Reproducible Publications with Julia and Quarto

J.J. Allaire (RStudio, PBC)

JuliaCon 2022

https://quarto.org

quarto

Overview

- Quarto Basics
- Scientific Markdown
- Output Formats
- Quarto and Julia

Quarto Basics

What is Quarto?

https://quarto.org

Quarto is an open-source scientific and technical publishing system that builds on standard markdown with features essential for scientific communication.

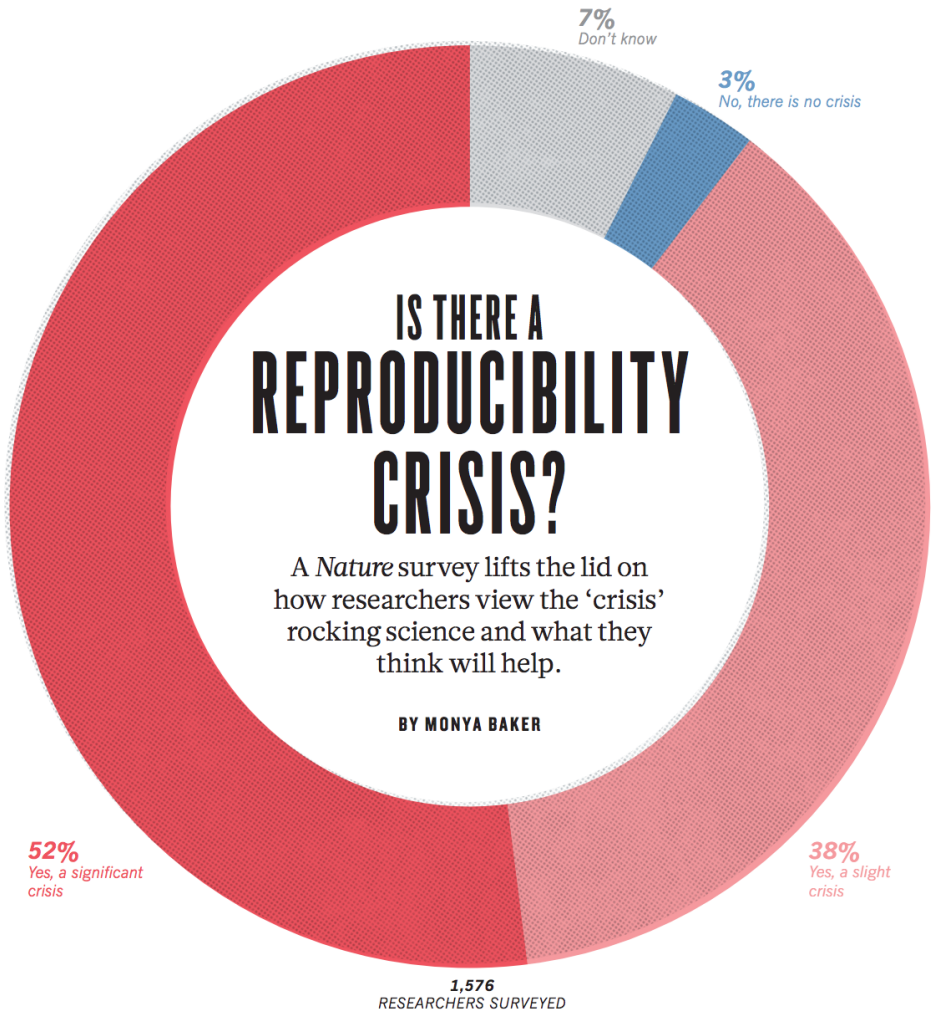
- Computations: Python, R, Julia, Observable JS
- Markdown: Pandoc w/ many enhancements
- Output: Documents, presentations, websites, books, blogs

Literate programming system in the tradition of Org-Mode, Sweave, Weave.jl, R Markdown, iPyPublish, Jupyter Book, etc.

Origins

- Open source project sponsored by RStudio, PBC.
- 10 years of experience with R Markdown (a similar system that was R-specific) convinced us that the core ideas were sound.
- The number of languages and runtimes used for scientific discourse is very broad.
- Quarto is a ground-up re-imagining of R Markdown that is fundamentally multi-language and multi-engine.

