le cnam

Conservatoire national des arts et métiers

ÉCOLE DOCTORALE SCIENCES DES MÉTIERS DE L'INGÉNIEUR

Laboratoire MACS, Cnam Paris

Thèse de doctorat

Présentée par : **Éric Bavu**
Soutenue le : **10 Octobre 1794**
Discipline : **60e section CNU — Mécanique, génie mécanique, génie civil**
Spécialité : **Acoustique**

USER GUIDE FOR THE QUARTO CNAM THESIS TEMPLATE

A self-documenting thesis document

Thèse dirigée par :
M. Henri Grégoire, Abbé constitutionnel, fondateur du Cnam, Paris

Jury		
M. Jean-Antoine Chaptal	Professeur des Universités, Chaire de Chimie Appliquée, Conservatoire national des arts et métiers, Paris	Président du jury
M. Donald Ervin Knuth	Professeur émérite, Department of Computer Science, Stanford University, Palo Alto, États-Unis	Rapporteur
Mme Ada Lovelace	Maîtresse de Conférences HDR, Department of Mathematics, University of London, Royaume-Uni	Rapportrice
M. Jacques de Vaucanson	Inspecteur général des manufactures, Académie royale des sciences, Paris	Évaluateur
M. Leslie Lamport	Senior Principal Researcher, Microsoft Research, Redmond, États-Unis	Évaluateur
M. Henri Grégoire	Abbé constitutionnel, fondateur du Cnam, Conservatoire national des arts et métiers, Paris	Directeur de thèse

To Donald Knuth, who transformed typography into a computational art form, and to all the graduate students who have wasted a day because of a missing LaTeX brace.

Acknowledgements

This work would not have been possible without the unwavering support of Donald Knuth's writings ; the *TEXbook* [1] has accompanied generations of doctoral students through long nights debugging their LaTeX code. I also thank Leslie Lamport for making TeX accessible to ordinary mortals with LaTeX [2], and the Quarto team — J.J. Allaire, Charles Teague, Carlos Scheidegger, Yihui Xie, and Charles Dervieux — for making reproducible scientific publishing a reality [3].

My thanks go to my virtual doctoral supervisor, Abbé Henri Grégoire, whose 1794 vision — bringing the arts and crafts within everyone's reach — resonates with the ideals of free software. I doubt he anticipated PDF/A-1b, but the spirit is there.

I thank the members of the virtual jury for their time and availability, in particular M. Vaucanson, whose expertise in mechanical automata proved more relevant than it might seem for understanding the pdflatex compilation pipeline and the magic behind Pandoc and Lua filters.

Finally, a heartfelt thought for all doctoral students at the Cnam and elsewhere who have opened a poorly constructed `.tex` template file for the first time and wondered why they had one too many left braces. This guide is for you: thanks to Quarto, writing becomes as simple as Markdown, yet with almost unlimited possibilities, and the elegance of generating the same document in multiple output formats through a modern approach.

Résumé

Ce travail présente et documente le template Quarto `quarto-cnam-thesis`, une extension permettant aux doctorants du Conservatoire national des arts et métiers (Cnam) de rédiger leur thèse en Quarto Markdown et de produire simultanément un PDF conforme à la maquette institutionnelle 2024-2025 (classe `book`, moteur pdflatex, conformité PDF/A-1b) et une version HTML navigable. La thèse est organisée en cinq chapitres documentant chacun une famille de fonctionnalités : structure et mise en forme, figures et mise en page, équations et tableaux, callouts et blocs de code, références et annotations collaboratives. La plupart des fonctionnalités est présentée selon le principe *code + rendu* : un bloc de code Markdown affiche la syntaxe exacte, immédiatement suivi du rendu réel dans le document. Cette approche auto-référentielle permet au document de servir à la fois de guide d'utilisation et de démonstration vivante du template.

Mots-clés : Quarto, LaTeX, pdflatex, PDF/A, thèse de doctorat, publication scientifique reproductible, Cnam

Abstract

This work presents and documents the `quarto-cnam-thesis` Quarto extension, which enables doctoral students at the Conservatoire national des arts et métiers (Cnam) to write their thesis in Quarto Markdown and simultaneously produce a PDF compliant with the 2024-2025 institutional template (book class, pdflatex engine, PDF/A-1b compliance) and a navigable HTML version. The thesis is organised in five chapters, each documenting a family of features: document structure and formatting, figures and layout, equations and tables, callouts and code blocks, and finally references, glossary, and collaborative annotations. Most features are presented following a *code + output* principle: a Markdown code block displays the exact syntax to use, immediately followed by the live rendering in the document. This self-referential approach allows the document to serve both as a user guide and a live demonstration of the template.

Keywords: Quarto, LaTeX, pdflatex, PDF/A, doctoral thesis, reproducible scientific publishing, Cnam

Contents

List of Tables

List of Figures

List of Acronyms

ABES Agence Bibliographique de l'Enseignement Supérieur

Cnam Conservatoire national des arts et métiers

CNU Conseil National des Universités

CSI Comité de Suivi Individuel

HDR Habilitation à Diriger des Recherches

LOF List of Figures

LOT List of Tables

QMD Quarto Markdown Document

TOC Table of Contents

YAML Yet Another Markup Language

Glossary

Callout Highlighted block designed to draw the reader's attention; produced by the ::: .callout-* syntax with five types: note, tip, warning, important, caution

Liminaire Set of pages placed before the main body of a thesis: acknowledgements, résumé, abstract, tables of contents, figures and tables, list of acronyms, glossary

Pandoc Universal document converter developed by John MacFarlane; technical foundation of Quarto

pdfx LaTeX package enabling PDF/A and PDF/X compliance; used in this template to produce a PDF/A-1b (option a-1b)

Postliminaire Set of pages placed after the main body of a thesis: general conclusion, bibliography, appendices

Quarto Open-source scientific publishing system based on Pandoc, developed by Posit PBC; produces PDF, HTML, EPUB, Word, slideshows, websites and books from QMD files

Introduction

Context and objectives

Doctoral students at the Conservatoire national des arts et métiers (Cnam) must submit their thesis as a PDF compliant with the institutional template, while also wanting a navigable HTML version for sharing and accessibility. This dual objective — archivable PDF/A and readable HTML — is precisely what the Quarto publishing system [3] offers, provided a template configured for the specific constraints of the Cnam is available.

The present document is both the *user guide* for the `quarto-cnam-thesis` template and its *live demonstration*: every feature described here is simultaneously explained, shown in raw syntax, and rendered before your eyes in this very document.

Structure of this guide

This guide is organised into five thematic chapters:

1. **Structure and formatting** — heading hierarchy, numbering, section cross-references, inline formatting, lists.
2. **Figures and layout** — static images, Python-generated figures, multi-column layout, PDF-only figures.
3. **Equations and tables** — mathematical syntax, simple Markdown tables, advanced tables with the `quartable` extension.
4. **Callouts and code blocks** — the five callout types, displayed code without execution, executed code with Python.
5. **References, glossary, and annotations** — bibliographic citations, acronym and term system, collaborative comments.

The guiding principle is constant: for each feature, a code block displays the exact Markdown syntax (not executed), immediately followed by the actual rendered output.

Prerequisites

The thesis source files are ordinary text files with the `.qmd` extension (Quarto Markdown). They can be opened and edited with any text editor — VS Code, RStudio, Neovim, Notepad++ or even the system notepad.

Quarto and LaTeX

This template requires **Quarto 1.4** or later. The compilation engine is pdflatex — XeLaTeX and LuaLaTeX are not supported due to constraints of the `pdfx` package used for PDF/A-1b compliance.

> **Step 1 — Install Quarto**
>
> https://quarto.org/docs/get-started/ — `.pkg`, `.msi` or `.deb` installer depending on the platform; step-by-step editor guide for VS Code, RStudio and more.

Step 2 — Install LaTeX. If you do not already have LaTeX, use TinyTeX, Quarto's built-in minimal distribution. It installs in seconds and automatically downloads any missing package on first render:

Code (bash)

```bash
quarto install tinytex
```

> **First render requires an internet connection**
>
> On the first render, TinyTeX downloads the packages required by the template (`pdfx`, `minitoc`, `tcolorbox`, `glossaries`...). This may take a few minutes. All subsequent renders run entirely offline.

Already have LaTeX? Any up-to-date **TeX Live 2023+** or **MiKTeX** installation works with no additional configuration. If your distribution is older, either update it or run `quarto install tinytex` to switch to TinyTeX.

System	Quarto	LaTeX
macOS	`.pkg` or `brew install quarto`	`quarto install tinytex` *(recommended)* or TeX Live / MacTeX
Windows	`.msi` (64-bit)	`quarto install tinytex` *(recommended)* or MiKTeX / TeX Live

System	Quarto	LaTeX
Linux (Debian/Ubuntu)	`sudo dpkg -i quarto-*.deb`	`quarto install tinytex` *(recommended)* or `sudo apt install texlive-full`

Executable code (Python, R. . .)

