CONTENTS Table of Contents

CHAPTER 1 Repository Layout and Build Entry Points

1.1 Top-Level Directory Map

The project root is intentionally shallow so a new contributor can discover the build targets without digging through source first. Each entry below lists the exact reason the directory exists.

README.md Human-facing overview. Mirrors the structure in this guide so GitHub visitors immediately understand how to get started.

.gitignore Keeps generated LaTeX artefacts out of version control. Every pattern is enumerated in Section 1.

docs/ Markdown quick-start and architecture notes for readers who prefer plain text over the LaTeX guide. These files echo the content of Chapter 1 and Chapter 3.

projects/ All compilable deliverables.

- **textbook/** The documentation-as-book project. `src/` hosts the LaTeX sources. Passing `-outdir=../output` keeps the book's artefacts in a sibling directory, mirroring the historic layout.
- **homework/** Content shared between the appendices and the standalone assignments. Each `homeworkXX.tex` file is self-contained but detects when it is embedded inside the textbook.

tex/ The reusable TeX library, organised as modules: `core/` (packages + helpers), `styles/` (visual identity), and `modules/` (small shims such as homework embedding).

1.2 Textbook Build Flow

The documentation book compiles through `projects/textbook/src/main.tex`. That file performs the following steps every time XeLaTeX runs:

1. **Resolve repository-relative paths.** Lines 11--35 compute the value of `\TexRoot` and `\HomeworkRoot` so the build works whether you call `latexmkmain.tex` from inside `src/` or from the repository root.

2. **Load the TeX library.** The helper macro `\TexInput` wraps every include so the exact module path stays declarative: `core/colors`, `core/base`, `styles/notes`, and finally the homework embedding shim from `modules/homework-embed`. Section 2 gives the rationale for each module.

3. **Render the documentation chapters.** The front matter styles and emits the live table of contents via `chapters/table-of-contents.tex` before inputting the curated chapters that form this tutorial.

4. **Switch into appendix mode.** The build captures the default header width, swaps to homework geometry, emits the appendix chapter, streams in every embedded assignment, and then restores the book layout.

Because every content include uses `\TexInput` or `\HomeworkIncludeInNotes`, there is no hidden state—each line points directly at its dependency. The only code outside the loop is the geometry swap that protects the textbook layout from homework overrides.

1.3 Homework Projects

Homework material now uses a single-source pattern: each file inside `projects/homework/` begins with `\textbackslashinput\{../../tex/system/document-homework.tex\}`. That wrapper loads the shared library, exposes `\HomeworkDocumentBegin`/`\HomeworkDocumentEnd`, and becomes a no-op when the textbook imports the same source via `\HomeworkIncludeInNotes`. You edit every problem once and choose whether to emit a standalone PDF or append it to the book simply by toggling the entry point. Discussion, lecture, and lab-report capture reuse the same philosophy via `tex/system/document-notes.tex` and `tex/system/document-report.tex`.

To compile Homework 5 on its own:

```
cd projects/homework
latexmk -xelatex -interaction=nonstopmode -halt-on-error
-file-line-error homework05.tex
```

The wrapper resolves `\TexRoot` identically to the textbook, so styling stays in sync across both targets and metadata helpers only live in one place. Running latexmk from any project folder automatically mirrors the source path by creating a sibling directory named after the `.tex` file (e.g., `projects/homework/homework05/homework05.*`), leaving the `projects/` tree tidy.

1.4 Output Hygiene and Path Helpers

Two small but important details keep the repository approachable:

1. **Output isolation.** Standalone projects keep artefacts inside a folder named after the `.tex` file in the same directory (while the textbook build can optionally target `../output/`). Nothing under those folders is committed—contributors rebuild on demand instead of trusting stale binaries.