Goal: Computational Documents

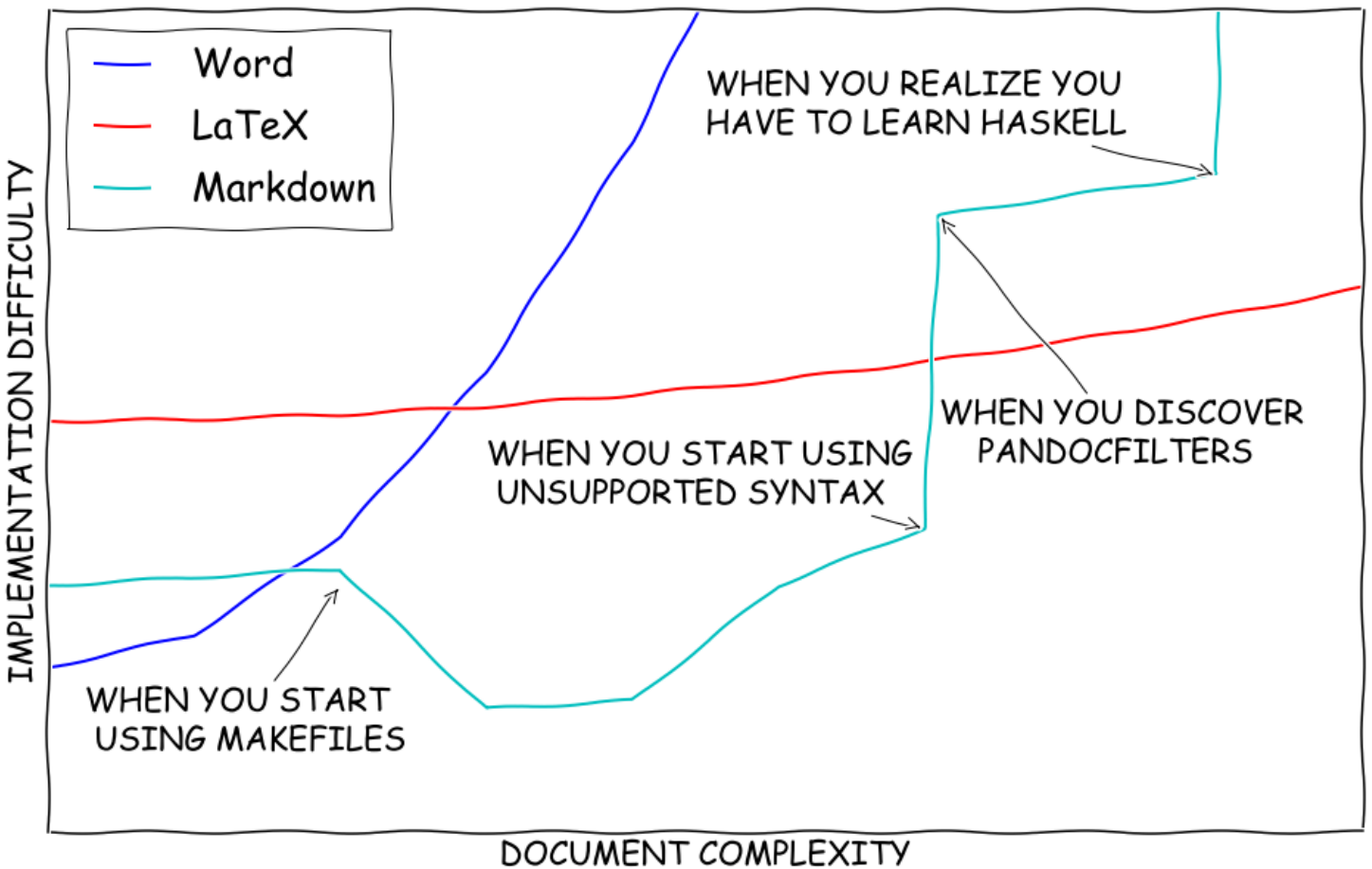


- Documents that incorporate the source code required for their production
- Notebook and plain text flavors
- Automation and reproducibility

https://quarto.org

quarto

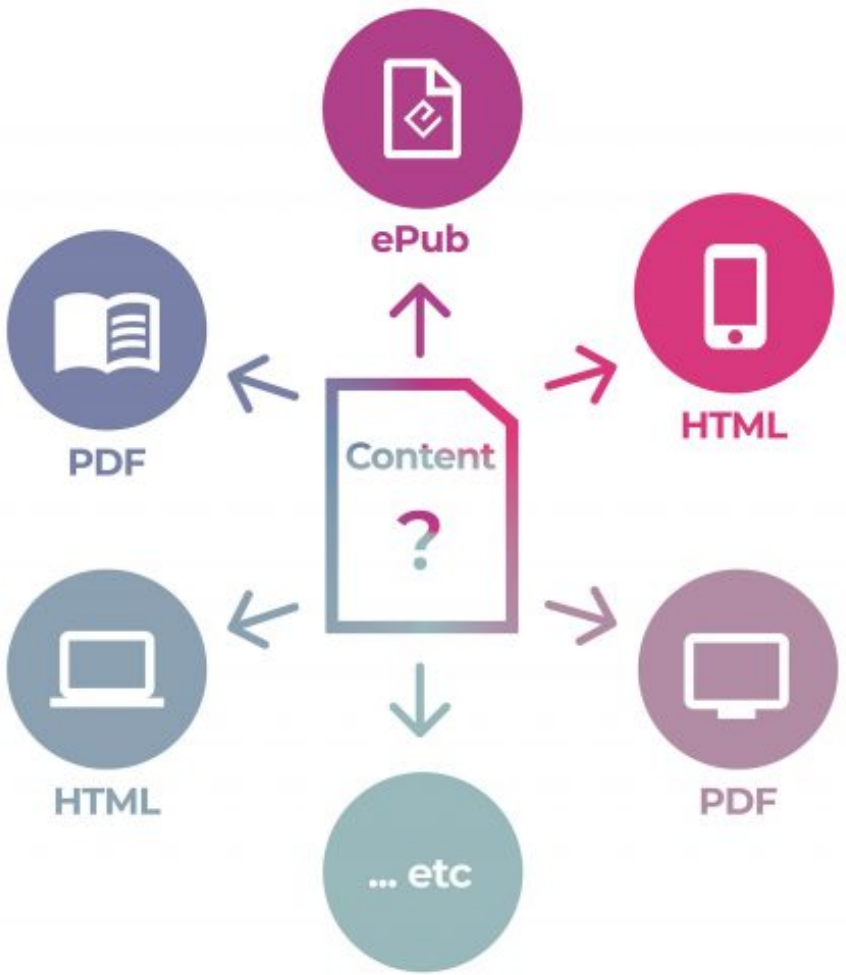
Goal: Scientific Markdown



quarto

Goal: Single Source Publishing

https://coko.foundation/articles/single-source-publishing.html



quarto

Simple Example

Markdown document with cross references and executable code blocks:

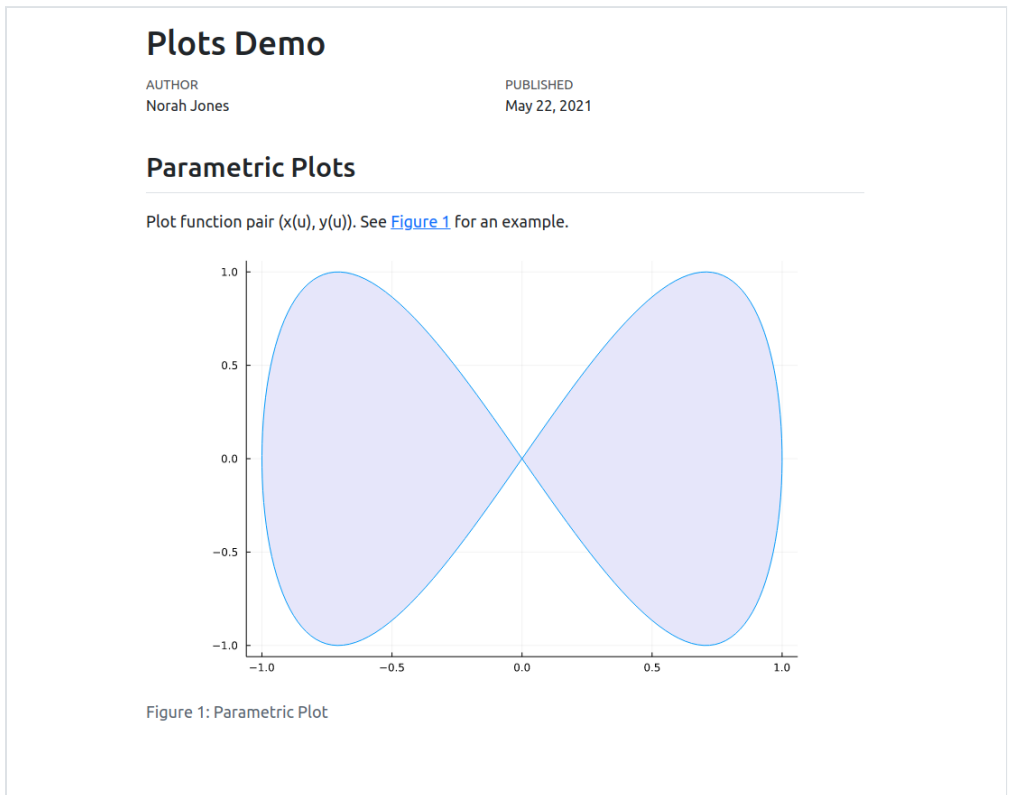
````
---
title: "Plots Demo"
author: "Norah Jones"
date: "5/22/2021"
jupyter: julia-1.7
---

## Parametric Plots

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

