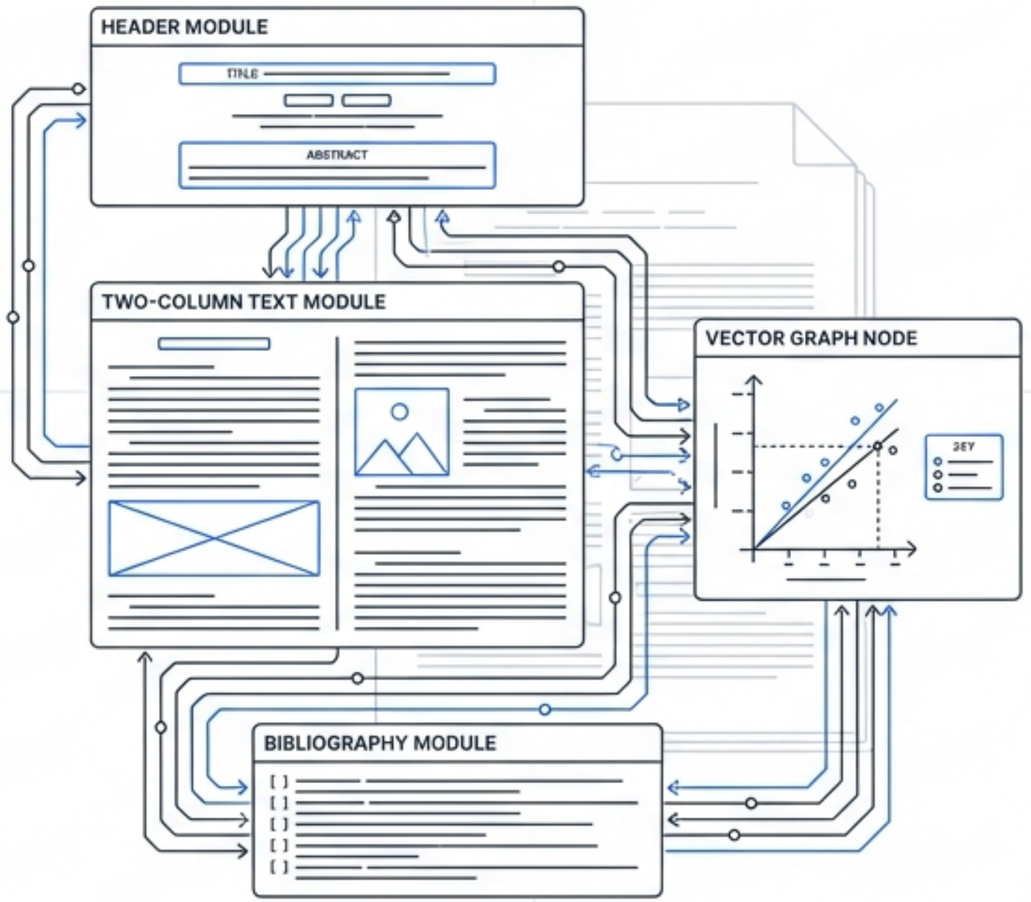
The Lab Standard: Architecting Research Papers

A guide to high-fidelity LaTeX workflows, version control stability, and publication-quality graphics.



Lab Guidelines for LaTeX & Graphics

We Don’t Just Write; We Collaborate.

A document without structure fails under the pressure of multiple authors. Our guidelines prioritize two engineering principles to prevent workflow collapse.

The Chaos of Unstructured Code

```
>>>>> HEAD
function calculate_velocity(t) {
    var v = u + a * t;
    return v;
} function calculate_velocity(t) {
    v = u + a * t;
    return v;
}
=========
function calculate_acceleration(v_i, v_f, t) {
    var a = (v_f - v_i) / t;
    return a;
}
<<<<<< FEATURE
```

Result: Merge Conflict Hell.