A scripting language is **only needed if the thesis contains executable code cells** — computed figures or tables generated directly in the document. A thesis written in pure text with static images requires **no scripting language at all**: the post-render script (`postrender.ts`) runs on Quarto's bundled Deno runtime.

Quarto supports several computation engines:

- **Python** (via Jupyter) — used in the examples of this guide. → Quarto documentation: Python computations
- **R** (via knitr) — well suited for statistics, econometrics, ecology and other disciplines where R is the dominant language. → Quarto documentation: R computations

The rest of this section covers **Python** setup, which powers the examples in this document. For R, refer to the Quarto documentation linked above.

The packages required by the examples in this document are listed in `requirements.txt` at the project root: `numpy`, `matplotlib`, `pandas`, `tabulate`.

Virtual environment — why it matters. A virtual environment isolates this project's packages from the rest of the system Python installation, avoids version conflicts, and facilitates reproducibility. Installing packages directly into the system Python is strongly discouraged.

uv — recommended

uv installs Python, creates the virtual environment, and manages packages in a single command, without requiring Python on the system beforehand. Quarto automatically detects the `.venv/` folder created — no manual activation is needed before `quarto render`.

Code (bash)

```bash
# Step 1 - install uv (once, on the system)
# macOS / Linux:
curl -LsSf https://astral.sh/uv/install.sh | sh
# Windows (PowerShell):
powershell -ExecutionPolicy ByPass -c "irm https://astral.sh/uv/install.ps1 | iex"
```

```

# Step 2 - in the thesis directory:
uv python install 3.12          # downloads and installs Python 3.12
uv venv                         # creates .venv/ in the current directory
uv pip install -r requirements.txt  # installs all packages + Jupyter
```

venv + pip — alternative

If Python 3.10+ is already installed on the system:

Code (bash)

```
python -m venv .venv
source .venv/bin/activate        # macOS / Linux
# .venv\Scripts\activate         # Windows (cmd)
# .venv\Scripts\Activate.ps1     # Windows (PowerShell)
pip install jupyter -r requirements.txt
```

Conda, pyenv...

Conda users can create a dedicated environment: `conda create -n mythesis python=3.12 && conda activate mythesis && pip install jupyter -r requirements.txt`. pyenv users install the target version with pyenv, then use `venv` as above.

Installing the template

Installing the template

Code (bash)

```
quarto use template zinc75/quarto-cnam-thesis
```

This command copies all source files into the current directory, including the `.qmd` files of this documentation, which serve as a starting point. The content files to modify are located in `content_fr/` (French profile) and `content_en/` (English profile), together with the `yaml` configuration files (see Annexe A).

Adapting the template to your thesis

Two files need to be edited before the first render.

`_quarto.yml` — thesis metadata: title, author, jury, defence date, discipline... This file is shared by both the PDF and HTML outputs. A complete reference is given in Annexe A.

_quarto-en.yml (or _quarto-fr.yml) — the list of chapters that make up the book. This is the first place to edit: replace the example chapters with your own .qmd files:

Code (yaml)

```yaml
book:
  chapters:
    - index.qmd                                    # do not remove
    - content_en/frontmatter/acknowledgements.qmd
    - content_en/frontmatter/resume.qmd
    - content_en/frontmatter/abstract.qmd
    - content_en/frontmatter/tables.qmd            # do not remove
    - content_en/chapters/00-introduction.qmd
    - content_en/chapters/01-my-chapter.qmd        # <- your chapters here
    - content_en/chapters/02-my-chapter.qmd
    - content_en/backmatter/conclusion.qmd
    - content_en/backmatter/bibliography.qmd
  appendices:
    - content_en/backmatter/appendices.qmd
    - content_en/backmatter/appendices-pdfa.qmd
```

To create a chapter: add a .qmd file in content_en/chapters/ and insert it into this list.

Rendering the document

In Quarto, this is called **rendering**: it triggers the generation of the .tex file, the compilation, and the creation of html files, since this template provides dual PDF and HTML output. To do this, run one of the following commands depending on the language in which you are writing the thesis (French or English, according to the rules of the doctoral schools and the composition of your jury):

Always use a language profile

Code (bash)

```bash
quarto render --profile fr   # → _these_fr/these_fr_<author>.pdf + HTML
quarto render --profile en   # → _thesis-en/these_en_<author>.pdf + HTML
```

quarto render without a profile fails: the chapter list is defined in the profiles _quarto-fr.yml and _quarto-en.yml, not in _quarto.yml.

Live preview

During writing, there is no need to rerun quarto render after every change. The quarto preview command watches the source files and automatically rebuilds the document whenever a

`.qmd` file is saved:

Code (bash)

```bash
quarto preview --profile en
```

Quarto then opens the HTML version in the browser and refreshes it within seconds on each save. Only the modified chapter is recompiled (incremental rebuild), which makes the edit–preview cycle very fast — even with Python code blocks, thanks to the `freeze: auto` cache.

Recommended workflow during writing

Use `quarto preview` daily to check formatting, cross-references, and figures. Run `quarto render` periodically (before a supervision meeting, before submission) to obtain the final PDF. Both outputs — HTML and PDF — are consistent: what renders correctly in HTML compiles correctly in PDF.
For **VS Code** or **RStudio** users, the Quarto extension integrates a *Render* button and a preview panel directly in the editor — without using the terminal.

CHAPTER

1

Document structure and formatting

Contenu

This chapter presents the basic building blocks of any Quarto document: heading hierarchy, numbering, section cross-references, and inline formatting.

1.1 Heading hierarchy

Quarto supports four levels of numbered headings in this template (parameter `number-depth: 4` in `_extension.yml`). The syntax uses Markdown hashes:

Code (`markdown`)

```
# Chapter heading (level 1)

## Section heading (level 2)

### Subsection heading (level 3)
```

```
6
7 #### Sub-subsection heading (level 4)
```

In PDF, these levels correspond respectively to the LaTeX commands `\chapter`, `\section`, `\subsection`, and `\subsubsection`. In HTML, they generate the tags `<h2>` through `<h5>` (with `<h1>` reserved for the book title).

1.1.1 Demonstration subsection

This is a `###` in context. The automatic numbering produces "1.1.1" here.

1.1.1.1 Demonstration sub-subsection

And this is a `####`, numbered "1.1.1.1". Beyond this level, Quarto stops numbering — use lists or bold paragraphs if you need a fifth level.

> ⚠ Recommended maximum depth
>
> A four-level numbered outline is generally the maximum acceptable in a thesis. A deeper outline is often a sign that the structure needs rethinking.

1.2 Numbered and unnumbered sections

By default, all headings from level 1 to 4 are numbered. To remove the numbering, add the class `{.unnumbered}`:

Code (markdown)

```
1 ## Section without number {.unnumbered}
```

For front matter and back matter pages (acknowledgements, abstract, conclusion. . .), combine `{.unnumbered}` with `{.toc-black}` — the latter class instructs the Lua filter to display the title in black in the table of contents, and triggers roman numeral page numbering in the front matter when `pagestyle-sections: true`:

Code (markdown)

```
1 # Conclusion {.unnumbered}
2
3 # Acknowledgements {.unnumbered .toc-black}
```

! `.unnumbered` vs `.unnumbered .toc-black`

- `{.unnumbered}` alone: unnumbered heading, displayed in colour in the TOC (chapters in the thesis body: introduction, conclusion).
- `{.unnumbered .toc-black}`: unnumbered heading, displayed in **black** in the TOC (front matter and back matter sections).

Never put `{.unnumbered}` on a numbered body chapter (chapters 1, 2, 3...) — it would be inconsistent with the overall structure.

1.3 Section cross-references (@sec-)

To create a cross-reference to a section, you need to: (1) give an identifier to the target heading with `{#sec-label}`, then (2) reference it with `@sec-label`:

Code (markdown)

```
## Heading hierarchy {#sec-headings}

As explained in @sec-headings, levels range from `#` to `####`.
```

Output:

As explained in Section 1.1, levels range from # to ####.

Quarto automatically resolves the number and hyperlink in both formats.

Label naming convention

Use the prefix `sec-` for sections, `fig-` for figures, `tbl-` for tables, `eq-` for equations. These prefixes are required by Quarto for cross-references to work correctly in both output formats.

1.4 Short title (`short-title`)

When a chapter or section title is too long to fit in the PDF running header or in the table of contents, use the `short-title` attribute:

Code (markdown)

```
# A really very long title that overflows into the header {short-title="Short title"}

## A subsection with an extremely long title {short-title="Short title"}
```

The short title is used in the running header (recto/verso page header) and in the table of contents. The long title remains unchanged in the body of the text.

Astuce

Aim for 40 to 50 characters maximum for a `short-title`. Beyond that, it may also overflow depending on the layout.

1.5 Inline formatting

1.5.1 Emphasis and code

Code (markdown)

```
**bold**, *italic*, ***bold italic***, `inline code`
~~strikethrough~~, ^superscript^, ~subscript~
:::
```

Output:

bold, *italic*, ***bold italic***, `inline code`, ~~strikethrough~~, [superscript], [subscript].