```{julia}
#| label: fig-parametric
#| fig-cap: "Parametric Plot"
#| echo: false

using Plots
plot(sin, x->sin(2x), 0, 2π, leg=false, fill=(0,:lavender))
```
````

Can be rendered to dozens of output formats (via Pandoc):

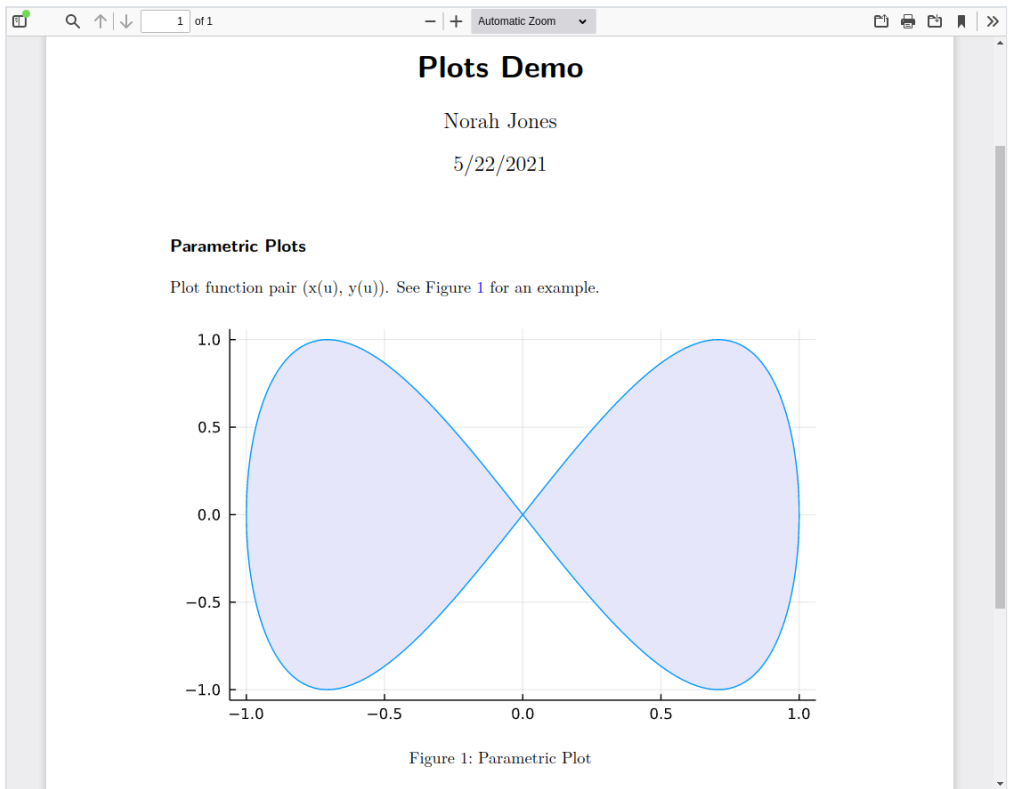
```
quarto render hello.qmd --to html
quarto render hello.qmd --to pdf
quarto render hello.qmd --to docx
quarto render hello.qmd --to pptx
```