The Order of Engineered Code

- **Fix:** Intro typo
- **Feat:** Add experiment data
- **Refactor:** Eq 2 labels

Result: Continuous Integration.

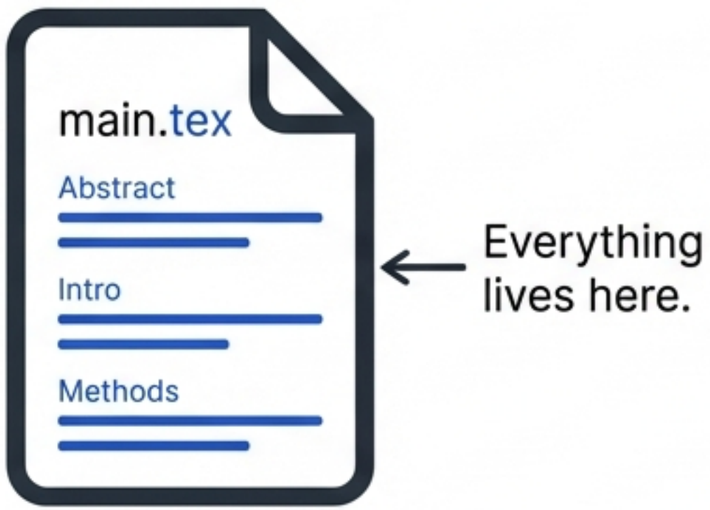
A messy codebase slows down science. A structured codebase accelerates publication.

Scale Your Document Architecture

Choose the structure that matches the complexity of your collaboration.

The Short Doc Stack

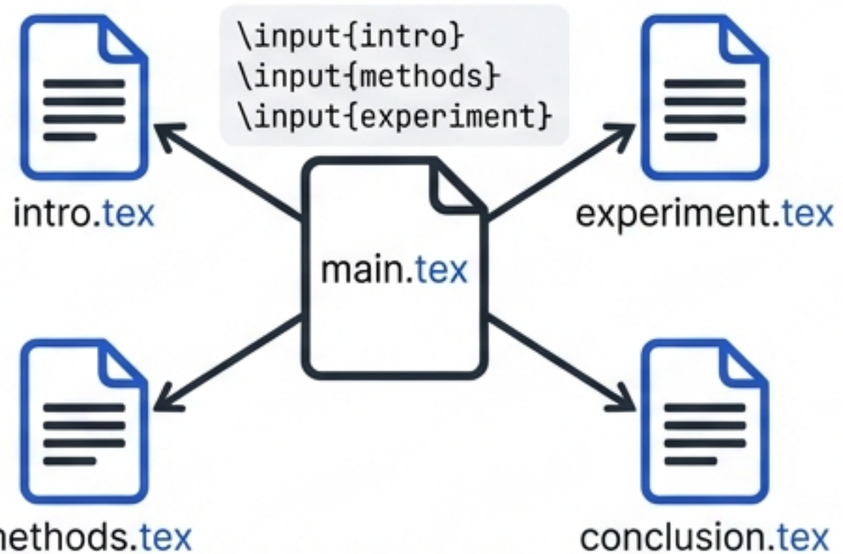
< 15 Pages | Single Primary Author



Warning: Even short docs need splitting if concurrent editing occurs.