2. **Path bootstrap.** The macros defined in `projects/textbook/src/main.tex` and `tex/system/document-*.tex` (homework, notes, reports) resolve the repository layout dynamically. They all expose the same helper:

   ```
   \newcommand {\TexInput }[1]{\input {\TexRoot ##1}}
   ```

   Any future document can reuse that pattern to opt into the shared TeX library without guessing relative paths. If the layout ever changes, update the detection logic in one place and the entire toolchain keeps working.

CHAPTER 2 TeX Library Reference

2.1 Module Overview

Everything under `tex/` is a reusable module. The directory is split into semantic groups so a reader can map code to effects immediately:

`core/` Foundational packages, typography helpers, and shared macros. These files define the vocabulary used by every project.

`styles/` Visual identity for the textbook and homework formats. The geometry, headers, and display environments all live here.

`modules/` Opt-in shims. Right now the only module toggles the homework style into ``embedded'' mode before streaming content into the book.

`system/` Bootstrap helpers and the standalone wrappers that hide documentclass boilerplate for homework, intake notes, and lab reports.

Each sub-section below documents why every line in these modules exists and how future changes should be staged.

2.2 `tex/core/colors.tex`: Palette Bootstrap

The colour module keeps the palette declarative:

1. `\usepackage[dvipsnames]{xcolor}` loads the extended colour names once for the entire toolkit.
2. Five `\providecommand` calls define override-friendly storage slots for the semantic colours (`Primary`, `Accent`, `RuleGray`, `Highlight`, and `Link`). Using `\providecommand` instead of `\newcommand` means downstream documents can reassign a colour before this file loads.
3. `\MathColorDefine` is a helper macro that converts the stored model/spec pair into a named colour via `\definecolor`. Calling it for each palette entry keeps the public API tiny.
4. `\MathColorOverride` exposes a single entry point for runtime overrides. It updates both the stored model/spec pair *and* calls `\MathColorDefine` so later macros see the new value.
5. The `\AtBeginDocument` block syncs `\hypersetup` with the palette so hyperlinks match the rest of the design.

Every document that wants a different palette does so by redefining the stored spec values before loading this file—no manual `\definecolor` clutter appears elsewhere.

2.3 `tex/core/base.tex`: Common Packages and Macros

This file centralises the dependencies shared by both projects.

Math & layout packages The opening `\usepackage` block imports the AMS suite, `mathtools` (fixes alignment quirks), list utilities (`enumitem`), float helpers, `needspace` for widow/orphan control, and quality-of-life packages such as `hyperref`. Loading them once prevents mismatched versions between the book and homework PDFs.

Graphics stack `graphicx`, `tikz`, and `pgfplots` are configured together so the compatibility level and libraries stay aligned. Any figure produced in one document will compile in the other.

Utility macros The `\vect`, `\mat`, `\R`, and similar shorthands keep mathematical expressions readable. Each macro exists because it appears somewhere in the sample content; if you add a new convenience wrapper, document it in this section and inside the homework guide.

Environment definitions `\newtheorem` entries align numbering with the current section, while the custom `\textbookproblem` and `\solutionbox` environments format exercises and answers consistently across projects.

Because all maths-centric dependencies sit here, downstream documents only need to manage content—they rarely touch this file unless a new global macro is introduced.

2.4 `tex/styles/notes.tex`: Book Layout

The notes style translates the project into a textbook aesthetic:

1. **Geometry block.** The `\usepackage[...]{geometry}` call defines the wide left margin (for commentary), narrow top margin, and increased `\marginparwidth`. Every numeric choice is the result of iterating on readability—changing it here updates the whole book.

2. **Font selection.** A trio of `\IfFontExistsTF` checks prefer JetBrains Mono, Helvetica, and Times New Roman while falling back to TeX Gyre alternatives. The macros `\NotesFontBody`, `\NotesFontHeader`, etc. centralise font usage so the rest of the code speaks in roles, not typefaces.

3. **Headers and footers.** The `\pagestyle{fancy}` block creates mirrored headers that swap chapter titles and page numbers on odd and even pages. Setting `\headheight` twice prevents common LaTeX warnings.