Rendered Output

HTML



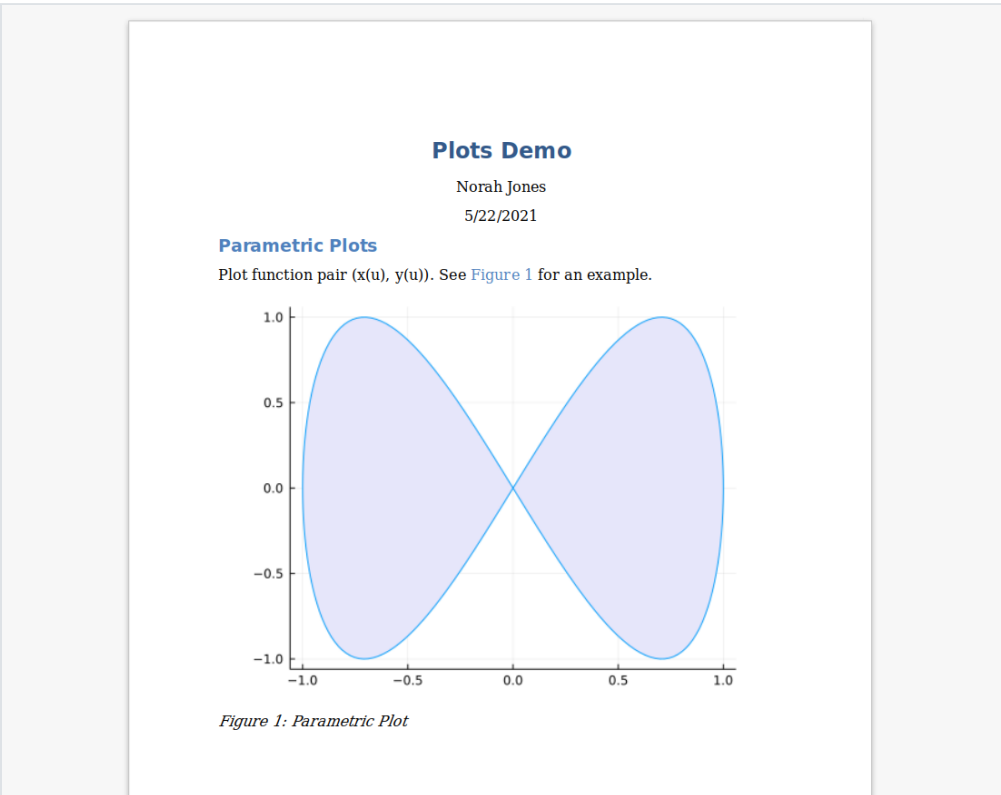
PDF



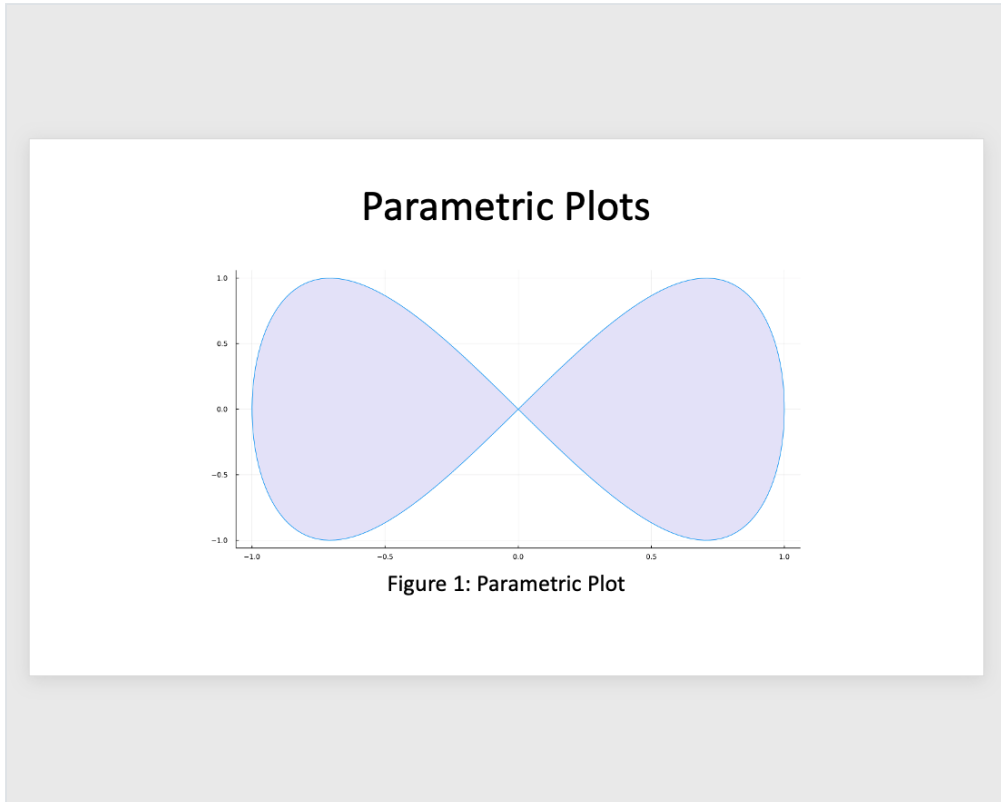
quarto

Rendered Output

MS Word



PowerPoint



quarto

Code Cells

Special syntax indicates that a code block is *executable*:

````
```{julia}
#| echo: false

using Plots
plot(sin, x->sin(2x), 0, 2π, leg=false, fill=(0,:lavender))
```
````

Code is executed and its output is included in the document.

Some useful options for code cells:

Option	Description
`echo`	Control whether source code is displayed.
`output`	Control whether output is displayed.
`warning`	Control whether warnings are displayed.
`code-fold`	Fold code using HTML `<details>` tag.
`layout`	Layout multiple plots side by side.

Rendering Pipeline

Plain text workflow (`.qmd` => `.ipynb` then execute cells):

Notebook workflow (no execution occurs by default):

Scientific Markdown

Brief History

- Org-Mode, Textile, AsciiDoc, and others pioneer plain text authoring.
- Markdown created by John Gruber and Aaron Swartz as an easier way to compose HTML
- Many flavors of Markdown emerge to build upon the core idea (MultiMarkdown, PHP Markdown Extra, GitHub Flavored Markdown, etc.)
- Pandoc created by John MacFarlane with deeper commitment to scholarly writing (e.g. citations) and publishing (LaTeX)

Citations

Pandoc includes robust support for citations and bibliographies in a wide variety of formats including BibTeX, CSL, and RIS.

Markdown Syntax	Output
`Blah Blah [see @knuth1984, pp. 33-35; also @wickham2015, chap. 1]`	Blah Blah (see Knuth 1984, 33–35; also Wickham 2015, chap. 1)
`Blah Blah [@knuth1984, pp. 33-35, 38-39 and passim]`	Blah Blah (Knuth 1984, 33–35, 38–39 and passim)
`Blah Blah [@wickham2015; @knuth1984].`	Blah Blah (Wickham 2015; Knuth 1984).
`Wickham says blah [-@wickham2015]`	Wickham says blah (2015)

More than 10,000 citation output styles supported via CSL (Citation Style Language):

```yaml
---
title: "My ARticle"
bibliography: references.bib
csl: nature.csl
---
```

Cross References

(a) Surus (b) Hanno

Figure 1: Famous Elephants

See fig. 1 for examples. In particular, fig. 1 (b).

Cross reference figures, tables, equations, sections, theorems, etc.