The Long Doc Stack

Theses, Books, Collab Reports



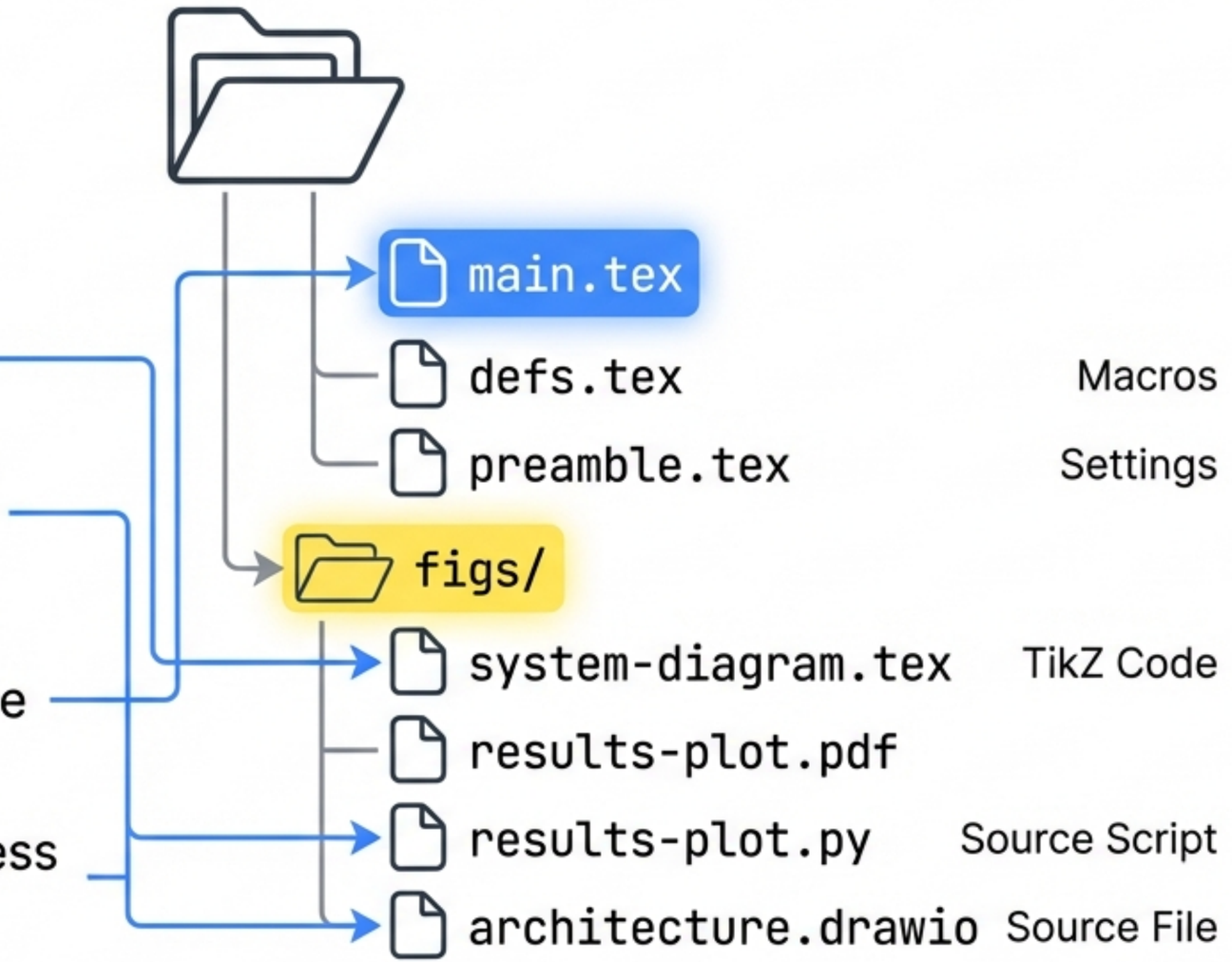
Anti-Pattern Alert:

1. Do not use generic names (e.g., '`section1.tex`').
2. Do not hardcode order in filenames (e.g., '`1-intro.tex`'). Order belongs in '`main.tex`'.

The Asset Pipeline & Directory Hygiene

Keep the workspace clean. Store non-text assets in dedicated directories to maintain a clear root level.

1. **Figures Folder:** Must be lowercase “`figs/`” or “`figures/`”. No spaces.
2. **Vector Sources:** If using TikZ, keep the `.tex` code in “`figs/`” and `\input` it.
3. **Source Preservation:** Save the editable source (Draw.io, Python script) alongside the exported PDF.
4. **Tables:** Write directly in “`main.tex`” unless massive. Use TablesGenerator.com.