4. **Counters.** `\numberwithin` and the custom counters (`textexample`, `notestheorem`, etc.) ensure every displayed element tracks the chapter number automatically.

5. **Environments.** The `\begin{textexample}` definition draws the highlight banner while preserving line breaks; the theorem and definition boxes share a `tcolorbox` style so they always inherit the palette.

6. **Headings.** `\ChapterHeading`, `\ContentsHeading`, and `\SectionBar` render the decorative chapter and section bars. They encapsulate TikZ drawing commands so authors never have to think about coordinates.

Whenever you adjust typography or spacing, update the explanatory bullets above and commit the rationale with the change. That practice keeps the guide in sync with the code.

2.5 `tex/styles/homework.tex`: Assignment Layout

The homework style mirrors the book without duplicating logic:

1. **Embedding toggle.** Lines 9--25 declare the boolean `\ifMathHomeworkEmbedded`. Standalone PDFs leave it false, but the embedding module flips the flag before loading this file.

2. **Palette adjustments.** `\HomeworkApplyStandalonePalette` recolours accents when printing assignments on their own so pages remain legible on monochrome printers.

3. **Geometry.** The `\usepackage{geometry}` block is conditionally loaded only when producing standalone PDFs. Embedded mode inherits the book geometry to avoid conflicting header widths.

4. **Header metadata.** `\HomeworkSetup` stores course and assignment information; `\HomeworkHeader` formats it within a `fancyhdr` header. The pagestyle is applied automatically at `\AtBeginDocument`.

5. **Print layout helper.** The optional print mode uses `\HomeworkEnablePrintLayout` to reserve vertical space after each `\solutionbox`, making handwritten answers neat.

6. **Embedding helper.** `\HomeworkIncludeInNotes` wraps every embedded assignment: it increments a counter for numbering, injects a table-of-contents entry, temporarily tweaks colours, and then restores the palette after the content finishes streaming in.

The implementation looks long because each behaviour is isolated and commented. This separation pays off when you only need to override one aspect—say, the header metadata—without touching print layout logic.

2.6 `tex/modules/homework-embed.tex`: Embedding Shim

This module contains just enough logic to reuse the homework style inside the book:

1. It ensures `\ifMathHomeworkEmbedded` exists and then forces it to `true`. That flag prevents the homework geometry from loading and lets the book own the page layout.

2. The module assumes `\TexInput` is defined. When the textbook loads it, `\TexInput` expands to the repository-aware import helper documented in Section 1. If you compile the module in isolation (for regression tests), the fallback definition simply delegates to `\input`.

3. Finally, it loads `styles/homework`. No other side effects occur—the caller remains in charge of geometry, counters, and headers.

Despite its size, this file is the connective tissue between the two deliverable types. Without it, homework exercises would either break the textbook layout or lose their consistent styling.

2.7 `tex/system/document-*.tex`: Standalone Wrappers

The wrappers under `tex/system/` eliminate repetitive guards in the project directories:

document-homework.tex Declares `\HomeworkDocumentBegin/\HomeworkDocumentEnd`, loads the homework style when compiled directly, and becomes a no-op when `\ifMathHomeworkEmbedded` is true. Homework sources now contain only content and metadata via `\HomeworkSetup`.

document-notes.tex Handles the lecture/discussion intake workflow. Authors set a title/author/date, input the wrapper, and start capturing notes—the wrapper selects the article class, loads `tex/styles/notes.tex`, applies the standalone header, and calls `\MathNotesDocumentEnd` for them.

document-report.tex Performs the same role for lab reports, wiring the shared `lab-report-template.sty` into both standalone PDFs and appendix inclusions.

bootstrap.tex Still exposes the repository-aware `\TexInput` helper, but most day-to-day files no longer need to reimplement its detection logic—the wrappers import it once.