Academic use of bold and italic

In a thesis, bold and italic serve to mark **terminology** at first introduction, or to cite a *book title*. Using them to decorate text or emphasise entire sentences is a practice to avoid.

1.5.2 Underline

Markdown has no native syntax for underlining. Quarto uses the span notation with the `.underline` class:

Code (markdown)

```
[underlined text]{.underline}
```

Output:

underlined text.

In PDF, Pandoc generates `\underline{...}`; in HTML, a `<span>` with `text-decoration: underline`. The `<u>text</u>` syntax works in HTML but is ignored in PDF.

> **Underlining should be avoided in a thesis**
>
> Underlining is a legacy of the typewriter era, when italic was not available. In a typeset document, it is reserved for hyperlinks. To highlight a term, use `*italic*` or `**bold**`.

1.5.3 Lists

Code (markdown)

```markdown
- First unordered item
- Second item
  - Sub-item (2-space indentation)

1. First ordered item
1. Second item
   a. Sub-item (3-space indentation)
```

Output:

- First unordered item
- Second item
  - Sub-item (2-space indentation)

1. First ordered item
2. Second item
   a. Sub-item (3-space indentation)

The written number does not matter

Pandoc only reads the number of the **first** item (to determine the counter start); all subsequent numbers are ignored and auto-incremented. These two syntaxes produce the same output:

Code (markdown)

```
1. First
2. Second
3. Third
```

Code (markdown)

```
1. First
1. Second
1. Third
```

The `1. 1. 1.` convention is even recommended: no need to renumber if an item is inserted or moved. The same logic applies to sub-lists: `a. a. a.` is sufficient to obtain `a.`, `b.`, `c.`...

1.5.4 Task lists

The `- [ ]` / `- [x]` syntax produces a task list. The case of `x` does not matter (`X` works too):

Code (markdown)

```
- [ ] Write the introduction
- [x] Choose the Quarto template
- [ ] Submit to the thesis supervisor
```

Output:

- ☐ Write the introduction
- ☒ Choose the Quarto template
- ☐ Submit to the thesis supervisor

Different rendering depending on the output format

In **HTML**, Quarto generates real `<input type="checkbox">` elements (disabled, non-interactive).
In **PDF** (pdflatex), Pandoc uses the mathematical symbols `$\square$` (empty box) and `$\boxtimes$` (checked box) from the `amssymb` package. The appearance is correct but typographically different from HTML checkboxes.

Use in a thesis

Task lists are suited to working notes, methodological appendices, or a README. In the thesis body, prefer a plain unordered list accompanied by a state verb (“completed”, “in progress”…): it is more neutral and more consistent with academic register.

1.5.5 Blockquotes

Code (markdown)

```
> "Typography is the craft of endowing human language with a durable visual form."
>
> - Robert Bringhurst [@Bringhurst2004]
```

Output:

> “Typography is the craft of endowing human language with a durable visual form.”
>
> — Robert Bringhurst [4]

Definition lists in pdflatex

Pandoc’s definition list syntax (`term\n:  definition`) is supported by Quarto but produces no particular visual style in pdflatex. If you need formatted definition lists, use a Markdown table (see Section 3.4) or a raw LaTeX environment.

1.5.6 Footnotes

Quarto offers two syntaxes for footnotes.

Inline syntax — the note is written directly in the text, between `^[` and `]`:

Code (markdown)

```
The Cnam was founded in 1794^[Decree of the National Convention of 19 Vendémiaire
Year III (10 October 1794).] on the initiative of Abbé Grégoire.
```

Output:

The Cnam was founded in 1794[1] on the initiative of Abbé Grégoire.

Reference syntax — the footnote call and its content are separated. Useful for long notes or to keep the main text readable:

Code (markdown)

```
The thesis must be deposited on theses.fr[^deposit] before the defence.

[^deposit]: National thesis portal managed by the ,
    accessible at <https://theses.fr>.
```

Output:

The thesis must be deposited on theses.fr[2] before the defence.

In PDF, both syntaxes produce a numbered footnote at the bottom of the page. In HTML, they produce an interactive footnote (clickable number with a back-reference to the call site).

Which syntax to choose?

Prefer the **inline syntax** `^[...]` for short notes (one sentence) — the content remains visible while writing. Use the **reference syntax** `[^label]` for long or multi-paragraph notes, and to avoid visually cluttering the Markdown source when the note exceeds two lines.

1.5.7 Page breaks

Quarto provides the shortcode to insert a page break compatible with both output formats:

Code (markdown)

```
End of this section's content.

{{< pagebreak >}}

Start of the next section.
```

In PDF, generates `\newpage`. In HTML, it inserts a `page-break-after: always` style rule — with no visible effect on screen, but useful if the HTML page is printed from the browser.

[1] Decree of the National Convention of 19 Vendémiaire Year III (10 October 1794).
[2] National thesis portal managed by the ABES, accessible at https://theses.fr.

\newpage vs \clearpage

If you need \clearpage (flushes pending floats before jumping) rather than \newpage, use a raw LaTeX block inside a {.content-visible when-format="pdf"}:

Code (markdown)

```
::: {.content-visible when-format="pdf"}
```{=latex}
\clearpage
```
:::
```

In a thesis with many figures, \clearpage is often preferable to prevent floats from accumulating and appearing well after their insertion point in the text.

Manual page breaks should be used sparingly

As a general rule, let LaTeX manage the layout. Manual page breaks solve a local problem but can create others further on in the document (blank pages, displaced floats). Prefer to adjust them at the very end of writing, once the content is fixed.

CHAPTER

2

Figures and layout

Contenu

This chapter documents the insertion of static figures, the generation of figures from Python code, multi-column layout, and the special case of PDF-only figures (TikZ, PGFplots).

2.1 Static image with caption and cross-reference

The basic syntax for inserting an image is:

Code (markdown)

```
1 ![Figure caption.](../../../images/my-image.png){#fig-label width=80%}
```

The identifier `#fig-label` allows a cross-reference with `@fig-label`. The `width` attribute accepts percentages or absolute values (`5cm`, `0.5\linewidth`).

Here is an example with the Cnam logo:

Code (markdown)

```
1 ![The official Cnam logo.](../../../images/Cnam.jpg){#fig-logo-cnam width=40%}
```

Figure 2.1 shows the official Cnam logo, loaded from the `images/` folder at the project root.

Figure 2.1: The official Cnam logo.

Recommended image formats

- **PDF**: prefer vector formats `.pdf` or `.eps` for diagrams and charts (sharp at all resolutions). For photographs, `.png` or `.jpg` at 300 dpi minimum.
- **HTML**: all standard web formats (`.png`, `.jpg`, `.svg`). `.svg` files are particularly well suited for diagrams.
- Quarto automatically handles format conversion depending on the render target.

Paths are relative to the `.qmd` file, not to the project root

Image paths are relative to the location of the `.qmd` file. From `content_en/chapters/`, the `images/` folder at the project root is at `../../images/myfile.png`. This behaviour is identical in PDF and HTML.

2.2 Python-generated figures

Quarto can execute Python cells (via Jupyter) and include their graphical output as numbered figures. The chunk syntax uses options prefixed with `#|`:

Code (markdown)

````
```{python}
#| label: fig-sincos
#| fig-cap: "Curves $\\sin(x)$ and $\\cos(x)$ over $[0, 2\\pi]$."
#| echo: true
import numpy as np
import matplotlib.pyplot as plt

x = np.linspace(0, 2 * np.pi, 300)
fig, axes = plt.subplots(1, 2, figsize=(8, 3))
axes[0].plot(x, np.sin(x), color='#d20025')
axes[0].set_title(r'$\sin(x)$')
axes[1].plot(x, np.cos(x), color='#005EA5')
axes[1].set_title(r'$\cos(x)$')
for ax in axes:
    ax.set_xlabel('$x$')
    ax.grid(True, alpha=0.3)
plt.tight_layout()
```
````

Output:

Code (python)

```python
import numpy as np
import matplotlib.pyplot as plt

x = np.linspace(0, 2 * np.pi, 300)
fig, axes = plt.subplots(1, 2, figsize=(8, 3))
axes[0].plot(x, np.sin(x), color='#d20025')
axes[0].set_title(r'$\sin(x)$')
axes[1].plot(x, np.cos(x), color='#005EA5')
axes[1].set_title(r'$\cos(x)$')
for ax in axes:
    ax.set_xlabel('$x$')
    ax.grid(True, alpha=0.3)
plt.tight_layout()
```