The Golden Rule of Version Control

New Sentence = New Line.

Git tracks changes line-by-line. If a paragraph is a single line of code, fixing one typo marks the whole paragraph as “modified” to your collaborators.

Bad Practice - Wrapped Paragraphs

```
- The system is robust. However, latency is high. We propose a
  fix for the architecture to mitigate this issue and improve
  overall performance in the next phase of the project.
+ The system is robust. However, latency is low. We propose a
  fix for the architecture to mitigate this issue and improve
  overall performance in the next phase of the project.
```

One typo fixed → 10 lines of merge conflict.

Lab Standard - Line Breaks

```
10   The system is robust.
11 - However, latency is high.
11 + However, latency is low.
12   We propose a fix.
```

One typo fixed → 1 line changed. Clean diff.

Syntax Details

- No commented lines (%%) between sentences.
- Hard breaks for commands like \begin{equation}.
- New paragraph = Blank line (not \\).

Namespace Logic: The Art of Labeling

Predictability beats creativity. Use the standard schema: **<prefix>:<description>**.

Lookup Table

Prefix	Object Type	Example
`fig:`	Figure	`fig:rmse-plot`
`tab:`	Table	`tab:hyperparameters`
`sec:`	Section	`sec:methodology`
`eq:`	Equation	`eq:objective-function`

The Rules of Scope

`fig:rmse-room-temp`

Forbidden (Bad)

- `fig:figure1` (Non-descriptive)
- `fig:rmse room temp` (Contains spaces)
- `fig:rmse_temp` (Underscore usage)

Accepted (Good)

- `sec:overview:dnn` (Hierarchical)
- `eq:mpc-state-constraint` (Descriptive)
- `tab:ablation-results` (Clear)

Automated Precision: Citations & Refs

Let LaTeX handle the formatting logic. You handle the structure.

The Tilde Tie (~)

Prevent hanging numbers or lonely references.

```
The authors of~\cite{paper} propose...
As seen in Figure~ref{fig:x}...
```

Equation Referencing

Automatic parentheses.

```
Bad:  (\ref{eq:loss}) -> Output: (1)
Good: \eqref{eq:loss} -> Output: (1)
```

Intelligent Referencing (cleveref)

Auto-detect type (Figure, Table, Eq).

```
\usepackage{cleveref}
Referencing \cref{fig:plot} and \cref{tab:data}.
Output Visualization: Referencing Figure 1 and Table 2.
```

Macro Strategy: Semantic Abstraction

“Define WHAT it is, not HOW it looks.”

The Criteria

Define a macro only if:

1. **Complexity:** It is error-prone to type (e.g., complex math layout).
2. **Frequency:** Used often AND improves consistency.
3. **Future-Proofing:** The notation might change (e.g., changing symbol L to J).

Implementation Strategy

Lazy / Bad

```
\newcommand{\tb}{\textbf}
// Just saves keystrokes. Adds no meaning.
```

Semantic / Good

```
\newcommand{\CovMatrix}{\Sigma}
// Encodes meaning. Allows global symbol change.
\newcommand{\LossFn}{L}
// If we switch to 'J', we change it in one place.
```

Macro Safety & Naming Conventions

Prevent spaghetti code and collisions.

Forbidden Actions

1. **NEVER override standard commands.**
   - Example: `\bb` for 'blackboard bold' board bold' breaks bibliography commands.
2. **NEVER use cryptic shorthand.**
   - Example: `\lam`, `\ba`, `\ea`.
3. **AVOID `\def`.**
   - Use `\newcommand` to prevent accidental overwrites.