Adding a new deliverable now means ``write content + point at the wrapper'', which keeps onboarding tight and enforces a single source of truth for every document type.

2.7 Cross-Referencing Intake Material

The snippet below mirrors the reference template: it combines a numbered figure, a styled definition, and a theorem box. The surrounding prose demonstrates how a curated chapter cites real intake material such as **??** from Homework 5 and the discussion scratch work in Problem 1.

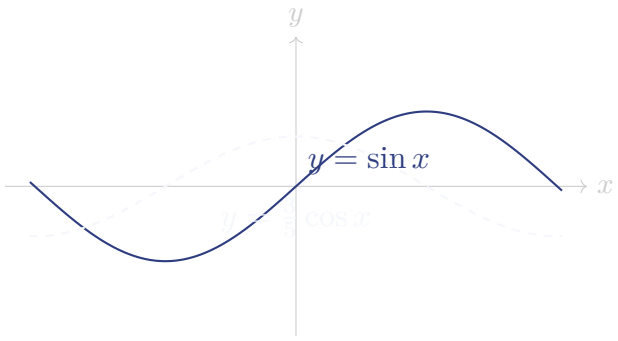

Figure 2.1: Sine and cosine responses used when discussing frequency-domain limits.

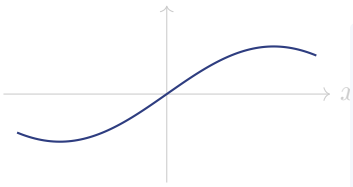

FIGURE 2.1. Margin sketch that mirrors the main plot for quick scanning.

DEFINITION 2.1 *Gain–Bandwidth Product*

The *gain--bandwidth product* is the closed-loop gain of an amplifier multiplied by the frequency at which that gain is measured. It captures how fast a controller may respond before stability erodes.

THEOREM 2.1 *Single-Pole Closed-Loop Response*

With low-frequency gain G_0 and dominant pole ω_p, driving the network captured in Fig. 2.1 produces the transfer function

$$G(j\omega) = \frac{G_0}{1 + j\omega/\omega_{\mathrm{p,eq}}}, \qquad \omega_{\mathrm{p,eq}} = \omega_p + \frac{1}{RC}.$$

Example 2.1 **Interpreting Frequency Sweeps**

$$|G(j\omega)| = \frac{G_0}{\sqrt{1 + (\omega/\omega_{\mathrm{p,eq}})^2}}$$

Solution Sample a handful of frequencies and compare the magnitudes against **??**. The homework entry spells out the algebra behind the matrix manipulations, while Problem 1 retains the scratch work that led to the approximation. Referencing Definition 2.1 keeps the terminology inline with the discussion notes.

Citing the homework entry via **??** lets the chapter summarise the takeaway without repeating every row operation. Likewise, Problem 1 preserves the spontaneous discussion notes that eventually produced Theorem 2.1. By keeping figures, theorem boxes, and cross-references aligned with the intake appendices, you preserve the original aesthetic while keeping the capture workflow fast.

CHAPTER 3 Authoring Workflow

3.1 Build Tooling and Commands

The toolkit relies on `xelatex` (for Unicode-friendly typesetting) and optionally `latexmk` (for incremental builds). The commands below assume you run them from the repository root; feel free to adapt them into shell aliases or editor tasks.

Compile the textbook once `cd projects/textbook/src`

```
xelatex -interaction=nonstopmode -output-directory ../output
main.tex
```

The explicit `-output-directory` flag mirrors the structure explained in Section 1: every build artefact ends up alongside the source in `../output/`.

Continuous documentation build `cd projects/textbook/src`

```
latexmk -xelatex -interaction=nonstopmode -outdir=../output
main.tex
```

`latexmk` watches the `src/` tree and only recompiles what changed. The flags mimic the single-run command so log parsing stays consistent across both approaches.