Markdown: https://quarto.org/docs/authoring/cross-references.html#subfigures

Julia cell: https://quarto.org/docs/computations/julia.html#code-blocks

Figure/Layout Panels

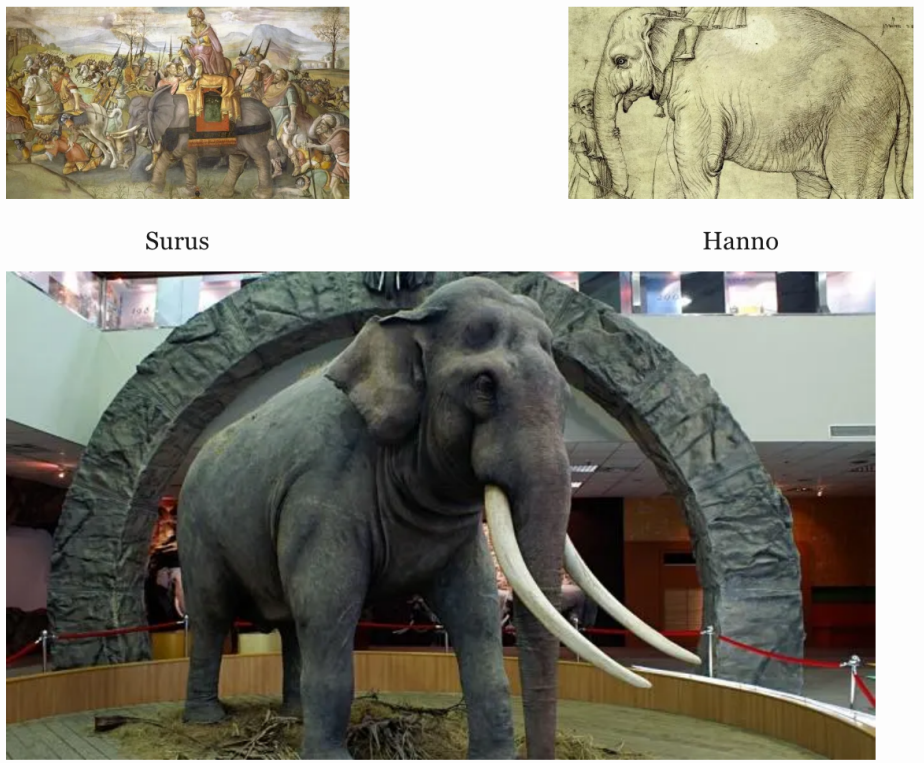
- Arbitrary layout of figures and tables:

```
::: {layout="[[40,-20,40], [100]]"}
![Surus](surus.png)
![Hanno](hanno.png)
![Lin Wang](lin-wang.png)
:::
```

- Shorthand syntax (`layout-ncol` or `layout-nrow`) available for simple cases:

```
::: {layout-ncol="2"}
Block 1

Block 2
:::
```



Also works for code chunks that produce figures and tables:
https://quarto.org/docs/authoring/figures.html#computations

Callouts

Work in HTML, PDF, MS Word, Revealjs, and ePub output

Note

Note that there are five types of callouts, including: `note`, `tip`, `warning`, `caution`, and `important`.

Tip With Caption

This is an example of a callout with a caption.

This is Important

Danger, callouts will really improve your writing.

Warning

Callouts provide a simple way to attract attention, for example, to this warning.

Expand To Learn About Collapse

This is an example of a 'collapsed' caution callout that can be expanded by the user. You can use `collapse="true"` to collapse it by default or `collapse="false"` to make a collapsible callout that is expanded by default.

Advanced Page Layout

Portable markdown syntax for advanced grid/column layout in HTML and PDF

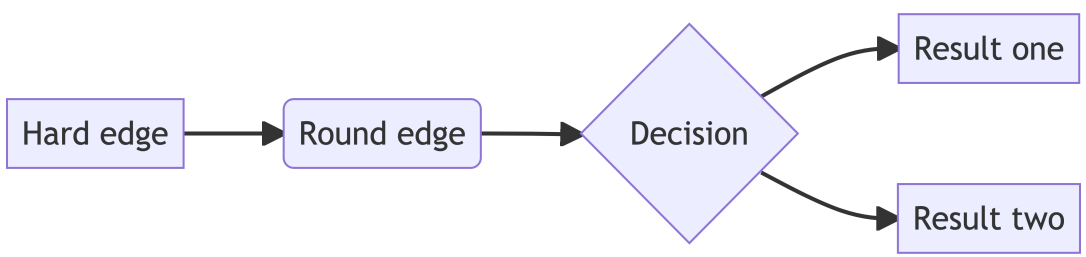
- Figures and tables that span the entire page
- Use of margin for figures, equations, captions, asides, footnotes, and citations
- Responsive show/hide of site navigation and TOC

Docs: https://quarto.org/docs/authoring/article-layout.html

Diagrams

Quarto has native support for embedding Mermaid and Graphviz diagrams.

````
```{mermaid}
flowchart LR
  A[Hard edge] --> B(Round edge)
  B --> C{Decision}
  C --> D[Result one]
  C --> E[Result two]
```
````



Output Formats

Documents

Pandoc supports a huge array of output formats, all of which can be used with Quarto.

- HTML
- PDF
- MS Word
- Open Office
- JATS (Journal Article Tag Suite)
- ConTeXt
- RTF
- AsciiDoc
- Markdown (gfm, hugo, etc.)

Example: GitHub Languages Analysis (PDF)

Presentations

Quarto supports a variety of formats for creating presentations, including:

- revealjs — reveal.js (HTML)
- pptx — PowerPoint (MS Office)
- beamer — Beamer (LaTeX/PDF)

Revealjs supports many advanced features including speaker notes, animations, custom backgrounds, and printing to PDF.

Example: JunctionTrees.jl: Efficient Bayesian Inference in Discrete Graphical Models

Websites

- Convenient way to publish groups of documents.
- Arbitrary content depth / organization
- Multi-level navigation (navbar / sidebar / hybrid)
- Full text search (client side or Algolia)

Example: Julia Workshop for Data Science

Example: Quarto Website

Books

- Inherit features of Quarto websites (navigation, search, etc.)
- Support cross references across chapters
- Formats include HTML, PDF, MS Word, and ePub

Example: Embrace Uncertainty: Mixed-effects models with Julia

Blogs

- Inherit features of Quarto websites (navigation, search, etc.)
- Automatically generated listing and RSS feed
- Customizable about page

Example: Patrick Altmeyer's Blog

Quarto and Julia

Prerequisites

Quarto executes Julia computations using its Jupyter engine (so Jupyter itself is a prereq):

```
pip3 install jupyter
```

The IJulia kernel is required, and Revise.jl recommended:

```
julia> using Pkg
julia> Pkg.add("IJulia")
julia> Pkg.add("Revise")
```

Basic Workflow

Rendering:

```
# plain text qmd
quarto render julia.qmd
quarto render julia.qmd --to pdf

## ipynb notebook
quarto render julia.ipynb
quarto render julia.ipynb --to docx
```

Preview:

```
# plain text qmd
quarto preview julia.qmd
quarto preview julia.qmd --to pdf

## ipynb notebook
quarto preview julia.ipynb
quarto preview julia.ipynb --to docx
```