Naming Standards Cheat Sheet

Math Objects

PascalCase

`\CovMatrix`, `\LossFn`

Text Commands

lowercase `\text...`

`\textblue`, `\texttodo`

Operators

`\DeclareMathOpposito`r

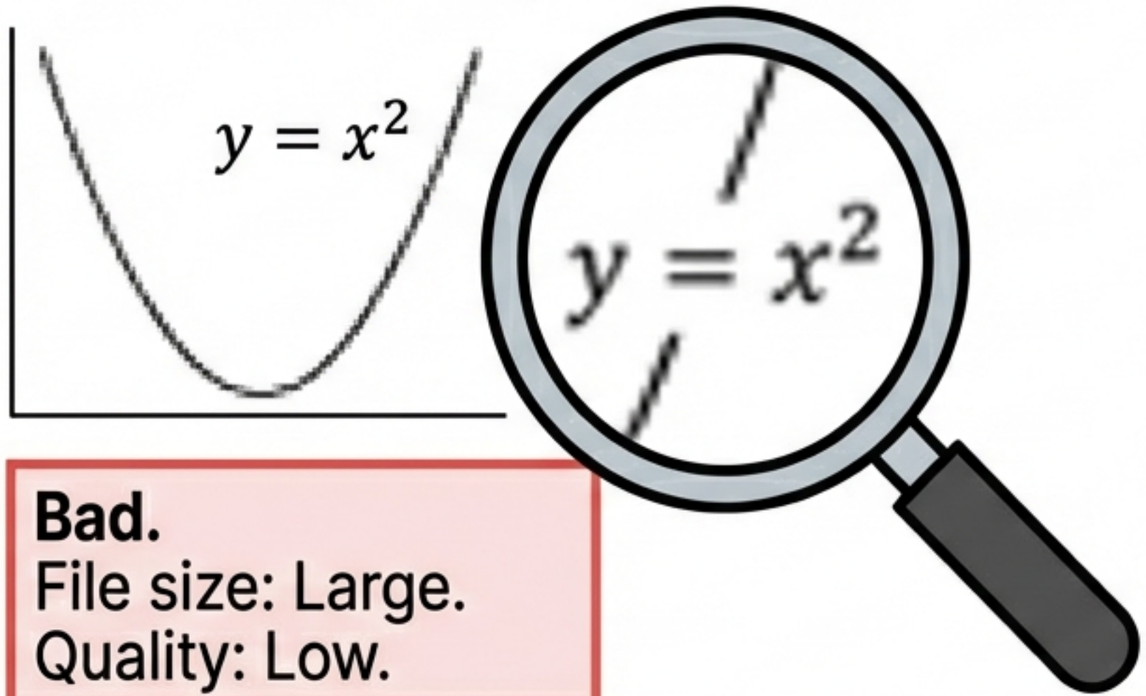
`\argmin` (instead of `\amn`)

Centralize all macros in 'defs.tex' or use 'icpslab.sty'.