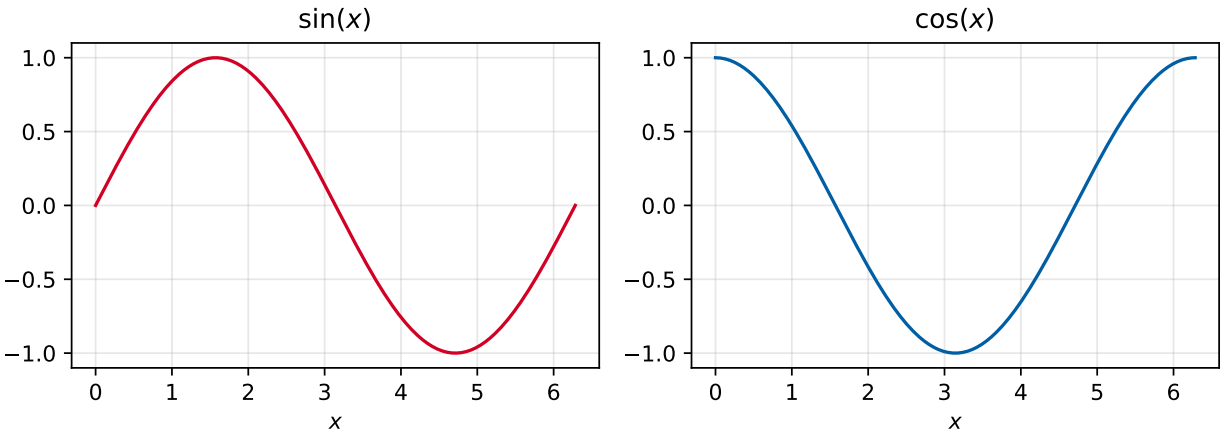


Figure 2.2: Curves $\sin(x)$ and $\cos(x)$ over $[0, 2\pi]$.

Figure 2.2 is generated at each compilation. With `execute: freeze: auto` in `_quarto.yml`, Quarto caches the results in `_freeze/` and only re-executes the code if the chunk has changed — useful for figures that take a long time to compute.

`plt.tight_layout()` is essential

Without `plt.tight_layout()`, axis labels frequently overlap in the PDF. Always end your matplotlib figures with this line.

`echo: false` for the final document

During writing, `echo: true` displays the code — convenient for review. For the version submitted to the doctoral school, switch figure chunks to `#| echo: false` to display only the graphic.

2.3 Multi-column layout

The `layout-ncol` directive places several figures side by side. Apply it to a div containing sub-figures:

Code (markdown)

````
::: {#fig-duo layout-ncol=2}

```{python}
#| label: fig-duo-a
#| fig-cap: "Function $x^2$"
import matplotlib.pyplot as plt, numpy as np
x = np.linspace(-2, 2, 200)
plt.plot(x, x**2, color='#d20025'); plt.grid(True, alpha=0.3); plt.tight_layout()
```

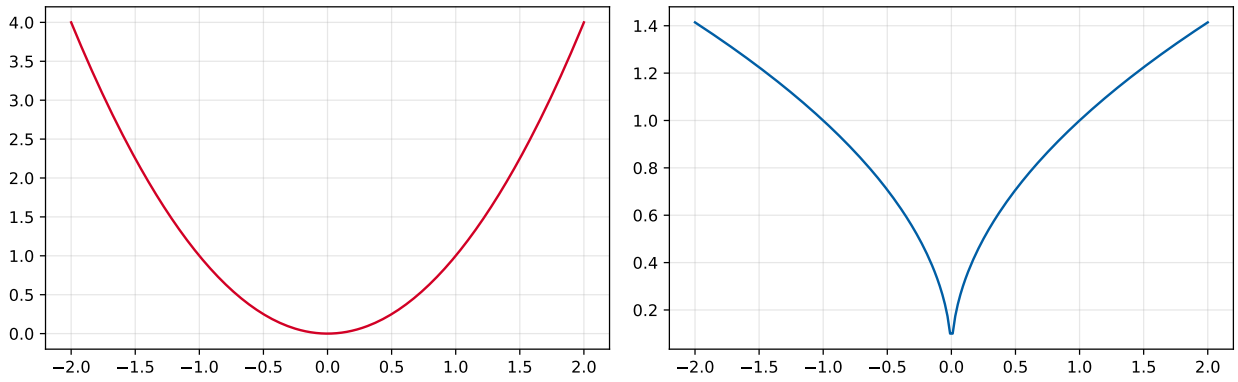
```{python}
#| label: fig-duo-b
#| fig-cap: "Function $\\sqrt{|x|}$"
import matplotlib.pyplot as plt, numpy as np
x = np.linspace(-2, 2, 200)
plt.plot(x, np.sqrt(np.abs(x)), color='#005EA5'); plt.grid(True, alpha=0.3);
↪  plt.tight_layout()
```

Two elementary functions.
:::
````

Output:

Code

```
import matplotlib.pyplot as plt, numpy as np
x = np.linspace(-2, 2, 200)
plt.plot(x, x**2, color='#d20025'); plt.grid(True, alpha=0.3);
↪  plt.tight_layout()
import matplotlib.pyplot as plt, numpy as np
x = np.linspace(-2, 2, 200)
plt.plot(x, np.sqrt(np.abs(x)), color='#005EA5'); plt.grid(True,
↪  alpha=0.3); plt.tight_layout()
```



(a) Function x^2

(b) Function $\sqrt{|x|}$

Figure 2.3: Two elementary functions.

Figure 2.3 groups the two sub-figures Figure 2.3a and Figure 2.3b. The last line of text in the div becomes the overall caption.

2.4 PDF-only figures (TikZ, PGFplots)

Some complex figures can only be generated in LaTeX (TikZ, PGFplots, `circuitikz`...). They are inserted via a `{=latex}` block inside a figure div, accompanied by replacement content visible in HTML:

Code (markdown)

````
:::{#fig-tikz fig-pos='h'}
```{=latex}
\centering
\begin{tikzpicture}
  \draw[thick, ->] (0,0) -- (3,0) node[right] {$x$};
  \draw[thick, ->] (0,0) -- (0,2) node[above] {$y$};
  \draw[blue, thick] plot[domain=0:2.8, samples=50] (\x, {sin(\x r)});
\end{tikzpicture}
```

::: {.content-visible when-format="html"}
*(TikZ figure - available in the PDF version only)*
:::

Example TikZ figure inserted as raw LaTeX.
:::
````

Output:

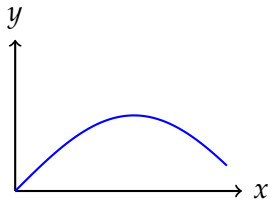


Figure 2.4: Example TikZ figure inserted as raw LaTeX.

Alternative: compile the TikZ figure as a separate PDF

For complex TikZ figures, a more robust approach is to compile them as a standalone `.pdf` file, then include them as a regular image:

Code (markdown)

```
1 ![Caption.](figures/my-tikz-diagram.pdf){#fig-tikz width=80%}
```

This avoids package conflicts and speeds up compilation.

2.5 Float positioning (PDF)

In LaTeX, figures and tables are *floats*: LaTeX decides their position to achieve the best possible layout. The `fig-pos` attribute (figures) or `tbl-pos` (tables) allows you to give it instructions.

Code (markdown)

```
1 ![Caption.](../../../images/Cnam.jpg){#fig-ex fig-pos='tb'}
```

Specifiers can be freely combined in a string:

Specifier	Meaning
`h`	*here* — at the exact point in the text (if space allows)
`t`	*top* — at the top of the current or next page
`b`	*bottom* — at the bottom of the current or next page
`p`	*page* — on a page dedicated to floats
`!`	Ignores LaTeX's internal constraints (density, float count)
`H`	Forces placement *here*, without any floating (requires `\usepackage{float}`)

The default value in Quarto is `tbp`. For a Python chunk, the specifier is passed as a cell option:

Code (markdown)

````
```{python}
#| label: fig-python
#| fig-cap: "My figure."
#| fig-pos: 'h'
```
````

To set a default specifier for the entire document, add to `_quarto.yml`:

Code (yaml)

```yaml
fig-pos: 'H'
```

h is not a guarantee

The `h` specifier asks LaTeX to place the float *here if possible*, but LaTeX may ignore it if there is insufficient space. The combination `ht` (here, then top of page) is generally more robust. `H` forces placement unconditionally, but can create nearly empty pages if the figure is large — reserve it for cases where position is truly critical.

Floats only apply to PDF

In HTML, `fig-pos` and `tbl-pos` are ignored: figures and tables are inserted in the document flow at their insertion point, like any other element.

CHAPTER

3

Equations and tables

Contenu

This chapter covers mathematical syntax (inline equations, numbered block equations, advanced environments) and two levels of tables: simple Markdown pipe tables and enriched tables via the `quartable` extension.

3.1 Inline equations

Inline equations are written between dollar signs: `$...$`. In PDF they use the native LaTeX math mode; in HTML they are rendered by KaTeX.

Code (markdown)

```
The Fourier transform of a function $f$ is defined by
$\hat{f}(\xi) = \int_{-\infty}^{\infty} f(x)\,e^{-2\pi i x\xi}\,\mathrm{d}x$.
```

Output:

The Fourier transform of a function f is defined by $\hat{f}(\xi) = \int_{-\infty}^{\infty} f(x)\, e^{-2\pi i x \xi}\,\mathrm{d}x$.