1 - JupyterLab

localhost:8888/lab

File Edit View Run Kernel Tabs Settings Help

hello.ipynb

Code

Julia 1.7.2

Parametric Plots

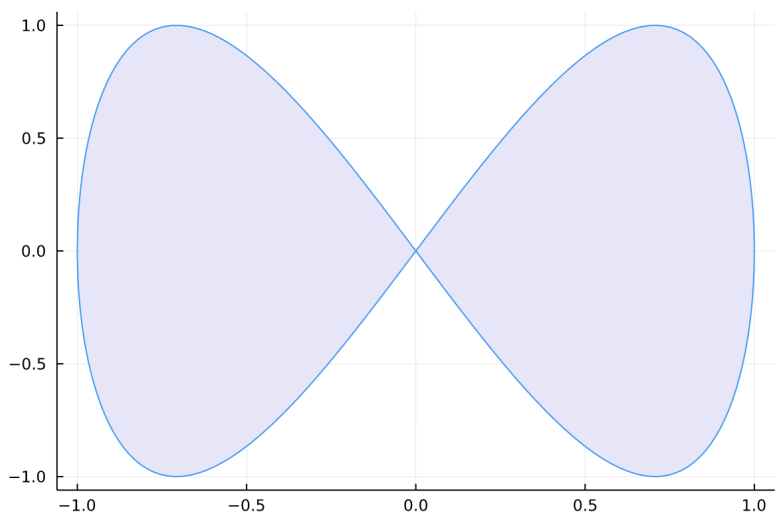
Plot function pair (x(u), y(u)). See @fig-parametric for an example.

```
[1]:
#| label: fig-parametric
#| fig-cap: "Parametric Plot"

using Plots

plot(
  sin,
  x->sin(2x),
  0, 2π,
  leg=false,
  fill=(0,:lavender)
)
```

[1]:



Terminal 1

```
  author: Norah Jones
  date: 5/22/2021

Output created: hello.html

Watching files for changes
```

Simple 1 1 Terminal 1

Plots Demo

localhost:5724

Plots Demo

AUTHOR
Norah Jones

PUBLISHED
May 22, 2021

Parametric Plots

Plot function pair (x(u), y(u)). See Figure 1 for an example.

▶ Code

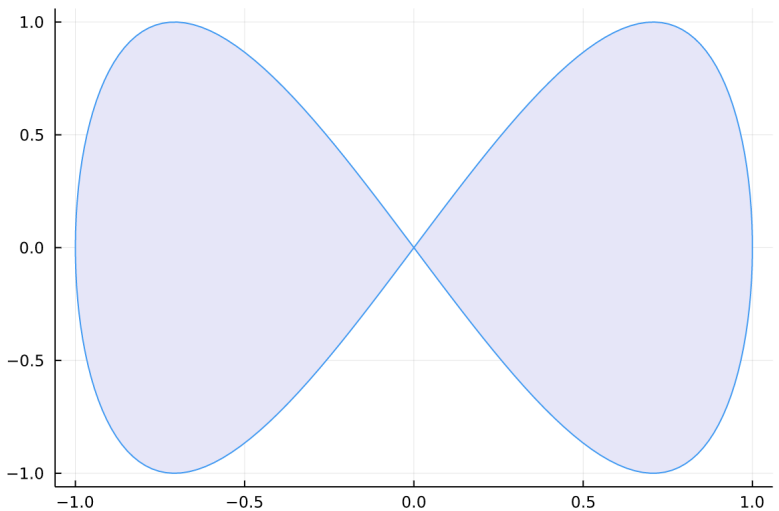


Figure 1: Parametric Plot

IJulia

- The IJulia Jupyter kernel is bound automatically when `{julia}` executable cells are present. A specific version of the kernel can be bound in document options:

```
jupyter: julia-1.7
```

- IJulia executes Julia code and transforms it to plain text, graphics, markdown, HTML, etc.
- For interactive sessions, Quarto keeps the Jupyter kernel resident as a daemon to mitigate long startup time.
- Revise.jl is (optionally) used to sync changes to dependent files/packages that occur while the daemon is running.

IJulia Figures

- Quarto provides default figure sizes and formats (e.g. PNG vs. SVG vs. PDF) that are format dependent (e.g. HTML, PDF, MS Word, PowerPoint, etc.)
- These are propagated to Julia in a setup cell that sets global defaults for some plotting packages (currently `Plots` and `CairoMakie`)
- Better approach would be if Julia plotting packages could automatically detect when they are running in Quarto and set their defaults accordingly

Performance Considerations

- **Startup performance** — How long does it take to load the interpreter + packages and how often do I need to do that?
- **Rendering performance** — How expensive are my computations and how frequently do I need to run them?

Startup Performance

- For interactive sessions, Jupyter kernels are kept alive to mitigate startup costs (`hello.qmd` example from earlier takes ~ 30 seconds on first run but less than 0.5 seconds on subsequent runs within the same session)
- This is convenient but creates the problem of stale code (e.g. referenced files are changed, packages are updated, etc.). The solution to this is Revise.jl, a library that helps you keep your Julia sessions running longer, reducing the need to restart when you make changes to code.

Revise.jl

Install Revise.jl:

```
julia> Pkg.add("Revise")
```

Configure to launch automatically within IJulia:

`.julia/config/startup_ijulia.jl`

```
try
  @eval using Revise
catch e
  @warn "Revise init" exception=(e, catch_backtrace())
end
```

Rendering Performance

1. Authoring in `.ipynb` enables you to control exactly when code execution occurs (and cache the results in the the notebook)
2. Jupyter Cache provides transient caching of cell outputs for a document (if any of the cells in the document change then all of the cells will be re-executed).
3. Quarto's Freeze feature enables you to permanently save and re-use computational outputs.

Why IJulia?

- Aligned with architectural investments we had already made for Jupyter (e.g. kernel daemonization, execution result caching)
- `IJulia.display` function provides support for MIME outputs from Julia results (including raw LaTeX/HTML)
- Ability to author in various popular notebook front ends (JupyterLab, VS Code, etc.)

Other literate programming systems (Pluto, Neptune, etc.) could certainly be integrated as an alternative to IJulia.