Visual Fidelity: The Vector Imperative

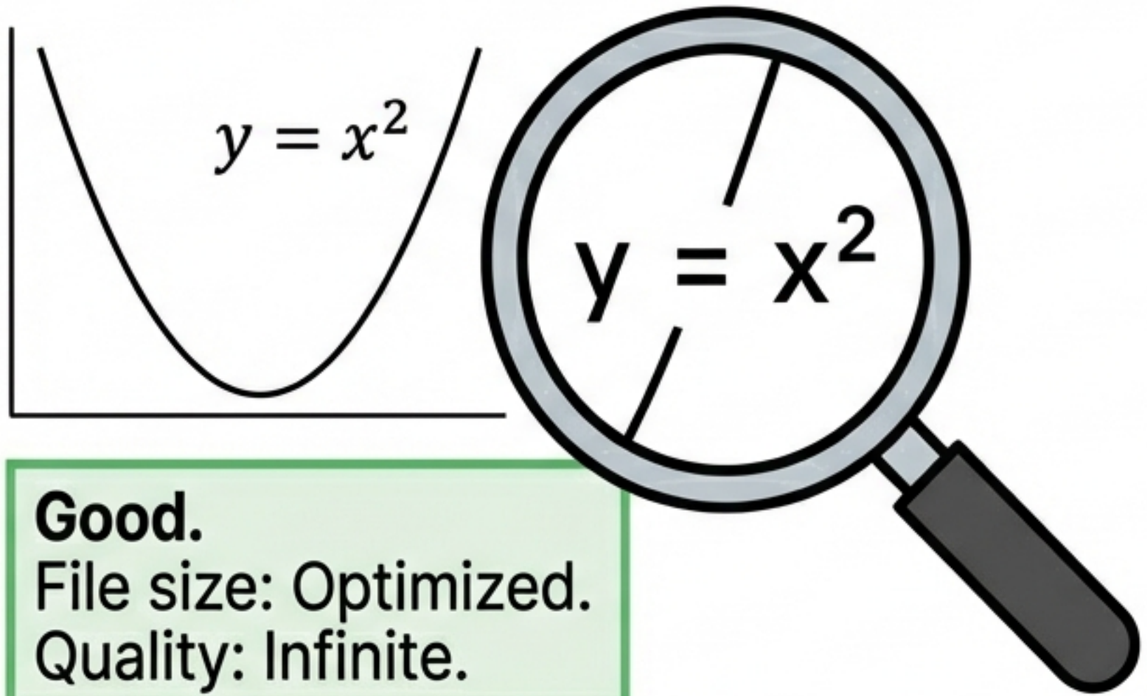
All diagrams and data plots must be resolution independent. No JPEGs for data.

Raster (PNG/JPG)



Bad.
File size: Large.
Quality: Low.

Vector (PDF/EPS)

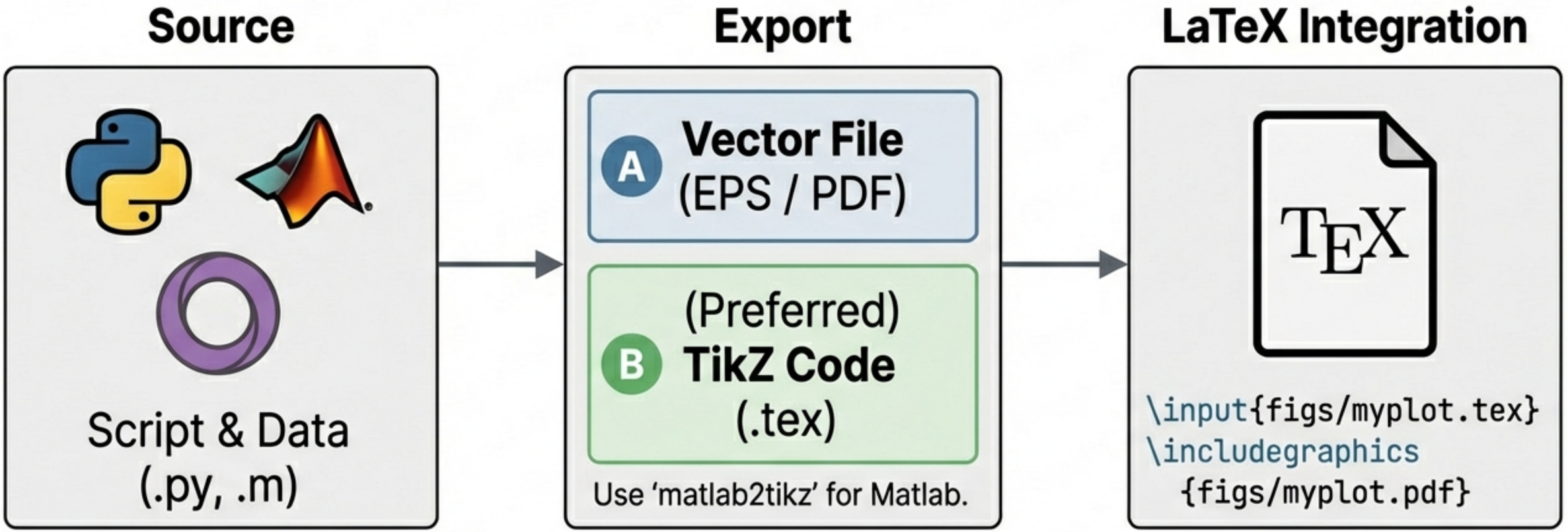


Good.
File size: Optimized.
Quality: Infinite.

