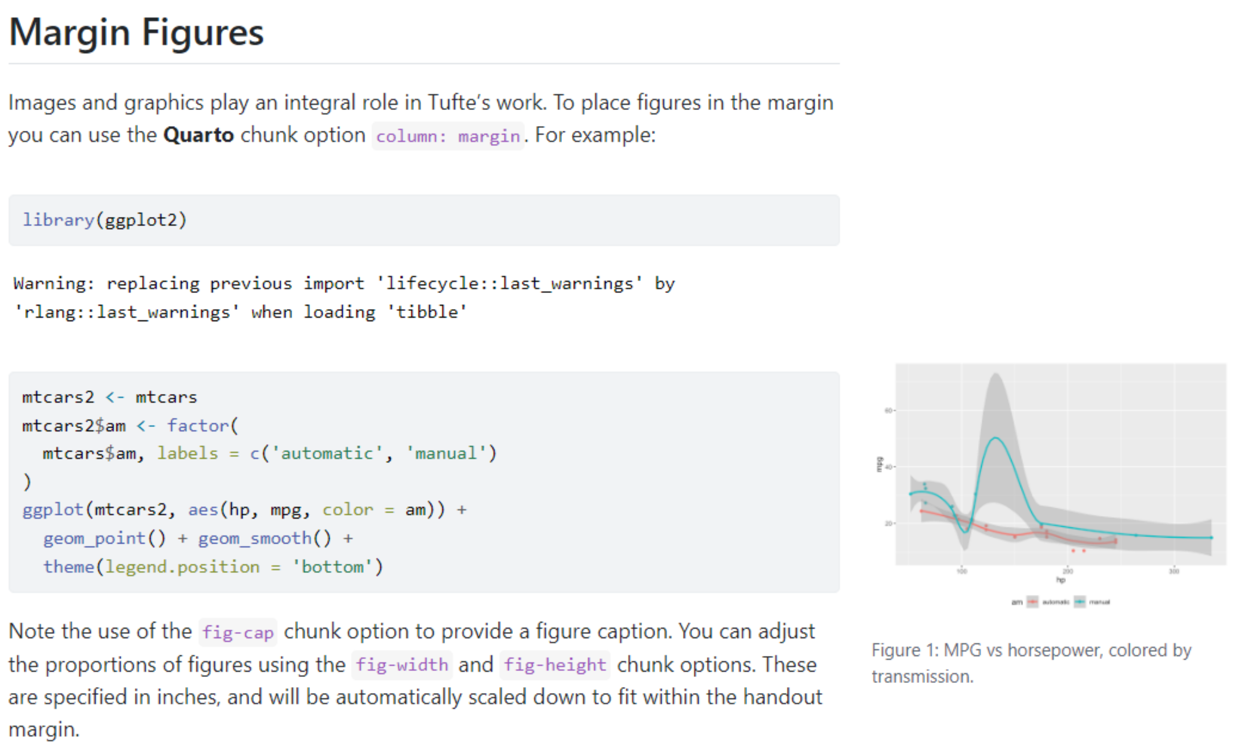
Use the `knitr::kable()` function to print tables as HTML:

```{r}
knitr::kable(head(ggplot2::diamonds))
```

LaTeX Math

You can also include LaTeX math:

$$
P\left(A=2\middle|\frac{A^2}{B}>4\right)
$$
```

This will produce the output seen here:

https://quarto-dev.github.io/quarto-gallery/articles/html/html.html

Below are some key notes:

1. Quarto uses reveal.js for HTML-presentations as opposed to the xaringan RMarkdown package. Quarto slides use pandoc-native markdown, and can be used with the RStudio Visual Editor. Quarto can also create slides in beamer (LaTeX) and the Microsoft Powerpoint format. reveal.js creates beautiful slides seen here:

   https://quarto.org/docs/presentations/revealjs/demo/#/title-slide

2. Quarto has many great layout options, especially for HTML, as seen here with the figure included as an "aside", in the margin next to the main body of the text:
```



## Margin Figures

Images and graphics play an integral role in Tufte's work. To place figures in the margin you can use the **Quarto** chunk option `column: margin`. For example:

```
library(ggplot2)
```

```
Warning: replacing previous import 'lifecycle::last_warnings' by
'rlang::last_warnings' when loading 'tibble'
```

```
mtcars2 <- mtcars
mtcars2$am <- factor(
  mtcars$am, labels = c('automatic', 'manual')
)
ggplot(mtcars2, aes(hp, mpg, color = am)) +
  geom_point() + geom_smooth() +
  theme(legend.position = 'bottom')
```

Figure 1: MPG vs horsepower, colored by transmission.

Note the use of the `fig-cap` chunk option to provide a figure caption. You can adjust the proportions of figures using the `fig-width` and `fig-height` chunk options. These are specified in inches, and will be automatically scaled down to fit within the handout margin.

https://quarto-dev.github.io/quarto-gallery/page-layout/tufte.html

3. Quarto has great code chunk options and ease of applying such as "for show and don't execute" etc.

~~~
```{r}
#| label: fig-pressure
#| fig-cap: "Pressure"
#| code-fold: true

library(ggplot2)
dat <- data.frame(cond = rep(c("A", "B"), each=10),
                  xvar = 1:20 + rnorm(20,sd=3),
                  yvar = 1:20 + rnorm(20,sd=3))

ggplot(dat, aes(x=xvar, y=yvar)) +
    geom_point(shape=1) +
    geom_smooth()
```
~~~

4. Bookdown is a great tool and many people have used it for dissertations etc. Quarto has implemented all of the scientific publishing workflow of bookdown but also has simplified some of the more challenging aspects. The biggest difference will likely be the way the way cross-references are handled. Here is an example book in Quarto:

https://jjallaire.github.io/hopr/

5. A popular open source language used in data science and pharmaceutical drug development is Javascript. Observable JS is a popular choice for interactive data exploration and analysis given its reactive runtime. You can read more about Observable here:

   https://quarto.org/docs/interactive/ojs/

   Quarto uses the core Observable JS (OJS) libraries along with a compiler that is run at render time to present the interactive visualizations. Here is an example:

   https://quarto.org/docs/interactive/ojs/examples/penguins.html

CONTACT & SUMMARY

The information above highlights the exciting developments for Quarto, and how statistical programmers can get started.

Phil Bowsher

phil@rstudio.com

https://github.com/philbowsher