Compile a standalone homework `cd projects/homework`

```
latexmk -xelatex -interaction=nonstopmode -halt-on-error
-file-line-error homework05.tex
```

Swap `hw05` for any other assignment file. latexmk creates a sibling directory named after the `.tex` file (e.g., `projects/homework/homework05/homework05.*`) so the command above keeps artefacts next to the source without littering the root. Discussion, lecture, and report templates pick up the same behaviour automatically.

Re-run XeLaTeX (or let `latexmk` handle it) after changing geometry, headers, or table-of-contents entries; a second pass updates cross-references and hyperlink targets.

3.2 Adding New Content

Both documentation and homework templates encourage copious inline explanation. Follow the steps in this section so the structure remains predictable.

Adding a documentation chapter

1. Duplicate an existing file under `projects/textbook/src/chapters/` and rename it (for example, `chapter_4.tex`).

2. Update the call to `\ChapterHeading`, set unique labels with `\label`, and immediately add a short paragraph explaining the goal of the chapter. Every code sample should mention which file it modifies so readers can trace the change.

3. No manual table-of-contents edits are necessary—the wrapper file simply provides the styled heading and defers to `\tableofcontents`.

4. Add an `\input` line to `projects/textbook/src/main.tex`, placing it with the other chapter imports. The build script intentionally keeps the chapter list in one place so reviewers can validate the reading order quickly.

Adding homework content

1. Create a new file in `projects/homework/`—the file should contain nothing but problems and solutions wrapped in the environments documented in Section 2. The guard at the top of each file is now the single line `\textbackslashinput\{../../tex/system/document-homework.tex\}`, so the wrapper decides whether to run the standalone preamble.

2. Update the appendix list in `projects/textbook/src/main.tex`. Append another `\HomeworkIncludeInNotes` entry with the display name and the relative path to the homework file. The order of these calls controls the numbering in the appendix table of contents.

All homework copies—embedded, standalone, and print-mode—share the same source file thanks to the document-homework wrapper. This design eliminates the drift that typically happens when exercises are maintained in multiple locations.

Capturing lecture or discussion notes

1. Duplicate the thin templates under `projects/lecture/lecture-example.tex` or `projects/discussion/discussion-example.tex`. Each file simply sets metadata and inputs `../../tex/system/document-notes.tex`, so the dual-use guard lives in one place.

2. Use `\newlecture` (or `\newdiscussion`) to start a section. The macros open a bullet list automatically; reach for `\LecturePoint` or `\DiscussionPoint` to append timestamped bullets without thinking about formatting.

3. When you need inline scratch work, drop into the lightweight `\begin\{LectureExample\}...\end\{LectureExample\}` or `\begin\{DiscussionDrill\}...\end\{DiscussionDrill\}` helpers. These counters are local to the intake templates so they never conflict with the numbered theorem/example environments in the polished chapters.

4. Grayscale versions of the chapter boxes—`\begin\{LectureDefinition\}`, `\begin\{LectureTheorem\}`, and `\begin\{LectureExample\}` (with matching discussion aliases)—live in `tex/styles/notes.tex`. They never touch the main chapter

counters, so you can stamp quick definitions or theorems during class without polluting the curated numbering.

3.3 Embedding Homework in the Documentation

The appendix logic documented in Section 2 and Section 2 is deliberate. When you add a new assignment:

1. Confirm the homework file compiles on its own (with the standalone preamble active). This sanity check ensures solution boxes, theorem counters, and colour overrides all behave as expected before the textbook pulls the file in.
2. Use `\HomeworkIncludeInNotes` with an optional display name if you need a shorter table-of-contents entry. The macro automatically:
   - bumps the appendix counter,
   - pushes a TOC entry,
   - inserts a section bar that matches the rest of the book, and
   - restores the standard palette when the homework finishes.
3. Keep spacing consistent. The macro temporarily redefines `\newpage` and `\clearpage` to avoid blank pages in the appendix. Do not reintroduce those commands inside the homework source unless a page break is absolutely necessary.