Thin spaces in mathematics

Use `\,` for thin spaces: `f(x)\,\mathrm{d}x` gives $f(x)\,\mathrm{d}x$. Use `\mathrm{d}` (roman) rather than `d` (italic) for differentials — this is the ISO 80000-2 standard.

3.2 Numbered block equations

Block equations are delimited by `$$...$$`. To number them and create a cross-reference, add `{#eq-label}`:

Code (markdown)

```
$$
i\hbar\frac{\partial \Psi}{\partial t} = \hat{H}\Psi
$$ {#eq-schrodinger}

The Schrödinger equation (@eq-schrodinger) describes the time evolution...
```

Output:

$$i\hbar\frac{\partial \Psi}{\partial t} = \hat{H}\Psi \tag{3.1}$$

The Schrödinger equation (Équation 3.1) describes the time evolution of a quantum system.

`@eq-label` without brackets

An equation cross-reference is written `@eq-label`, **without brackets**. Brackets `[@eq-label]` are reserved for bibliographic citations and will not work for equations.

3.3 Advanced mathematical environments

LaTeX environments such as `aligned`, `cases`, `bmatrix` work natively in Quarto, both in PDF (pure LaTeX) and in HTML (KaTeX):

Code (markdown)

```
$$
\begin{cases}
  \dot{x}(t) = Ax(t) + Bu(t) \\
```

```
  y(t) = Cx(t) + Du(t)
\end{cases}
$$ {#eq-statespace}
```

Output:

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t) \\ y(t) = Cx(t) + Du(t) \end{cases} \tag{3.2}$$

The state-space representation (Équation 3.2) is the standard form in control theory.

Code (markdown)

```
$$
\begin{aligned}
  \nabla \cdot \mathbf{E}  &= \frac{\rho}{\varepsilon_0} \\
  \nabla \cdot \mathbf{B}  &= 0 \\
  \nabla \times \mathbf{E} &= -\frac{\partial \mathbf{B}}{\partial t} \\
  \nabla \times \mathbf{B} &= \mu_0\mathbf{J} + \mu_0\varepsilon_0\frac{\partial
↪  \mathbf{E}}{\partial t}
\end{aligned}
$$ {#eq-maxwell}
```

Output:

$$\begin{aligned} \nabla \cdot \mathbf{E} &= \frac{\rho}{\varepsilon_0} \\ \nabla \cdot \mathbf{B} &= 0 \\ \nabla \times \mathbf{E} &= -\frac{\partial \mathbf{B}}{\partial t} \\ \nabla \times \mathbf{B} &= \mu_0\mathbf{J} + \mu_0\varepsilon_0\frac{\partial \mathbf{E}}{\partial t} \end{aligned} \tag{3.3}$$

Maxwell's equations (Équation 3.3) in vector notation — a single number for the entire block.

To assign **one number per line**, use `align` (without the final `d`) instead of `aligned`. The `align` environment is self-contained — it does not nest inside `$$...$$`. Pandoc wraps every `$$...$$` in `\[...\]` or `\begin{equation}`, and nesting `\begin{align}` inside is invalid (amsmath error). A raw `{=latex}` block is needed for PDF and `$$...$$` for HTML:

Code (markdown)

````
::: {.content-visible when-format="pdf"}
```{=latex}
\begin{align}
  \nabla \cdot \mathbf{E}  &= \frac{\rho}{\varepsilon_0} \\
  \nabla \cdot \mathbf{B}  &= 0 \\
  \nabla \times \mathbf{E} &= -\frac{\partial \mathbf{B}}{\partial t} \\
  \nabla \times \mathbf{B} &= \mu_0\mathbf{J}
    + \mu_0\varepsilon_0\frac{\partial \mathbf{E}}{\partial t}
\end{align}
```
:::

::: {.content-visible when-format="html"}
$$
\begin{align}
  \nabla \cdot \mathbf{E}  &= \frac{\rho}{\varepsilon_0} \\
  \nabla \cdot \mathbf{B}  &= 0 \\
  \nabla \times \mathbf{E} &= -\frac{\partial \mathbf{B}}{\partial t} \\
  \nabla \times \mathbf{B} &= \mu_0\mathbf{J}
    + \mu_0\varepsilon_0\frac{\partial \mathbf{E}}{\partial t}
\end{align}
$$
:::
````

Output:

$$\nabla \cdot \mathbf{E} = \frac{\rho}{\varepsilon_0} \tag{3.4}$$

$$\nabla \cdot \mathbf{B} = 0 \tag{3.5}$$

$$\nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t} \tag{3.6}$$

$$\nabla \times \mathbf{B} = \mu_0\mathbf{J} + \mu_0\varepsilon_0\frac{\partial \mathbf{E}}{\partial t} \tag{3.7}$$

Each `\\` receives its own sequential number. To suppress the number on a line, add `\notag` before the `\\`. To reference an individual line, use `\label{my-label}` inside the `{=latex}` block and `\eqref{my-label}` in the text (PDF only — KaTeX does not generate addressable labels).

⚠ `\begin{align}` is incompatible with `$$...$$` in Quarto

Quarto (Pandoc) wraps every `$$...$$` in `\[...\]` (without label) or `\begin{equation}...\end{equation}` (with `{#eq-}`). Both forms create invalid nesting with `\begin{align}`, which amsmath rejects:

`Package amsmath Error: Erroneous nesting of equation structures`

The systematic solution is the dual `{.content-visible}` pattern: raw `{=latex}` block for PDF, `$$\begin{align}...\end{align}$$` for HTML (KaTeX supports `align` natively).

`aligned` vs `align` — what is the difference?

Environment	Quarto syntax	Numbering	`@eq-`
`aligned`	`$$\begin{aligned} ... \end{aligned}$$`	one number for the **block**	✓
`align`	`{=latex}` + `$$...$$` (dual)	one number **per line**	×
`align*`	`{=latex}` + `$$...$$` (dual)	none	—

Practical rule: calculation development → `aligned` (one number, `@eq-` possible). Independent equations → `align` with the dual pattern.

Hierarchical numbering (3.1a), (3.1b) with `subequations`

For **(3.1a)**, **(3.1b)**… numbering, combine `subequations` and `align` in a `{=latex}` block (KaTeX does not support `subequations`):

Code (markdown)

```
::: {.content-visible when-format="pdf"}
```{=latex}
\begin{subequations}\label{eq-navier}
\begin{align}
  \rho\,\frac{D\mathbf{u}}{Dt}
    &= -\nabla p + \mu\,\nabla^2\mathbf{u} + \mathbf{f}
    \label{eq-navier-momentum} \\
  \nabla \cdot \mathbf{u} &= 0
    \label{eq-navier-incomp}
\end{align}
\end{subequations}
```
:::
```

`\label{eq-navier}` references the group; `\label{eq-navier-momentum}` and `\label{eq-navier-incomp}` reference the lines — accessible with `\eqref{}`. Add a sibling `{.content-visible when-format="html"}` block with `$$\begin{align}...\end{align}$$` for HTML rendering.

3.4 Simple Markdown tables

Pandoc/Quarto's *pipe table* syntax creates tables with per-column alignment. The caption, preceded by `:`, must be placed immediately after the table:

Code (markdown)

```markdown
| Command           | PDF format     | HTML format   |
|:------------------|:--------------:|:-------------:|
| `@fig-label`      | figure number  | hyperlink     |
| `@tbl-label`      | table number   | hyperlink     |
| `@eq-label`       | eq. number     | hyperlink     |
| `@sec-label`      | section num.   | hyperlink     |

: Quarto cross-reference prefixes. {#tbl-crossref}
```

Output:

Table 3.1: Quarto cross-reference prefixes.

Command	PDF format	HTML format
`@fig-label`	figure number	hyperlink
`@tbl-label`	table number	hyperlink
`@eq-label`	eq. number	hyperlink
`@sec-label`	section num.	hyperlink

Table 3.1 summarises the four prefixes to know.

Column alignment

- `:---` — left-aligned
- `:---:` — centred
- `---:` — right-aligned

The line `|:---|:---:|---:|` defines the alignment of each column.

The caption must be flush against the table

The `: Caption. {#tbl-label}` line must be placed **immediately** after the table, with no blank line in between. A blank line breaks the table-caption association and produces an orphan caption with no numbering.

3.5 Advanced tables with quartable

The quartable extension (listed in the filters: key of _quarto.yml) adds three capabilities to Markdown tables:

- **Rowspan**: [text]{rs=N} merges N rows; ^ continues the merged cell
- **Colspan**: [text]{cs=N} merges N columns
- **Midrules**: === inserts a horizontal separator; ===2-4 a partial rule spanning columns 2 to 4.

Full documentation for the quartable extension

To see all the features of this extension (inspired by the LaTeX booktabs package), see https://github.com/zinc75/quarto-quartable/ and https://zinc75.github.io/quarto-quartable/test_quartable.pdf/.