The Mandate: Photos = JPEG/PNG. Plots & Diagrams = PDF/EPS/SVG.

The Graphics Workflow (Data to Image)

A plot is not just a picture; it is Data + Code. Ensure reproducibility.



Never take **screenshots** of your data.

Choosing the Right Tool

Data Plots

- TikZ / Pgfplots (Native)
- Python / Matlab (Scripted)

Rule: Export to Vector or TikZ.

Diagrams & Flows

- Draw.io (Quick)
- IPE (Precise GUI)
- Inkscape (Complex)

Rule: Save source XML/SVG.

Network / Graph

- GraphViz
- yEd

Rule: Automated layout engines.

Forbidden

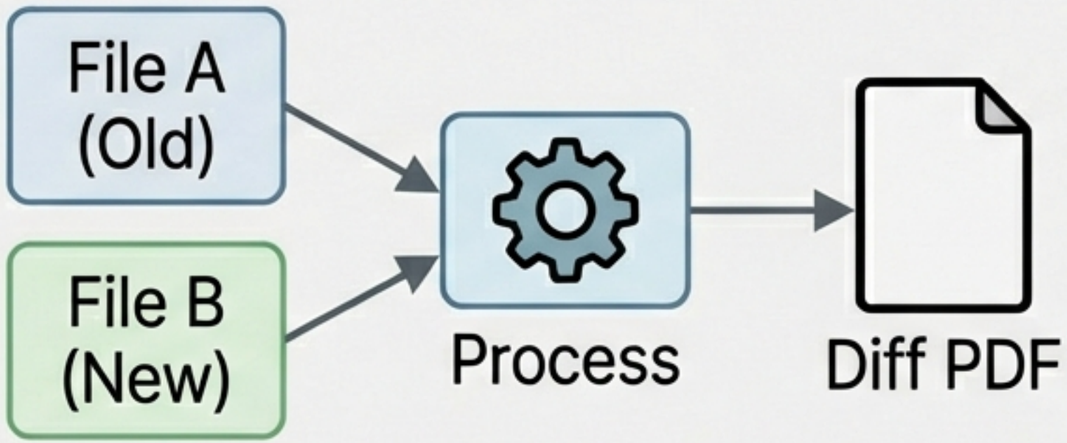
Do not use for publication graphics.

Managing Revisions

Two paths to transparent iteration.

Automated (Git-Based)

`latexdiff`



Zero planning required. Generates visual diff from commit history.

Manual (Collaborative)

`package {changes}`

```
This is \added{new text} and
this is \deleted{old text}.
```

Output Preview

This is new text (in blue) and this is ~~old text (struck out~~ in red).

High control, high effort.
Good for specific feedback loops.

The ‘Pre-Flight’ Checklist

Confirm these items before submission or pull request.

- ✅ **Structure:** Document split logically? No “`section1.tex`”?
- ✅ **Diff-Ready:** One sentence per line? No massive paragraphs?
- ✅ **Labels:** Namespace syntax used (`fig:name`, `sec:name`)?
- ✅ **Macros:** Semantic names? Centralized in `defs.tex`?
- ✅ **Visuals:** All plots are Vector (PDF/TikZ)? No pixelated PNGs?
- ✅ **Source:** Are raw scripts (`Python`/`Matlab`) and figure sources (`Draw.io`) committed?

Create Work That Lasts.

Adopting these standards ensures your research is readable, reproducible, and professional. It transforms your paper from a static document into a robust, collaborative codebase.

Get the Package: `icpslab.sty` View the Repo Tools: tablesgenerator.com | matlab2tikz