Treat each assignment as documentation in its own right: annotate non-obvious solution steps and reference any shared macros the problems rely on.

3.4 Per-Document Overrides

When the shared modules are almost—but not quite—what you need, apply local overrides that respect the layering described in Chapter 2.

- **Palette experiments.** Redefine colour specs before calling `\TexInput\{core/colors\}` (Section 2). For single-document experiments, wrap the override in a group so the change does not leak into subsequent inputs.
- **Geometry tweaks.** Use the pattern from `projects/textbook/src/main.tex`: capture the current `\headwidth`, enter a `\newgeometry` block, and restore everything afterwards. This keeps embedded content from mutating the global layout.
- **Conditional content.** The homework style exposes the boolean `\ifMathHomeworkEmbedded`. Use it to swap instructions or hide print-only banners when the same content is embedded in the book.
- **Custom macros.** Add small helpers to `tex/core/base.tex` only when multiple documents need them. Otherwise, define them near the content block and annotate why they are local to that document.

Always add a brief comment when applying an override; the comment should explain both the symptom and the fix. Future maintainers can then confirm whether the original constraint still holds.

3.5 Maintenance Checklist

Before opening a pull request or tagging a release, walk through this list:

1. **Clean builds.** Run the textbook and every affected homework through `latexmk`. Confirm that each per-file output directory (and `../output/` if you keep using it for the book) only contains fresh artefacts.
2. **Log audit.** Skim the generated `.log` files for overfull boxes, missing references, and font warnings. Appendix sections are most prone to margin issues because they reuse the homework typography.
3. **Documentation sync.** Keep the prose in `projects/textbook/src/chapters/` or the Markdown notes under `docs/` up to date so every new macro or workflow is described in at least one place.
4. **Output hygiene.** Delete stray PDFs, logs, and Synctex files that slipped outside the per-file directories. The root `.gitignore` already blankets common extensions, but explicit cleanup keeps reviewers focused on the meaningful diffs.
5. **Cross-project parity.** If you changed a shared module, recompile at least one standalone homework to ensure the update behaves identically outside the textbook.

Following this routine preserves the repository's goal: every file, directory, and macro exists for a documented reason, and collaborators can reproduce the build without guessing at hidden conventions.

Appendix A

Homework Collection

A.1 Homework Template

Section 0.0

Problem 00

Begin problem prompt here.

Solution

This is an example solution.

Problem 00 — Optional Subtitle

Begin problem prompt with subtitle here.

Solution

Additional solution example.

Appendix B

Discussion Notes

Discussion <pack here> — <sheet or topic> (<YYYY-MM-DD>)

Warm-up Drill

- Capture the raw bullet points from the session.
- **Tag:** Use the optional label to add a cue such as Warm-up/Action.

Main Worksheet

- Call back to earlier sections with Section 1 to keep continuity.
- **Check:** Note any grading or rubric cues.

DEFINITION 1 *Optional subtitle*

Use this grayscale box for quick definitions or terminology that surfaced during the discussion.

THEOREM 1 *Optional subtitle*

Drop rules-of-thumb or impromptu proofs here.

Example 1 **Optional subtitle**

Keep scratch calculations or sketches near the discussion log.

Tip: Add labels (`\label\{...\}`) inside any block to make cross-referencing easy, e.g., Definition 1.

Disc W7 • Problem Q3 — GBW warm-up

Estimate the settling time for a 1% error band when the amplifier from Fig. 2.1 drives a unit-step input. Reference Definition 2.1 for terminology.

Solution. Approximate the closed-loop response with the dominant pole from Theorem 2.1 and apply the standard $4/\zeta\omega_n$ heuristic. Compare the estimate with the measured values in the lecture log.

Appendix C

Lecture Quick Notes

<YYYY-MM-DD> — <topic here>

Warm-up Sweep

- Capture quick bullet points here.
- **Tag:** Use the optional argument to emphasize a lead word.