Tools: VS Code

https://marketplace.visualstudio.com/items?itemName=quarto.quarto

- Render command with integrated preview pane
- Syntax highlighting for markdown and embedded languages
- Completion for embedded languages (e.g. Python, R, Julia, LaTeX, etc.)
- Completion and diagnostics for project files and document/cell options
- Commands and key-bindings for running cells and selected line(s)
- Live preview for embedded mermaid and graphviz diagrams
- Assist panel for contextual help, image preview, and math preview
- Code snippets for common markdown constructs
- Code folding and document outline for navigation within documents
- Workspace symbol provider for navigation across project files

hello.qmd M

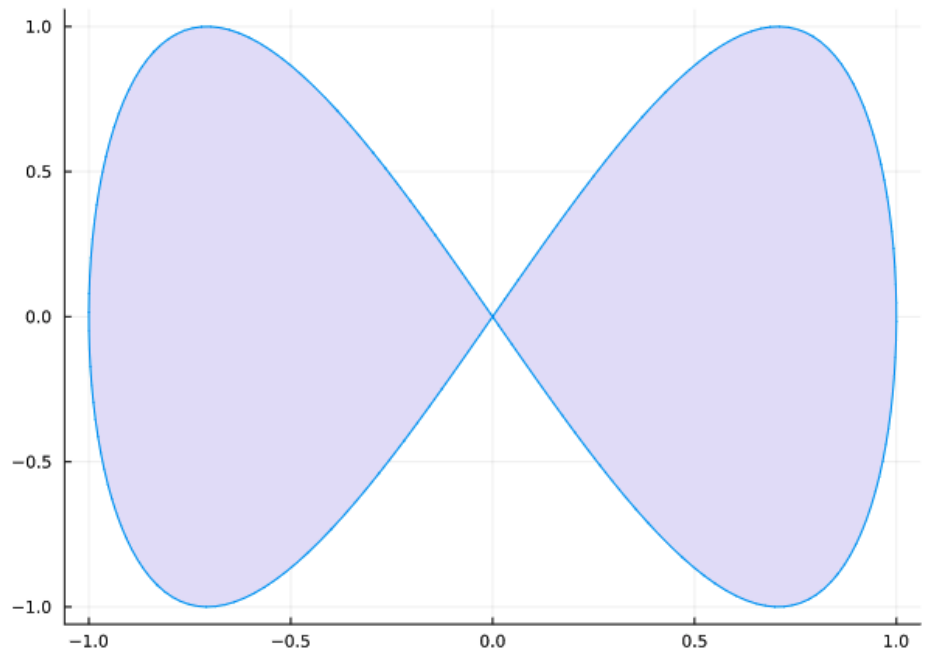
Render

Users > jjallaire > presentations > juliaconf-2022 > demo > hello.qmd > Parametric Plots > fig-p

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

## Parametric Plots

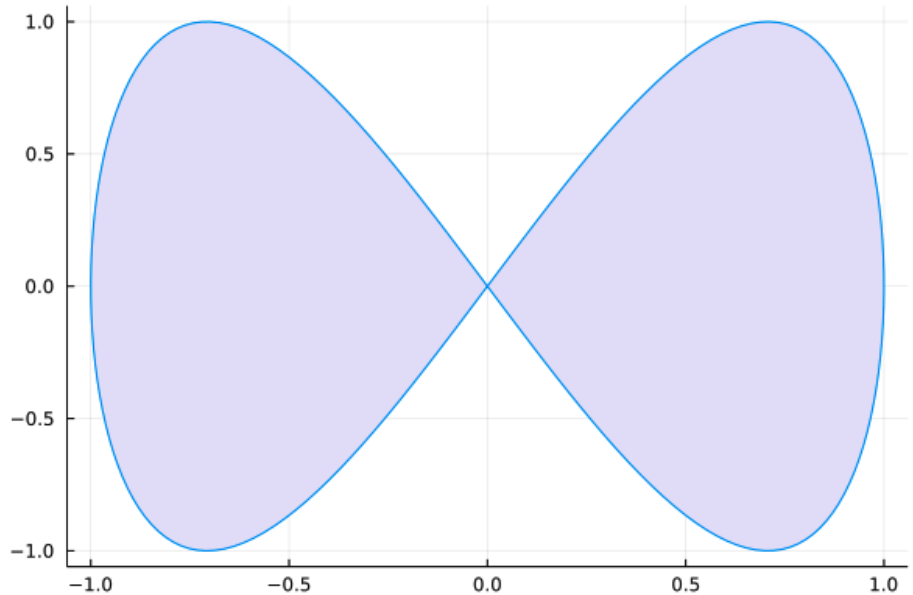
Plot function pair (x(u), y(u)). See @fig-parametric for an example.

▷ Run Cell
```{julia}
#| label: fig-parametric
#| fig-cap: "Parametric Plot"

using Plots

plot(
  sin,
  x->sin(2x),
  0, 2π,
  leg=false,
  fill=(0,:lavender)
)
```
```

Quarto Preview

http://localhost:5778/

Plots Demo

AUTHOR
Norah Jones

PUBLISHED
May 22, 2021

Parametric Plots

Plot function pair (x(u), y(u)). See Figure 1 for an example.

▶ Code



Figure 1: Parametric Plot

TERMINAL PROBLEMS OUTPUT DEBUG CONSOLE JUPYTER

Quarto Preview - demo

```
Output created: hello.html

Watching files for changes
```

hello.qmd M

Render

Quarto Preview

Julia Plots (2/2)

Users > jjallaire > presentations > juliaconf-2022 > demo > hello.qmd > Parametric Plots

```
---
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.

▷ Run Cell
```{julia}
#| label: fig-parametric
#| fig-cap: "Parametric Plot"

using Plots

plot(
  sin,
  x->sin(2x),
  0, 2π,
  leg=false,
  fill=(0,:lavender)
)
```
```



TERMINAL PROBLEMS OUTPUT DEBUG CONSOLE JUPYTER

```
julia> #| label: fig-parametric
       #| fig-cap: "Parametric Plot"

       using Plots

       plot(
         sin,
         x->sin(2x),
         0, 2π,
         leg=false,
         fill=(0,:lavender)
       )

julia>
```

Quarto Preview demo

Julia REPL jjallaire

hello.qmd M

Render

Quarto Preview

Julia Plots (2/2)

jjallaire > presentations > juliaconf-2022 > demo > hello.qmd > Parametric Plots > fig-parametric

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

## Parametric Plots

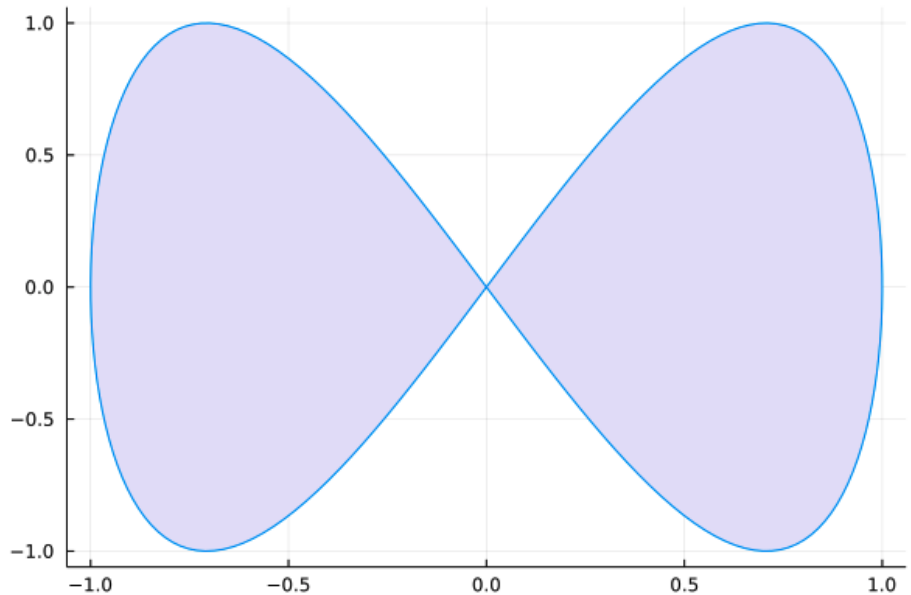
Plot function pair (x(u), y(u)). See @fig-parametric for an example.

▷ Run Cell
```{julia}
#| label: fig-parametric
#| fig-cap: "Parametric Plot"