Code (markdown)

```
| Family               | Feature               | Chapter |
|:---------------------|:----------------------|:-------:|
| ===                  |                       |         |
| [Structure]{rs=3}    | Headings and levels   | 1       |
| ^                    | Cross-references      | 1       |
| ^                    | Inline formatting     | 1       |
| ===                  |                       |         |
| [Visuals]{rs=3}      | Static figures        | 2       |
| ^                    | Python figures        | 2       |
| ^                    | Multi-column          | 2       |
| ===                  |                       |         |
| [Math & data]{rs=3}  | Equations             | 3       |
| ^                    | Simple tables         | 3       |
| ^                    | `quartable`           | 3       |

: Summary of features by family. {#tbl-recap}
```

Output:

Table 3.2: Summary of features for chapters 1 to 3.

Family	Feature	Chapter
Structure	Headings and levels	1
	Cross-references	1
	Inline formatting	1
Visuals	Static figures	2
	Python figures	2
	Multi-column	2

Table 3.2: Summary of features for chapters 1 to 3.

Family	Feature	Chapter
Math & data	Equations	3
	Advanced equations	3
	Simple tables	3

`quartable` must be in `filters:`

The `[text]{rs=N}` syntax is only recognised if `quartable` is listed in the `filters:` key of `_quarto.yml`. Without this, the curly braces remain displayed as-is in the output.

Partial rules with `===N-M`

`===2-4` inserts a rule that spans only columns 2 to 4 (inclusive). Useful for underlining a sub-group of columns without separating the entire row.

CHAPTER

4

Callouts and code blocks

Contenu

Callouts allow important information to be highlighted. Code blocks allow code to be displayed or executed. This chapter presents both with examples drawn from a doctoral context.

4.1 The five callout types

Quarto defines five callout types, each with a precise semantic:

Code (markdown)

```
::: {.callout-specification}
## Title (optional) of the callout
Content of the callout, with `specification` taking one of the values
`note`, `tip`, `warning`, `important`, or `caution`, each associated with
a colour code and an icon in the callout header.
:::
```

4.1.1 `callout-note` — neutral information

Code (markdown)

```
::: {.callout-note}
## Jury composition

A Cnam thesis jury comprises at minimum a chair, two reviewers
(with ) and one or two examiners. The defence lasts approximately
45 minutes of presentation, followed by questions from the jury.
:::
```

Output:

> **Jury composition**
>
> A Cnam thesis jury comprises at minimum a chair, two reviewers (with HDR) and one or two examiners. The defence lasts approximately 45 minutes of presentation, followed by questions from the jury.

4.1.2 `callout-tip` — actionable advice

Code (markdown)

```
::: {.callout-tip}
## Start writing early

Start writing your thesis from the second year of doctoral study. Late writing is
the primary cause of exceeding the contractual thesis duration. Even imperfect drafts
are better than empty sections.
:::
```

Output:

> **Start writing early**
>
> Start writing your thesis from the second year of doctoral study. Late writing is the primary cause of exceeding the contractual thesis duration. Even imperfect drafts are better than empty sections.

4.1.3 `callout-warning` — pitfall to avoid

Code (markdown)

```
::: {.callout-warning}
## Do not rename chapters without updating the profile

If you rename a `.qmd` file, update the `chapters:` list in
`_quarto-en.yml` (or `_quarto-fr.yml`). Quarto does not automatically detect
missing files and will fail silently.
:::
```

Output:

> **Do not rename chapters without updating the profile**
>
> If you rename a `.qmd` file, update the `chapters:` list in `_quarto-en.yml` (or `_quarto-fr.yml`). Quarto does not automatically detect missing files and will fail silently.

4.1.4 `callout-important` — non-negotiable point

Code (markdown)

```
::: {.callout-important}
## PDF/A-1b mandatory for theses.fr

The  requires a PDF/A for archiving on theses.fr. This template
automatically generates a PDF/A-1b via `\usepackage[a-1b]{pdfx}`. Do not remove
this package from the template.
:::
```

Output:

> **PDF/A-1b mandatory for theses.fr**
>
> The ABES requires a PDF/A for archiving on theses.fr. This template automatically generates a PDF/A-1b via `\usepackage[a-1b]{pdfx}`. Do not remove this package from the template.

4.1.5 `callout-caution` — danger with consequences

Code (markdown)

```
::: {.callout-caution}
## Do not use `date:` for the defence date

Quarto interprets the `date:` key as a JavaScript date and produces
"Invalid Date" for French date strings (e.g. "12 June 2026"). Use exclusively
the custom key `date-soutenance:` provided in this template.
:::
```

Output:

Do not use `date:` for the defence date

Quarto interprets the `date:` key as a JavaScript date and produces "Invalid Date" for French date strings (e.g. "12 June 2026"). Use exclusively the custom key `date-soutenance:` provided in this template.

4.2 Callouts with title and collapsing

The title is added with a `## Title` line inside the callout (as in the examples above). To make a callout collapsed by default (click to open), use the `collapse` attribute:

Code (markdown)

```
::: {.callout-tip collapse="true"}
## Click to see the tip (collapsed by default)

This callout is collapsed when the HTML document opens. In PDF, it always appears
in full (collapsing is an HTML-only feature).
:::
```

Output:

Click to see the tip (collapsed by default)

This callout is collapsed when the HTML document opens. In PDF, it always appears in full (collapsing is an HTML-only feature).

4.3 Code blocks — display only

To display code without executing it, use either a block delimited by backticks without a language specifier in curly braces, or a chunk with `eval: false`:

Code (markdown)

````
```python
# This block is displayed but not executed
import numpy as np
x = np.linspace(0, 10, 100)
print(x.mean())
```
````

Output:

Code (python)

```python
# This block is displayed but not executed
import numpy as np
x = np.linspace(0, 10, 100)
print(x.mean())
```

To display the code of an executable chunk without executing it (useful for installation or configuration examples):

Code (markdown)

````
```{python}
#| eval: false
#| echo: true
# Installing dependencies
pip install matplotlib numpy pandas
```
````

Output:

Code (python)

```python
# Installing template dependencies
# pip install matplotlib numpy pandas
```

Most useful chunk options

Table 4.1: Frequently used Quarto chunk options.

Option	Values	Effect
`echo`	`true/false`	shows or hides the code
`eval`	`true/false`	executes or skips the code
`output`	`true/false`	shows or hides the output
`warning`	`true/false`	shows or hides warnings
`fig-cap`	text	caption for the produced figure
`label`	`fig-...` or `tbl-...`	identifier for cross-reference

4.4 Code blocks — with Python execution

In this document, all executable code examples — including the matplotlib figures of Section 2.2 — use `#| echo: true` to display both the source code and the result. In a finalised thesis, code is generally hidden by switching to `#| echo: false`: only the figure or table appears, without the code that produced it.

Languages natively supported by Quarto

Quarto supports three scientific computing languages **without additional configuration**: **Python**, **R**, and **Julia**. The chunk option syntax (`#| echo:`, `#| fig-cap:`, etc.) and the cross-reference mechanism (`#| label:`) are identical for all three languages. This document illustrates Python only.
Official documentation:

- Python: https://quarto.org/docs/computations/python.html
- R: https://quarto.org/docs/computations/r.html
- Julia: https://quarto.org/docs/computations/julia.html

A Python chunk with `eval: true` (the default) executes the code and displays the output. Here is an example with `pandas`:

Code (markdown)

````
```{python}
#| echo: true
#| label: tbl-pandas
#| tbl-cap: "Descriptive statistics of trigonometric functions."
import numpy as np, pandas as pd
from IPython.display import Markdown

x = np.linspace(0, 2 * np.pi, 1000)
df = pd.DataFrame({r'$\sin(x)$':           np.sin(x),
                   r'$\cos(x)$':           np.cos(x),
                   r'$\tan(x)$ (clipped)': np.clip(np.tan(x), -5, 5)})
Markdown(df.describe().round(3).to_markdown())
```
````

Output:

Code (python)

```
import numpy as np, pandas as pd
from IPython.display import Markdown

x = np.linspace(0, 2 * np.pi, 1000)
df = pd.DataFrame({r'$\sin(x)$':           np.sin(x),
                   r'$\cos(x)$':           np.cos(x),
                   r'$\tan(x)$ (tronqué)': np.clip(np.tan(x), -5, 5)})
Markdown(df.describe().round(3).to_markdown())
```

Table 4.2: Descriptive statistics of trigonometric functions.

	$\sin(x)$	$\cos(x)$	$\tan(x)$ (tronqué)
count	1000	1000	1000
mean	0	0.001	-0
std	0.707	0.708	2.335
min	-1	-1	-5
25%	-0.706	-0.705	-0.997
50%	-0	0.002	-0
75%	0.706	0.708	0.997
max	1	1	5

`freeze: auto` in `_quarto.yml`

With `execute: freeze: auto`, Quarto stores cell results in `_freeze/` and only recomputes them if the chunk code has changed. This considerably speeds up successive renders for theses with many computations.