Main Topic

- Reference earlier sections inline --- see Section 1 for the baseline sweep.
- **Action:** Drop follow-up actions or TODOs that surface during the lecture.

DEFINITION 1 *Optional subtitle*

Drop a grayscale definition box here whenever you need lightweight structure in your intake notes.

THEOREM 1 *Optional subtitle*

Use the theorem block for rules-of-thumb or ad-hoc proofs that are easier to keep near the raw lecture log.

Example 1 **Optional subtitle**

Keep scratch derivations or numeric sanity checks in this example box.

Tip: Add `\label\{...\}` inside any lecture block (e.g., Definition 1) to cross-reference it from chapters or appendices.

Appendix D

Lab Reports

[REPORT NAME OR NUMBER]

Type Your Title Here:

Concise and Informative Subtitle

[Author]

[Lab Partners]

[Course]

[Institution]

[Term]

Prepared on November 7, 2025

Abstract

[150–250 words summarizing purpose, method, quantitative results (with units), and conclusion. No citations.]

Introduction

[Provide scientific context, relevant theory, and objectives. Use equations like $\Delta T_f = iK_f m$. Cite key references.[1]]

Experimental

[Summarize procedure in paragraph form, past tense. Include chemicals, instruments, and conditions. Enough detail for reproducibility.]

Results

[Present data clearly with tables and figures. Units in headers. Include one sample calculation. Summarize results concisely.]

Table D.1: Measured freezing points of solutions

Solution	Concentration (mol/kg)	Freezing Point (°C)
NaCl	0.50	-1.85
NaCl	1.00	-3.72
Urea	1.00	-1.86

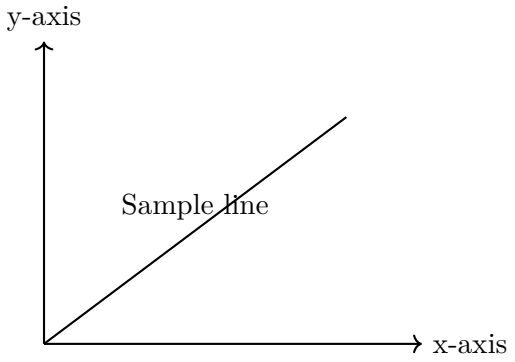

Figure D.1: Simple diagram drawn with TikZ.