using Plots

plot(
  si
  x-
  0,
  le
  fi
)
```
```

plot
plot!
plot3d
plot3d!
plot_color
plotarea
plotattr
plotly
plotlyjs
plots_heatmap
plots_heatmap!
Plots._add_plot_title!



TERMINAL PROBLEMS OUTPUT DEBUG CONSOLE JUPYTER

```
julia> #| label: fig-parametric
       #| fig-cap: "Parametric Plot"

       using Plots

       plot(
         sin,
         x->sin(2x),
         0, 2π,
         leg=false,
         fill=(0,:lavender)
       )

julia>
```

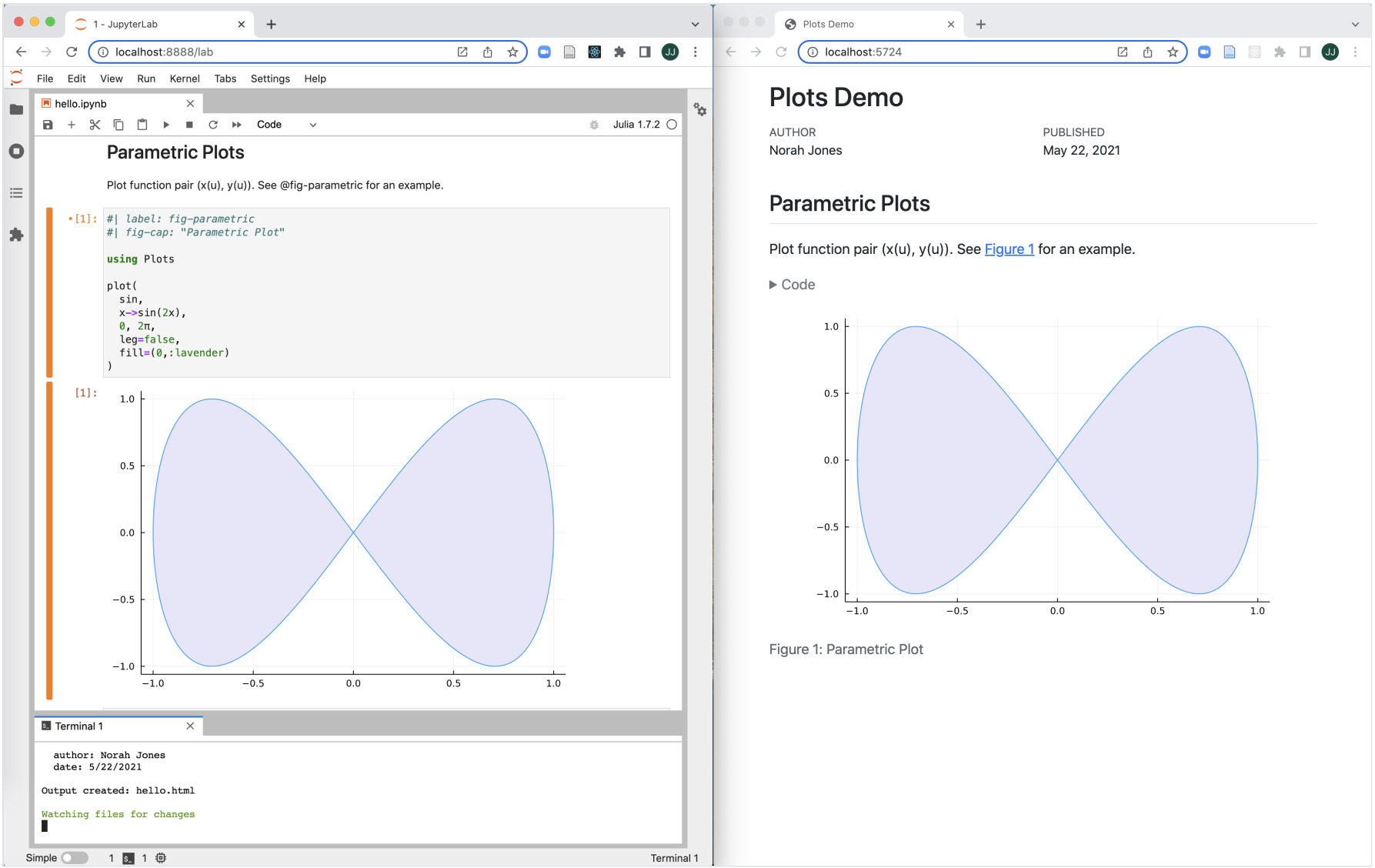
Quarto Preview demo

Julia REPL jjallaire

https://quarto.org

arto®

Tools: Jupyter Lab



quarto

Tools: Text Editors

- Use `quarto preview` with any text editor:

```
$ quarto preview julia.qmd
```

  Live reloading for documents and websites/books.

- Quarto modes/extensions for various editors:
  - Emacs: https://github.com/quarto-dev/quarto-emacs
  - Vim: https://github.com/quarto-dev/quarto-vim
  - Neovim: https://github.com/jmbuhr/quarto-nvim

Thank You!

- Slides: https://jjallaire.github.io/quarto-juliacon-2022/
- Learning More
  - Getting Started: https://quarto.org/docs/get-started/
  - Quarto and Julia: https://quarto.org/docs/computations/julia.html
  - About the Project: https://quarto.org/about.html
- Talk to us at https://github.com/quarto-dev/quarto-cli/discussions
- Questions?