Remember to version the `_freeze/` folder

If you version your thesis on GitHub and want reviewers to be able to compile without Python installed, commit the `_freeze/` folder. The cached results are stored there and will be used instead of re-executing the code.

CHAPTER

5

References, glossary, and annotations

Contenu

This chapter presents the bibliographic citation system (IEEE style via `IEEEtran-francais.bst`), the glossary and acronym system, collaborative comments, and special front matter pages.

5.1 Bibliographic citations

5.1.1 Basic syntax

Citations are inserted with `[@key]`. The key corresponds to the BibTeX identifier defined in `references.bib`:

Code (markdown)

```
Computer typesetting was revolutionised by Knuth [@Knuth1984],
whose \TeX{} system was made accessible by Lamport with \LaTeX{} [@Lamport1994].
Quarto [@Allaire2022] unifies these tools in a modern publishing environment.
```

Output:

Computer typesetting was revolutionised by Knuth [1], whose TEX system was made accessible by Lamport with LATEX [2]. Quarto [3] unifies these tools in a modern publishing environment.

5.1.2 Multiple references

Code (markdown)

```
Python visualisation tools [@Hunter2007] and typographic reference works
[@Bringhurst2004; @Mittelbach2023] complement the author's toolkit.
```

Output:

Python visualisation tools [5] and typographic reference works [4, 6] complement the author's toolkit.

Always use `[@key]` with brackets

The `IEEEtran-francais.bst` style uses `natbib`. The `@key` syntax without brackets calls `\citet{}` which is broken with IEEEtran — it produces the author name followed by `[?]`. **Always use `[@key]`** to obtain `[N]` (IEEE numbering).

5.1.3 The `references.bib` file

The `references.bib` file must be located **at the project root**. Pandoc in natbib mode generates `\bibliography{references}` (without a path), and BibTeX looks for the file in the current directory. A file in `content_en/references.bib` will not be found.

Bibliography managers

Zotero (with the Better BibTeX connector), JabRef, or Mendeley export directly to `.bib` format. The `@IEEEtranBSTCTL` entry at the top of the file configures the IEEEtran style (number of authors before "et al.", etc.) — do not remove it.

5.2 Glossary and acronyms

5.2.1 Declaring entries in `glossaire-entries.qmd`

All entries are declared in `content_en/frontmatter/glossaire-entries.qmd` via two shortcodes:

Code (markdown)

```
{{< acr label ABBREV "Full expansion" >}}
{{< terme label "word" "Complete definition of the term" >}}
```

The shortcodes automatically write `_glossaire-entries.tex` (read by LaTeX) and populate the HTML tables. Labels must be unique.

5.2.2 Using acronyms in the text

Four shortcodes allow referencing an acronym in the desired form:

Table 5.1: Glossary reference shortcodes.

Shortcode	PDF output	HTML output
`{{< gls cnam >}}`	short form (1st use: long)	short form
`{{< acrs cnam >}}`	always short form	always short form
`{{< acrl cnam >}}`	always long form	always long form
`{{< acrf cnam >}}`	long (short)	long (short)

Examples in context:

Code (markdown)

```
The {{< acrs cnam >}} ({{< acrl cnam >}}) was founded in 1794.
The {{< acrf qmd >}} format is Quarto's native format.
The  is generated automatically by Quarto.
```

Output:

The Cnam (Conservatoire national des arts et métiers) was founded in 1794. The Quarto Markdown Document (QMD) format is Quarto's native format.

PDF vs HTML behaviour

In PDF, expands on the first occurrence (long form + abbreviation in parentheses) then uses the short form. In HTML, this first-occurrence logic is not handled by the shortcode — it always produces the short form. For consistency, prefer on the first occurrence in the text, and thereafter.

5.3 Collaborative comments (quarto-comments)

5.3.1 Configuration in _quarto.yml

The quarto-comments extension (folder _extensions/comments/) is configured via the extensions.quarto-c key in _quarto.yml:

Code (yaml)

```yaml
extensions:
  quarto-comments:
    enabled: true
    authors:
      eb:
        name: "Eric Bavu"
      ag:
        name: "Abbé Grégoire"
      # Colors auto-assigned from Bootstrap 5 palette; override if needed:
      # nc:
      #   name: "Nicolas Conté"
      #   color_html: "#198754"
      #   color_latex: "green!30"
```

5.3.2 The four comment types

Code (markdown)

```
{{< comment "Rephrase this sentence, it is too long." author="ag" >}}
{{< todo "Add a figure illustrating this point." author="eb" >}}
{{< note "See also Bringhurst 2004, §2.4." author="ag" >}}
{{< question "Is this result reproducible on another dataset?" author="ag" inline=true
↪  >}}
```

Rendered in context:

Here is a sentence with a comment from the thesis supervisor (who added it directly in the `.qmd` file, because that is how it works!).

These annotations are only visible when `enabled: true` is set under `extensions.quarto-comments` in `_quarto.yml` — which is the case for this documentation.

An inline note can be added without pushing the surrounding text to the margin: a bibliographic reference

Abbé Grégoire: See Knuth 1984, chapter 12.

stays in the text flow.

Abbé Grégoire: This is a 'comment' type annotation by the supervisor.

Eric Bavu: Remember to set this option to 'false' before submitting to the reviewers!

`inline=true` occupies the full line in PDF

In HTML, `inline=true` annotations appear as a compact coloured span, exactly as wide as their content. In PDF, the `todonotes` package renders `\todo[inline]` as a full-width coloured block — this is a design constraint of the package that cannot be worked around without losing the ability to wrap long annotations across lines. Use `inline=true` for short notes that must stay in the text flow; for longer annotations, omit `inline` to place the note in the margin instead.

Disable comments before final submission

Before submitting the thesis to the doctoral school and depositing it on theses.fr, set `extensions.quarto-comments.enabled: false` in `_quarto.yml`. Annotations must not appear in the official archived version.

5.4 Web annotations with Hypothesis (opt-in)

For remote proofreading, the HTML output can be enhanced with web annotations via Hypothesis. Supervisors and reviewers select text in the browser and add comments without touching the source files. Annotations are stored on Hypothesis servers and tied to the page URL — they persist across renders as long as the URL stays stable (GitHub Pages deployment

recommended).

To enable, uncomment the following block in `_quarto.yml`:

Code (yaml)

```yaml
format:
  cnam-thesis-html:
    comments:
      hypothesis:
        theme: clean   # clean = discreet ; classic = sidebar always visible
```

Then create a private group on hypothes.is and share the invite link with your reviewers. Hypothesis is disabled by default in all profiles.

5.5 Special front matter pages

5.5.1 Résumé and abstract

The `resume.qmd` and `abstract.qmd` pages use CSS classes that add a left border in HTML:

Code (markdown)

```markdown
::: {.resume-block}
Text of the résumé in French.

**Mots-clés :** mot1, mot2, mot3
:::
```

! French résumé mandatory for all Cnam theses

Even for a thesis written in English, a French résumé is **mandatory** according to the ABES rules for deposit on theses.fr. This is the role of the file `content_en/frontmatter/resume.qmd` in the English profile.

5.5.2 The Individual Monitoring Committee (CSI)

i French doctoral process — the CSI

The Comité de Suivi Individuel (CSI) has been mandatory since 2016 for all doctoral students in France. It meets at least once a year, independently of the thesis supervisor, to verify progress, working conditions, and adherence to the schedule. The minutes of the CSI meeting must be included in the defence dossier.
The HTML version of this thesis template is particularly well suited for CSI meetings: it allows sharing a link to the online version, which is easier to navigate than a 200-page PDF.

Conclusion

This guide has presented the features of the `quarto-cnam-thesis` template following the *code + output* principle: each syntax is shown exactly as typed, then immediately rendered in the document. This self-referential approach aims to make the template self-sufficient as documentation.

Feature summary

Table 5.2: Summary of all documented features.

Feature	Key syntax	Chapter
Structure	Hierarchy `#` / `##` / `###` / `####`	1
	`{.unnumbered}`, `{.toc-black}`	1
	`{#sec-label}` + `@sec-label`	1
	`short-title="..."`	1
Figures	`![cap](img){#fig-label width=X%}`	2
	Chunk `{python}` with `#|` `label: fig-...`	2
	`layout-ncol=2`	2
	`{=latex}` block in `:::{#fig-}` div	2
Math & tables	`$...$` / `$$...$$` `{#eq-label}`	3
	`\begin{aligned}`, `\begin{cases}`	3
	Pipe table + `: Caption.` `{#tbl-label}`	3
	`[text]{rs=N}`, `^`, `===` (`quartable`)	3
Callouts	`{.callout-note}`	4
	`{.callout-tip}`	4
	`{.callout-warning}`	4
	`{.callout-important}`	4
	`{.callout-caution}`	4
References	`[@key]` / `[@key1; @key2]`	5

Table 5.2: Summary of all documented features.