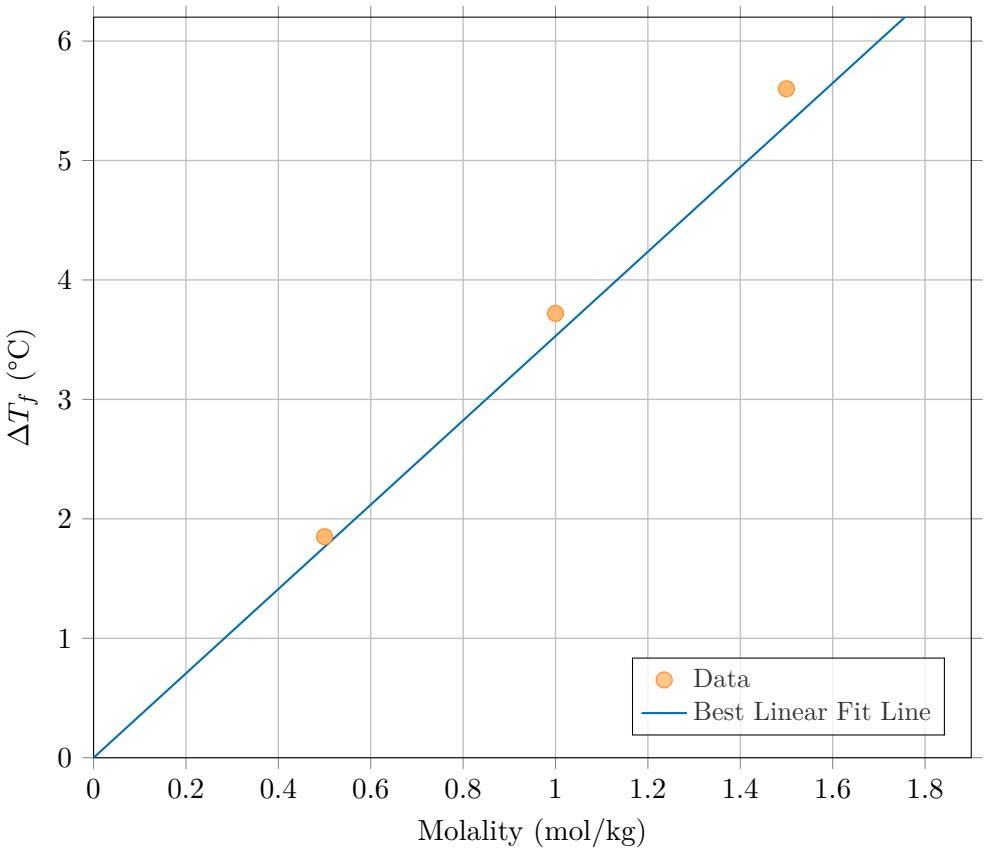

Figure D.2: Freezing-point depression versus molality with best-fit line.

Discussion

[Interpret data. Compare with literature. Discuss sources of error. Connect back to theory and objectives.]

As shown in Figure D.3, the pH rises sharply near the equivalence point, consistent with the expected titration behavior.

Conclusion

[Summarize findings concisely. Confirm whether objectives were achieved. No new data.]

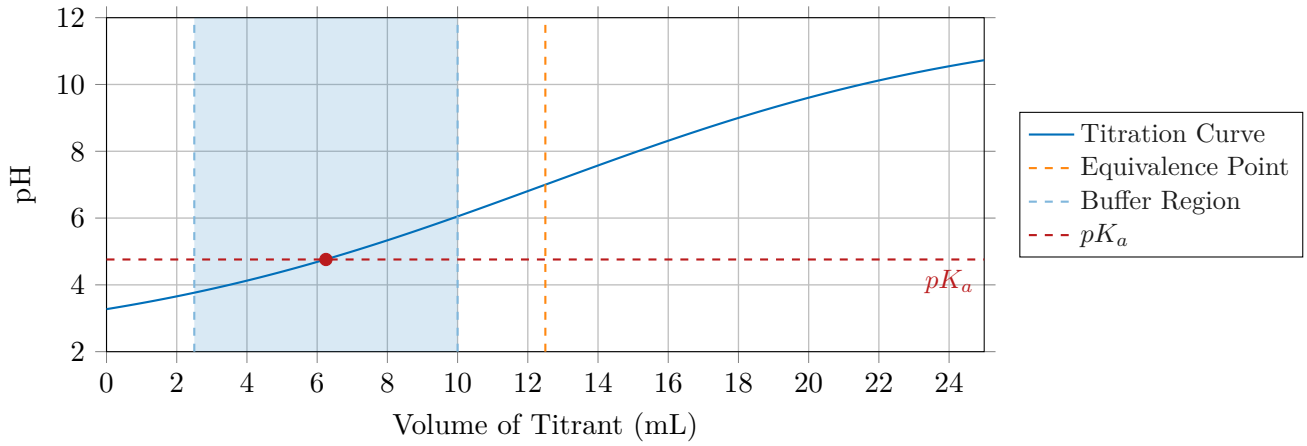

Figure D.3: Example titration curve with key regions highlighted.

Bibliography

[1] Author, A. B.; Author, C. D. *Journal Name* **Year**, *Volume*, page–page.

[2] Author, E. F. *Book Title*; Publisher: Place, Year.

[3] Author, G. H. Title of Webpage. URL (accessed Sept 29, 2025).