Feature	Key syntax	Chapter
	`{{{< acr >}}}`, `{{{< terme >}}}`, `{{{< acrs >}}}`	5
	`{{{< comment >}}}`, `{{{< todo >}}}`, `{{{< note >}}}`	5
	`.resume-block`, `.abstract-block`	5

Recommended workflow

Daily writing cycle

1. **Write in HTML**: `quarto render --profile en --to cnam-thesis-html --no-clean` — fast rendering, easy navigation, ideal for proofreading.
2. **Check the PDF** before each CSI meeting: `quarto render --profile en` — compiles everything and generates the PDF/A-1b.
3. **Version regularly**: commit at least after each writing session, ideally before and after each supervision meeting.
4. **Disable comments** before final submission: `extensions.quarto-comments.enabled: false` in `_quarto.yml`.

The template is maintained on GitHub. To report an issue or propose an improvement: zinc75/quarto-cnam-thesis.

Bibliography

[1] D. E. Knuth, *The TEXbook*, ser. Computers & Typesetting. Reading, Massachusetts: Addison-Wesley, 1984, vol. A.

[2] L. Lamport, *LATEX: A Document Preparation System*, 2e éd. Reading, Massachusetts: Addison-Wesley Professional, 1994.

[3] J. Allaire, C. Teague, Y. Xie et C. Dervieux, "Quarto," 2022.

[4] R. Bringhurst, *The Elements of Typographic Style*, 3e éd. Vancouver: Hartley & Marks, 2004.

[5] J. D. Hunter, "Matplotlib: A 2D graphics environment," *Computing in Science & Engineering*, vol. 9, no. 3, p. 90–95, 2007.

[6] F. Mittelbach et U. Fischer, *The LATEX Companion*, 3e éd. Reading, Massachusetts: Addison-Wesley Professional, 2023.

APPENDIX

A

YAML configuration

Contenu

This appendix documents the two main configuration files of the template: `_quarto.yml` (metadata shared across all renders) and `_quarto-fr.yml` / `_quarto-en.yml` (metadata specific to each language profile).

A.1 Metadata in `_quarto.yml` — cover page

`_quarto.yml` holds the fields **shared across all profiles**: jury, supervisors, institutional information. Title, author, and defence date belong in the language profile (see Language profiles below).

Code (yaml)

```yaml
book:
  downloads: [pdf]                    # PDF button in the HTML sidebar

discipline: "60th CNU section - Mechanics, mechanical engineering, civil engineering"
specialite: "Acoustics"
ecole-doctorale: "Sciences des Métiers de l'Ingénieur"
```

```yaml
laboratoire: "MACS Laboratory, Cnam Paris"
directeur: "Prof. First LAST, Full Professor, Cnam Paris"
cotutelle: false

dedicace: "Optional dedication..."       # remove if absent
```

A.2 Jury composition for the cover page

Code (yaml)

```yaml
jury:
  - nom: "Ms First LAST"
    titre: "Title, Research Unit, Institution"
    role: "Chair"
  - nom: "Mr First LAST"
    titre: "Title, Research Unit, Institution"
    role: "Reviewer"
  - nom: "Ms First LAST"
    titre: "Title, Research Unit, Institution"
    role: "Reviewer"
  - nom: "Mr First LAST"
    titre: "Associate Professor HDR, Unit, Institution"
    role: "Examiner"
  - nom: "Mr First LAST"
    titre: "Full Professor, MACS, Cnam Paris"
    role: "Thesis supervisor"
```

A.3 Infrastructure section options

Code (yaml)

```yaml
glossaire: true          # true = loads the LaTeX glossaries package
pagestyle-sections: true # true = roman/arabic/Roman numbering per section
bibliography-manual: true
minitoc-newpage: false   # true = page break after each mini-TOC
chapter-openright: true  # false = for digital version without right-hand page
index: false             # true = generates an index via \index{term}
```

A.4 Collaborative comments (`quarto-comments`)

Code (yaml)

```yaml
extensions:
  quarto-comments:
    enabled: true    # false before final deposit on theses.fr
    authors:
      doc:
        name: "First LAST"       # doctoral student
```

```yaml
      dir:
        name: "Supervisor"
      # Colors auto-assigned from Bootstrap 5 palette; override if needed:
      # codir:
      #   name: "Co-supervisor"
      #   color_html: "#198754"
      #   color_latex: "green!30"
```

See Section 5.3 for shortcode syntax and PDF/HTML rendering details.

A.5 Web annotations with Hypothesis (opt-in)

To allow reviewers to annotate the HTML version directly in their browser, uncomment this block in `_quarto.yml`:

Code (yaml)

```yaml
format:
  cnam-thesis-html:
    comments:
      hypothesis:
        theme: clean   # clean = discreet ; classic = sidebar always visible
```

Create a private group on hypothes.is and share the invite link. Annotations are tied to the page URL — works best on a stable deployment (GitHub Pages). See Section 5.3 for more details.

A.6 Python execution

Code (yaml)

```yaml
execute:
  freeze: auto    # recomputes only if the chunk code changes
  echo: true      # shows code by default (overridable per chunk)
```

A.7 Language profiles: `_quarto-fr.yml` and `_quarto-en.yml`

Language profiles **override** `_quarto.yml` when rendering with `--profile fr` or `--profile en`. They define the language-specific metadata — **title, author, defence date** — as well as the chapter list and technical render options.

A.7.1 Chapter list (first thing to customise)

This is the first thing to change when taking the template in hand. The full list of files that make up the book is defined in `_quarto-en.yml` (or `_quarto-fr.yml`), under the `book.chapters` key:

Code (yaml)

```yaml
book:
  chapters:
    - index.qmd                                    # HTML home page - do not remove
    - content_en/frontmatter/acknowledgements.qmd
    - content_en/frontmatter/resume.qmd            # French abstract (Cnam
↪  requirement)
    - content_en/frontmatter/abstract.qmd
    - content_en/frontmatter/tables.qmd            # TOC + PDF lists - do not remove
    - content_en/frontmatter/glossaire-entries.qmd # remove if glossaire: false
    - content_en/frontmatter/acronyms.qmd          # remove if glossaire: false
    - content_en/frontmatter/glossary.qmd          # remove if no glossary terms
    - content_en/chapters/00-introduction.qmd
    - content_en/chapters/01-chapter1.qmd          # <- add your chapters here
    - content_en/chapters/02-chapter2.qmd
    - content_en/backmatter/conclusion.qmd
    - content_en/backmatter/bibliography.qmd
  appendices:
    - content_en/backmatter/appendices.qmd
    - content_en/backmatter/appendices-pdfa.qmd
```

To add a chapter: create a `.qmd` file in `content_en/chapters/` and insert it in this list at the desired position.

Do not remove `index.qmd` or `tables.qmd`

`index.qmd` is the Quarto book entry point — removing it breaks the HTML render. `tables.qmd` generates the table of contents, list of figures and list of tables in the PDF.

A.7.2 Other profile options

Code (yaml)

```yaml
thesis-lang: en      # thesis language: fr or en
validate: false      # set to true before depositing to theses.fr (see @sec-pdfa)

book:
  title: "Thesis title"
  subtitle: "Optional subtitle"        # remove if absent
  author: "First LAST"
  output-file: these_en               # base name of the generated PDF
↪  (these_en_<author>.pdf)

```

```yaml
date-soutenance: "10 October 1794"    # DO NOT use date:

project:
  output-dir: _thesis-en    # output directory
  post-render:
    - ./_scripts/postrender.ts en _thesis-en

format:
  cnam-thesis-html:
    toc-title: "Table of Contents"   # English sidebar TOC title
```

APPENDIX

B

PDF/A-1b validation

Contenu

B.1 Final deposit on theses.fr

The French national thesis repository theses.fr requires a PDF/A-1b file (ISO 19005-1). The PDF produced by this template is designed to meet the criteria used by CINES — the body responsible for institutional archiving of French doctoral theses.

B.2 Official validator: facile.cines.fr

facile.cines.fr (CINES) is the reference validator for deposit on theses.fr. It — not third-party tools — determines whether your PDF will be accepted. Simply upload the PDF to obtain an immediate report.

B.3 Automated check from the post-render script

The post-render script can query the CINES web service automatically. Simply set `validate: true` in `_quarto-en.yml`:

Code (yaml)

```yaml
# In _quarto-en.yml - before depositing to theses.fr:
validate: true
```

Then render as usual:

Code (bash)

```bash
quarto render --profile en --to cnam-thesis-pdf
```

The script sends the PDF to the CINES web service and displays the result:

```
Validating PDF/A on facile.cines.fr (CINES)...
[OK]  PDF/A-1b valide -- archivable sur theses.fr.
```

If the PDF fails:

```
[!!]  PDF/A-1b non valide.
      -> Corriger via : https://facile.cines.fr/#correction
```

After validation

Set `validate: false` back in `_quarto-en.yml` for day-to-day builds — the web service call is only needed before the final deposit.

B.4 If validation fails

Use the official correction service: **facile.cines.fr/#correction**. It automatically corrects the most common non-conformities with no additional software to install.